Games as Mathematics of Logic and Computation

Norihiro Yamada
Balliol College
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy

Trinity 2017

This thesis is dedicated to my parents
Muneyoshi and Yumiko
for their enduring love and support.

Acknowledgements

First and foremost, I am grateful to my supervisor Samson Abramsky. It all began rather mysteriously with the accidental meeting with him during my travel to Cambridge, without which I would not have come to Oxford, let alone become who I am today. He profoundly understands and supports my views and interests in mathematical and foundational problems, and has given me a complete freedom to pursue them. Also, he has been offering me accurate guidance with his extraordinary depth and breath of knowledge as well as flexibility to embrace new ideas. I learnt a lot from his insights, viewpoints and ways of thinking. It is largely his supervision that led me to find topics, which I feel are *mine*, and become an independent researcher with own passion and perspective.

I would like to express my heartfelt appreciation to my second supervisor Bob Coecke. His care and support encouraged me numerous times, and made the research group more than a working place to me.

I owe important debts to people in relevant academic fields, whom I met directly or indirectly during my DPhil years. Special thanks to Thorsten Altenkirch, Ulrich Berger, Valentin Blot, Pierre Clairambault, Martin Hyland, Alex Kavvos, Kobi Kremnitzer, John Longley, Yoshihiro Maruyama, Luke Ong, Erik Palmgren, Thomas Streicher and Matthijs Vákár. Their comments, feedbacks and encouragements were invaluable to this thesis.

I would like to thank Julie Sheppard, the graduate studies administrator of the department. She has been greatly helpful and supportive so that I could focus on my research as much as possible.

My deepest gratitude goes to Funai Overseas Scholarship, which has been supporting my days in Oxford not only financially but also emotionally through personal interactions. I am grateful to their decision to support me who had not completed a master degree, just believing in my potential and ambition, without which I could not study at Oxford in the first place. I am also grateful to the financial supports from Balliol College.

Friends I met in Oxford enriched my life and developed me as a person, without whom the thesis would not be the same. Special thanks to Kate Brabon, Francesca Bianchi, Filippo Cervelli, Takeshi Chino, Aris Filos-Ratsikas, Atsuyoshi Ishizumi, Jonghoon Kim, Johannes Kleiner, Philip Lazos, Niccolo Lollini, Akira Marusaki, Aiko Morii, Kang Feng Ng, Viviana Ponta, Atsushi Shimada, Mattia Sormani, Fumiko Takahashi, Kei Takahashi and Coco Wing Nam Yiu. In addition, I would like to express my special thanks to my office mate Robin Piedeleu, with whom I shared a number of fun and exciting moments both in academic and private lives.

Finally, I would like to express my deepest appreciation to my family, whose enduring love and support are most valuable and essential to me.

Abstract

Mathematical logic and *theoretical computer science* are the mathematical studies of *logic* and *computation*, respectively, which largely correspond to each other notably by the *Curry-Howard isomorphism*. However, logic and computation have been captured mainly *syntactically*, which may be criticized as conceptually unclear and mathematically cumbersome.

For this point, the field of *semantics* has been developed, i.e., its main aim is to explain logic and computation by mathematical and in particular *syntax-independent* concepts. However, logic and computation have not yet been completely captured semantically, where a main obstacle lies in the point that *proofs* and *programs* are central in these fields, and they are *dynamic*, *intensional* concepts, but mainstream mathematics has been concerned mainly with *static*, *extensional* objects such as sets and functions, lacking in dynamic, intensional structures.

Motivated by this foundational problem, the present thesis is written for the aim of establishing mathematically rigorous, syntax-independent, conceptually natural semantics of some central aspects of logic and computation, particularly their *dynamics* and *intensionality*, that have not been captured very well from a conceptual or mathematical viewpoint, for which our primary goal is to obtain a deeper understanding of logic and computation. Specifically, the thesis is concerned with concepts such as *proof-normalization* in logic (or *reduction* in computation), *higher-order computability*, and *predicates* in logic (or *dependent types* in computation).

Our approach is based on mathematical structures developed in the field of *game semantics* for they are mathematically rigorous, syntax-independent, conceptually natural and flexible enough to model various kinds of logic and computation in a systematic manner; also, they have a good potential to capture dynamics and intensionality of logic and computation. More concretely, we generalize, for each concept mentioned above, an existing variant of *games* to capture the concept, where each case is confirmed by a

certain *soundness* or sometimes *completeness* theorem. Since each of the generalizations is orthogonal to one another, we may combine them into a single framework that provides a unified view on these developments.

As technical achievements of the thesis, many of the generalizations give the first game-semantic interpretations of the corresponding concepts, for which we have introduced a number of novel mathematical structures and proved their properties. Also, conceptually, our games give dynamic, intensional semantics that explains various concepts and phenomena in a natural, intuitive yet mathematically precise manner, e.g., normalization of reduction, Π-, Σ- and Id-types as well as universes in Martin-Löf type theory, incompatibility of Σ-types and classical reasoning, and non-constructivity of Univalence Axiom in homotopy type theory.

Last but not least, since our mathematical structures capture logic and computation more completely, well beyond conventional game semantics, one of their implications would be the *semantics-first-view*: Logic and computation may be understood abstractly and syntax-independently as mathematics, viz., *games*, rather than as syntactic entities to be analyzed by various semantics, which is our intention behind the title of the thesis. In particular, our game-semantic approach enables an *analytic* study of logic and computation, as opposed to more standard *synthetic* ones such as categorical semantics, in the sense that it reduces the two concepts to the highly primitive notions of '(dialogical) arguments' between prover and refuter mathematicians, and '(computational) processes' between a computational agent and an environment, respectively, giving a deeper understanding of the very notion of logic and computation. The former can be seen particularly as a unity of *proof theory* and *model theory*: It gives a syntax-independent formulation of *proofs* that is also a computational description of *models* (by which we mean what formal languages refer to in a loose sense), namely *strategies*, where *provability* and *validity* of a formula simply coincide as the *existence of a winning strategy* on the corresponding game. By this unity, the central questions of *soundness* and *completeness* in logic just disappear. Furthermore, *consistency* of the logic which our games and strategies embody is immediate. These points demonstrate certain technical advantages (in addition to the conceptual naturality mentioned above) of our game-semantic approach to logic.

Contents

Chapter 1

Introduction

On the one hand, *logic* is the study of *reasoning*, i.e., making a conclusion possibly under premises in a sensible way, and it also refers to a method or a law of reasoning. The origin of logic goes back to ancient Greece, China and India in the 5-6th century BC, or even before in Egypt and Babylonia [30]. Its subfield, called *mathematical logic*, is the mathematical study of reasoning in mathematics, whose modern, in particular *symbolic*, treatment began in the late 19th century [61, 149, 50], providing rigorous formalizations of logic and mathematics; the 20th century saw intrinsic powers and limitations of logic with profound consequences [187, 88, 79, 80, 175, 64].

On the other hand, *theoretical computer science* is the mathematical study of *computation*, i.e., mechanical procedures to process some 'information', which was born as a by-product of foundational work in mathematical logic in 1930's [183, 34]. Conceptually by the *BHK-interpretation* [181], which identifies proofs in *intuitionistic logic* [180] with computation in an informal sense, as well as syntactically by the *Curry-Howard isomorphism* [172], which establishes mutual translations for several different pairs of a (formal) logical system (i.e., a syntactic formalization of logic) and a programming language, logic and computation largely correspond to each other.[1]

As implied above, logic (in the sense of a method or a law of reasoning) and computation have been captured mainly via *syntactic* concepts. However, although syntactic approaches give convenient embodiments of and practical tools for logic and computation, they are not very suitable for conceptual or mathematical clarification of them, which the present thesis is primarily concerned with, for the following reasons:

- From the *semantics-first-view*, i.e., the view that semantic concepts must come first, and syntax merely provides formal notations and symbolic calculi for them,

[1]The Curry-Howard isomorphism, also called the *Curry-Howard correspondence*, may be thought of *a posteriori* as a particular (syntactic) formalization of the BHK-interpretation: The former takes programs as a formal counterpart of computation in the sense of the latter.

which is our principle and also conventional in mathematics, syntax *per se* (i.e., without what it refers to) is rather mysterious and conceptually hard to capture;

- From our viewpoint on mathematics which is a rigorous study of *abstract essence* of various concepts and phenomena, syntactic definitions often contain superfluous details, where non-canonical design choices are usually involved, and thus they are not abstract enough for a mathematical study of logic and computation.

In other words, logic and computation have not yet been completely captured since they have been formalized mainly syntactically, and it is in general difficult for a syntactic formalization to provide a conceptual understanding or a mathematical abstraction in a satisfactory fashion.[2] In particular, their *dynamics* and *intensionality* have been rarely formulated syntax-independently, though they are intrinsic parts of logic and computation, mainly because traditionally mathematics has been concerned mostly with *static*, *extensional* objects such as sets and functions.

Motivated by this point, the present thesis is written for the aim of capturing, in a mathematically rigorous, syntax-independent, conceptually natural manner, some central aspects of logic and computation that have not yet been completely clarified in that way. Specifically, the thesis addresses the following four concepts:

1. *Dynamics* and *intensionality* of logic and computation;

2. *Mathematical models of computation* beyond classical computation;

3. The *meaning explanation of Martin-Löf type theory*;

4. The relation between *classical*, *intuitionistic* and *linear* logics (only casually).

Remark. It is admittedly difficult to formalize the distinction between syntactic (or syntax-dependent) and semantic (or syntax-independent) concepts; nevertheless, our general guideline in this thesis, though informal, is as follows:

- The substance of syntactic concepts is *symbols*, which *per se* do not denote anything, and operations and properties on them are defined in terms of symbol manipulations, e.g., substitution, α-equivalence, confluence and consistency;

- Semantic concepts do not consist of symbols, where notation for them is just for convenience for mathematicians and inessential for their 'ontology' (for instance, natural numbers are invariant however they are written, e.g., binary or decimal).

[2]In fact, this perspective prohibited Kurt Gödel from accepting the λ-calculus as a foundation of computability, and also led Dana Scott and Jean-Yves Girard to develop denotational semantics and geometry of interaction, respectively, which we shall briefly explain in Sections 1.1 and 1.2.

1.1 Dynamics and Intensionality

Dynamics and *intensionality* are intrinsic parts of logic and computation, but they have been rarely formalized syntax-independently. Thus, for a deeper understanding of logic and computation, we need new mathematics of dynamics and intensionality.

1.1.1 Dynamics of Logic and Computation

Among programming languages, we focus on *functional* languages [24, 49] in this thesis since they are defined solely in terms of syntax (as opposed to *imperative* languages for which *states* of a computer are essential [188]), and thus conceptually closer to formal logical systems and mathematically simpler. In a functional language $\mathscr{L}$, computation proceeds by *evaluating* a program (or a *term*) P to a *value* V:

$$\mathsf{P} \rightarrow \mathsf{P}_1 \rightarrow \mathsf{P}_2 \rightarrow \cdots \rightarrow \mathsf{P}_\mathsf{n} = \mathsf{V}$$

where $\rightarrow$ represents the one-step evaluation or *small-step operational semantics* [152] of the language $\mathscr{L}$. For instance, assume that $\mathscr{L}$ has the *numeral* $\underline{\mathsf{n}}$ for each natural number $n \in \mathbb{N}$, where $\mathbb{N}$ is the set of all natural numbers, as well as operations succ and double that satisfy respectively $\mathsf{succ}(\underline{\mathsf{n}}) \rightarrow \underline{\mathsf{n+1}}$ and $\mathsf{double}(\underline{\mathsf{n}}) \rightarrow \underline{\mathsf{2n}}$ for all $n \in \mathbb{N}$. Then, the computation of the program $\mathsf{double}(\mathsf{succ}(\underline{\mathsf{5}}))$ would be:

$$\mathsf{double}(\mathsf{succ}(\underline{\mathsf{5}})) \rightarrow \mathsf{double}(\underline{\mathsf{6}}) \rightarrow \underline{\mathsf{12}}.$$

This kind of process is called *reduction* or *rewriting*, which is a syntactic formalization of what we call *dynamics of computation*. Note that such dynamics may be seen as giving *semantics* of programs in the sense that it describes how each program is executed in the language $\mathscr{L}$; in fact, there is such an approach to formal semantics of programming languages, called *operational semantics* [188, 82]. There are two variants of operational semantics: *big-step* and *small-step* ones; the former evaluates each program to its value at a time, while the latter does in a step-by-step fashion (as seen above). Although the former is often simpler, we focus on the latter in this thesis as it gives a finer description of dynamics of computation.

Similarly, there is what should be called *dynamics of logic* as well. In a *natural deduction* style logical system, proposed by Gerhard Gentzen in his PhD thesis [65, 66], if a proof of a formula contains a *detour*, then we may obtain a shorter proof of the same formula by eliminating the detour; this meta-theoretic process is called *proof-normalization* [180], embodying dynamics of logic. Moreover, a *sequent calculus* style system, also proposed in the same PhD thesis, includes such detours 'internally' as a rule of inference, called *cut*, and thus dynamics corresponds to the meta-theoretic process to eliminate cuts from a proof, called *cut elimination* [180]. However, note that unlike programming languages dynamics of logic is not formalized inside a logical system though they are rigorously defined as meta-theoretic algorithms.

1.1.2 Intensionality of Logic and Computation

Next, let us point out that computation is an *intensional* concept in the sense that not only what its result (or *extension*) is but also *how it computes*, i.e., its *algorithm*, matters in an informal sense. For instance, the programs $\mathsf{double}(\mathsf{succ}(\underline{5}))$ and $\mathsf{succ}(\mathsf{succ}(\mathsf{double}(\underline{5})))$ clearly have the same value, namely $\underline{12}$, but different algorithms: The former first calculates the successor of 5 and then doubles the result, while the latter first doubles 5, and then apply the successor operation twice to the result. Thus, these two programs are *extensionally equal* but *intensionally different*. Let us call informally the equality of algorithms *intensional equality of computation*.

One may wonder if *α-equivalence* [85] of programs (i.e., the equality of programs up to renaming of bound variables) would be a syntactic counterpart of intensional equality of computation. However, syntactic formalization of intensional equality is rather difficult, and it has not been completely established. The point here is that α-equivalence describes *how each program is constructed*, but two different programs with respect to α-equivalence may describe the same algorithm, e.g., consider the two programs $(\lambda \mathsf{x}.\underline{0})\underline{1}$ and $(\lambda \mathsf{y}.\underline{0})\underline{2}$, which are not α-equivalent, but their algorithms are the same, namely to just output zero.[3] Hence, intensional equality of computation does not have an established counterpart in syntax; it is finer than *$\beta\eta$-equivalence* [85] (i.e., the extensional equality in syntax) but coarser than α-equivalence.

Similarly, we may consider *intensional equality of proofs* again in an informal, non-syntactic sense. However, in contrast to the case of computation, a proof should be identified in a sense with *how it derives its conclusion*, i.e., *how it is constructed.* Therefore, there is a syntactic counterpart of intensional equality of proofs, namely the syntactic equality of proofs *on the nose* in a logical system. But if we identify proofs with their algorithmic behavior in the spirit of the BHK-interpretation, then there would be no established counterpart of the equality in syntax (though there have been some approaches in the literature, e.g., *proof-nets* [70]) by the same reason as in the case of computation.

[3]This point will be clearer when we give a formal semantics of this phenomenon in Chapter 3.

1.1.3 Denotational Semantics

For a 'true "understanding" of a program' [161], which is similar to our motivation, Dana Scott and Christopher Strachey initiated in 1970's the approach to semantics of programming languages by mathematical objects, called *denotational semantics* [164]. Based on Scott's beautiful *domain theory* [162, 67, 11], this mathematical approach has been highly successful, giving a deeper understanding of programs as well as mathematical tools for language analysis, verification and design. Moreover, via the Curry-Howard isomorphism, it has been applied to logical systems as well.

However, the conventional view of denotational semantics is *static* and *extensional* in the following sense: A basic requirement of denotational semantics is *soundness*, i.e., it identifies a program or a proof with its value, and thus it has an intrinsic limitation in capturing dynamics and intensionality of logic and computation.

1.1.4 Towards Semantics of Dynamics and Intensionality

To sum up, on the one hand we have the *operational* side of logic and computation, which captures their dynamics and intensionality (though not perfectly as we have seen above) but not their conceptual underpinning or abstract essence, and on the other hand we have their *denotational* side, which gives a deeper understanding of logic and computation but limited to their static, extensional aspects. Therefore, what we need for a more comprehensive understanding of logic and computation would be a mathematical, in particular *syntax-independent*, semantics of their dynamics and intensionality, which combines the operational and denotational approaches.

In fact, this perspective is nothing new. In [78], Jean-Yves Girard mentions the dichotomy between the *static* and *dynamic* viewpoints in logic and computation; the former identifies proofs and programs with their *denotations* (i.e., *results* of their computations in an ideal sense), while the latter focuses on their *senses* (i.e., *algorithms* or *intensionality*) and *dynamics*. He points out that a *mathematical* formulation of the former has been relatively well-developed, but it is not the case for the latter; the treatment of senses has been based on ad-hoc *syntactic manipulations*. He then emphasizes the importance of a *mathematical formulation of senses*:

> The establishment of a truly operational semantics of algorithms is perhaps the most important problem in computer science [78].

The operational aspect of logic and computation is the first concept which the present thesis captures mathematically and syntax-independently in Chapter 3. More specifically, we shall establish an interpretation $[\![\_]\!]_{\mathcal{D}}$ of a programming language $\mathscr{L}$ with a small-step operational semantics $\rightarrow$ and a *syntax-independent* operation $\blacktriangleright$ that satisfy the following ***dynamic correspondence property (DCP)***: $\mathsf{M}_1 \rightarrow \mathsf{M}_2$ only if $[\![\mathsf{M}_1]\!]_{\mathcal{D}} \neq [\![\mathsf{M}_2]\!]_{\mathcal{D}}$ and $[\![\mathsf{M}_1]\!]_{\mathcal{D}} \blacktriangleright [\![\mathsf{M}_2]\!]_{\mathcal{D}}$ for any programs M_1 and M_2 of $\mathscr{L}$. Note that the 'only if' direction corresponds to a certain *soundness* property of the interpretation $\blacktriangleright$ of $\rightarrow$.[4] Note also that the interpretation $[\![\_]\!]_{\mathcal{D}}$ is *finer* than the usual (sound) denotational semantics because $\mathsf{M}_1 \rightarrow \mathsf{M}_2$ implies $[\![\mathsf{M}_1]\!]_{\mathcal{D}} \neq [\![\mathsf{M}_2]\!]_{\mathcal{D}}$. Thus, the interpretation $[\![\_]\!]_{\mathcal{D}}$ and the operation $\blacktriangleright$ capture *intensionality* and *dynamics* of computation, respectively (n.b., we do not have soundness, let alone completeness, for intensional equality as it has not been captured in syntax in the first place as mentioned before though our semantic equality per se appears a reasonable one).

Although our framework in this thesis is intended to be a *general* approach, and it should be applicable to a wide range of logics and computations, as the first step, we shall focus on a finite fragment of the prototypical functional programming language *PCF* [163, 151] customized for our purpose.

1.2 Mathematical Models of Computation

Various *mathematical models of computation* have been proposed in the literature; however, there are only a few models of *non-classical computation* that investigate the very notion of computation in an as primitive, intrinsic manner as *Turing machines*. Moreover, there has been no established theory of computation beyond symbolic or low-level computation. We shall develop a more general, abstract model of computation to solve these two problems.

1.2.1 Turing Machines

Turing machines (TMs) introduced in the classic work [183] by Alan Turing have been widely accepted as giving a mathematical foundation of '(effective) computability' or 'effectivity' of functions on (finite sequences of) natural numbers, which we call ***classical computability***, ***Church-Turing computability*** or ***recursiveness*** in this thesis. This is because 'computability' of a function intuitively means the very existence of an algorithm that implements the function's input/output-behavior, and TMs are none other than a mathematical formulation of this informal concept.

[4]On the other hand, the opposite, 'completeness' property does not hold, e.g., $(\lambda \mathsf{x}.\underline{0})((\lambda \mathsf{y}.\mathsf{y})\underline{1}) \rightarrow (\lambda \mathsf{x}.\underline{0})\underline{1}$ and $(\lambda \mathsf{x}.\underline{0})((\lambda \mathsf{y}.\mathsf{y})\underline{1}) \not\rightarrow (\lambda \mathsf{x}.\underline{0})\underline{2}$, but $[\![(\lambda \mathsf{x}.\underline{0})\underline{1}]\!]_{\mathcal{D}} = [\![(\lambda \mathsf{x}.\underline{0})\underline{2}]\!]_{\mathcal{D}}$ in our interpretation $[\![\_]\!]_{\mathcal{D}}$.

Note that other pioneering approaches to classical computability do not have this foundational nature: The *λ-calculus* [34], *combinatory logic* [159, 45] and *Post canonical systems* [153] are *syntactic* entities, and so they do not formalize how to 'effectively' achieve the specified symbol manipulations in an as primitive level as TMs (e.g., *β-reduction* [85] in the λ-calculus does not describe how to execute itself, which is in fact a motivation for *explicit substitution* [158]); *partial recursive functions* [107] are *axiomatic*, i.e., without defining what algorithms are, they inductively define 'computable functions' directly by certain axioms, and so they are not as foundational as TMs. For instance, Kurt Gödel found Alonzo Church's proposal to take the λ-calculus as a foundation for classical computability 'thoroughly unsatisfactory', but he enthusiastically accepted TMs as such a foundation [171]; also Church stated:

> Computability by a Turing machine . . . has the advantage of making the identification with effectiveness in the ordinary (not explicitly defined) sense evident immediately [35].

Later in the literature, various mathematical models of computation similar to TMs such as *register machines (RMs)* [46] have been proposed. However, since their ideas and features are similar to those of TMs, below we just mention TMs as a representative of these similar models.

1.2.2 Beyond Classical Computation

Today, however, there are various kinds of computation in practice *whose aim is processes per se*, rather than to implement functions, such as operating systems and web browsers. They are often *interactive* and *non-terminating*, for which TMs do not fit very well [26, 4, 2, 51]. Also in mathematics, there are various kinds of *non-classical computation*, where by ***classical computation*** we mean what merely implements a function on natural numbers, for which TMs again have some limitations.

As an example of non-classical computation, consider *higher-order computation* [122], i.e., computation that may take (as an input) or produce (as an output) another computation, which abounds in mathematics, e.g., quantification in mathematical logic, differentiation in analysis or simply an application $(f, a) \mapsto f(a)$ of a function $f : A \to B$ to an element $a \in A$. However, TMs cannot capture higher-order computation in a natural or systematic manner. In fact, although TMs may compute on symbols that *encode* other TMs, e.g., consider *universal TMs* [116], they cannot compute on 'external behavior' of input computation, which in particular implies that the input is limited to a recursive one; however it makes perfect sense to consider computation on non-recursive objects such as non-recursive real numbers [122]. For this point, one may wonder *oracle TMs* [115] may treat an input computation as an *oracle*, a 'black-box-like' computation that does not have to be recursive, but it is like a function (rather than a process) that computes in just a single step, which appears conceptually mysterious and mathematically ad-hoc (another approach is to give an input computation as a possibly infinite sequence of symbols on the input tape, but it may be criticized in a similar manner).

On the other hand, most of the other existing models of higher-order computation are either syntactic (such as programming languages and *nested sequential procedures* [122]), axiomatic (such as *Kleene's schemata* S1-S9 [108, 109]) or extrinsic (i.e., reduced to classical computation by *numbering* whose 'effective computability' is often left imprecise [46, 122]); thus, they are not as primitive or foundational as TMs. Moreover, unlike classical computability whose various definitions turn out to be equivalent, such an equivalence result for higher-order computability has been rarely established [122], i.e., we have not yet captured higher-order computability, let alone higher-order computation, in a satisfactory fashion. For this reason, we believe that it would be a key step towards a 'correct' notion of higher-order computation and computability to establish a 'TMs-like' model of computation beyond classical one.

1.2.3 High-Level vs. Low-Level Computational Processes

Perhaps more crucially than the limitation for non-classical computation mentioned above, one may argue that TMs are not appropriate as mathematics of *high-level* computational processes[5] for their computational steps are often too *low-level* to see what they are supposed to compute. This point is relatively ignorable if one is only concerned with classical computation as it suffices to read off inputs and outputs; however, it is no longer the case for non-classical, e.g., higher-order, one. Thus, we need mathematics of high-level computational processes that explain contents or purposes of low-level computational processes such as TMs'[6]. Also, what TMs formulate is *symbol manipulations*, but the content of computation on mathematical, semantic, non-symbolic objects seems completely independent of its symbolic representation, e.g., to add numbers or to take the union of sets (as a *process* not as a function).

[5]Throughout the present thesis, we informally use the terms *computational processes* almost as synonyms of *computation*, but they put more emphasis on 'processes'.

[6]This idea is similar to that of denotational semantics of programming languages, but there is an important difference: Denotational semantics interprets programs usually by (extensional) functions and identifies them with their values, but we are concerned with (intensional) *processes*.

Hence, it would be rather appropriate, at least from conceptual and mathematical points of view, to formulate high-level computational processes in a more abstract, in particular *syntax-independent*, manner, in order to explain low-level computational processes, and regard the latter as executable *symbolic implementations* of the former that witness its 'effective computability'.

1.2.4 Towards Mathematics of Computational Processes

To summarize, it seems appropriate, fruitful and meaningful for both conceptual and technical reasons to develop, in an intrinsic, syntax-independent, non-axiomatic, non-inductive manner, mathematics of abstract, high-level computational processes as well as symbolic, low-level ones beyond classical computation such that the former is defined to be 'effectively computable' if it is representable by the latter. In fact, this (or similar) perspective is nothing new, and shared with various prominent researchers; for instance, Robin Milner stated:

> ... we should have achieved a mathematical model of computation, perhaps highly abstract in contrast with the concrete nature of paper and register machines, but such that programming languages are merely executable fragment of the theory ... [133]

We shall address this problem in Chapter 4. However, since there are so many kinds of computation, e.g., parallel, concurrent, probabilistic, non-deterministic, quantum, etc., as the first step, we shall focus on a certain kind of higher-order, *sequential* (i.e., at most one computational step may be performed at a time) computation which the programming language PCF embodies.

1.3 Meaning Explanation of MLTT

Martin-Löf type theory is one of the most prominent foundations of constructive mathematics, which can be seen in a sense as a syntactic formalization of its *meaning explanation*. Thus, a mathematical, syntax-independent semantics of MLTT that is in accordance with the meaning explanation should be considered as its *standard model*, which is essential for a deeper understanding of MLTT, and also it would give a mathematical foundation of *constructivism* in the philosophy of mathematics.

1.3.1 Martin-Löf Type Theory

Martin-Löf type theory (MLTT) [125, 126, 127] is an extension of the simply-typed λ-calculus [36] whose logical part, under the Curry-Howard isomorphism, corresponds to intuitionistic predicate logic (whose predicates are higher-order but *predicative* [58]), for which the extension is made mainly by *dependent types*, i.e., types depending on terms [172]. It actually *internalizes* the isomorphism in the sense that its types and programs may be read respectively as formulas and proofs; it is not only a programming language but also a logical system.

For instance, to prove that each natural number $n \in \mathbb{N}$ has a unique pair of a *quotient* q and a *remainder* r with respect to a given integer $m > 0$, i.e.,

$$\begin{aligned}&\forall n, m \in \mathbb{N}.\, m > 0 \Rightarrow \exists q, r \in \mathbb{N}.\, r < m \wedge n = q \cdot m + r \\ &\qquad\qquad\qquad\qquad \wedge (\forall \tilde{q}, \tilde{r} \in \mathbb{N}.\, \tilde{r} < m \wedge n = \tilde{q} \cdot m + \tilde{r} \Rightarrow q = \tilde{q} \wedge r = \tilde{r})\end{aligned}$$

in MLTT is to write a program of type

$$\begin{aligned}&\mathsf{\Pi}_{\mathsf{n},\mathsf{m}:\mathsf{N}}(\mathsf{\Sigma}_{\mathsf{l}:\mathsf{N}}\mathsf{m} = \mathsf{l} + 1) \to \mathsf{\Sigma}_{\mathsf{q},\mathsf{r}:\mathsf{N}}\mathsf{\Sigma}_{\mathsf{k}:\mathsf{N}}\mathsf{m} = \mathsf{r} + \mathsf{k} + 1 \wedge \mathsf{Id}_{\mathsf{N}}(\mathsf{n}, \mathsf{q} \cdot \mathsf{m} + \mathsf{r}) \\ &\qquad\qquad \wedge(\mathsf{\Pi}_{\tilde{\mathsf{q}},\tilde{\mathsf{r}}:\mathsf{N}}\mathsf{\Sigma}_{\tilde{\mathsf{k}}:\mathsf{N}}\mathsf{m} = \tilde{\mathsf{r}} + \tilde{\mathsf{k}} + 1 \wedge \mathsf{Id}_{\mathsf{N}}(\mathsf{n}, \tilde{\mathsf{q}} \cdot \mathsf{m} + \tilde{\mathsf{r}}) \to \mathsf{Id}_{\mathsf{N}}(\tilde{\mathsf{q}}, \mathsf{q}) \wedge \mathsf{Id}_{\mathsf{N}}(\tilde{\mathsf{r}}, \mathsf{r}))\end{aligned}$$

that compute the quotient q and the remainder r, and demonstrate that they are in fact a quotient and a remainder that are unique with respect to n and m, where $\mathsf{\Pi}$, $\mathsf{\Sigma}$ and Id_{N} are the type-theoretic universal and existential quantifications and (intensional) equality on natural numbers, respectively (see Section 5.2 for the details).

MLTT was proposed by Per Martin-Löf in 1970-1980's through several revisions as a foundation of constructive mathematics (in the sense that it plays a similar role for constructive mathematics to the *Zermelo-Fraenkel set theory with the Axiom of Choice (ZFC)* for classical mathematics [54]). For instance, MLTT is proof-theoretically much stronger than *Heyting Arithmetic (HA)* (even than *higher-order HA*) [178]; it is even possible to interpret the *constructive Zermelo-Fraenkel set theory (CZF)* in MLTT equipped with *well-founded tree types* [16, 182]. Furthermore, based on the novel homotopy-theoretic interpretation, an extension of MLTT, called *homotopy type theory (HoTT)*, has been proposed, connecting a priori different areas of mathematics and having a potential to be a powerful, practical foundation of mathematics [184].

Strictly speaking, there are *intensional* and *extensional* variants of MLTT; the latter is an extension of the former by the principle that identifies judgmental and propositional equalities (see Section 5.2). We shall focus on the intensional variant in this thesis because it is computationally more desirable than the extensional version, e.g., type checking is decidable in the former but not in the latter, while retaining proof-theoretic strength. Furthermore, the new development of HoTT is possible only for the intensional one, i.e., the intensional version is not only a 'computational compromise' of the extensional one but also a *positive* theory in its own right.

In addition, MLTT has been also a subject of active research in computer science because it can be seen as a programming language, and one may extract programs from its proofs in such a way that extracted programs are 'correct by construction' [38, 160]. In fact, MLTT and similar *dependent type theories*, i.e., type theories with dependent types, have been implemented as proof assistants such as *Nuprl* [38], *Coq* [177] and *Agda* [141]. They are used to formalize mathematical theorems and proofs by programming as well as to give formal correctness of computer software.

1.3.2 Meaning Explanation

Importantly, MLTT is conceptually based on the *meaning explanation* [126] in the sense that Martin-Löf himself explains and justifies its axioms and (inference) rules by the meaning explanation though it is informal or *pre-mathematical* [54]. In fact, the meaning explanation is usually considered to be an essential ingredient of MLTT (see, e.g., [154]). Thus, we believe that a mathematical, syntax-independent semantics of MLTT that is in accordance with the meaning explanation can be considered as its *standard model*, which would be essential for a deeper understanding of MLTT.

Another point is that the meaning explanation *per se* (without MLTT) is a highly reasonable, systematic approach to foundations of constructive mathematics in the following sense. First, the meaning explanation is a fundamental conceptual underpinning of the very notion of constructive mathematics, whose argument is quite convincing, though pre-mathematical, being a *refinement* as well as an *extension* of the BHK-interpretation (see Section 5.2.11 for more on this point). Here, syntax is inessential, and one may even say that MLTT is merely a syntactic embodiment of the meaning explanation. Next, unlike other foundations of constructive mathematics such as *constructive set theory* [138] and *explicit mathematics* [57], the meaning explanation does not separate logic from mathematics; rather, it provides a unifying framework, in which logic becomes a particular part of mathematics; in this sense, it gives a more systematic perspective than other approaches.

1.3.3 Towards a Mathematical Foundation of Constructivism

Thus, a mathematical, syntax-independent formalization of the meaning explanation, if achieved, would be a fundamental underpinning of the very notion of constructive mathematics or *constructivism* in the philosophy of mathematics [181, 20].

However, such a standard model of MLTT has not been completely established, e.g., *realizability models* have been either syntactic or too low-level to capture type-theoretic phenomena (see Section 5.1.3). We shall address this problem in Chapter 5.

1.4 Classical, Intuitionistic and Linear Logics

Classical, *intuitionistic* and *linear* logics[7] are perhaps the three most established logics. Syntactically, their definitions and relations are clear; however, for a deeper understanding of these logics, it is desirable to have a syntax-independent framework that systematically explains each of them as well as their relations.

1.4.1 Classical, Intuitionistic and Linear Logics

Classical logic [167] has been, as the name suggests, the most traditional, standard logic for working mathematicians. As its standard *Tarskian semantics* [176] indicates, it is the logic that regards *validity* or *truth* as independent of (especially *constructive*) reasoning by mathematicians; it is in accordance with *Platonism* in the philosophy of mathematics [121]. Therefore, it contains *non-constructive* laws such as the *law of excluded middle (LEM)* and the *double negation elimination (DNE).* This point also explains, though informally, why classical logic in sequent calculus 'collapses' in the sense that any two classical proofs of a formula must be equal [118, 78].

On the other hand, *intuitionistic logic* [181] was motivated by *intuitionism* in the philosophy of mathematics initiated by Luitzen Egbertus Jan Brouwer, and it was later formalized as a logical system by Arend Heyting [87]. Opposing to Platonism, intuitionism regards mathematics as an activity of 'mental construction', and a formula as true iff its proof has been constructed; thus, it prohibits non-constructive laws. Today, intuitionism is regarded as an approach to constructivism [181] along with MLTT though it (at least what Brouwer considered) is an informal view rather than a formal logical system. Syntactically, e.g., in *Hilbert-style logical systems* [185], intuitionistic logic is obtained from classical logic by eliminating LEM or DNE [180].

[7]It is customary to say a *logic* to refer to an unspecified formal logical system that embodies the logic, and we adopt this convention in this thesis too whenever it would not bring any confusion.

Finally, *linear logic* [70] proposed by Girard is often called a *resource-conscious logic* in the sense that it requires each premise to be used exactly once to produce a conclusion; in fact, it can be seen informally but very naturally as the logic of *resource consumption/production* [74]. It has both classical and intuitionistic variants, which are syntactically obtained respectively from the classical and intuitionistic sequent calculi by eliminating the *weakening* and *contraction* rules. Also, linear logic can be seen as a *linear refinement* of classical and intuitionistic logics in the sense that both logics may be embedded into (or represented in) linear logic, which adds an explicit control on the use of premises in classical and intuitionistic logics.

1.4.2 Towards a Unified View on the Logics

As explained above, their syntactic definitions and relations are clear; however, for a deeper understanding of what classical (resp. intuitionistic, linear) reasoning is as well as the relation between the three kinds of reasoning, we need a systematic framework to interpret and relate the three in a *syntax-independent* manner.

In the literature, there have been some *proof-theoretic* approaches to this problem. For instance, it is well-known that a classical proof can be translated into an intuitionistic one by the *negative translation* [180]. Moreover, classical reasoning corresponds, under the Curry-Howard isomorphism, to *control operators* in programming, and the negative translation of proofs to the *CPS-translation* of programs [172].

However, although these results are significant achievements, they are something like to mechanically translate a Japanese sentence into the corresponding French one without clarifying the underlying 'meaning'. That is, proof-theoretic approaches are not very suitable for a 'true understanding' of the relation between the three logics.

More generally, conventional *proof theory* [180] in general does not give a syntax-independent, non-inductive, non-axiomatic or intrinsic explanation of the logics.

Although our focus in the present thesis is mostly on intuitionistic logic, motivated by this problem, we briefly sketch our idea on a syntax-independent framework that systematically explains and relates the three logics in Sections 2.4.7 and 6.3, leaving its thorough development as future work.

1.5 Our Approach: Games and Strategies

Our approach is based on mathematical structures developed in the field of *game semantics* for they are dynamic and intensional (though not completely, which we will fix) as well as conceptually natural, mathematically precise and syntax-independent.

Game semantics [5, 14, 101] refers to a particular kind of semantics of logic and computation [188, 82] in which formulas (or types) and proofs (or programs) are interpreted as *games* and *strategies*, respectively. Historically, having its roots in 'games-based' approaches in mathematical logic to capture *validity* [155, 59], *higher-order computability* [110, 111, 112, 106, 113, 62] and *proofs in linear logic* [25, 7, 99], combined with ideas from *sequential algorithms* [22], *process calculi* [134, 90] and *geometry of interaction* [71, 72, 73, 75, 76, 77] in computer science, several variants of game semantics in its modern form were developed in the early 1990's to give the first syntax-independent characterization of the programming language PCF [9, 100, 139]; since then a variety of games and strategies have been proposed to model various programming features [14].

An advantage of game semantics is this flexibility: It models a wide range of logical systems and programming languages by just varying constraints on strategies [14], which enables one to systematically compare and relate different formalisms, ignoring superficial syntactic details. Another strong point of game semantics is its conceptual naturality: It interprets syntax as 'dynamic interactions' between the participants of games, providing a dynamic, intensional (though not completely, which again we shall resolve) explanation of syntax in a natural, intuitive (yet mathematically precise) manner. Note that its dynamic and intensional natures stand in sharp contrast to the traditional *set-theoretic* semantics [188, 82] which, e.g., cannot capture *sequentiality* of PCF (but the game semantics [9, 100, 139] can).

Taking advantage of the flexibility and conceptual naturality of game semantics, the present thesis aims to develop a single mathematical framework that clarifies, both conceptually and mathematically, the concepts and phenomena described in Sections 1.1, 1.2 and 1.3 (the problem of Section 1.4 is addressed only casually).

1.6 Main Results of the Thesis

The main results of the present thesis are summarized as follows, where comparison with related work is left to the end of each main chapter.

Dynamics and Intensionality (Chapter 3). We define a certain class of bicategories, called *cartesian closed β-categories of computation (CCBoCs)*, such that there is the 'extensionally collapsing' functor from a CCBoC to the corresponding *cartesian closed category (CCC)*. We then refine the standard semantics of programming languages in CCCs by CCBoCs to capture their dynamics and intensionality, and axiomatize a sufficient condition for the resulting interpretation of our variant of finitary PCF to satisfy the DCP. Finally, we define a *dynamic* variant of games and strategies, and show that they give rise to a CCBoC that satisfies the axioms, providing the first *dynamic game semantics* that captures dynamics and intensionality of computation.

Mathematical Models of Computation (Chapter 4). We define 'effective computability', called *viability*, of strategies in an *intrinsic*, *non-axiomatic*, *non-inductive* manner, and shows that viable dynamic strategies are *Turing complete*. As immediate corollaries, some of the well-known theorems in computability theory such as the *smn-theorem* and the *first recursion theorem* [46] are generalized. As a consequence, we have given a mathematical foundation of computation in the same sense as TMs but beyond classical computation in a more abstract fashion. Moreover, this game-semantic framework distinguishes high-level computational processes that operate directly on mathematical objects such as natural numbers (not on their symbolic representations) and their concrete symbolic implementations or low-level processes that define their 'effectivity', which sheds new light on the very notion of computation.

Meaning Explanation of MLTT (Chapter 5). We propose a novel variant of games and show that they induce an *injective* (for types built without N- and Id-types) and *surjective* interpretation of the *intensional* variant of MLTT equipped with $\mathsf{1}$-, $\mathsf{0}$-, N-, $\mathsf{\Pi}$-, $\mathsf{\Sigma}$- and Id-types as well as a cumulative hierarchy of universes. Moreover, by incorporating the framework of dynamic games and strategies, we obtain a semantics that can be seen as a mathematical formalization of the meaning explanation.

Classical, Intuitionistic and Linear Logics (Sections 2.4.7 and 6.3). We give a new construction $?$ on games, which is based on *why not* in linear logic [70], and prove that it forms a monad on the category of games. Conceptually, $?A$ is essentially A except that Player can make 'another try' anytime and any number of times; thus, we propose that a strategy on $?A$ is a *classical* proof of A. Moreover, based on $?$ and *exponential* $!$ on games [9], we propose a systematic method to relate classical and classical linear reasonings to intuitionistic and intuitionistic linear reasonings. Then, in particular, the intuitionistic implication $!A \multimap B$ can be seen as an asymmetric restriction of $?(A \multimap B)$ since we have the equation $?(A \multimap B) = !A \multimap ?B$. Also, as a corollary, we can see why $\mathsf{\Sigma}$-types in MLTT are incompatible with classical reasoning.

Nevertheless, we shall not fully develop these ideas, leaving the full account as future work, and therefore they are not a main contribution of the present thesis.

1.7 Philosophical Implications

As explained above, one of the main achievements of the thesis is a mathematical, syntax-independent formalization of the meaning explanation of MLTT, or more broadly constructivism in the philosophy of mathematics. In particular, this can be seen as a new approach to logic that unifies logical systems and models:

> Games and strategies form a logical system (i.e., formulas, proofs and operations on them) as well as a model (i.e., what formal languages refer to in a loose sense) and a theory of computation (in the sense of TMs).

This point arguably has some philosophical implications.

1.7.1 Semantic, Analytic Study of Logic

The first point is the *semantic*, *analytic* study of logic.

We have already explained the importance of the semantic study of logic. By their conceptual naturality and more complete formulation of logic than conventional games and strategies, in particular its dynamics and intensionality, one may reasonably, though arguably, regard our games and strategies developed in the present thesis as a mathematical formalization of the very notion of logic, rather than a semantics of logic as an a priori syntactic entity. In other words, we may regard the game-semantic approach as a mathematical *foundation* of logic, i.e., logic is internal (in the sense of *internal logic* [105, 123, 130]) to games.

On the other hand, a mathematical theory is *analytic* if its mathematical objects are defined in terms of more primitive ones such as sets, and its theorems are derived from axioms of a theory of such primitive objects such as ZFC via a logical system. In fact, most of the mainstream mathematics has been analytic; it is commonly said that in principle most part of mathematics can be reduced to ZFC and classical logic.

Importantly, this point implies that logic as well as theories of primitive objects must be *synthetic*, i.e., their mathematical objects are defined not in terms of more primitive ones but via what operations and properties they have, since otherwise the process of reducing mathematics to more primitive objects and theories would never stop. For this reason, logic in particular has been formulated synthetically.

Nevertheless, although a synthetic approach gives an abstract formalization of mathematics, whose generality and convenience are often strong advantages, e.g., category theory is a good example, it is in general not suitable for a conceptual underpinning of mathematics. In other words, a synthetic theory would be much more convincing if it can be explained and justified by an analytic approach. For this reason, we believe that an *analytic* study of logic would be fruitful and meaningful.

One may argue that category theory and *topos theory* [105, 104, 123, 130] formulate logic elegantly in terms of categorical notions as internal logic. As already mentioned, however, they are intrinsically synthetic because it is hard to say that categories are more primitive than logic; also, internal logic formulates *only provability*, not proofs themselves, let alone their dynamics or intensionality. Note that *algebraic logic* [19] pioneered by George Bool may be seen a posteriori as a particular kind of categorical semantics, in which again one may talk about only provability.

Also, one may point out that *model theory* [32, 91] can be seen as an analytic study of logic, which is again certainly true. However, it basically studies logical systems in comparison with their models; thus, it does not directly capture *proof-theoretic* aspects of logic, lacking in frameworks to go beyond provability and capture proofs themselves. On the other hand, main approaches in proof theory are syntactic, not completely satisfactory from our standpoint as already explained.

Finally, denotational semantics may be regarded as an analytic study of logic. We take this opinion without doubt since its aim is to capture formulas and proofs by mathematical objects. However, let us emphasize again the point that conventional (domain-theoretic) denotational semantics is intrinsically *static* and *extensional*.

In this context, the present thesis has given a stepping stone towards a more comprehensive analytic theory of logic (and constructive mathematics) than existing approaches (including conventional game semantics), especially of its dynamics and intensionality, by reducing logic to the primitive notions of games and strategies, which naturally embody '(dialogical) arguments' of truths of formulas (see Section 2.1).

At this point, for clarity of our standpoint, let us quote, from [126], the following:

> Mathematical logic and the relation between logic and mathematics have been interpreted in at least three different ways:
>
> 1. Mathematical logic as symbolic logic, or logic using mathematical symbolism;
> 2. Mathematical logic as foundations (or philosophy) of mathematics;

3. Mathematical logic as logic studied by mathematical methods, as a branch of mathematics.

The traditional and major viewpoint in mathematical logic and theoretical computer science is the first one as already pointed out, while our perspective in the present thesis is definitely the third one. We are aiming at capturing logic (and computation) by mathematics, viz., games and strategies.

1.7.2 Unity of Proof, Model and Recursion Theories

Another point is the *unity* of subfields of mathematical logic, specifically proof theory, model theory and recursion theory, in the sense described below.[8]

In mathematical logic, one usually formalizes a mathematical theory by a *logical system*, i.e., by giving a formal language of mathematics and postulating axioms and rules to derive theorems, while *models* are possible interpretations of the language [168, 56]. Hence, logical systems and models are traditionally *a priori completely separated*, and thus *soundness* and *completeness* are central problems: Axioms and rules ought to be *sound* so that theorems are all 'correct' as well as *complete* so that every 'true' statement can be proved. Even a more elementary question is *consistency*: One should not be able to prove both a statement A and its negation $\neg\mathsf{A}$ since otherwise the logic would be completely unreliable and useless, proving any formula.

In our game-semantic approach, in contrast, logical systems and models are simply *unified* via games and strategies; thus, the questions of soundness and completeness just disappear: Validity and provability of a formula just coincide, namely as the existence of a *winning strategy* on the underlying game, where *winning* roughly means 'justice' or 'reasonability' of the argument by the strategy. In other words, a winning strategy on a game embodies a proof of the formula which the game models, and also it defines validity of the formula (according to the meaning explanation) at the same time. Moreover, consistency of the logic which our games and strategies embody follows immediately from the *composability* of winning strategies: If there is a winning strategy on a game A, then there is none on its negation $\neg A = A \Rightarrow \mathbf{0}$ since otherwise the composition of the two would be a winning strategy on the *empty game* $\mathbf{0}$, a contradiction, and vice versa.

Note that the present work formalizes only a slight fragment of mathematics, and thus it is unclear how large part of mathematics can be formalized in this manner;

[8]This point holds also for MLTT and the meaning explanation. However, as already pointed out, the former is *syntactic* and the latter is *pre-mathematical.* Thus, the significance of the present work is in the point that it formulates the unity by a semantic and mathematical method.

however, it has certain *technical* advantages over conventional approaches to logic as explained above (in addition to the *conceptual* points described in Section 1.3.2), and therefore we regard it as worth pursuing further as future work.

In addition, note also that mathematical foundations of computability such as TMs have been extrinsic to proofs and models except that they have been applied to decision problems in proof theory [183, 34] and employed as realizers. Nevertheless, we shall formulate a mathematical model of computation based on our games and strategies in Chapter 4, and therefore our approach incorporates recursion theory too.

1.8 Thesis Outline

The rest of the thesis is structured as follows. First, Chapter 2 recalls an existing variant of games and strategies, on which our variants in the subsequent chapters are all based. Specifically, giving an informal introduction to games and strategies in Section 2.1, we review in detail Guy McCusker's games in Section 2.2 and strategies in Section 2.3. We then establish categories of games and strategies in Section 2.4.

Main contents of the thesis begin with Chapter 3, which gives a game semantics of dynamics and intensionality of computation, specifically a variant of finitary PCF (FPCF). We first explain the main idea and contributions of the chapter in Section 3.1, and then define our FPCF and axiomatize our semantics of FPCF that satisfies the DCP via certain bicategories in Section 3.2 so that it suffices to give its game-semantic instance. Next, we define a dynamic variant of games and strategies in Section 3.3 and show that they induce our bicategorical interpretation of FPCF in Section 3.4. We finally draw a conclusion of the chapter and propose future work in Section 3.5.

Next, based on Chapter 3, we develop a new mathematical model of higher-order computation in Chapter 4. Explaining the main idea and contributions of the chapter in Section 4.1, we first define our game-semantic 'high-level' computational processes in Section 4.2. Then, we formalize 'tags' for disjoint union of sets in Section 4.3 in order to define 'effective computability' of strategies in a rigorous manner. Next, as the main contributions of the chapter, we define a novel notion of 'computable' or *viable* strategies in an intrinsic, non-axiomatic, non-inductive fashion, and show that viable dynamic strategies are in fact Turing complete in Section 4.4. Finally, we make a conclusion of the chapter and propose further work in Section 4.5.

As the last main content of the thesis, we develop a game semantics of MLTT in Section 5, which is rather orthogonal to the successive development of the previous two chapters. We first explain the main obstacle in giving a game semantics of MLTT, our solution in a nutshell and contributions of the chapter in Section 5.1. Next, we briefly recall the syntax of MLTT in Section 5.2 and define our generalization of games, called *predicative games*, in Section 5.3. We then give an interpretation of MLTT equipped with various types mentioned before by predicative games in Section 5.4. However, the model cannot interpret universes; we overcome this point in Section 5.5. We also analyze the degree of intensionality of the game semantics in Section 5.6. Finally, we draw a conclusion of the chapter and propose future work in Section 5.7.

As a climax, Chapter 6 pieces together the rather orthogonal developments of the last three chapters. In Section 6.1, we combine the methods of Chapters 3 and 5 to capture the meaning explanation of MLTT more thoroughly. We extend it further in Section 6.2 by incorporating the framework of Chapter 4 so that we obtaine a mathematical model of higher-order computation that embodies the computation of MLTT. Finally, we briefly sketch some consequences of applying our method of modeling classical reasoning to the game semantics of MLTT in Section 6.3.

At last, we draw a conclusion of the entire thesis and propose some further work in Chapter 7.

Chapter 2

Preliminary: Games and Strategies

In this chapter, we recall an existing variant of games and strategies on which our variants will be based; specifically, it is Guy McCusker's one introduced in [14, 129]. Strictly speaking, the variants of [14] and [129] are slightly different; *legal positions* must satisfy *well-bracketing* in [129] but not in [14]. We have chosen the variant of [14] equipped with equivalence relations on *(valid) positions* introduced in [9] and sketched in Section 3.6 of [129], which we call ***games*** and ***strategies*** in this thesis.

We have selected McCusker's variant for it is relatively less restrictive, which is important to interpret various constructions of MLTT (e.g., it interprets *(weak) sum types* for the first time as games [129]) in Chapter 5. Also, the variant of [14] gives a unifying view on various logics and computations because it models many different calculi by simply varying constraints on strategies; in particular, under the Curry-Howard isomorphism, the presence/absence of the well-bracketing condition corresponds to intuitionistic/classical logic; see Section 2.4. Moreover, the equivalence relations on positions enable us to employ the well-behaving *exponential* defined in Section 3.6 of [129] (it is originally introduced in [9]), rather than the ad-hoc one defined in [14], without distinguishing inessential details of 'tags' for disjoint union of sets. This exponential is categorically well-behaving in the sense that it gives rise to a *comonad* that turns in the standard manner a categorical semantics of (a fragment of) linear logic into a categorical semantics of intuitionistic logic, while the ad-hoc one does not; see Section 2.4. Thus, our choice of games and strategies is mathematically well-behaving, and it gives a unifying view on classical, intuitionistic and linear logics.

The present chapter is structured as follows. Giving an informal introduction to games and strategies in Section 2.1, we recall the basic definitions and properties of games in Section 2.2 and of strategies in Section 2.3. Finally, we conclude the chapter by defining various categories of games and strategies, some of which embody computation, and the others do logic, in Section 2.4.

2.1 Pre-Mathematical Introduction

Before recalling the formal definitions, let us first give an informal or pre-mathematical introduction to games and strategies without technical details for it may be useful for a reader who is not very familiar with game semantics.

A *game*, roughly, is a certain kind of a rooted forest whose branches correspond to possible developments or *(valid) positions* of the 'game in the usual sense' (such as chess, poker, etc.) which it represents. These branches are finite sequences of *moves* of the game, where some moves are distinguished and called *initial*; only initial moves can be the first element (or occurrence) of a position of the game. A *play* of a game is an (finitely or infinitely) increasing sequence $\boldsymbol{\epsilon}, m_1, m_1 m_2, \dots$ of positions of the game, where $\boldsymbol{\epsilon}$ is the *empty sequence*. For the aim of the present thesis, it suffices to focus on rather standard *sequential* (as opposed to *concurrent* [15]) and *unpolarized* (as opposed to *polarized* [119]) games played by two participants, *Player*, who represents a 'mathematician' (or a 'computational agent'), and *Opponent*, who represents a 'rebutter' (or an 'environment'), in each of which Opponent always starts a play (i.e., unpolarized), and then they alternately (i.e., sequential) perform moves allowed by the rules of the game. Strictly speaking, a position of each game is not just a sequence of moves: Each occurrence m of Opponent's or O- (resp. Player's or P-) non-initial move in a position points to a previous occurrence m' of P- (resp. O-) move in the position, representing that m is performed specifically as a response to m'. Thus, a game is what specifies possible interactions between the participants, which may be seen as a *formula* in logic defining possible '(dialogical) arguments' on the truth of the formula or as a *type* in computation defining possible 'computational processes' of the type; in fact, in game semantics, we interpret formulas and types as games. If a game is to be regarded as a formula in logic, then it is natural to define that Player (resp. Opponent) *wins* a play (or an 'argument') of the game if the next move is to be performed by Opponent (resp. Player) but there is no possible one, and a participant *loses* a play if the other wins it. We shall extend this definition later to determine which participant wins even in the case where a play keeps growing *infinitely*, but for the moment it suffices to focus on the case where a play always terminates.

On the other hand, a *strategy* on a game is, informally, what tells Player which move (with a pointer) she should perform at each of her turns in the game, and thus it interprets a *proof* of the formula (or a *program* of the type) which the game models. If a game is to be regarded as a formula in logic, then we define the game to be *true* if there is a strategy by which Player wins every play of the game and *false* otherwise.

Convention. Pointers of strategies are often obvious, and so we usually omit them.

In a *game semantics* $[\![\_]\!]_{\mathcal{G}}$ of a logical system (resp. a programming language) $\mathcal{L}$, a formula (resp. a type) A in $\mathcal{L}$ is interpreted as the game $[\![\mathsf{A}]\!]_{\mathcal{G}}$, and a proof (resp. a program) M of A in $\mathcal{L}$, as a strategy $[\![\mathsf{M}]\!]_{\mathcal{G}}$ on $[\![\mathsf{A}]\!]_{\mathcal{G}}$[1]. Then, an 'execution' of the proof (resp. the program) M is interpreted as a play of $[\![\mathsf{A}]\!]_{\mathcal{G}}$, where Player follows $[\![\mathsf{M}]\!]_{\mathcal{G}}$.

Let us consider some simple examples. The simplest game of a true (resp. false) formula is the *unit game* $\mathbf{1}$ (resp. the *empty game* $\mathbf{0}$) in which the empty sequence $\boldsymbol{\epsilon}$ (resp. the singleton sequence q of any element q) is the only possible maximal position. Clearly, there is the trivial strategy on $\mathbf{1}$ for Player to win but not on $\mathbf{0}$.

Next, let us consider simple games for computation. The game N of natural numbers looks like the following tree (which is infinite in width):

$$\begin{tikzcd} & & q \arrow[dll] \arrow[dl] \arrow[d] \arrow[dr] & \cdots \\ 0 & 1 & 2 & 3 & \cdots \end{tikzcd}$$

in which a play starts with Opponent's question q ('What is your number?') and ends with Player's answer $n \in \mathbb{N}$ ('My number is n!'), where n points to q. A strategy $\underline{10}$ on N for $10 \in \mathbb{N}$, for instance, is represented by the map $q \mapsto 10$, or diagrammatically:

$$\begin{tikzcd} \underline{10} \\ q \\ 10 \arrow[u, bend left] \end{tikzcd}$$

where the arrow $q \leftarrow 10$ in the diagram represents the pointer, which is a notation we shall employ throughout the present thesis.

As another example, consider the game $N \multimap N$ of *linear functions* [70] (also written informally $N_{[0]} \multimap N_{[1]}$) on natural numbers, whose typical maximal position is $q_{[1]}q_{[0]}n_{[0]}m_{[1]}$, where $n, m \in \mathbb{N}$, and $(\_)_{[i]}$ for $i = 0, 1$ are arbitrary, unspecified 'tags' to distinguish the two copies of N (in the rest of the thesis, we employ a similar notation for three or more copies of N in the obvious manner too), or diagrammatically[2]:

$$\begin{tikzcd} N_{[0]} & \multimap & N_{[1]} \\ & & q_{[1]} \\ q_{[0]} \arrow[urr] & & \\ n_{[0]} \arrow[u, bend left] & & \\ & & m_{[1]} \arrow[uuu, bend right] \end{tikzcd}$$

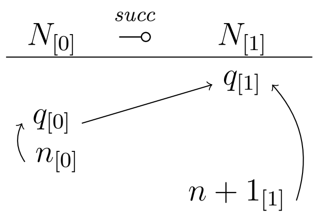
[1]Strictly speaking, a program in a (functional) programming language is usually of the form $\mathsf{\Gamma} \vdash \mathsf{M} : \mathsf{A}$, where $\mathsf{\Gamma}$ is a *context*, and the strategy $[\![\mathsf{M}]\!]_{\mathcal{G}}$ is on the game $[\![\mathsf{\Gamma}]\!]_{\mathcal{G}} \Rightarrow [\![\mathsf{A}]\!]_{\mathcal{G}}$ of *implication* from $[\![\mathsf{\Gamma}]\!]_{\mathcal{G}}$ to $[\![\mathsf{A}]\!]_{\mathcal{G}}$. However, here we focus on the case where contexts are *empty* for brevity.

[2]The diagram is depicted as above only to clarify which component game (i.e., the domain or the codomain) each move belongs to; it should be read just as a finite sequence, namely, $q_{[1]}q_{[0]}n_{[0]}m_{[1]}$, equipped with pointers.

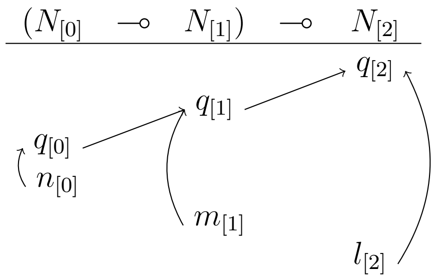
which can be read as follows:

1. Opponent's question $q_{[1]}$ for an output ('What is your output?');
2. Player's question $q_{[0]}$ for an input ('Wait, what is your input?);
3. Opponent's answer, say, $n_{[0]}$, to $q_{[0]}$ ('OK, here is an input n.');
4. Player's answer, say, $m_{[1]}$, to $q_{[1]}$ ('Alright, the output is then m.').

A strategy *succ* on this game that corresponds to the (linear) successor function can be represented by the map $q_{[1]} \mapsto q_{[0]}, q_{[1]}q_{[0]}n_{[0]} \mapsto n+1_{[1]}$, or diagrammatically:



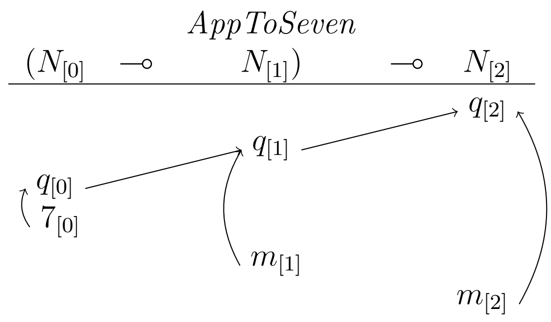
As yet another example, consider the game $(N \multimap N) \multimap N$ of higher-order (in particular *second-order*) linear functions, whose typical maximal position is:



where $n, m, l \in \mathbb{N}$, which can be read as follows:

1. Opponent's first question $q_{[2]}$ for an output ('What is your output?');
2. Player's question $q_{[1]}$ for an input function ('Wait, what is your input?');
3. Opponent's second question $q_{[0]}$ for an input ('What is your input then?');
4. Player's answer, say, $n_{[0]}$, to $q_{[0]}$ ('OK, here is an input n.');
5. Opponent's answer, say, $m_{[1]}$, to $q_{[1]}$ ('Sure, then here is an input m.');
6. Player's answer, say, $l_{[2]}$, to $q_{[2]}$ ('Alright, the output is then l.').

A strategy *AppToSeven* on this game that applies an input linear function to $7 \in \mathbb{N}$ can be represented by the map $q_{[2]} \mapsto q_{[1]}$, $q_{[2]}q_{[1]}q_{[0]} \mapsto 7_{[0]}$, $q_{[2]}q_{[1]}q_{[0]}7_{[0]}m_{[1]} \mapsto m_{[2]}$:



Even in these simple examples, we may see the dynamic and intensional natures of games and strategies as well as their conceptual naturality; also, it is clear that they are syntax-independent, i.e., notation for them is secondary and inessential. However, as we shall see shortly, they are actually not fully dynamic or intensional, and thus we shall define a more dynamic, intensional variant in Chapter 3.

Also, in some sense Opponent plays the role of an *oracle* [169], and it is a part of the formalization, which is a distinguishing feature of games and strategies. Note also that Player may compute on the 'external behavior' of Opponent, even when he plays the role of a higher-order input, rather than some 'encoding' of the input. Thus, unlike other mathematical models of computation such as TMs, games and strategies may model higher-order computation in a general, systematic fashion.

Finally, we may see that computational processes in games and strategies are clearly more 'high-level' than TMs, e.g., the participants of the game N handle a natural number as a *single* move; we shall define in Chapter 4 the remaining game-semantic 'low-level' computational processes as promised in the introduction.

2.2 Games

Let us review *games* in detail. Games are based on two preliminary concepts: *arenas* and *legal positions*. An arena defines the basic components of a game, which in turn induces its legal positions to specify the basic rules of the game in the sense that *(valid) positions* of the game must be legal positions. We first recall these notions.

Notation. We use the following notation throughout the present thesis:

- We use bold letters $\boldsymbol{s}, \boldsymbol{t}, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}$, etc. for sequences, in particular $\boldsymbol{\epsilon}$ for the *empty sequence*, and letters $a, b, c, d, m, n, x, y, z$, etc. for elements of sequences;

- We often abbreviate a finite sequence $\boldsymbol{s} = (x_1, x_2, \dots, x_{|\boldsymbol{s}|})$ as $x_1 x_2 \dots x_{|\boldsymbol{s}|}$, where $|\boldsymbol{s}|$ denotes the *length* (i.e., the number of elements) of $\boldsymbol{s}$, and write $\boldsymbol{s}(i)$, where $i \in \{1, 2, \dots, |\boldsymbol{s}|\}$, as another notation for x_i;

- A *concatenation* of sequences is represented by the juxtaposition of them, but we often write $a\boldsymbol{s}$, $\boldsymbol{t}b$, $\boldsymbol{u}c\boldsymbol{v}$ for $(a)\boldsymbol{s}$, $\boldsymbol{t}(b)$, $\boldsymbol{u}(c)\boldsymbol{v}$, etc., and also $\boldsymbol{s}.\boldsymbol{t}$ for $\boldsymbol{st}$; we define $\boldsymbol{s}^n \overset{\text{df.}}{=} \underbrace{\boldsymbol{ss}\cdots\boldsymbol{s}}_{n}$ for any sequence $\boldsymbol{s}$ and natural number $n \in \mathbb{N}$;

- We write $\mathsf{Even}(\boldsymbol{s})$ (resp. $\mathsf{Odd}(\boldsymbol{s})$) iff $\boldsymbol{s}$ is of even-length (resp. odd-length), and for any set S of sequences $S^{\mathsf{P}} \overset{\text{df.}}{=} \{\boldsymbol{s} \in S \mid \mathsf{P}(\boldsymbol{s})\}$, where $\mathsf{P} \in \{\mathsf{Even}, \mathsf{Odd}\}$;

- We write $\boldsymbol{s} \preceq \boldsymbol{t}$ iff $\boldsymbol{s}$ is a *prefix* of $\boldsymbol{t}$, and given a set S of sequences, $\mathsf{Pref}(S)$ for the set of all prefixes of sequences in S, i.e., $\mathsf{Pref}(S) \overset{\text{df.}}{=} \{\boldsymbol{s} \mid \exists \boldsymbol{t} \in S.\, \boldsymbol{s} \preceq \boldsymbol{t}\}$;

- Given a poset P and a subset $S \subseteq P$, $\mathsf{Sup}(S)$ denotes the *supremum* of S;

- $X^* \overset{\text{df.}}{=} \{x_1x_2\dots x_n \mid n \in \mathbb{N}, \forall i \in \{1, 2, \dots, n\}.\, x_i \in X\}$ for each set X;

- Given a function $f : A \to B$ and a subset $S \subseteq A$, we define $f \upharpoonright S : S \to B$ to be the *restriction* of f to S and $f^* : A^* \to B^*$ by $f^*(a_1a_2\dots a_n) \overset{\text{df.}}{=} f(a_1)f(a_2)\dots f(a_n) \in B^*$ for all $a_1a_2\dots a_n \in A^*$;

- Given sets A and B, we write B^A for the set of all functions from A to B;

- Given sets $X_1, X_2, \dots, X_n$, and $i \in \{1, 2, \dots, n\}$, we write $\pi_i^{(n)}$ or π_i for the i^{th}-*projection function* $X_1 \times X_2 \times \cdots \times X_n \to X_i$ that maps $(x_1, x_2, \dots, x_n) \mapsto x_i$;

- We write $x \downarrow$ if an element x is defined, and $x \uparrow$ otherwise; $\simeq$ denotes the *Kleene equality*, i.e., $x \simeq y \overset{\text{df.}}{\Leftrightarrow} (x \downarrow \wedge\, y \downarrow \wedge\, x = y) \vee (x \uparrow \wedge\, y \uparrow)$;

- We use symbols $\Rightarrow$ (*implication*), $\wedge$ (*conjunction*), $\vee$ (*disjunction*), $\forall$ (*universal quantification*), $\exists$ (*existential quantification*) and $\Leftrightarrow$ (*if and only if*) informally in our meta-language, whose intended meanings are as written in the following parentheses; $\forall$ and $\exists$ precedes any other symbols; $\wedge$ and $\vee$ precede $\Rightarrow$ and $\Leftrightarrow$; $\wedge$ and $\vee$ are left associative, while $\Rightarrow$ and $\Leftrightarrow$ are right associative;

- If P refers to a *predicate*, then $\mathsf{P}(x)$ means that an object x satisfies P.

2.2.1 Arenas and Legal Positions

Definition 2.2.1 (Arenas [14, 129])**.** An ***arena*** is a triple $G = (M_G, \lambda_G, \vdash_G)$, where:

- M_G is a set whose elements are called ***moves***;

- λ_G is a function from M_G to $\{\mathsf{O}, \mathsf{P}\} \times \{\mathsf{Q}, \mathsf{A}\}$, called the ***labeling function***, in which O, P, Q and A are arbitrarily fixed symbols, called the ***labels***;

- $\vdash_G$ is a subset of $(\{\star\} \cup M_G) \times M_G$, where $\star$ is an arbitrarily fixed element such that $\star \notin M_G$, called the ***enabling relation***, that satisfies:
  - (E1) If $\star \vdash_G m$, then $\lambda_G(m) = \mathsf{OQ}$ and $n \vdash_G m \Leftrightarrow n = \star$;
  - (E2) If $m \vdash_G n$ and $\lambda_G^{\mathsf{QA}}(n) = \mathsf{A}$, then $\lambda_G^{\mathsf{QA}}(m) = \mathsf{Q}$;
  - (E3) If $m \vdash_G n$ and $m \neq \star$, then $\lambda_G^{\mathsf{OP}}(m) \neq \lambda_G^{\mathsf{OP}}(n)$

  in which $\lambda_G^{\mathsf{OP}} \stackrel{\text{df.}}{=} \pi_1 \circ \lambda_G : M_G \to \{\mathsf{O}, \mathsf{P}\}$ and $\lambda_G^{\mathsf{QA}} \stackrel{\text{df.}}{=} \pi_2 \circ \lambda_G : M_G \to \{\mathsf{Q}, \mathsf{A}\}$.

Convention. A move $m \in M_G$ of an arena G is called:

- ***initial*** if $\star \vdash_G m$;
- an ***O-move*** if $\lambda_G^{\mathsf{OP}}(m) = \mathsf{O}$, and a ***P-move*** if $\lambda_G^{\mathsf{OP}}(m) = \mathsf{P}$;
- a ***question*** if $\lambda_G^{\mathsf{QA}}(m) = \mathsf{Q}$, and an ***answer*** if $\lambda_G^{\mathsf{QA}}(m) = \mathsf{A}$.

Notation. Given an arena G, we define $M_G^{\mathsf{Init}} \stackrel{\text{df.}}{=} \{ m \in M_G \mid \star \vdash_G m \} \subseteq M_G$.

Thus, an arena G defines moves of a game, each of which is Opponent's/Player's question/answer, and which move n can be performed for each move m by the enabling relation $m \vdash_G n$, where $\star \vdash_G m$ means that Opponent can initiate a play by m (for this point, see Definitions 2.2.5, 2.2.9 and 2.2.10 below).

The axioms of arenas are to be read as follows:

- E1 sets the convention that an initial move must be Opponent's question, and an initial move cannot be performed for a previous move;
- E2 states that an answer must be performed for a question;
- E3 mentions that an O-move must be performed for a P-move, and vice versa.

Example 2.2.2. The ***terminal arena*** T is given by $T \stackrel{\text{df.}}{=} (\emptyset, \emptyset, \emptyset)$.

Example 2.2.3. The ***flat arena*** *flat*(S) on a given set S is given by $M_{\mathit{flat}(S)} \stackrel{\text{df.}}{=} \{q\} \cup S$, where q is any element with $q \notin S$; $\lambda_{\mathit{flat}(S)} : q \mapsto \mathsf{OQ}, (m \in S) \mapsto \mathsf{PA}$; $\vdash_{\mathit{flat}(S)} \stackrel{\text{df.}}{=} \{(\star, q)\} \cup \{(q, m) \mid m \in S\}$. For instance, $N \stackrel{\text{df.}}{=} \mathit{flat}(\mathbb{N})$ is the arena of natural numbers, and $\mathbf{2} \stackrel{\text{df.}}{=} \mathit{flat}(\mathbb{B})$, where $\mathbb{B} \stackrel{\text{df.}}{=} \{\mathit{tt}, \mathit{ff}\}$, is the arena of booleans.

As already mentioned, interactions between Opponent and Player in a game are represented by certain sequences of moves of the underlying arena, equipped with *pointers* (Definition 2.2.5) that specify the occurrence of a move in the sequence which each occurrence of a non-initial move (Definition 2.2.4) in the sequence is performed for. Technically, pointers are to distinguish similar but different computations; see [14, 43] for this point. In addition, they play an important role in Chapter 4.

Definition 2.2.4 (Occurrences of moves)**.** Given a finite sequence $\boldsymbol{s} \in M_G^*$ of moves of an arena G, an ***occurrence (of a move)*** in $\boldsymbol{s}$ is a pair $(\boldsymbol{s}(i), i)$ such that $i \in \{1, 2, \dots, |\boldsymbol{s}|\}$. More specifically, we call the pair $(\boldsymbol{s}(i), i)$ an ***initial occurrence*** (resp. a ***non-initial occurrence***) in $\boldsymbol{s}$ if $\star \vdash_G \boldsymbol{s}(i)$ (resp. otherwise).

Definition 2.2.5 (J-sequences [100, 14, 143])**.** A ***justified (j-) sequence*** of an arena G is a pair $\boldsymbol{s} = (\boldsymbol{s}, \mathcal{J}_{\boldsymbol{s}})$ of a finite sequence $\boldsymbol{s} \in M_G^*$ and a map $\mathcal{J}_{\boldsymbol{s}} : \{1, 2, \dots, |\boldsymbol{s}|\} \rightarrow \{0, 1, 2, \dots, |\boldsymbol{s}| - 1\}$ such that for all $i \in \{1, 2, \dots, |\boldsymbol{s}|\}$ $\mathcal{J}_{\boldsymbol{s}}(i) = 0$ if $\star \vdash_G \boldsymbol{s}(i)$, and $0 < \mathcal{J}_{\boldsymbol{s}}(i) < i \wedge \boldsymbol{s}(\mathcal{J}_{\boldsymbol{s}}(i)) \vdash_G \boldsymbol{s}(i)$ otherwise. The occurrence $(\boldsymbol{s}(\mathcal{J}_{\boldsymbol{s}}(i)), \mathcal{J}_{\boldsymbol{s}}(i))$ is called the ***justifier*** of a non-initial occurrence $(\boldsymbol{s}(i), i)$ in $\boldsymbol{s}$. We also say that $(\boldsymbol{s}(i), i)$ is ***justified*** by $(\boldsymbol{s}(\mathcal{J}_{\boldsymbol{s}}(i)), \mathcal{J}_{\boldsymbol{s}}(i))$, or there is a ***pointer*** from the former to the latter.

The idea is that each non-initial occurrence in a j-sequence must be performed for a specific previous occurrence, viz., its justifier, in the j-sequence.

Convention. By abuse of notation, we usually keep the pointer structure $\mathcal{J}_{\boldsymbol{s}}$ of each j-sequence $\boldsymbol{s} = (\boldsymbol{s}, \mathcal{J}_{\boldsymbol{s}})$ implicit and often abbreviate occurrences $(\boldsymbol{s}(i), i)$ in $\boldsymbol{s}$ as $\boldsymbol{s}(i)$. Moreover, we usually write $\mathcal{J}_{\boldsymbol{s}}(\boldsymbol{s}(i)) = \boldsymbol{s}(j)$ if $\mathcal{J}_{\boldsymbol{s}}(i) = j$. This convention is mathematically imprecise, but it does not bring any serious confusion in practice.

Notation. We write $\mathscr{J}_G$ for the set of all j-sequences of an arena G.

Definition 2.2.6 (J-subsequences)**.** Given an arena G and a j-sequence $\boldsymbol{s} \in \mathscr{J}_G$, a ***j-subsequence*** of $\boldsymbol{s}$ is a j-sequence $\boldsymbol{t} \in \mathscr{J}_G$ that satisfies:

- $\boldsymbol{t}$ is a subsequence of $\boldsymbol{s}$, for which we write $\boldsymbol{t} = (\boldsymbol{s}(i_1), \boldsymbol{s}(i_2), \dots, \boldsymbol{s}(i_{|\boldsymbol{t}|}))$;
- $\mathcal{J}_{\boldsymbol{t}}(\boldsymbol{s}(i_r)) = \boldsymbol{s}(i_l)$ iff there are occurrences $\boldsymbol{s}(j_1), \boldsymbol{s}(j_2), \dots, \boldsymbol{s}(j_k)$ in $\boldsymbol{s}$ eliminated in $\boldsymbol{t}$, where $l, r, k \in \mathbb{N}$ and $1 \leqslant l < r \leqslant |\boldsymbol{t}|$, such that $\mathcal{J}_{\boldsymbol{s}}(\boldsymbol{s}(i_r)) = \boldsymbol{s}(j_1) \wedge \mathcal{J}_{\boldsymbol{s}}(\boldsymbol{s}(j_1)) = \boldsymbol{s}(j_2) \wedge \cdots \wedge \mathcal{J}_{\boldsymbol{s}}(\boldsymbol{s}(j_{k-1})) = \boldsymbol{s}(j_k) \wedge \mathcal{J}_{\boldsymbol{s}}(\boldsymbol{s}(j_k)) = \boldsymbol{s}(i_l)$.

For later developments of the thesis (specifically Chapter 5), we need to define the following notion precisely though it is not conventional in the literature:

Definition 2.2.7 (Equality of j-sequences)**.** Given arenas G and H, we define j-sequences $\boldsymbol{s} \in \mathscr{J}_G$ and $\boldsymbol{t} \in \mathscr{J}_H$ to be ***equal***, written $\boldsymbol{s} = \boldsymbol{t}$, if they are the same not only as finite sequences but also their labels and justifiers are the same, i.e.,

$$\forall i, j \in \{1, 2, \dots, |\boldsymbol{s}|\}. \lambda_G(\boldsymbol{s}(i)) = \lambda_H(\boldsymbol{t}(i)) \wedge (\mathcal{J}_{\boldsymbol{s}}(\boldsymbol{s}(i)) = \boldsymbol{s}(j) \Leftrightarrow \mathcal{J}_{\boldsymbol{t}}(\boldsymbol{t}(i)) = \boldsymbol{t}(j)).$$

Convention. A j-sequence is identified by the equality defined in Definition 2.2.7, for which it is rather appropriate to assume that OP- and QA-labels are attached to each occurrence in a j-sequence.

Next, let us recall the notion of 'relevant' part of previous occurrences:

Definition 2.2.8 (Views [100, 14])**.** The ***Player (P-) view*** $\lceil \boldsymbol{s} \rceil_G$ and the ***Opponent (O-) view*** $\lfloor \boldsymbol{s} \rfloor_G$ of a j-sequence $\boldsymbol{s} \in \mathscr{J}_G$ of an arena G are the j-subsequences of $\boldsymbol{s}$ given by the following induction on $|\boldsymbol{s}|$:

- $\lceil \boldsymbol{\epsilon} \rceil_G \overset{\mathrm{df.}}{=} \boldsymbol{\epsilon}$;
- $\lceil \boldsymbol{s}m \rceil_G \overset{\mathrm{df.}}{=} \lceil \boldsymbol{s} \rceil_G.m$ if m is a P-move;
- $\lceil \boldsymbol{s}m \rceil_G \overset{\mathrm{df.}}{=} m$ if m is initial;
- $\lceil \boldsymbol{s}m\boldsymbol{t}n \rceil_G \overset{\mathrm{df.}}{=} \lceil \boldsymbol{s} \rceil_G.mn$ if n is an O-move such that m justifies n;
- $\lfloor \boldsymbol{\epsilon} \rfloor_G \overset{\mathrm{df.}}{=} \boldsymbol{\epsilon}$;
- $\lfloor \boldsymbol{s}m \rfloor_G \overset{\mathrm{df.}}{=} \lfloor \boldsymbol{s} \rfloor_G.m$ if m is an O-move;
- $\lfloor \boldsymbol{s}m\boldsymbol{t}n \rfloor_G \overset{\mathrm{df.}}{=} \lfloor \boldsymbol{s} \rfloor_G.mn$ if n is a P-move such that m justifies n

where the justifiers of occurrences in $\lceil \boldsymbol{s} \rceil_G$ (resp. $\lfloor \boldsymbol{s} \rfloor_G$) are unchanged if they occur in $\lceil \boldsymbol{s} \rceil_G$ (resp. $\lfloor \boldsymbol{s} \rfloor_G$), and undefined otherwise. A ***P-view*** (resp. an ***O-view***) ***of G*** is the P-view (resp. the O-view) of a j-sequence of G. A ***view*** is a P- or O-view.

Notation. We often omit the subscripts G in $\lceil \_ \rceil_G$ and $\lfloor \_ \rfloor_G$ when they are obvious.

The idea behind the notion of views is as follows. Given a j-sequence $\boldsymbol{s}m$ of an arena G such that m is a P-move (resp. an O-move), the P-view $\lceil \boldsymbol{s} \rceil$ (resp. the O-view $\lfloor \boldsymbol{s} \rfloor$) is intended to be the currently 'relevant' part of previous occurrences for Player (resp. Opponent). That is, Player (resp. Opponent) is concerned only with the last occurrence of an O-move (resp. a P-move), its justifier and that justifier's 'concern', i.e., P-view (resp. O-view), which then recursively proceeds. See [100, 43, 42] for an explanation of justifiers and views in terms of their correspondences with syntax.

Remark. A view may not be a j-sequence, motivating the *visibility* condition below.

We are now ready to introduce the notion of *legal positions* of an arena:

Definition 2.2.9 (Legal positions [14])**.** A ***legal position*** of an arena G is a j-sequence $\boldsymbol{s} \in \mathscr{J}_G$ that satisfies the following two conditions:

- (Alternation) If $\boldsymbol{s} = \boldsymbol{s_1}mn\boldsymbol{s_2}$, then $\lambda_G^{\mathsf{OP}}(m) \neq \lambda_G^{\mathsf{OP}}(n)$;
- (Visibility) If $\boldsymbol{s} = \boldsymbol{t}m\boldsymbol{u}$ with m non-initial, then $\mathcal{J}_{\boldsymbol{s}}(m)$ occurs in $\lceil \boldsymbol{t} \rceil_G$ if m is a P-move, and in $\lfloor \boldsymbol{t} \rfloor_G$ otherwise.

Notation. The set of all legal positions of an arena G is denoted by $\mathscr{L}_G$.

The visibility condition is technically to guarantee that the P-view and the O-view of a j-sequence of an arena are both j-sequences of the arena [129], and conceptually to ensure that the justifier of each non-initial occurrence belongs to the 'relevant' part of previous occurrences. Legal positions are to specify the basic rules of a game in the sense that every position of the game must be a legal position (Definition 2.2.10):

- During a play of the game, Opponent performs the first move by a question, and then Player and Opponent alternately play (by alternation), where each non-initial move is performed for a specific previous occurrence, viz., the justifier;
- The justifier of each non-initial occurrence belongs to the 'relevant' part, viz., the view, of previous occurrences (by visibility) in a position of the game.

2.2.2 Games

We are now ready to recall the central notion of *games*:

Definition 2.2.10 (Games [14, 129])**.** A ***game*** is a quintuple

$$G = (M_G, \lambda_G, \vdash_G, P_G, \simeq_G)$$

such that:

- The triple $(M_G, \lambda_G, \vdash_G)$ forms an arena (also denoted by G);
- P_G is a subset of $\mathscr{L}_G$, whose elements are called ***(valid) positions*** of G, that satisfies:

  (P1) P_G is non-empty and *prefix-closed* (i.e., $\forall \boldsymbol{s}m \in P_G.\, \boldsymbol{s} \in P_G$);
- $\simeq_G$ is an equivalence relation on P_G, called the ***identification of (valid) positions***, that satisfies:
  - (I1) $\boldsymbol{s} \simeq_G \boldsymbol{t} \Rightarrow |\boldsymbol{s}| = |\boldsymbol{t}|$;
  - (I2) $\boldsymbol{s}m \simeq_G \boldsymbol{t}n \Rightarrow \boldsymbol{s} \simeq_G \boldsymbol{t} \wedge \lambda_G(m) = \lambda_G(n) \wedge (m, n \in M_G^{\mathsf{Init}} \vee (\exists i \in \{1, 2, \dots, |\boldsymbol{s}|\}.\, \mathcal{J}_{\boldsymbol{s}m}(m) = \boldsymbol{s}(i) \wedge \mathcal{J}_{\boldsymbol{t}n}(n) = \boldsymbol{t}(i)))$;
  - (I3) $\boldsymbol{s} \simeq_G \boldsymbol{t} \wedge \boldsymbol{s}m \in P_G \Rightarrow \exists \boldsymbol{t}n \in P_G.\, \boldsymbol{s}m \simeq_G \boldsymbol{t}n$.

A ***play*** of G is a (finite or infinite) sequence $\boldsymbol{\epsilon}, m_1, m_1m_2, \dots$ of positions in G.

The axiom P1 corresponds to the natural phenomenon that a non-empty 'moment' (or position) of a game must have the previous 'moment'. Identifications of positions are originally introduced in [9] and also employed in Section 3.6 of [129]. They are to identify positions up to inessential details of 'tags' for disjoint union, particularly for *exponential* (Definition 2.2.25); each position $\boldsymbol{s} \in P_G$ of a game G is a representative of the equivalence class $[\boldsymbol{s}] \overset{\text{df.}}{=} \{\boldsymbol{t} \in P_G \mid \boldsymbol{t} \simeq_G \boldsymbol{s}\} \in P_G/\simeq_G$ which we take as primary. For this underlying idea, the three axioms I1, I2 and I3 should make sense.

Remark. In [14, 129], positions of a game G are required to be closed under taking *threads* for their rather simple exponential; however, we do not need this axiom.

Example 2.2.11. The ***terminal game*** $T \overset{\text{df.}}{=} (\emptyset, \emptyset, \emptyset, \{\boldsymbol{\epsilon}\}, \{(\boldsymbol{\epsilon}, \boldsymbol{\epsilon})\})$ is the simplest game, which is actually the ***unit game*** $\mathbf{1}$ sketched in Section 2.1.

Example 2.2.12. The ***flat game*** $\mathit{flat}(S)$ on a given set S is defined as follows. The triple $\mathit{flat}(S) = (M_{\mathit{flat}(S)}, \lambda_{\mathit{flat}(S)}, \vdash_{\mathit{flat}(S)})$ is the flat arena in Example 2.2.3, $P_{\mathit{flat}(S)} \overset{\text{df.}}{=} \{\boldsymbol{\epsilon}, q\} \cup \{qm \mid m \in S\}$, and $\simeq_{\mathit{flat}(S)} \overset{\text{df.}}{=} \{(\boldsymbol{s}, \boldsymbol{s}) \mid \boldsymbol{s} \in P_{\mathit{flat}(S)}\}$. For instance, $N \overset{\text{df.}}{=} \mathit{flat}(\mathbb{N})$ is the game of natural numbers sketched in the introduction, and $\mathbf{2} \overset{\text{df.}}{=} \mathit{flat}(\mathbb{B})$ is the game of booleans. Also, $\mathbf{0} \overset{\text{df.}}{=} \mathit{flat}(\emptyset)$ is the ***empty game***.

Next, let us define a substructure relation between games:

Definition 2.2.13 (Subgames)**.** A game H is a ***subgame*** of a game G, written $H \trianglelefteq G$, if $M_H \subseteq M_G$, $\lambda_H = \lambda_G \upharpoonright M_H$, $\vdash_H \subseteq \vdash_G \cap ((\{\star\} \cup M_H) \times M_H)$, $P_H \subseteq P_G$, and $\simeq_H = \simeq_G \cap (P_H \times P_H)$.

Example 2.2.14. The terminal game T (Example 2.2.11) is clearly a subgame of any game. As another example, consider the flat games $2N \overset{\text{df.}}{=} \mathit{flat}(\{2n \mid n \in \mathbb{N}\})$ and $2N + 1 \overset{\text{df.}}{=} \mathit{flat}(\{2n + 1 \mid n \in \mathbb{N}\})$. Clearly, they are both subgames of N (Definition 2.2.12), but neither is a subgame of the other.

Note that a game G may have a move $m \in M_G$ that does not occur in any position of G, or an *enabling pair* $m \vdash_G n$ not used for a justification in a position of G. For technical convenience (e.g., for Lemma 5.3.7), we prohibit such unused structures:

Definition 2.2.15 (Economical games)**.** A game G is ***economical*** if every move $m \in M_G$ occurs in a position of G, and every pair $m \vdash_G n$ in the enabling relation occurs as a non-initial occurrence n and its justifier m in a position of G.

Example 2.2.16. The terminal game, the unit game and flat games are economical.

Convention. Henceforth, ***games*** refer to *economical* games by default (n.b., all the constructions on games defined in the present thesis preserve this property).

We shall later focus on *well-opened*, *well-founded* games. Roughly, a game is well-opened if every initial occurrence of the game is the first element of a position, and it is well-founded if so is the enabling relation. Formally, they are defined as follows:

Definition 2.2.17 (Well-opened games [9, 14, 129])**.** A game G is ***well-opened*** if:

(WO) $\boldsymbol{s}m \in P_G$ with m initial implies $\boldsymbol{s} = \boldsymbol{\epsilon}$.

Definition 2.2.18 (Well-founded games [37])**.** A game G is ***well-founded*** if:

(WF) The enabling relation $\vdash_G$ is well-founded *downwards*, i.e., there is no countably infinite sequence $(m_i)_{i \in \mathbb{N}}$ of moves $m_i \in M_G$ such that $\star \vdash_G m_0 \wedge \forall i \in \mathbb{N}. \, m_i \vdash_G m_{i+1}$.

2.2.3 Constructions on Games

Next, let us review existing constructions on games. It is straightforward to show that these constructions are well-defined, and so we leave the proofs to [14, 129].

Convention. For brevity, we usually omit 'tags' for disjoint union of sets. For instance, we write $x \in A + B$ iff $x \in A$ or $x \in B$; also, given relations $R_A \subseteq A \times A$ and $R_B \subseteq B \times B$, we write $R_A + R_B$ for the relation on the disjoint union $A + B$ such that $(x, y) \in R_A + R_B \overset{\text{df.}}{\Leftrightarrow} (x, y) \in R_A \vee (x, y) \in R_B$.

Let us begin with *tensor (product)* $\otimes$. Roughly, a position $\boldsymbol{s}$ of the tensor $A \otimes B$ of games A and B is an interleaving mixture of a position $\boldsymbol{t}$ of A and a position $\boldsymbol{u}$ of B developed 'in parallel without communication'. Formally:

Definition 2.2.19 (Tensor of games [7, 14])**.** Given games A and B, the ***tensor (product)*** $A \otimes B$ of A and B is defined by:

- $M_{A \otimes B} \overset{\text{df.}}{=} M_A + M_B$;
- $\lambda_{A \otimes B} \overset{\text{df.}}{=} [\lambda_A, \lambda_B]$;
- $\vdash_{A \otimes B} \overset{\text{df.}}{=} \vdash_A + \vdash_B$;
- $P_{A \otimes B} \overset{\text{df.}}{=} \{\boldsymbol{s} \in \mathscr{L}_{A \otimes B} \mid \boldsymbol{s} \upharpoonright A \in P_A, \boldsymbol{s} \upharpoonright B \in P_B\}$;
- $\boldsymbol{s} \simeq_{A \otimes B} \boldsymbol{t} \overset{\text{df.}}{\Leftrightarrow} \boldsymbol{s} \upharpoonright A \simeq_A \boldsymbol{t} \upharpoonright A \wedge \boldsymbol{s} \upharpoonright B \simeq_B \boldsymbol{t} \upharpoonright B \wedge \forall i \in \mathbb{N}. \, \boldsymbol{s}(i) \in M_A \Leftrightarrow \boldsymbol{t}(i) \in M_A$

where $\boldsymbol{s} \upharpoonright A$ (resp. $\boldsymbol{s} \upharpoonright B$) denotes the j-subsequence of $\boldsymbol{s}$ that consists of occurrences of moves of A (resp. B).

As explained in [5], it is easy to see that during a play of a tensor $A \otimes B$ only Opponent can switch between the component games A and B (by alternation).

Example 2.2.20. Consider the tensor $N \otimes N$ of the natural number game N with itself, whose maximal position is either of the following forms:

$$\begin{tikzcd}
N_{[0]} & \otimes & N_{[1]} & & N_{[0]} & \otimes & N_{[1]} \\
q_{[0]} & & & & & & q_{[1]} \\
n_{[0]} \arrow[u, bend left] & & & & & & m_{[1]} \arrow[u, bend left] \\
& & q_{[1]} & & q_{[0]} & & \\
& & m_{[1]} \arrow[u, bend left] & & n_{[0]} \arrow[u, bend left] & &
\end{tikzcd}$$

where $n, m \in \mathbb{N}$, and $(\_)_{[i]}$ $(i = 0, 1)$ are again arbitrary, unspecified 'tags' such that $[0] \neq [1]$ to distinguish the two copies of N, and the arrows represent pointers.

Convention. We shall keep using the notation in Example 2.2.20 (also for more than two copies of a game in the obvious manner), and often omit 'tags' $(\_)_{[i]}$.

Next, let us recall *linear implication* $\multimap$. A linear implication $A \multimap B$ is the space of *linear functions* from A to B in the sense of linear logic [70], i.e., they consume exactly one input in A (strictly speaking *at most once* since it is possible for Player not to play in A at all, i.e., $\multimap$ is actually *affine implication*, but we shall stick to the convention; it is linear if strategies are all *strict* [3]) to produce an output in B:

Definition 2.2.21 (Linear implication [7, 14]). The ***linear implication*** $A \multimap B$ from a game A to a game B is defined by $A \multimap B \stackrel{\text{df.}}{=} T$ if $B = T$, and otherwise by:

- $M_{A \multimap B} \stackrel{\text{df.}}{=} M_A + M_B$;
- $\lambda_{A \multimap B} \stackrel{\text{df.}}{=} [\overline{\lambda_A}, \lambda_B]$, where $\overline{\lambda_A} \stackrel{\text{df.}}{=} \langle \overline{\lambda_A^{\mathsf{OP}}}, \lambda_A^{\mathsf{QA}} \rangle$ and $\overline{\lambda_A^{\mathsf{OP}}}(m) \stackrel{\text{df.}}{=} \begin{cases} \mathsf{P} & \text{if } \lambda_A^{\mathsf{OP}}(m) = \mathsf{O} \\ \mathsf{O} & \text{otherwise} \end{cases}$;
- $\star \vdash_{A \multimap B} m \stackrel{\text{df.}}{\Leftrightarrow} \star \vdash_B m$;
- $m \vdash_{A \multimap B} n \ (m \neq \star) \stackrel{\text{df.}}{\Leftrightarrow} (m \vdash_A n) \vee (m \vdash_B n) \vee (\star \vdash_B m \wedge \star \vdash_A n)$;
- $P_{A \multimap B} \stackrel{\text{df.}}{=} \{\boldsymbol{s} \in \mathscr{L}_{A \multimap B} \mid \ \boldsymbol{s} \upharpoonright A \in P_A, \boldsymbol{s} \upharpoonright B \in P_B\}$;
- $\boldsymbol{s} \simeq_{A \multimap B} \boldsymbol{t} \stackrel{\text{df.}}{\Leftrightarrow} \boldsymbol{s} \upharpoonright A \simeq_A \boldsymbol{t} \upharpoonright A \wedge \boldsymbol{s} \upharpoonright B \simeq_B \boldsymbol{t} \upharpoonright B \wedge \forall i \in \mathbb{N}. \boldsymbol{s}(i) \in M_A \Leftrightarrow \boldsymbol{t}(i) \in M_A$

where pointers between initial occurrences from A and from B in $\boldsymbol{s}$ are deleted.

In the domain A of a linear implication $A \multimap B$, the roles of Player and Opponent are interchanged; it is only the difference between $A \multimap B$ and the tensor $A \otimes B$. Dually to $A \otimes B$, it is easy to see that during a play of $A \multimap B$ only Player may switch between A and B. Note that the case distinction on whether or not $B = T$ for $A \multimap B$ is just to preserve *economy* of games (Definition 2.2.15).

Example 2.2.22. A maximal position of the linear implication $N \multimap N$ is either of the following forms:

$$\begin{tikzcd}[row sep=small]
N & \multimap & N \\
 & & q \\
q \arrow[urr] & & \\
n \arrow[u, bend left] & & \\
 & & m \arrow[uuu, bend right]
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}[row sep=small]
N & \multimap & N \\
 & & q \\
 & & m \arrow[u, bend right]
\end{tikzcd}$$

where $n, m \in \mathbb{N}$. The left diagram represents a *strict* linear function as it asks an input before producing an output, while the right diagram does a non-strict one.

Next, *product* & forms the categorical product in the categories of games given in Section 2.4. A position of the product $A \& B$ is simply a position of A or B:

Definition 2.2.23 (Product [7, 14])**.** Given games A and B, the ***product*** $A \& B$ of A and B is defined by:

- $M_{A\&B} \overset{\mathrm{df.}}{=} M_A + M_B$;
- $\lambda_{A\&B} \overset{\mathrm{df.}}{=} [\lambda_A, \lambda_B]$;
- $\vdash_{A\&B} \overset{\mathrm{df.}}{=} \vdash_A + \vdash_B$;
- $P_{A\&B} \overset{\mathrm{df.}}{=} \{\boldsymbol{s} \in \mathscr{L}_{A\&B} \mid (\boldsymbol{s} \upharpoonright A \in P_A \wedge \boldsymbol{s} \upharpoonright B = \boldsymbol{\epsilon}) \vee (\boldsymbol{s} \upharpoonright A = \boldsymbol{\epsilon} \wedge \boldsymbol{s} \upharpoonright B \in P_B)\}$;
- $\boldsymbol{s} \simeq_{A\&B} \boldsymbol{t} \overset{\mathrm{df.}}{\Leftrightarrow} \boldsymbol{s} \simeq_A \boldsymbol{t} \vee \boldsymbol{s} \simeq_B \boldsymbol{t}$.

Example 2.2.24. A maximal position of the product $\mathbf{2} \& N$ is either of the following forms:

$$\begin{tikzcd}[row sep=small]
\mathbf{2} & \& & N \\
q & & \\
b \arrow[u, bend right] & &
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}[row sep=small]
\mathbf{2} & \& & N \\
 & & q \\
 & & n \arrow[u, bend right]
\end{tikzcd}$$

where $b \in \mathbb{B}$ and $n \in \mathbb{N}$.

Now, let us recall *exponential* !, which is intuitively the countably infinite iteration of tensor, i.e., $!A \cong A \otimes A \otimes \dots$ for any game A.

Definition 2.2.25 (Exponential [7, 9, 129])**.** Given a game A, the ***exponential*** $!A$ of A is defined by:

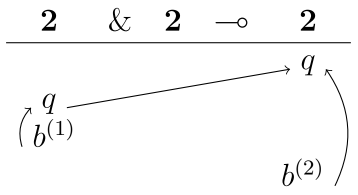
- $M_{!A} \overset{\mathrm{df.}}{=} M_A \times \mathbb{N}$;
- $\lambda_{!A} : (a, i) \mapsto \lambda_A(a)$;
- $\star \vdash_{!A} (a, i) \overset{\mathrm{df.}}{\Leftrightarrow} \star \vdash_A a$;
- $(a, i) \vdash_{!A} (a', j) \overset{\mathrm{df.}}{\Leftrightarrow} i = j \wedge a \vdash_A a'$;
- $P_{!A} \overset{\mathrm{df.}}{=} \{\boldsymbol{s} \in \mathscr{L}_{!A} \mid \forall i \in \mathbb{N}.\, \boldsymbol{s} \upharpoonright i \in P_A\}$;
- $\boldsymbol{s} \simeq_{!A} \boldsymbol{t} \overset{\mathrm{df.}}{\Leftrightarrow} \exists \varphi \in \mathcal{P}(\mathbb{N}).\, \forall i \in \mathbb{N}.\, \boldsymbol{s} \upharpoonright \varphi(i) \simeq_A \boldsymbol{t} \upharpoonright i \wedge \pi_2^*(\boldsymbol{s}) = (\varphi \circ \pi_2)^*(\boldsymbol{t})$

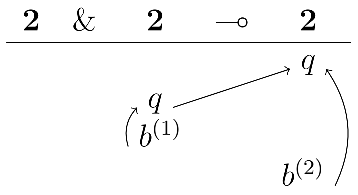
where $\boldsymbol{s} \upharpoonright i$ is the j-subsequence of $\boldsymbol{s}$ that consists of occurrences of moves of the form (a, i) but changed into a, and $\mathcal{P}(\mathbb{N})$ is the set of all permutations of natural numbers.

Now, it should be clear, from the definition of $\simeq_{!A}$, why we have equipped each game with an identification of positions: A particular choice of 'tags' $(\_, i)$ for an exponential $!A$ should not matter; since this identification may occur *locally* in games in a *nested* form, e.g., $!(!A \otimes B)$, $!A \multimap B$, etc., it gives a neat solution to define a *tailored* identification $\simeq_G$ of positions as part of the structure of each game G.

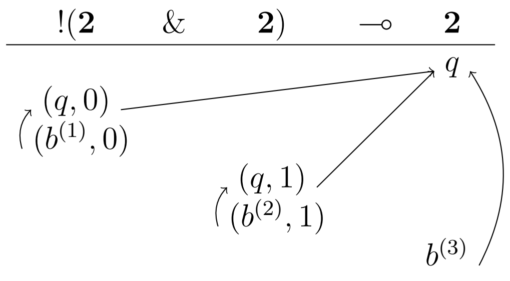
Exponential enables us, via *Girard's translation* [70] $A \Rightarrow B \overset{\mathrm{df.}}{=} !A \multimap B$, to model the construction $\Rightarrow$ of the usual ***implication*** (or the ***function space***).

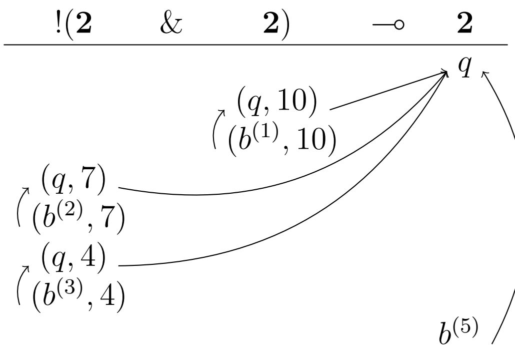
Example 2.2.26. In the linear implication $\mathbf{2}\&\mathbf{2} \multimap \mathbf{2}$, Player may play only in one $\mathbf{2}$ of the domain $\mathbf{2}\&\mathbf{2}$:





where $b^{(1)}, b^{(2)} \in \mathbb{B}$. On the other hand, positions of the implication $\mathbf{2}\&\mathbf{2} \Rightarrow \mathbf{2} =$ $!!(\mathbf{2}\&\mathbf{2}) \multimap \mathbf{2}$ are of the expected forms; for instance:





where $b^{(1)}, b^{(2)}, b^{(3)}, b^{(4)}, b^{(5)} \in \mathbb{B}$. Hence, e.g., Player may play as conjunction $\wedge : \mathbb{B} \to \mathbb{B}$ or disjunction $\vee : \mathbb{B} \to \mathbb{B}$ on the implication $\mathbf{2}\&\mathbf{2} \Rightarrow \mathbf{2}$ in the obvious manner, but not on the linear implication $\mathbf{2}\&\mathbf{2} \multimap \mathbf{2}$. This example illustrates why the standard notion of functions corresponds in game semantics to the construction $\Rightarrow$, not $\multimap$.

Note that on the game $\mathbf{2}\&\mathbf{2} \Rightarrow \mathbf{2}$ it is possible for Player to 'intentionally' keep playing in the domain $!(2\&2)$ *infinitely*, but it should be seen as a *loss* or a *defeat* of Player since intuitively a 'valid argument' should not just prolong a play. We shall formalize this idea precisely in Section 2.3 via the notion of *winning strategies*.

Finally, let us consider *weak sum* $+$ of games. The first proposal of weak sum of games is introduced by McCusker in [129], but it is defined only on *well-opened* games. He has later overcome this point in [128] by the following solution:

Definition 2.2.27 (Weak sum [128])**.** Given games A and B, the ***weak sum*** $A+B$ of A and B is defined by:

- $M_{A+B} \stackrel{\mathrm{df.}}{=} \{q_+, l, r\} + M_A + M_B$, where q_+, l and r are any pairwise distinct elements;
- $\lambda_{A+B} : q_+ \mapsto \mathsf{OQ}, l \mapsto \mathsf{PA}, r \mapsto \mathsf{PA}, (a \in M_A) \mapsto \lambda_A(a), (b \in M_B) \mapsto \lambda_B(b)$;
- $\star \vdash_{A+B} m \stackrel{\mathrm{df.}}{\Leftrightarrow} m = q_+$;
- $m \vdash_{A+B} n \ (m \neq \star) \stackrel{\mathrm{df.}}{\Leftrightarrow} (m = q_+ \wedge (n = l \vee n = r))$
  $\vee (m = l \wedge \star \vdash_A n) \vee (m = r \wedge \star \vdash_B n) \vee m \vdash_A n \vee m \vdash_B n$;
- $P_{A+B} \stackrel{\mathrm{df.}}{=} \mathsf{Pref}(\{q_+ l a_1 \boldsymbol{s_1} q_+ l a_2 \boldsymbol{s_2} \dots q_+ l a_k \boldsymbol{s_k} \in \mathscr{L}_{A+B} \mid a_1 \boldsymbol{s_1} a_2 \boldsymbol{s_2} \dots a_k \boldsymbol{s_k} \in P_A, \forall i \in \{1, 2, \dots, k\}. \star \vdash_A a_i \wedge \mathsf{NonInit}_A(\boldsymbol{s_i})\ \} \cup \{q_+ r b_1 \boldsymbol{t_1} q_+ r b_2 \boldsymbol{t_2} \dots q_+ r b_k \boldsymbol{t_k} \in \mathscr{L}_{A+B} \mid b_1 \boldsymbol{t_1} b_2 \boldsymbol{t_2} \dots b_k \boldsymbol{t_k} \in P_B, \forall i \in \{1, 2, \dots, k\}. \star \vdash_B b_i \wedge \mathsf{NonInit}_B(\boldsymbol{t_i})\ \})$, where q_+ justifies l and r, l (resp. r) justifies initial occurrences of A (resp. B), and $\mathsf{NonInit}_G(\boldsymbol{s}) \stackrel{\mathrm{df.}}{\Leftrightarrow}$ there is no initial move of the arena G occurring in $\boldsymbol{s}$;
- $\simeq_{A+B} \stackrel{\mathrm{df.}}{=} \{(\boldsymbol{s}, \boldsymbol{s'}) \in P_{A+B} \times P_{A+B} \mid \boldsymbol{s} \upharpoonright A \simeq_A \boldsymbol{s'} \upharpoonright A \vee \boldsymbol{s} \upharpoonright B \simeq_B \boldsymbol{s'} \upharpoonright B\}$, where $\boldsymbol{s} \upharpoonright A$ (resp. $\boldsymbol{s} \upharpoonright B$) is the j-subsequence of $\boldsymbol{s}$ that consists of moves different from q_+ and l (resp. r).

Thus, a play of a weak sum $A+B$ begins with Opponent's question q_+ ('A or B?') followed by Player's answer l ('I select A!') or r ('I select B!'), and then a play of A (resp. B) follows if the answer is l (resp. r), in which the initial protocol $q_+ l$ (resp. $q_+ r$) is inserted everytime Opponent performs an initial move in A (resp. B).

Note that this weak sum is almost the same as the one defined in [129] except that an initial protocol occurs at most once in a play of the latter. Thus, the two notions of weak sum coincide on *well-opened* games, but for the latter we cannot form an *innocent copairing* of innocent strategies unless the codomain is well-opened, while it is not the case for former. Another advantage of the weak sum in [128] over the one in [129] is that the former gives a unifying treatment of weak coproducts in both linear and intuitionistic settings, while the latter needs additional work to adapt it for the intuitionistic case. For these reasons, we have adopted the weak sum in [128].

Nevertheless, as explained in [129], the weak sum $+$ is *weak* in the sense that it does not satisfy the universal property of coproduct; we shall explain it in Section 2.4.

Example 2.2.28. Consider the weak sum $\mathbf{2} + N$. Its maximal position is either of the following forms:

$$\begin{tikzcd}
\mathbf{2} & + & N \\
 & q_+ & \\
 & l \arrow[u, bend right] & \\
q & & \\
b \arrow[u, bend right] & &
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathbf{2} & + & N \\
 & q_+ & \\
 & r \arrow[u, bend left] & \\
 & & q \\
 & & n \arrow[u, bend left]
\end{tikzcd}$$

where $b \in \mathbb{B}$ and $n \in \mathbb{N}$.

Theorem 2.2.29 (Constructions on games)**.** *Games are closed under tensor $\otimes$, linear implication $\multimap$, product $\&$, exponential $!$ and weak sum $+$; and these constructions preserve the subgame relation $\trianglelefteq$. Moreover, $\multimap$, $\&$ and $+$ preserve economy, well-openness and well-foundedness, while $\otimes$ and $!$ preserve economy and well-foundedness.*

Proof. See [14, 129] for the proof of the closure of games under the constructions. The preservation of the subgame relation and the constraints under the constructions is straightforward to verify, and thus we leave the details to the reader. □

Notation. Exponential precedes any other constructions on games; tensor, product and weak sum all precede linear implication. Tensor, product and weak sum are all left associative, while linear implication is right associative. For instance, $!A \multimap B = (!A) \multimap B$, $A \otimes !B \multimap !C\&D = (A \otimes (!B)) \multimap ((!C)\&D)$, $A\&B\&C = (A\&B)\&C$ and $A \multimap B \multimap C = A \multimap (B \multimap C)$.

2.3 Strategies

In this section, let us recall another central notion of *strategies*.

2.3.1 Strategies

Strategies are usually formulated as follows:

Definition 2.3.1 (Strategies [7, 9, 100, 14])**.** A ***strategy*** on a game G is a subset $\sigma \subseteq P_G^{\mathsf{Even}}$, written $\sigma : G$, that satisfies:

- (S1) Non-empty and *even-prefix-closed* (i.e., $\forall \boldsymbol{s}mn \in \sigma . \, \boldsymbol{s} \in \sigma$);
- (S2) *Deterministic*, i.e., $\forall \boldsymbol{s}mn, \boldsymbol{s}mn' \in \sigma . \, \boldsymbol{s}mn = \boldsymbol{s}mn'$.

Example 2.3.2. There is only the trivial strategy $\_ \overset{\text{df.}}{=} \{\boldsymbol{\epsilon}\}$ on the terminal game T. There is the strategy $\underline{n} \overset{\text{df.}}{=} \{\boldsymbol{\epsilon}, q.n\}$ on the game N for each $n \in \mathbb{N}$, and there are the strategies $\underline{b} \overset{\text{df.}}{=} \{\boldsymbol{\epsilon}, q.b\}$ on the game $\mathbf{2}$ for each $b \in \mathbb{B}$.

As positions of a game G are identified up to the identification $\simeq_G$, we must identify strategies on G if they behave in the same manner up to $\simeq_G$, leading to:

Definition 2.3.3 (Identification of strategies [9, 129])**.** The ***identification of strategies*** on a game G, written $\simeq_G$, is the relation between strategies on G given by:

$$\forall \sigma, \tau : G . \, \sigma \simeq_G \tau \overset{\text{df.}}{\Leftrightarrow} \forall \boldsymbol{s} \in \sigma, \boldsymbol{t} \in \tau . \, \boldsymbol{s}m \simeq_G \boldsymbol{t}l \Rightarrow \forall \boldsymbol{s}mn \in \sigma . \, \exists \boldsymbol{t}lr \in \tau . \, \boldsymbol{s}mn \simeq_G \boldsymbol{t}lr \\ \wedge \forall \boldsymbol{t}lr \in \tau . \, \exists \boldsymbol{s}mn \in \sigma . \, \boldsymbol{t}lr \simeq_G \boldsymbol{s}mn.$$

It is not hard to see that the identification $\simeq_G$ of strategies on each game G forms a *partial equivalence relation (PER)*, i.e., a symmetric and transitive relation:

Lemma 2.3.4 (First PER lemma)**.** *Given a game G, let $\sigma, \tau : G$ such that $\sigma \simeq_G \tau$. Then, $(\forall \boldsymbol{s} \in \sigma . \, \exists \boldsymbol{t} \in \tau . \, \boldsymbol{s} \simeq_G \boldsymbol{t}) \wedge (\forall \boldsymbol{t} \in \tau . \, \exists \boldsymbol{s} \in \sigma . \, \boldsymbol{t} \simeq_G \boldsymbol{s})$.*

Proof. By symmetry, it suffices to show $\forall \boldsymbol{s} \in \sigma . \, \exists \boldsymbol{t} \in \tau . \, \boldsymbol{s} \simeq_G \boldsymbol{t}$. We prove it by induction on $|\boldsymbol{s}|$. The base case is trivial; for the inductive step, let $\boldsymbol{s}mn \in \sigma$. By the induction hypothesis, there exists some $\boldsymbol{t} \in \tau$ such that $\boldsymbol{s} \simeq_G \boldsymbol{t}$. Then, by I3 on $\simeq_G$, there exists some $\boldsymbol{t}l \in \tau$ such that $\boldsymbol{s}m \simeq_G \boldsymbol{t}l$. Finally, since $\sigma \simeq_G \tau$, there exists some $\boldsymbol{t}lr \in \tau$ such that $\boldsymbol{s}mn \simeq_G \boldsymbol{t}lr$, completing the proof. □

Corollary 2.3.5 (PERs on strategies)**.** *Given a game G, the identification $\simeq_G$ of strategies on G is a PER.*

Proof. We just show the transitivity as the symmetry is obvious. Let $\sigma, \tau, \mu : G$ such that $\sigma \simeq_G \tau$ and $\tau \simeq_G \mu$. Assume that $\boldsymbol{s}mn \in \sigma$, $\boldsymbol{u} \in \mu$ and $\boldsymbol{s}m \simeq_G \boldsymbol{u}p$. By Lemma 2.3.4, there exists some $\boldsymbol{t} \in \tau$ such that $\boldsymbol{s} \simeq_G \boldsymbol{t}$. By I3 on $\simeq_G$, there exists some $\boldsymbol{t}l \in P_G$ such that $\boldsymbol{s}m \simeq_G \boldsymbol{t}l$, whence $\boldsymbol{t}l \simeq_G \boldsymbol{u}p$. Also, since $\sigma \simeq_G \tau$, there exists some $\boldsymbol{t}lr \in \tau$ such that $\boldsymbol{s}mn \simeq_G \boldsymbol{t}lr$. Finally, since $\tau \simeq_G \mu$, there exists some $\boldsymbol{u}pq \in \mu$ such that $\boldsymbol{t}lr \simeq_G \boldsymbol{u}pq$, whence $\boldsymbol{s}mn \simeq_G \boldsymbol{u}pq$, completing the proof. □

We are particularly concerned with strategies *identified with themselves*:

Definition 2.3.6 (Validity of strategies [9, 129]). A strategy $\sigma : G$ is ***valid*** if $\sigma \simeq_G \sigma$.

Remark. This notion is introduced in [9, 129], but these papers do not call it validity.

More explicitly, a strategy $\sigma : G$ is valid iff it satisfies:

$$(\textsc{Val})\ \forall \boldsymbol{s}, \boldsymbol{t} \in \sigma, \boldsymbol{s}m, \boldsymbol{t}l \in P_G.\, \boldsymbol{s}m \simeq_G \boldsymbol{t}l \Rightarrow \forall \boldsymbol{s}mn \in \sigma.\, \exists \boldsymbol{t}lr \in \sigma.\, \boldsymbol{s}mn \simeq_G \boldsymbol{t}lr.$$

Note that any strategy $\sigma : G$ identified with a strategy $\tau : G$ is valid: $\sigma \simeq_G \tau \simeq_G \sigma$.

Now, let us recall two constraints on strategies: *innocence* and *well-bracketing*. One of the highlights of *HO-games* [100] is to establish a one-to-one correspondence between terms of the programming language PCF in a certain *η-long normal form*, known as *PCF Böhm trees* [18], and innocent, well-bracketed strategies (on games modeling types of PCF). That is, the two conditions narrow down the hom-sets of the codomain of the interpretation functor, i.e., the category of HO-games, so that the interpretation becomes *full*. Roughly, a strategy is *innocent* if its computation depends only on P-views, and *well-bracketed* if every 'question-answering' by the strategy is achieved in the 'last-question-first-answered' fashion. Formally:

Definition 2.3.7 (Innocence of strategies [100, 14]). A strategy $\sigma : G$ is ***innocent*** if:

$$(\textsc{Inn})\ \forall \boldsymbol{s}mn, \boldsymbol{t} \in \sigma, \boldsymbol{t}m \in P_G.\, \lceil \boldsymbol{t}m \rceil_G = \lceil \boldsymbol{s}m \rceil_G \Rightarrow \boldsymbol{t}mn \in \sigma.$$

Definition 2.3.8 (Well-bracketing of strategies [14, 129]). A strategy $\sigma : G$ is ***well-bracketed (wb)*** if:

(WB) Given $\boldsymbol{s}q\boldsymbol{t}a \in \sigma$, where $\lambda_G^{\mathsf{QA}}(q) = \mathsf{Q}$, $\lambda_G^{\mathsf{QA}}(a) = \mathsf{A}$ and $\mathcal{J}_{\boldsymbol{s}q\boldsymbol{t}a}(a) = q$, each occurrence of a question in $\boldsymbol{t}'$, defined by $\lceil \boldsymbol{s}q\boldsymbol{t} \rceil_G = \lceil \boldsymbol{s}q \rceil_G . \boldsymbol{t}'$[3], justifies an occurrence of an answer in $\boldsymbol{t}'$.

The bijective correspondence holds also for the game model of [14]. Moreover, it corresponds respectively to modeling *states* and *control operators* in programming languages to relax innocence and well-bracketing in the model; in this sense, the two conditions characterize purely *functional* computation [14].

Next, recall that a programming language is *total* if its computation always terminates in a finite period of time. This phenomenon is interpreted in game semantics by *totality* of strategies in a sense similar to the totality of partial functions [5]:

[3]Note that $\lceil \boldsymbol{s}q\boldsymbol{t} \rceil_G$ must be of the form $\lceil \boldsymbol{s}q \rceil_G . \boldsymbol{t}'$ by visibility on $\boldsymbol{s}q\boldsymbol{t}a$ (Definition 2.2.9).

Definition 2.3.9 (Totality of strategies [5]). A strategy $\sigma : G$ is ***total*** if it satisfies:

$$(\textsc{Tot})\ \forall \boldsymbol{s} \in \sigma, \boldsymbol{s}m \in P_G.\, \exists \boldsymbol{s}mn \in \sigma.$$

Nevertheless, it is well-known that totality of strategies is *not* preserved under composition (Definition 2.3.14) due to the problem of 'infinite chattering' [5, 37]. For this point, one usually imposes a condition on strategies stronger than totality, e.g., *winning* [5], that is preserved under composition. We may certainly just apply the winning condition of [5], but it requires an additional structure on games, which may be criticized as extrinsic and/or ad-hoc; thus, we prefer another, simpler solution. A natural idea is then to require that strategies should not contain any strictly increasing (with respect to $\preceq$) infinite sequence of positions. However, we have to relax this constraint: The dereliction der_A (Definition 2.3.26), the identity on a game A in the categories of games and strategies given in Section 2.4, satisfies it iff so does the game A, but we cannot impose it on games as the binary operation $\Rightarrow\ =\ !(\_) \multimap (\_)$ on games, which is the exponential construction (in the sense of the function space) in the categories, does not preserve it.

Instead, we apply the same idea to *P-views*, arriving at:

Definition 2.3.10 (Noetherianity of strategies [37]). A strategy $\sigma : G$ is ***noetherian*** if it satisfies:

> ($\textsc{Noe}$) σ does not contain any strictly increasing (with respect to $\preceq$) infinite sequence of P-views of G.

2.3.2 Constructions on Strategies

Next, let us review existing constructions on strategies. Although it is straightforward to prove that they are well-defined and preserve validity, innocence, well-bracketing, totality and noetherianity of strategies, we present this fact explicitly as lemmata since our games and strategies in Chapter 5 are based on this fact.

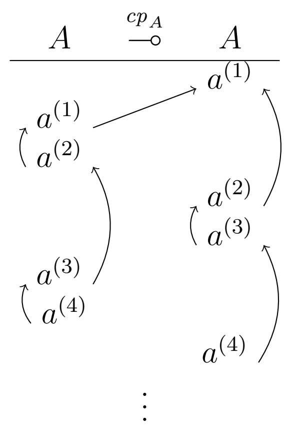
One of the most basic strategies is *copy-cats*, which, as the name suggests, simply 'copy-cats' the last O-moves:

Definition 2.3.11 (Copy-cats [7, 9, 100, 129]). The ***copy-cat (strategy)*** cp_A on a game A is defined by:

$$cp_A \overset{\text{df.}}{=} \{\boldsymbol{s} \in P^{\mathsf{Even}}_{A_{[0]} \multimap A_{[1]}} \mid \forall \boldsymbol{t} \preceq \boldsymbol{s}.\ \mathsf{Even}(\boldsymbol{t}) \Rightarrow \boldsymbol{t} \upharpoonright A_{[0]} = \boldsymbol{t} \upharpoonright A_{[1]}\}$$

where the subscripts $[i]$ on A ($i = 0, 1$) are to distinguish the two copies of A.

Diagrammatically, the copy-cat cp_A plays as follows:



Lemma 2.3.12 (Well-defined copy-cats [7, 129]). *Given a game A, cp_A is a valid, innocent, wb, total strategy on $A \multimap A$. It is noetherian if A is well-founded.*

Proof. We just show that cp_A is noetherian if A is well-founded for the other points are trivial, e.g., validity of cp_A is immediate from the definition of $\simeq_{A \multimap A}$. Given $\boldsymbol{s}mm \in cp_A$, it is easy to see by induction on $|\boldsymbol{s}|$ that the P-view $\lceil \boldsymbol{s}m \rceil$ is of the form $m_1 m_1 m_2 m_2 \dots m_k m_k m$, and thus there is a sequence $\star \vdash_A m_1 \vdash_A m_2 \cdots \vdash_A m_k \vdash_A m$ of enabling pairs. Therefore, if A is well-founded, then cp_A must be noetherian. $\square$

Next, to formulate *composition* of strategies, it is convenient to first define the following intermediate concept:

Definition 2.3.13 (Parallel composition of strategies [5]). Given games A, B and C, and strategies $\phi : A \multimap B$ and $\psi : B \multimap C$, the ***parallel composition*** $\phi \| \psi$ of ϕ and ψ is given by:

$$\phi \| \psi \stackrel{\text{df.}}{=} \{\boldsymbol{s} \in \mathscr{J}_{((A \multimap B_{[0]}) \multimap B_{[1]}) \multimap C} \mid \boldsymbol{s} \upharpoonright A, B_{[0]} \in \phi, \boldsymbol{s} \upharpoonright B_{[1]}, C \in \psi, \boldsymbol{s} \upharpoonright B_{[0]}, B_{[1]} \in pr_B \}$$

where the subscripts $[i]$ on B $(i = 0, 1)$ are to distinguish the two copies of B, $\boldsymbol{s} \upharpoonright A, B_{[0]}$ (resp. $\boldsymbol{s} \upharpoonright B_{[1]}, C$, $\boldsymbol{s} \upharpoonright B_{[0]}, B_{[1]}$) is the j-subsequence of $\boldsymbol{s}$ that consists of moves of A and $B_{[0]}$ (resp. $B_{[1]}$ and C, $B_{[0]}$ and $B_{[1]}$) as in Definition 2.2.21, and $pr_B \stackrel{\text{df.}}{=} \{\boldsymbol{s} \in P_{B_{[0]} \multimap B_{[1]}} \mid \forall \boldsymbol{t} \preceq \boldsymbol{s}.\ \mathsf{Even}(\boldsymbol{t}) \Rightarrow \boldsymbol{t} \upharpoonright B_{[0]} = \boldsymbol{t} \upharpoonright B_{[1]}\}$.

Remark. Parallel composition is just a preliminary for composition given below; it does *not* preserve the structure of strategies.

Now, we are ready to recall *composition* of strategies.

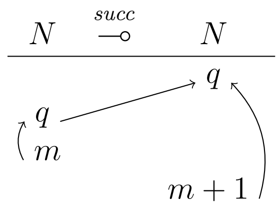
Definition 2.3.14 (Composition of strategies [100, 9, 14])**.** Given games A, B and C, and strategies $\phi : A \multimap B$ and $\psi : B \multimap C$, the ***composition*** $\phi ; \psi$ of ϕ and ψ (also written $\psi \circ \phi$) is defined by:

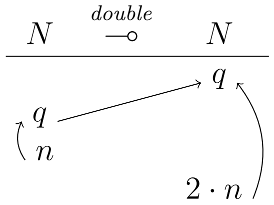
$$\phi ; \psi \stackrel{\mathrm{df.}}{=} \{ \boldsymbol{s} \upharpoonright A, C \mid \boldsymbol{s} \in \phi \| \psi \}.$$

That is, the composition $\phi ; \psi : A \multimap C$ of strategies $\phi : A \multimap B$ and $\psi : B \multimap C$ plays implicitly on $((A \multimap B_{[0]}) \multimap B_{[1]}) \multimap C$, employing ϕ if the last O-move is of A or $B_{[0]}$, and ψ otherwise, while Opponent plays on $A \multimap C$, where ϕ and ψ communicate with each other via moves of $B_{[0]}$ or $B_{[1]}$, but the communication is 'hidden' from Opponent. This idea originally comes from *parallel composition plus hiding* [5] in process calculi [90, 134].

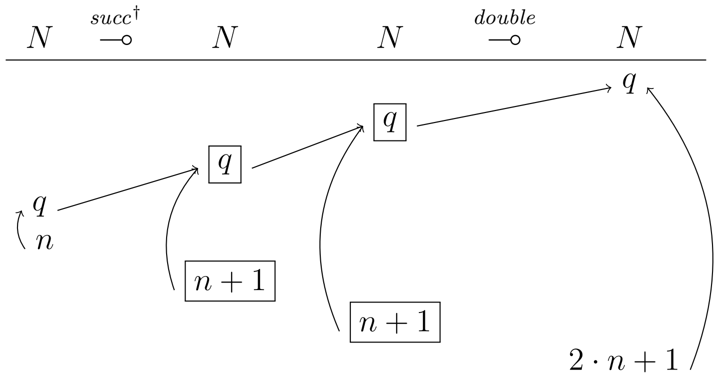
Notation. Given a strategy $\sigma : G$, we write σ^T for the obvious strategy on $T \multimap G$ that coincides with σ up to 'tags'; we write $(\_)^{\ominus T}$ for the inverse of $(\_)^T$. We define the ***composition*** $\phi \circ \alpha : B$ of strategies $\phi : A \multimap B$ and $\alpha : A$ by $\phi \circ \alpha \stackrel{\mathrm{df.}}{=} (\phi \circ \alpha^T)^{\ominus T}$. Note that the exponential $!T$ coincides with T on the nose, and therefore we define the ***composition*** $\varphi \circ \tilde{\alpha} : A \Rightarrow B$ of strategies $\varphi : A \Rightarrow B$ and $\tilde{\alpha} : !A$ by $\phi \circ \tilde{\alpha} \stackrel{\mathrm{df.}}{=} (\phi \circ \tilde{\alpha}^T)^{\ominus T}$.

Example 2.3.15. Consider the composition *succ*; *double* : $N \multimap N$ of strategies *succ* : $N \multimap N$ and *double* : $N \multimap N$ (in linear form) that play as in the following diagrams:

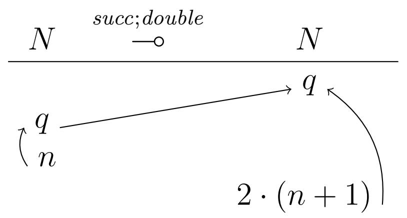




where $m, n \in \mathbb{N}$. The parallel composition *succ*$\|$*double* then plays as follows:


$N \overset{succ^\dagger}{\multimap} N$
$N \overset{double}{\multimap} N$
q
$\boxed{q}$
$\boxed{q}$
q
n
$\boxed{n+1}$
$\boxed{n+1}$
$2 \cdot n + 1$

where moves for internal communication are marked by square boxes just for clarity. Then, the composition is computed from the parallel composition by deleting the moves with the square boxes, resulting in the following diagram as expected:



Lemma 2.3.16 (Well-defined composition of strategies [129, 37])**.** *Given games A, B and C, and strategies $\phi : A \multimap B$ and $\psi : B \multimap C$, $\phi;\psi$ is a strategy on $A \multimap C$. If ϕ and ψ are innocent, total and noetherian (resp. wb), then so is $\phi;\psi$. Given $\phi' : A \multimap B$ and $\psi' : B \multimap C$ with $\phi \simeq_{A \multimap B} \phi'$ and $\psi \simeq_{B \multimap C} \psi'$, $\phi;\psi \simeq_{A \multimap C} \phi';\psi'$.*

Proof. It is well-known that strategies are closed under composition, and composition preserves innocence and well-bracketing; see [14, 129]. Also, it is shown in [37] that the conjunction of innocence, totality and noetherianity is preserved under composition. Finally, it is easy to see that composition preserves identification of strategies. □

Next, let us recall *tensor* $\otimes$ of strategies:

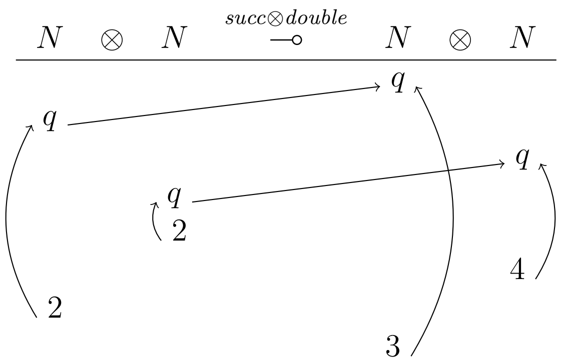
Definition 2.3.17 (Tensor of strategies [7, 129])**.** Given games A, B, C and D, and strategies $\phi : A \multimap C$ and $\psi : B \multimap D$, the ***tensor (product)*** $\phi \otimes \psi$ of ϕ and ψ is given by:

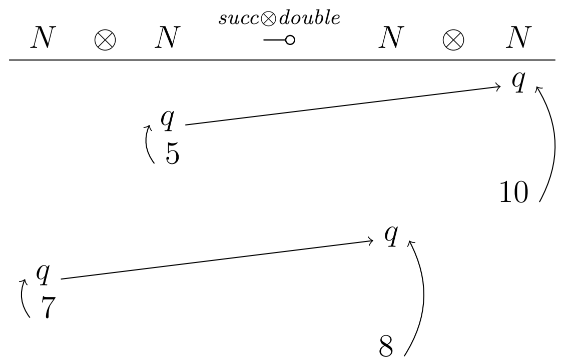
$$\phi \otimes \psi \stackrel{\text{df.}}{=} \{\boldsymbol{s} \in \mathscr{L}_{A \otimes B \multimap C \otimes D} \mid \boldsymbol{s} \upharpoonright A, C \in \phi, \boldsymbol{s} \upharpoonright B, D \in \psi\}.$$

where $\boldsymbol{s} \upharpoonright A, C$ (resp. $\boldsymbol{s} \upharpoonright B, D$) is the j-subsequence of $\boldsymbol{s}$ that consists of moves of A and C (resp. B and D).

Intuitively the tensor $\phi \otimes \psi : A \otimes B \multimap C \otimes D$ of $\phi : A \multimap C$ and $\psi : B \multimap D$ plays by ϕ if the last O-move is of A or C, and by ψ otherwise.

Example 2.3.18. The tensor $succ \otimes double : N \otimes N \multimap N \otimes N$ plays, e.g., as follows:


$N \otimes N \overset{succ \otimes double}{\multimap} N \otimes N$
q
q
q
q
2
4
2
3


$N \otimes N \overset{succ \otimes double}{\multimap} N \otimes N$
q
q
5
10
q
q
7
8

Lemma 2.3.19 (Well-defined tensor of strategies [7, 129])**.** *Given games A, B, C and D, and strategies $\phi : A \multimap C$ and $\psi : B \multimap D$, $\phi \otimes \psi$ is a strategy on $A \otimes B \multimap C \otimes D$. If ϕ and ψ are innocent (resp. wb, total, noetherian), then so is $\phi \otimes \psi$. Given $\phi' : A \multimap C$ and $\psi' : B \multimap D$ with $\phi \simeq_{A \multimap C} \phi'$ and $\psi \simeq_{B \multimap D} \psi'$, $\phi \otimes \psi \simeq_{A \otimes B \multimap C \otimes D} \phi' \otimes \psi'$.*

Proof. Straightforward; see [7, 14, 129, 9]. □

We proceed to recall the construction of *pairing* of strategies:

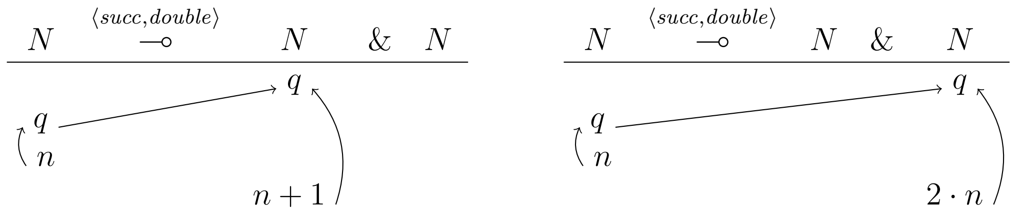
Definition 2.3.20 (Pairing of strategies [9, 129])**.** Given games A, B and C, and strategies $\phi : C \multimap A$ and $\psi : C \multimap B$, the ***pairing*** $\langle \phi, \psi \rangle$ of ϕ and ψ is defined by:

$$\langle \phi, \psi \rangle \overset{\mathrm{df.}}{=} \{ \boldsymbol{s} \in \mathscr{L}_{C \multimap A \& B} \mid (\boldsymbol{s} \upharpoonright C, A \in \phi \wedge \boldsymbol{s} \upharpoonright B = \boldsymbol{\epsilon}) \vee (\boldsymbol{s} \upharpoonright C, B \in \psi \wedge \boldsymbol{s} \upharpoonright A = \boldsymbol{\epsilon}) \}.$$

That is, the pairing $\langle \phi, \psi \rangle : C \multimap A \& B$ of $\phi : C \multimap A$ and $\psi : C \multimap B$ plays by ϕ if the play is of $C \multimap A$, and by ψ otherwise.

Notation. The ***pairing*** $\langle \alpha, \beta \rangle : A \& B$ of strategies $\alpha : A$ and $\beta : B$ is given by $\langle \alpha, \beta \rangle \overset{\mathrm{df.}}{=} \langle \alpha^T, \beta^T \rangle^{\ominus T}$.

Example 2.3.21. The pairing $\langle \mathit{succ}, \mathit{double} \rangle : N \multimap N \& N$ plays as either of the following forms:



$N \overset{\langle succ, double \rangle}{\multimap} N \& N$
q
q
n
$n+1$
$N \overset{\langle succ, double \rangle}{\multimap} N \& N$
q
q
n
$2 \cdot n$

where $n \in \mathbb{N}$, depending on Opponent's behavior.

Lemma 2.3.22 (Well-defined pairing of strategies [129])**.** *Given games A, B and C, and strategies $\phi : C \multimap A$ and $\psi : C \multimap B$, $\langle \phi, \psi \rangle$ is a strategy on $C \multimap A \& B$. If ϕ and ψ are innocent (resp. wb, total, noetherian), then so is $\langle \phi, \psi \rangle$. Given $\phi' : C \multimap A$ and $\psi' : C \multimap B$ with $\phi \simeq_{C \multimap A} \phi'$ and $\psi \simeq_{C \multimap B} \psi'$, $\langle \phi, \psi \rangle \simeq_{C \multimap A \& B} \langle \phi', \psi' \rangle$.*

Proof. Straightforward; see [14, 129, 9]. □

Next, let us recall *promotion* of strategies:

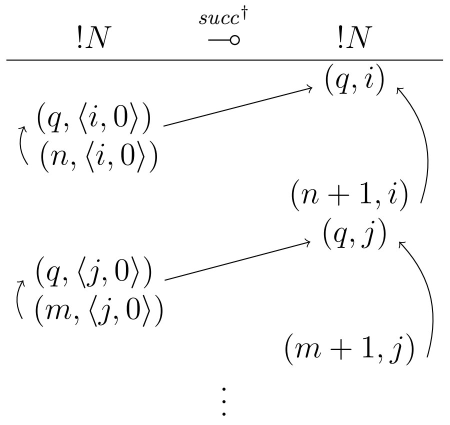
Definition 2.3.23 (Promotion of strategies [9, 129])**.** Given games A and B, and a strategy $\varphi : !A \multimap B$, the ***promotion*** $\varphi^\dagger$ of φ is defined by:

$$\varphi^\dagger \overset{\mathrm{df.}}{=} \{ \boldsymbol{s} \in \mathscr{L}_{!A \multimap !B} \mid \forall i \in \mathbb{N}. \, \boldsymbol{s} \upharpoonright i \in \varphi \}$$

where $\boldsymbol{s} \upharpoonright i$ denotes the j-subsequence of $\boldsymbol{s}$ that consists of moves of the form (b, i) with $b \in M_B$ or $(a, \langle i, j \rangle)$ with $a \in M_A$ changed into b and (a, j), respectively, and $\langle \_, \_ \rangle : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ is any bijection fixed throughout the present thesis.

That is, a promotion $\varphi^\dagger : !A \multimap !B$ plays during a play $\boldsymbol{s}$ of $!A \multimap !B$ as φ for each j-subsequence $\boldsymbol{s} \upharpoonright i$. We could have defined noetherianity of strategies in terms of positions, but then it would not be preserved under promotion; this is why we have defined it in terms of P-views.

Example 2.3.24. Let $succ : !N \multimap N$ be the successor strategy (n.b., it is on the implication, not the linear implication), which specifically selects, say, the 'tag' $(\_, 0)$ in the domain $!N$. Then, its promotion $succ^\dagger : !N \multimap !N$ plays as follows:



$!N$
$\overset{succ^\dagger}{\multimap}$
$!N$
(q, i)
$(q, \langle i, 0 \rangle)$
$(n, \langle i, 0 \rangle)$
$(n+1, i)$
(q, j)
$(q, \langle j, 0 \rangle)$
$(m, \langle j, 0 \rangle)$
$(m+1, j)$
$\vdots$

where $i, j, n, m \in \mathbb{N}$. Note that $succ^\dagger$ repeatedly and consistently plays as $succ$.

Lemma 2.3.25 (Well-defined promotion [129])**.** *Given games A and B, and a strategy $\varphi : !A \multimap B$, its promotion $\varphi^\dagger$ is a strategy on $!A \multimap !B$. If φ is innocent (resp. wb, total, noetherian), then so is $\varphi^\dagger$. Given $\tilde{\varphi} : !A \multimap B$ with $\varphi \simeq_{!A \multimap B} \tilde{\varphi}$, $\varphi^\dagger \simeq_{!A \multimap !B} \tilde{\varphi}^\dagger$.*

Proof. Straightforward; see [129, 9]. $\square$

Now, let us recall:

Definition 2.3.26 (Derelictions [9, 129])**.** The ***dereliction*** $der_A : !A \multimap A$ on a game A is defined by:

$$der_A \overset{\text{df.}}{=} \{ \boldsymbol{s} \in P^{\mathsf{Even}}_{!A \multimap A} \mid \forall \boldsymbol{t} \preceq \boldsymbol{s}. \, \mathsf{Even}(\boldsymbol{t}) \Rightarrow (\boldsymbol{t} \upharpoonright !A) \upharpoonright 0 = \boldsymbol{t} \upharpoonright A \}.$$

Thus, der_A plays essentially in the same manner as cp_A. Note that any 'tag' $(\_, i)$ such that $i \in \mathbb{N}$ would work; our particular choice $(\_, 0)$ does not matter.

Notation. Given a strategy $\sigma : G$, the ***promotion*** $\sigma^\dagger : !G$ is given by $\sigma^\dagger \overset{\text{df.}}{=} ((\sigma^T)^\dagger)^{\ominus T}$.

Lemma 2.3.27 (Well-defined derelictions [9, 129]). *Given a game A, der_A is a valid, innocent, wb, total strategy on $!A \multimap A$. It is noetherian if A is well-founded.*

Proof. Essentially the same as the proof of Lemma 2.3.12. $\square$

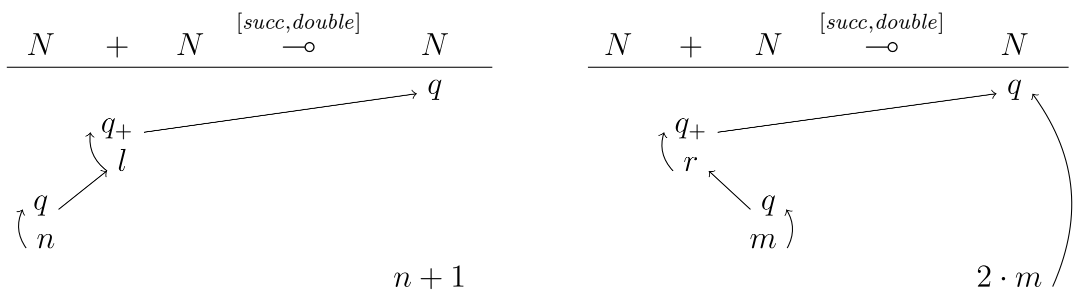
At the end of the present section, let us recall *copairing* and *injections*:

Definition 2.3.28 (Copairing of strategies [128]). Given games A, B and C, and strategies $\phi : A \multimap C$ and $\psi : B \multimap C$, the ***copairing*** $[\phi, \psi]$ of ϕ and ψ is defined by:

$$\begin{aligned}[\phi, \psi] \overset{\text{df.}}{=} \mathsf{Pref}(&\{c_0 q_+ l \boldsymbol{s_0} a_1 \boldsymbol{s_1} q_+ l a_2 \boldsymbol{s_2} \dots q_+ l a_k \boldsymbol{s_k} \in \mathscr{L}_{A+B \multimap C} \mid c_0 \boldsymbol{s_0} a_1 \boldsymbol{s_1} a_2 \boldsymbol{s_2} \dots a_k \boldsymbol{s_k} \in \phi,\\ &\forall i \in \{1, 2, \dots, k\}. \star \vdash_A a_i \wedge \mathsf{NonInit}_{A \multimap C}(\boldsymbol{s_i}) \}\\ &\cup \{c_0 q_+ r \boldsymbol{s_0} b_1 \boldsymbol{s_1} q_+ r b_2 \boldsymbol{s_2} \dots q_+ r b_k \boldsymbol{s_k} \in \mathscr{L}_{A+B \multimap C} \mid c_0 \boldsymbol{s_0} b_1 \boldsymbol{s_1} b_2 \boldsymbol{s_2} \dots b_k \boldsymbol{s_k} \in \psi,\\ &\forall i \in \{1, 2, \dots, k\}. \star \vdash_B b_i \wedge \mathsf{NonInit}_{B \multimap C}(\boldsymbol{s_i}) \})^{\mathsf{Even}}.\end{aligned}$$

That is, the copairing $[\phi, \psi] : A + B \multimap C$ of $\phi : A \multimap C$ and $\psi : B \multimap C$ initially asks Opponent about his choice on the domain, i.e., A or B, and then plays by ϕ if the answer is A (via l), and by ψ otherwise, where it inserts an initial protocol $q_+ l$ or $q_+ r$ whenever it performs an initial move of A or B.

Example 2.3.29. The copairing $[succ, double] : N + N \multimap N$ plays as either of the following forms:



where $n, m \in \mathbb{N}$, depending on Opponent's behavior.

Lemma 2.3.30 (Well-defined copairing of strategies [129]). *Given games A, B and C, and strategies $\phi : A \multimap C$ and $\psi : B \multimap C$, $[\phi, \psi]$ is a strategy on $A + B \multimap C$. If ϕ and ψ are innocent (resp. wb, total, noetherian), then so is $[\phi, \psi]$. Given $\phi' : A \multimap C$ and $\psi' : B \multimap C$ with $\phi \simeq_{A \multimap C} \phi'$ and $\psi \simeq_{A \multimap B} \psi'$, $[\phi, \psi] \simeq_{A+B \multimap C} [\phi', \psi']$.*

Proof. Straightforward; see [129]. $\square$

Definition 2.3.31 (Injections [129])**.** Given games A and B, the ***left injection*** $\iota_l^{A,B}$ and the ***right injection*** $\iota_r^{A,B}$ are defined by:

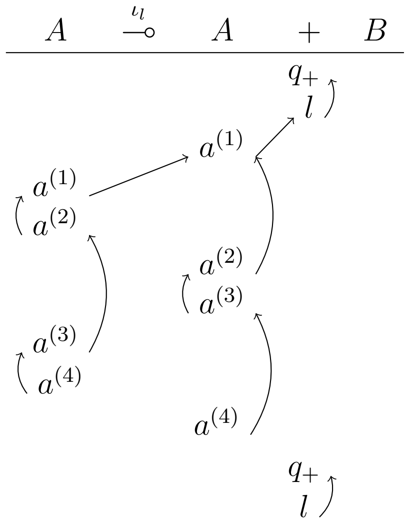
$$\iota_l^{A,B} \stackrel{\mathrm{df.}}{=} \mathsf{Pref}(\{q_+ l a_1 a_1 \boldsymbol{s_1} q_+ l a_2 a_2 \boldsymbol{s_2} \dots q_+ l a_k a_k \boldsymbol{s_k} \mid a_1 a_1 \boldsymbol{s_1} a_2 a_2 \boldsymbol{s_2} \dots a_k a_k \boldsymbol{s_k} \in cp_A, \forall i \in \{1, 2, \dots, k\}. \star \vdash_A a_i \wedge \mathsf{NonInit}_{A \multimap A}(\boldsymbol{s_i})\})^{\mathsf{Even}};$$

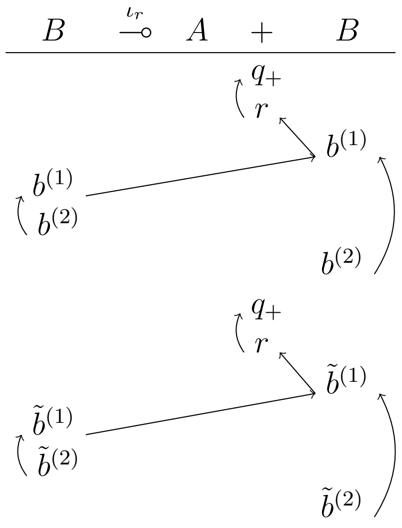
$$\iota_r^{A,B} \stackrel{\mathrm{df.}}{=} \mathsf{Pref}(\{q_+ r b_1 b_1 \boldsymbol{s_1} q_+ r b_2 b_2 \boldsymbol{s_2} \dots q_+ r b_k b_k \boldsymbol{s_k} \mid b_1 b_1 \boldsymbol{s_1} b_2 b_2 \boldsymbol{s_2} \dots b_k b_k \boldsymbol{s_k} \in cp_B, \forall i \in \{1, 2, \dots, k\}. \star \vdash_B b_i \wedge \mathsf{NonInit}_{B \multimap B}(\boldsymbol{s_i})\})^{\mathsf{Even}}.$$

Notation. We often omit the superscripts A, B on $\iota_l^{A,B}$ and $\iota_r^{A,B}$.

That is, the left injection $\iota_l : A \multimap A + B$ and the right injection $\iota_r : B \multimap A + B$ are the copy-cats cp_A and cp_B up to the initial protocols $q_+ l$ and $q_+ r$, respectively.

Example 2.3.32. Given games A and B, the injections $\iota_l^{A,B}$ and $\iota_r^{A,B}$ play for instance as follows:





where a_1 and a'_1 (resp. b_1 and b'_1) are initial moves of A (resp. B).

Lemma 2.3.33 (Well-defined injections [129])**.** *Given games A and B, the left (resp. right) injection $\iota_l^{A,B}$ (resp. $\iota_r^{A,B}$) is a valid, innocent, wb, total strategy on $A \multimap A+B$ (resp. $B \multimap A + B$). It is noetherian if A (resp. B) is well-founded.*

Proof. Essentially the same as the proof of Lemma 2.3.12. $\square$

Notation. Promotion precedes any other operation. Tensor and composition are left associative. For instance, $\phi \circ \sigma^\dagger = \phi \circ (\sigma^\dagger)$ and $\sigma \otimes \tau \otimes \mu = (\sigma \otimes \tau) \otimes \mu$.

2.4 Categories of Games and Strategies

We conclude the chapter by establishing twelve categories of games and strategies, eight for computation of *simple* type theories and the remaining four for intuitionistic linear, classical linear, intuitionistic and classical *propositional* logics. It is a striking and beautiful aspect of game semantics that the essentially combinatorial concepts of games and strategies give rise to neat categorical structures; it makes game semantics not only conceptually natural but also mathematically elegant and reasonable.

Note that morphisms in these categories are valid strategies *up to identifications* induced by the underlying games (Definition 2.3.3), similarly to the categories in [9] and Section 3.6 of [129]. It conceptually makes sense for concrete implementations of 'tags' for disjoint union of sets of moves should not matter; also technically, this level of identification of strategies matches the syntactic equality of programs as seen in the literature and also in Chapter 5. Nevertheless, it is important to keep in mind that it is valid strategies, not their equivalence classes, that guide Player on games.

2.4.1 Logic vs. Computation

We first describe the general difference between categories of games and strategies for logic and those for computation. Since games (resp. strategies) are to model formulas (resp. proofs) in logic and types (resp. programs) in computation (see Section 2.1), following the standard categorical approach [102, 150, 41], in both of the categories, objects are games and morphisms are (equivalence classes of valid) strategies.

As the first approximation, one may say that a category of games and strategies is for logic or *logical* if every strategy is *total*, and for computation or *computational* otherwise.[4] This is because a proof or an 'argument' for the truth of a formula should not get 'stuck', i.e., it must always have the next move, and in contrast, even such a non-responding strategy may be seen as a reasonable computational process.

In addition, since logic is concerned with truths of formulas, which are invariant with respect to passage of time, proofs should not depended on *states*. Hence, it makes sense to impose *innocence* on strategies to model proofs. Also, recall that totality per se is not preserved under composition of strategies, and our solution is to add innocence and noetherianity (see Section 2.3). The conjunction of these three conditions is conceptually reasonable too because if a play by an innocent, noetherian strategy keeps growing *infinitely*, then it cannot be Player's intention, and thus it should result in *win* for Player. In fact, however, it is win even in a stronger sense:

[4]This distinction corresponds in syntax to if every term terminates in a finite period of time.

Every play by an innocent, noetherian strategy is finite [37]. For this reason, we define a strategy to be ***winning*** if it is total, innocent and noetherian.

Definition 2.4.1 (Winning of strategies)**.** A strategy is ***winning*** if it is total, innocent and noetherian.

In summary, we regard a category of games and strategies as embodying some logic if its morphisms are all winning, and as modeling some computation otherwise.

On the other hand, it corresponds in programming to modeling states and control operators, respectively, to relax innocence and well-bracketing on morphisms in a computational category of games and strategies, as already explained in Section 2.3.1.

2.4.2 Cartesian Closure via New-Seely Categories

The present thesis is primarily concerned with *intuitionistic* logic (in Chapter 5) and *functional* computation (in Chapters 3 and 4), which mutually correspond to each other by the Curry-Howard isormorphism [172]. Categorically, their basic structures are *cartesian closed categories (CCCs)*, where products (resp. exponentials) model conjunction (resp. implication) in logic and pair-types (resp. function-types) in computation [118, 102]. Therefore, we review, as the main focus of the present section, both of the logical and computational CCCs of games and strategies.

We obtain these CCCs by following a general categorical recipe, namely as the *co-Kleisli categories* $\mathcal{C}_!$ of *new-Seely categories (NSCs)* [23]:

Definition 2.4.2 (NSCs [23])**.** A ***new-Seely category (NSC)*** is a symmetric monoidal closed category (SMCC) $\mathcal{C} = (\mathcal{C}, \otimes, I, \multimap)$ with finite products $(1, \times)$ equipped with a comonad $! = (!, \epsilon, \delta)$ on $\mathcal{C}$ and two natural isomorphisms $\eta : !A \otimes !B \xrightarrow{\sim} !(A \times B)$ and $\rho : I \xrightarrow{\sim} !1$ such that the canonical adjunction between $\mathcal{C}$ and $\mathcal{C}_!$ is monoidal.

Theorem 2.4.3 (CCCs via NSCs [165, 23])**.** *The co-Kleisli category $\mathcal{C}_!$ of a NSC $\mathcal{C}$ forms a CCC.*

Proof (sketch). We actually do not need all the components of a given NSC $\mathcal{C} = (\mathcal{C}, \otimes, I, \multimap, 1, \times, !, \epsilon, \delta, \eta, \rho)$; we only need that $\mathcal{C}$ is a *Seely category (SC)* [165]. Given $A, B \in \mathcal{C}_!$, we define the exponential $A \Rightarrow B \in \mathcal{C}_!$ to be $!A \multimap B$. Then, we have:

$$\begin{aligned}
\mathcal{C}_!(A \times B, C) &= \mathcal{C}(!(A \times B), C) \\
&\cong \mathcal{C}(!A \otimes !B, C) \\
&\cong \mathcal{C}(!A, !B \multimap C) \\
&= \mathcal{C}_!(A, B \Rightarrow C)
\end{aligned}$$

leaving the required naturality condition to the reader. □

Gavin Bierman [23] showed that SCs are *not* sound for *intuitionistic multiplicative exponential linear logic (IMELL)* though they give rise to CCCs, and proposes NSCs as a remedy; in fact, he proved that NSCs are a categorical model of IMELL in [23].

The charm of this construction is that it may be seen as the categorical counterpart of Girard's translation $A \Rightarrow B = !A \multimap B$ of intuitionistic logic into linear logic [70]. Although linear logic is not a main focus of the present thesis, we employ this construction of CCCs from NSCs since it shows that:

- The CCCs of games and strategies are mathematically reasonable (not ad-hoc);
- Games and strategies may give, to some degree, a unifying view on linear and intuitionistic logics.

2.4.3 Computational CCCs of Games and Strategies

Now, let us construct the CCCs of games and strategies:

Definition 2.4.4 ($\mathcal{CLMG}$)**.** The NSC $\mathcal{CLMG} = (\mathcal{CLMG}, \otimes, T, \multimap, T, \&, !, \eta, \rho)$ of ***computational linear McCusker games*** consists of:

- The category $\mathcal{CLMG}$ such that objects are games, morphisms $A \to B$ are the equivalence classes $[\phi]$ of valid strategies $\phi : A \multimap B$ with respect to $\simeq_{A \multimap B}$, the composition $[\psi] \circ [\phi] : A \to C$ of morphisms $[\phi] : A \to B$ and $[\psi] : B \to C$ is $[\psi \circ \phi]$ and the identity $id_A : A \to A$ on each object A is $[cp_A]$;
- The functor $\otimes : \mathcal{CLMG} \times \mathcal{CLMG} \to \mathcal{CLMG}$ consists of tensors on games and strategies;
- T is the terminal game;
- The natural isomorphisms η and ρ are the obvious ones;
- & is product on games, and ! is exponential on games.

Theorem 2.4.5 (The NSC $\mathcal{CLMG}$)**.** *$\mathcal{CLMG}$ forms a NSC.*

Proof. Straightforward; see [101] for the proof outline. □

Corollary 2.4.6 (The CCC $\mathcal{CLMG}_!$)**.** *The co-Kleisli category $\mathcal{CLMG}_!$ forms a CCC.*

Proof. By Theorems 2.4.3 and 2.4.5. □

Definition 2.4.7 ($\mathcal{CMG}$)**.** We define $\mathcal{CMG} \stackrel{\text{df.}}{=} \mathcal{CLMG}_!$ and call it the CCC of ***computational McCusker games***.

Explicitly, $\mathcal{CMG}$ is the CCC such that:

- Objects are games;
- Morphisms $A \to B$ are the equivalence classes $[\phi]$ of valid strategies $\phi : !A \multimap B$ with respect to $\simeq_{!A \multimap B}$;
- The composition $[\psi] \bullet [\phi] : A \to C$ of $[\phi] : A \to B$ and $[\psi] : B \to C$ is $[\psi \bullet \phi] \stackrel{\text{df.}}{=} [\psi \circ \phi^\dagger] : A \to C$;
- The identity $id_A : A \to A$ on each object A is $[der_A]$;
- The terminal object is the terminal game T;
- The unique morphism $A \to T$ is the trivial strategy $\_ \stackrel{\text{df.}}{=} \{\boldsymbol{\epsilon}\}$ for any object A;
- The product of A and B is the product $A\&B$ of games;
- The projections $A \stackrel{\pi_1}{\leftarrow} A\&B \stackrel{\pi_2}{\rightarrow} B$ are $[der_A]$ and $[der_B]$ up to 'tags';
- The pairing $\langle [\alpha], [\beta] \rangle : C \to A\&B$ of $[\alpha] : C \to A$ and $[\beta] : C \to B$ is $[\langle \alpha, \beta \rangle]$;
- The exponential $A \Rightarrow B$ is $!A \multimap B$;
- The evaluation map $ev_{A,B} : (A \Rightarrow B)\&A \to B$ is $[der_{A \Rightarrow B}]$ up to 'tags';
- The currying $\Lambda([\kappa]) : A \to (B \Rightarrow C)$ of $[\kappa] : A\&B \to C$ is $[\kappa]$ up to 'tags'.

By Lemmata 2.3.16, 2.3.22, 2.3.25 and 2.3.27, we may impose both or either of innocence and well-bracketing on morphisms (strictly speaking, on the *representatives* of morphisms) in $\mathcal{CLMG}$ and $\mathcal{CMG}$, prohibiting the behavior of *states* and/or *control operators*. Thus, we have defined eight different categories of games and strategies for computation. Among these categories, the CCC $\mathcal{CMG}_{\mathsf{InnWB}}$ whose morphisms are all innocent and wb is particularly central in the present thesis.

2.4.4 Logical CCCs of Games and Strategies

As explained in Section 2.4.1, we may obtain categories of games and strategies for logic from those for computation by imposing *winning* on morphisms:

Definition 2.4.8 ($\mathcal{LLMG}$)**.** The NSC $\mathcal{LLMG}$ of ***logical linear McCusker games*** is the subNSC of $\mathcal{CLMG}$ whose objects are all well-founded, and morphisms are equivalence classes of winning strategies.

Definition 2.4.9 ($\mathcal{LMG}$)**.** The NSC $\mathcal{LMG}$ of ***logical McCusker games*** is the subNSC of $\mathcal{CMG}$ whose objects are all well-founded, and morphisms are equivalence classes of winning strategies.

We have required that games are all *well-founded* in both $\mathcal{LLMG}$ and $\mathcal{LMG}$ in order to ensure that identities are all *noetherian*; see Lemmata 2.3.12 and 2.3.27.

Remark. Although well-bracketing, totality and noetherianity of strategies on a game G are all preserved under $\simeq_G$, it is not the case for innocence. Thus, we cannot define winning on the equivalence classes of strategies on G.

Corollary 2.4.10 (The NSC $\mathcal{LLMG}$ and the CCC $\mathcal{LMG}$)**.** *$\mathcal{LLMG}$ and $\mathcal{LMG}$ form a NSC and a CCC, respectively.*

Proof. By Theorem 2.4.5, Corollary 2.4.6 and Lemmata 2.3.16, 2.3.22, 2.3.25, 2.3.12 and 2.3.27. □

Note that $\mathcal{LMG} = \mathcal{LLMG}_!$. The explicit definitions of $\mathcal{LLMG}$ and $\mathcal{LMG}$ should be now clear, and thus we do not describe them here.

Before arguing how these categories embody logics (in Section 2.4.6), let us briefly consider the problem of modeling *coproducts* in the next section.

2.4.5 Coproducts

Naturally, one may wonder whether the categories of games and strategies introduced so far have *(strong) coproducts*. However, the answer is *negative*: They have *weak* coproducts but not *strong* ones, as we now explain below.

First of all, there is no initial object in $\mathcal{CLMG}$, $\mathcal{CMG}$, $\mathcal{LLMG}$, $\mathcal{LMG}$ or their subcategories for there is more than one morphism $A \to B$ in any of the categories if so is the codomain B, e.g., take $B \stackrel{\text{df.}}{=} \mathbf{2}$. At best, the empty game $\mathbf{0}$ is a *weak* initial object in each of these categories: There is the trivial (resp. two-move) morphism $\mathbf{0} \to B$ if $B = T$ (resp. if $B \neq T$), but it is in general not unique in the latter case.

Next, as shown in [98], if a CCC $\mathcal{C}$ has both *fixed-points* and *binary coproducts*, then $\mathcal{C}$ must be a trivial category in the sense that each object of $\mathcal{C}$ is isomorphic to a terminal object. This implies that $\mathcal{CMG}$ does not have binary coproducts unless it is trivial since it has fixed-points by *fixed-point strategies* (see Chapter 4).

On the other hand, it seems still possible for the logical categories of games and strategies to have binary coproducts for the winning, more specifically noetherianity, condition on morphisms excludes fixed-point strategies. However, it is unfortunately impossible as argued in [129, 128]; we illustrate this point by showing that our weak sum (Definition 2.2.27) is, as the name suggests, in fact *weak* as follows. First, the copairing $[\iota_l, \iota_l] : T + T \multimap T + T$ of the first injection $\iota_l : T \multimap T + T$ clearly satisfies:

$$\iota_l; [\iota_l, \iota_l] = \iota_l = \iota_r; [\iota_l, \iota_l]. \tag{2.1}$$

However, there is a strategy $\underline{l} \stackrel{\mathrm{df.}}{=} \mathsf{Pref}(\{q_+ l\})^{\mathsf{Even}} : T + T \multimap T + T$ that satisfies:

$$\iota_l; \underline{l} = \iota_l = \iota_r; \underline{l}$$

but $\underline{l} \neq [\iota_l, \iota_l]$. Thus, the copairing $[\iota_l, \iota_l]$ is not a unique morphism that satisfies the equation (2.1), the universal property of (strong) coproduct.

By the same argument, we may show that if there is a non-trivial play of $T \oplus T$ for any variant of sum $\oplus$, then $\oplus$ is weak. Hence, we should define $T \oplus T \stackrel{\mathrm{df.}}{=} T$, but then there cannot be a copairing $[\underline{n_1}, \underline{n_2}] : T \oplus T \multimap N$ of $\underline{n_1}^T, \underline{n_2}^T : T \multimap N$ if $n_1 \neq n_2$.

Also, it should be now clear how to adapt copairings and injections in SMCCs to CCCs; see [128, 129] for the details. Nevertheless, by the same argument, we may see that our sum $+$ is weak in the CCCs setting too.

Therefore, we may conclude that it is impossible to give binary coproducts in any of the categories of games and strategies defined so far. More generally, it has been an open problem in the field of game semantics to give games and strategies that form a category with (strong) coproducts. Note that although the *Fam-construction* in [13] gives a CCC with coproducts or a *bicartesian closed category (BCC)*, its objects and morphisms are *families* of games and families of strategies, respectively i.e., it does not provide constructions on games and strategies that form coproducts.

On the other hand, however, the BCC *Fam*$(\mathcal{G})$ can be seen as equivalent to the subcategory of the underlying CCC $\mathcal{G}$ of conventional games and strategies with weak coproducts such as $\mathcal{CMG}$ and $\mathcal{LMG}$ whose objects are of the form $\sum_{i \in I} A_{i \in I}$ and morphisms are *strict* ones of the form $\phi : \sum_{i \in I} !A_{i \in I} \rightarrow \sum_{j \in J} !B_{j \in J}$, where $(A_i)_{i \in I}$ and $(B_j)_{j \in J}$ are families of objects of $\mathcal{G}$; see [13] for the details. Somewhat similarly to this phenomenon (though independently discovered), we shall obtain a BCC of games and strategies in Chapter 5.

2.4.6 Logic of Games and Strategies

Since a BCC is a categorical model of *intuitionistic propositional logic (IPL)* [118, 102], $\mathcal{LMG}$ models IPL except that it does not identify some proofs equated in syntax.

In fact, $\mathcal{LMG}$ not only embodies the structural and algebraic aspects of IPL but also gives a fundamental underpinning of IPL in a mathematically precise, syntax-independent, conceptually natural manner, which we now explain as follows. First, the implication $A \Rightarrow B$ may be read as a *formula* B under the *premise* A, where the case $A = T$ corresponds to 'no premise'. Thus, we call each object $A \in \mathcal{LMG}$ a ***formula*** and define it to be ***true*** if there is a morphism in $\mathcal{LMG}(A) \overset{\mathrm{df.}}{=} \mathcal{LMG}(T, A)$, called a ***proof*** of A, and ***false*** otherwise. As explained before, a morphism in $\mathcal{LMG}$ can be regarded reasonably as a proof or an 'argument' for the truth of a formula since it is total (i.e., it never gets 'stuck'), innocent (i.e., it is not 'concerned' with passage of time) and noetherian (i.e., it never 'intentionally stalls' an argument).

Following the standard formulation of intuitionistic logic, we define the ***negation*** $\neg A$ of each object $A \in \mathcal{LMG}$ by $\neg A \overset{\mathrm{df.}}{=} A \Rightarrow \mathbf{0}$. It is easy to see that $\neg A$ is true iff A is false. Notice that the logic of $\mathcal{LMG}$ is *intuitionistic*, not classical, since there is no proof of $\neg\neg A \Rightarrow A$ in $\mathcal{LMG}$ for some $A \in \mathcal{LMG}$ (see Section 2.4.7). Moreover, product & and weak sum $+$ respectively capture *conjunction* $\wedge$ and *disjunction* $\vee$ in a conceptually natural manner, e.g., consider $\mathbf{2}\&\mathbf{2}$ and $\mathbf{2} + \mathbf{2}$.

Now, the *consistency* of the logic of $\mathcal{LMG}$ immediately follows from Corollary 2.4.10: If a formula $A \in \mathcal{LMG}$ and its negation $\neg A$ are both true, witnessed by proofs $[\alpha] : T \to A$ and $[\beta] : T \to (A \Rightarrow \mathbf{0})$, then $\mathbf{0}$ would be true, witnessed by the proof $[\beta^{\ominus T}] \circ [\alpha^{\dagger}] : T \to \mathbf{0}$, a contradiction.

Also, *validity* and *provability* of the logic simply *coincide*: A formula has been defined to be *true* iff there is a proof of a formula, i.e., *provable*. Moreover, games and strategies explain not only logical systems (as described above) but also what formal languages refer to, i.e., *models* in a loose sense. Hence, one may say that the CCC $\mathcal{LMG}$ unifies a logical system and a model of IPL.

What we have just explained is a more accurate description of the content of Section 1.7, specialized in IPL. Nevertheless, $\mathcal{LMG}$ cannot formulate *predicates* in logic or *dependent types* in computation. We shall address this point in Chapter 5.

2.4.7 Classical Linear and Classical Logics?

Now, let us emphasize again that the logics of the categories of games and strategies given so far are *intuitionistic*: $\mathcal{LLMG}$ for IMELL and $\mathcal{LMG}$ for IPL. Accordingly, we would like to extend our game-semantic approach to classical linear and classical logics; however, the problem of giving semantics of these logics is highly non-trivial.

For instance, it is well-known that one must go beyond the usual *sequential* games, i.e., games in which Opponent always starts a play, and then the two participants alternately perform moves, to model classical linear logic, e.g., *concurrent games* [15] and *asynchronous games* [131] have been proposed to model classical linear logic.

Also, semantics of classical logic is subtle even in the categorical level: A CCC $\mathcal{C}$ with an initial object 0 such that $A \cong (A \Rightarrow 0) \Rightarrow 0$ for all $A \in \mathcal{C}$ is a preorder category, i.e., any two parallel morphisms in $\mathcal{C}$ are equal [118]. Fortunately, a category of games and strategies usually has only a *weak* initial object to interpret falsity, and thus it is not necessarily a preorder category. In the literature, e.g., there are game models of classical logic [120, 27] based on HO-games [100].

Hence, we do not give a game semantics of classical linear or classical logic in this thesis since it would require another thesis. Nevertheless, we have observed some connections between intuitionistic linear, classical linear, intuitionistic and classical logics by a method similar to the co-Kleisli construction in Section 2.4.2. We just briefly introduce the idea, which will give some useful insights later in the thesis.

The following is the key construction motivated by *why not* ? in linear logic [70]:

Definition 2.4.11 (Why not). Given a game A, the ***why not*** $?A$ of A is defined by:

- $M_{?A} \stackrel{\text{df.}}{=} \{(a,0) \mid a \in M_A^{\mathsf{Init}}\} \cup \{(a,i) \mid a \in M_A \setminus M_A^{\mathsf{Init}}, i \in \mathbb{N}\}$;
- $\lambda_{?A} : (a,i) \mapsto \lambda_A(a)$;
- $\star \vdash_{?A} (a,i) \stackrel{\text{df.}}{\Leftrightarrow} \star \vdash_A a$;
- $(a,i) \vdash_{?A} (a',j)$ $((a,i) \neq \star) \stackrel{\text{df.}}{\Leftrightarrow} (\star \vdash_A a \wedge a \vdash_A a') \vee (\star \not\vdash_A a \wedge i = j \wedge a \vdash_A a')$;
- $P_{?A} \stackrel{\text{df.}}{=} \mathsf{Pref}(\{(a_1,0)\boldsymbol{s_1}(a_2,0)\boldsymbol{s_2}\dots(a_k,0)\boldsymbol{s_k} \in \mathscr{L}_{?A} \mid (\forall i \in \{1,2,\dots,k\}. \star \vdash_A a_i$ $\wedge\, \mathsf{NonInit}_{?A}(\boldsymbol{s_i}) \wedge \forall j \in \mathbb{N}.\, a_1(\boldsymbol{s_1} \upharpoonright j)a_2(\boldsymbol{s_2} \upharpoonright j)\dots a_k(\boldsymbol{s_k} \upharpoonright j) \in P_A\})$;
- $\simeq_{?A} \stackrel{\text{df.}}{=} \{((a_1,0)\boldsymbol{s_1}(a_2,0)\boldsymbol{s_2}\dots(a_k,0)\boldsymbol{s_k}, (a'_1,0)\boldsymbol{t_1}(a'_2,0)\boldsymbol{t_2}\dots(a'_k,0)\boldsymbol{t_k}) \mid \exists \varphi \in \mathcal{P}(\mathbb{N}).$ $\forall j \in \mathbb{N}.\, a_1(\boldsymbol{s_1} \upharpoonright \varphi(j))a_2(\boldsymbol{s_2} \upharpoonright \varphi(j))\dots a_k(\boldsymbol{s_k} \upharpoonright \varphi(j)) \simeq_A a'_1(\boldsymbol{t_1} \upharpoonright j)a'_2(\boldsymbol{t_2} \upharpoonright j)\dots$ $a'_k(\boldsymbol{t_k} \upharpoonright j) \wedge \forall i \in \{1,2,\dots,k\}.\, \pi_2^*(\boldsymbol{s_i}) = (\varphi \circ \pi_2)^*(\boldsymbol{t_i})\}$, where recall that $\mathcal{P}(\mathbb{N})$ is the set of all permutations of natural numbers.

Lemma 2.4.12 (Well-defined why not). *Games are closed under why not ?.*

Proof. Straightforward. □

Intuitively, a why not $?A$ is essentially the game A except that Player may restart a play of A from the second occurrence as many times as she likes. In other words, $?A$ is a 'Player-friendly' version of A for Player may *backtrack* any number of times.

In the literature, there have been game-semantic approaches to classical logic by allowing such backtracking, e.g., [40, 21]. In the same spirit, we now propose:

- (Intuitionistic Linear Implication from A to B) $A \multimap B$;
- (Classical Linear Implication from A to B) $?A \multimap ?B$;
- (Intuitionistic Implication from A to B) $!A \multimap B$;
- (Classical Implication from A to B) $!?A \multimap ?B$.

Accordingly, we define a formula (i.e., a game) A is:

- ***intuitionistically true*** if there is a winning strategy on $T \multimap A$;
- ***classically true*** if there is a winning strategy on $T \multimap ?A$

where the linear/non-linear distinction does not matter for $!?T = T = ?T$.

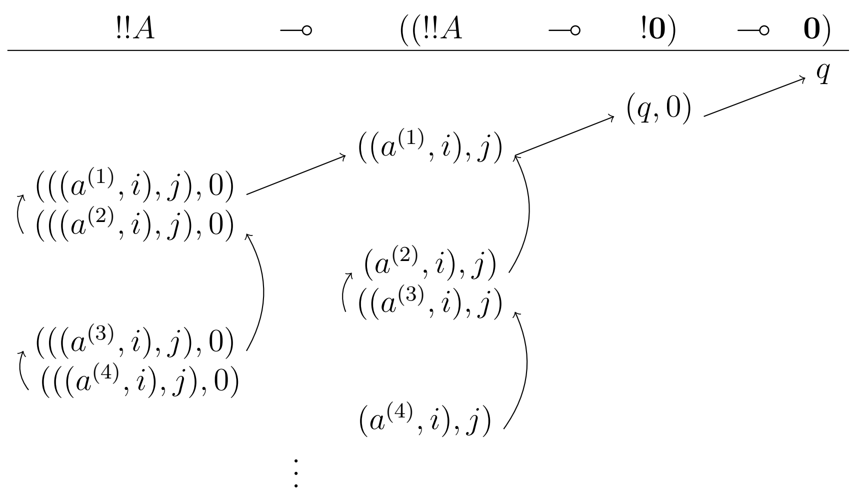
The intuitionistic linear and the intuitionistic implications are from linear logic. Let us explain the two new implications: classical linear and classical ones. A classical linear implication $?A \multimap ?B$ is *linear* for it allows Player to refer to the domain $?A$ just once (again, strictly speaking, *at most once*), and it is *classical* for the domain $?A$ and the codomain $?B$ allow backtracking. We then obtain the classical one $!?A \multimap ?B$ from $?A \multimap ?B$ by applying Girard's translation, relaxing the resource sensitivity.

Nevertheless, we leave it as future work to develop this naive idea further to give semantics of classical and classical linear logics in both game-semantic and categorical levels. For now, we have just observed:

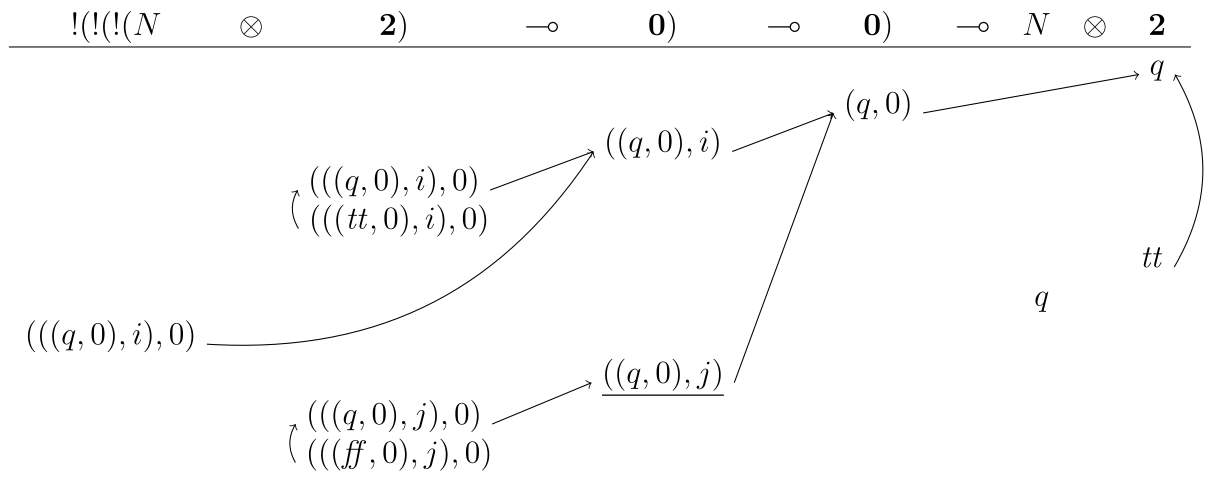
Lemma 2.4.13 (Why not monad)**.** *Why not ? is a monad on $\mathcal{CLMG}$ and on $\mathcal{LLMG}$ such that $?(A \multimap B) = !A \multimap ?B$ and $!?A \cong ?!A$ for any games A and B.*

Proof. Very similar to the proof that exponential ! forms a comonad [101]. □

As expected, an intuitionistic proof of a formula A is a restricted type of a classical proof of A, namely it never backtracks. To see its reasonability, let us consider some examples. First, consider the *double negation elimination (DNE)* $\neg\neg A \Rightarrow A$ for any given formula A. Note that the converse $A \Rightarrow \neg\neg A$ is intuitionistically true since there is the dereliction preceded by the two canonical moves:



However, as expected, there is no intuitionistic proof of DNE. Note that there is only one possible strategy on $\neg\neg A \Rightarrow A$ for any given formula A, namely the dereliction on A intervened by the two canonical moves. However, it is *not* total; for instance, consider the following play of the tensor $A \stackrel{\text{df.}}{=} N \otimes \mathbf{2}$:


$!(!(!(N \otimes \mathbf{2}) \multimap \mathbf{0}) \multimap \mathbf{0}) \multimap N \otimes \mathbf{2}$
q
$(q, 0)$
$((q, 0), i)$
$(((q, 0), i), 0)$
$(((tt, 0), i), 0)$
tt
q
$(((q, 0), i), 0)$
$\underline{((q, 0), j)}$
$(((q, 0), j), 0)$
$(((ff, 0), j), 0)$

where $i \neq j$. The point is that when Opponent performs the move $((q, 0), j)$ with underline just for clarity, initiating a new thread, the only conceivable general strategy is to 'copy-cat' the first move q as the next move; however, Opponent can perform the next move $(((ff, 0), j), 0)$, which Player cannot 'copy-cat' to the codomain $N \otimes \mathbf{2}$.

On the other hand, there *is* a classical proof of DNE because multiple threads initiated by Opponent as in the above example may be 'copy-catted' to the codomain:

$$!?(!?(!?A \quad \multimap \quad ?\mathbf{0}) \quad \multimap \quad ?\mathbf{0}) \quad \multimap \quad ?A$$

$$\begin{tikzcd}[row sep=small]
 & & & a_0^{(1)} \\
 & & q_{0,0,0} \arrow[ur] & \\
 & q_{0,0,i,j,0} \arrow[ur] & & \\
a^{(1)}_{0,0,0,i,j,0} \arrow[ur] & & & \\
a^{(2)}_{k,0,0,i,j,0} \arrow[u, bend left] & & & a_k^{(2)} \arrow[uuuu, bend right] \\
 & & & a_k^{(3)} \arrow[u, bend left] \\
a^{(3)}_{k,0,0,i,j,0} \arrow[uu, bend right] & & & \\
 & \underline{q_{0,0,l,r,0}} \arrow[uuuuuur] & & \\
a^{(1)}_{0,0,0,l,r,0} \arrow[ur] & & & \\
\tilde{a}^{(2)}_{p,0,0,l,r,0} \arrow[u, bend left] & & & \tilde{a}_p^{(2)} \arrow[uuuuuuuuu, bend right] \\
 & & & \tilde{a}_p^{(3)} \arrow[u, bend left] \\
\tilde{a}^{(3)}_{p,0,0,l,r,0} \arrow[uu, bend right] & & & \\
 & \underline{q_{0,0,x,y,0}} \arrow[uuuuuuuuuuur] & & \\
a^{(1)}_{0,0,0,x,y,0} \arrow[ur] & & & \\
\hat{a}^{(2)}_{z,0,0,x,y,0} \arrow[u, bend left] & & & \hat{a}_z^{(2)} \arrow[uuuuuuuuuuuuuu, bend right] \\
 & & \vdots &
\end{tikzcd}$$

where the pairs (i, j), (l, r) and (x, y) are pairwise distinct, and we abbreviate a finite sequence $(\dots((a, x_1), x_2), \dots, x_n)$ as $a_{x_1,x_2,\dots,x_n}$. Similarly, the *law of excluded middle (LEM)* is not intuitionistically but classically true.

We leave a thorough development of these ideas as future work; however, they give useful insights, e.g., why Σ-types is incompatible with classical reasoning in Chapter 6.

Chapter 3

Dynamic Game Semantics

3.1 Introduction to the Chapter

The main contents of the thesis begin with the present chapter, which aims to give semantics of *dynamics* and *intensionality* of computation as explained in Section 1.1, specifically by games and strategies.

3.1.1 Existing Game Semantics Is Static

Game semantics is often said to be a *dynamic*, *intensional* semantics for a category of games is usually not well-pointed, and plays of a game may be regarded as 'dynamic interactions' between the participants of the game [5, 101]. However, it has been employed as *denotational semantics* (see Section 1.1.3), and thus it has been *sound*: If two programs evaluate to the same value, then their denotations are identical. As a consequence, existing game semantics $[\![\_]\!]_{\mathcal{G}}$ is actually *static* and *extensional* in the sense that if there is a reduction $\mathsf{M}_1 \to \mathsf{M}_2$ in syntax, then the equation $[\![\mathsf{M}_1]\!]_{\mathcal{G}} = [\![\mathsf{M}_2]\!]_{\mathcal{G}}$ holds in the semantics (i.e., it does not capture the dynamics $\mathsf{M}_1 \to \mathsf{M}_2$ or the intensional difference between M_1 and M_2). In other words, it is *not* dynamic or intensional in the sense that it cannot satisfy a *DCP* (see Section 1.1.4).

Therefore, for achieving the research aim described in Section 1.1.4, we need to introduce a more dynamic, intensional variant of games and strategies so that they satisfy DCPs for logical systems and programming languages. To get some insights to develop such games and strategies, let us see how existing game semantics fails to be dynamic or intensional. The point in a word is that the 'internal communication' between strategies for their composition is *a priori* 'hidden', and thus, the resulting strategy is always in *normal form*. For instance, as seen in Chapter 2, the composition $succ^{\dagger}; double : N \Rightarrow N$ of the two strategies $succ : N \Rightarrow N$ and $double : N \Rightarrow N$

$$\begin{tikzcd}
!N_{[0]} & \overset{succ}{\multimap} & N_{[1]} \\
 & & q_{[1]} \\
(q,0)_{[0]} \arrow[urr] & & \\
(m,0)_{[0]} \arrow[u, bend left] & & \\
 & & m+1_{[1]} \arrow[uuu, bend right]
\end{tikzcd}
\qquad
\begin{tikzcd}
!N_{[2]} & \overset{double}{\multimap} & N_{[3]} \\
 & & q_{[3]} \\
(q,0)_{[2]} \arrow[urr] & & \\
(n,0)_{[2]} \arrow[u, bend left] & & \\
 & & 2 \cdot n_{[3]} \arrow[uuu, bend right]
\end{tikzcd}$$

is calculated as follows. First, by 'internal communication', we mean that Player plays the role of Opponent in the intermediate component games $!N_{[1]}$ and $!N_{[2]}$ just by 'copy-catting' her last moves, resulting in the following play:

$$\begin{tikzcd}
!N_{[0]} & \overset{succ^{\dagger}}{\multimap} & !N_{[1]} & !N_{[2]} & \overset{double}{\multimap} & N_{[3]} \\
 & & & & & q_{[3]} \\
 & & & \boxed{(q,0)_{[2]}} \arrow[urr] & & \\
 & & \boxed{(q,0)_{[1]}} \arrow[ur] & & & \\
(q,\langle 0,0\rangle)_{[0]} \arrow[urr] & & & & & \\
(n,\langle 0,0\rangle)_{[0]} \arrow[u, bend left] & & & & & \\
 & & \boxed{(n+1,0)_{[1]}} \arrow[uuu, bend left] & & & \\
 & & & \boxed{(n+1,0)_{[2]}} \arrow[uuuuu, bend left] & & \\
 & & & & & 2 \cdot (n+1)_{[3]} \arrow[uuuuuuu, bend right]
\end{tikzcd}$$

where moves for 'internal communication' are marked by square boxes just for clarity, and a pointer from $(q,0)_{[1]}$ to $(q,0)_{[2]}$ is added because the move $(q,0)_{[1]}$ is no longer initial. Importantly, it is assumed that Opponent plays on the game $!N_{[0]} \multimap N_{[3]}$, 'seeing' only moves of $!N_{[0]}$ or $N_{[3]}$. The resulting play is to be read as follows:

1. Opponent's question $q_{[3]}$ for an output in $!N_{[2]} \multimap N_{[3]}$ ('What is your output?');

2. Player's question $\boxed{(q,0)_{[2]}}$ by *double* for an input in $!N_{[2]} \multimap N_{[3]}$ ('Wait, what is an input?');

3. $\boxed{(q,0)_{[2]}}$ in turn triggers the question $\boxed{(q,0)_{[1]}}$ for an output in $!N_{[0]} \multimap !N_{[1]}$ ('What is an output?');

4. Player's question $(q,\langle 0,0\rangle)_{[0]}$ by $succ^{\dagger}$ for an input in $!N_{[0]} \multimap !N_{[1]}$ ('Wait, what is an input?');

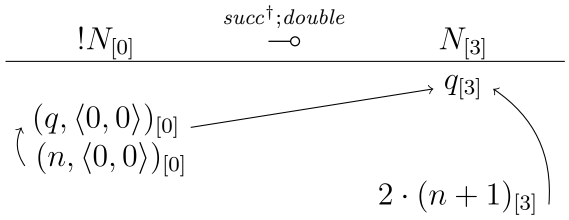
5. Opponent's answer, say, $(n,\langle 0,0\rangle)_{[1]}$, to the question $(q,\langle 0,0\rangle)_{[0]}$ in $!N_{[0]} \multimap !N_{[3]}$ ('Here is an input n.');

6. Player's answer $\boxed{(n+1,0)_{[1]}}$ to the question $\boxed{(q,0)_{[1]}}$ by $succ^{\dagger}$ in $!N_{[0]} \multimap !N_{[1]}$ ('The output is then $n+1$.');

7. $\boxed{(n+1,0)_{[1]}}$ in turn triggers the answer $\boxed{(n+1,0)_{[2]}}$ to the question $\boxed{(q,0)_{[2]}}$ in $!N_{[2]} \multimap N_{[3]}$ ('Here is the input $n+1$.');

8. Player's answer $2 \cdot (n+1)_{[3]}$ to the initial question $q_{[3]}$ by *double* in $!N_{[2]} \multimap N_{[3]}$ ('The output is then $2 \cdot (n+1)$!').

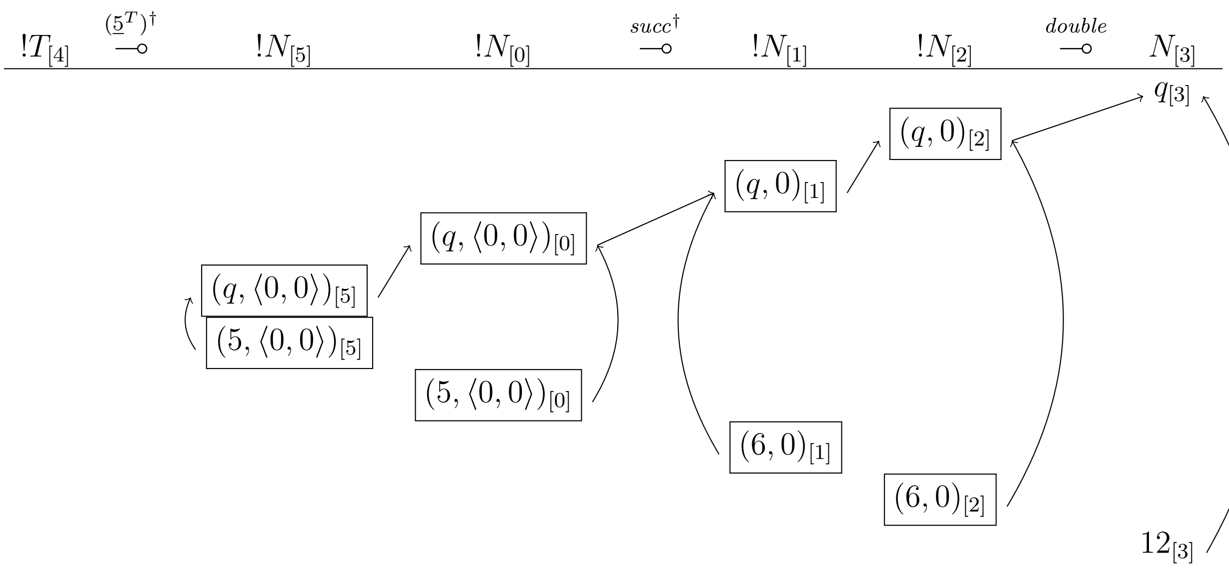
Next, 'hiding' means to hide or delete every move with a square box from the play, resulting in the strategy for the function $n \mapsto 2 \cdot (n+1)$ as expected:



$!N_{[0]} \overset{succ^{\dagger};double}{\multimap} N_{[3]}$
$q_{[3]}$
$(q,\langle 0,0\rangle)_{[0]}$
$(n,\langle 0,0\rangle)_{[0]}$
$2\cdot(n+1)_{[3]}$

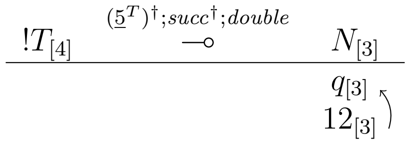
Note that it is 'hiding' that makes the resulting play a valid one on the game $N \Rightarrow N$.

Now, let us plug in the strategy $\underline{5}^T : q \mapsto 5$ on the game $T \Rightarrow N$ (n.b., recall that $T \Rightarrow N$ coincides with N up to 'tags'). The composition $(\underline{5}^T)^{\dagger}; succ^{\dagger}; double : T \Rightarrow N$ of $\underline{5}^T$, *succ* and *double*[1] is computed again by 'internal communication':



$!T_{[4]} \overset{(\underline{5}^T)^{\dagger}}{\multimap} !N_{[5]}$
$!N_{[0]} \overset{succ^{\dagger}}{\multimap} !N_{[1]}$
$!N_{[2]} \overset{double}{\multimap} N_{[3]}$
$q_{[3]}$
$\boxed{(q,0)_{[2]}}$
$\boxed{(q,0)_{[1]}}$
$\boxed{(q,\langle 0,0\rangle)_{[0]}}$
$\boxed{(q,\langle 0,0\rangle)_{[5]}}$
$\boxed{(5,\langle 0,0\rangle)_{[5]}}$
$\boxed{(5,\langle 0,0\rangle)_{[0]}}$
$\boxed{(6,0)_{[1]}}$
$\boxed{(6,0)_{[2]}}$
$12_{[3]}$

plus 'hiding':



$!T_{[4]} \overset{(\underline{5}^T)^{\dagger};succ^{\dagger};double}{\multimap} N_{[3]}$
$q_{[3]}$
$12_{[3]}$

[1]Composition of strategies is *associative*; thus, the order of applying composition does not matter.

In syntax, on the other hand, assuming that there are a (ground) type ι of natural numbers, a numeral $\underline{\mathsf{n}}$ of type ι for each $n \in \mathbb{N}$, and constants succ and double of type ι for the successor and the doubling functions, respectively, equipped with the operational semantics $\mathsf{succ}\,\underline{\mathsf{n}} \to \underline{\mathsf{n+1}}$ and $\mathsf{double}\,\underline{\mathsf{n}} \to \underline{\mathsf{2 \cdot n}}$ for all $n \in \mathbb{N}$ in an arbitrary functional programming language, the program $\mathsf{p}_1 \overset{\mathrm{df.}}{\equiv} \lambda \mathsf{x}.(\lambda \mathsf{y}.\,\mathsf{double}\,\mathsf{y})((\lambda \mathsf{z}.\,\mathsf{succ}\,\mathsf{z})\,\mathsf{x})$ represents the syntactic composition $\mathsf{succ}; \mathsf{double}$. When it is applied to the numeral $\underline{5}$, we have the following chain of reductions:

$$\begin{aligned} \mathsf{p}_1\,\underline{5} &\to^* (\lambda \mathsf{x}.\,\mathsf{double}\,(\mathsf{succ}\,\mathsf{x}))\,\underline{5} \\ &\to^* \mathsf{double}\,(\mathsf{succ}\,\underline{5}) \\ &\to^* \mathsf{double}\,\underline{6} \\ &\to^* \underline{12}. \end{aligned}$$

Therefore, it seems that reduction in syntax corresponds in game semantics to 'hiding internal communication'. As seen in the above example, however, this game-semantic normalization is *a priori* executed and thus invisible in conventional game semantics $[\![\_]\!]_{\mathcal{G}}$. As a result, the two programs $\mathsf{p}_1\,\underline{5}$ and $\underline{12}$ are interpreted by $[\![\_]\!]_{\mathcal{G}}$ as the same strategy. Moreover, observe that moves with a square box describe *intensionality* or *step-by-step processes* to compute an output from an input, but they are invisible after 'hiding'. Thus, e.g., a program $\mathsf{p}_2 \overset{\mathrm{df.}}{\equiv} \lambda \mathsf{x}.(\lambda \mathsf{y}.\,\mathsf{succ}\,\mathsf{y})(\lambda \mathsf{v}.(\lambda \mathsf{z}.\,\mathsf{succ}\,\mathsf{z})((\lambda \mathsf{w}.\,\mathsf{double}\,\mathsf{w})\,\mathsf{v})\,\mathsf{x})$, representing the same function as p_1 yet a different algorithm $\mathsf{double}; \mathsf{succ}; \mathsf{succ}$ is modeled as:

$$[\![\mathsf{p}_2]\!]_{\mathcal{G}} = [\![\mathsf{double}; \mathsf{succ}; \mathsf{succ}]\!]_{\mathcal{G}} = [\![\mathsf{succ}; \mathsf{double}]\!]_{\mathcal{G}} = [\![\mathsf{p}_1]\!]_{\mathcal{G}}.$$

To sum up, we have observed the following:

1. (Reduction as hiding). Reduction in syntax corresponds in game semantics to 'hiding intermediate moves (i.e., moves with a square box)';
2. (A priori normalization). However, the 'hiding' process is *a priori* executed in conventional game semantics, and thus strategies are always in 'normal form';
3. (Intermediate moves as intensionality). 'Intermediate moves' constitute *intensionality* of computation, but they are not captured in conventional game semantics again due to the a priori execution of the 'hiding' operation.

3.1.2 Dynamic Games and Strategies

From these observations, we have obtained a promising solution: to define a variant of games and strategies, in which 'intermediate moves' are not a priori 'hidden', representing intensionality of computation, and the *hiding operations* $\mathcal{H}$ on the games and strategies 'hide intermediate moves' in a step-by-step fashion, interpreting dynamics of computation. Let us call such a variant of games (resp. strategies) ***dynamic games*** (resp. ***dynamic strategies***).

In doing so, we have tried to develop new structures and axioms that are *conceptually natural* and *mathematically elegant*. This is in order to inherit the natural, intuitive nature of existing game semantics so that the resulting interpretation would be insightful, convincing and useful. Also, mathematics often leads to a 'correct' formulation: If a definition gives rise to neat mathematical structures, then it is likely to succeed in capturing the essence of concepts and phenomena of concern, and subsume various instances. In fact, dynamic games and strategies are a natural generalization of existing games and strategies, and they satisfy beautiful algebraic laws: They form a *cartesian closed bicategory (CCB)* in the sense of [145][2] $\mathcal{LDG}$ (Definition 3.4.1), in which 0- (resp. 1-) cells are certain dynamic games (resp. dynamic strategies), and 2-cells correspond to the *extensional equivalence* between 1-cells; and the hiding operation $\mathcal{H}$ induces the functor $\mathcal{H}^\omega : \mathcal{LDG} \to \mathcal{CMG}$, where the CCC $\mathcal{CMG}$ (Definition 2.4.7) can be seen as an 'extensionally collapsed' $\mathcal{LDG}$.

3.1.3 Dynamic Game Semantics

We shall then give a game semantics $[\![\_]\!]_{\mathcal{DG}}$ of a logical calculus or a programming language in $\mathcal{LDG}$ that together with the hiding operation $\mathcal{H}$ satisfies a DCP, which we call ***dynamic game semantics*** as it captures dynamics and intensionality of computation better than existing ones. Since a simple language would be appropriate for the first work on dynamic game semantics, let us select *finitary PCF* (i.e., the simply-typed λ-calculus equipped with the boolean type) as the target language.

Note that it does not make much sense to ask whether *full abstraction* [44] holds for dynamic game semantics as its aim is to capture intensionality of computation.

Also, the model does not satisfy *faithfulness*: Our semantic equation is of course *finer than β-equivalence* but also *coarser than α-equivalence*, e.g., non-α-equivalent terms $(\lambda \mathsf{x}.\, \underline{0})\, \underline{1}$ and $(\lambda \mathsf{x}.\, \underline{0})\, \underline{2}$ are interpreted to be the same in our dynamic game

[2]N.b., for the present chapter, it suffices to know that a CCB is a generalized CCC in the sense that the equational axioms of CCCs are required to hold only *up to 2-cell isomorphisms*.

semantics. It is because, as explained in Section 1.1.2, the semantic equation captures *algorithmic difference* of programs, while α-equivalence distinguishes how programs are constructed even if their algorithms are the same.

On the other hand, it makes sense to ask whether *full completeness* [7, 44] holds for dynamic game semantics as well. In fact, we shall establish it in the present work.

3.1.4 Related Work and Contribution of the Chapter

To the best of our knowledge, the present work is the first syntax-independent characterization of dynamics of computation in the sense that it satisfies DCPs.

The work closest in spirit is Girard's *geometry of interaction (GoI)* [71, 72, 73, 75, 76, 77]. However, GoI appears mathematically *ad-hoc* for it does not conform to the standard categorical semantics of type theories [118, 150, 41, 102]; also, it does not capture the *step-by-step* process of reduction in the sense of DCPs. In contrast, dynamic game semantics refines the standard semantics and satisfies DCPs.

Next, the idea of exhibiting 'intermediate moves' in the composition of strategies is *nothing new*; there are game-semantic approaches [52, 81, 29, 144] that give such moves an official status. However, because their aims are rather to develop a tool for program analysis and verification, they do not study in depth mathematical structures thereof, give an intensional game semantics that follows the standard categorical semantics of type theories or formulate a *step-by-step* 'hiding' process. Therefore, our contribution for this point is to study algebraic structures of games and strategies when we do not a priori 'hide intermediate moves' and refine the standard categorical semantics of computation in such a way that satisfies DCPs.

Also, there are several different approaches to model dynamics of computation by *2-categories* [166, 89, 132]. In these papers, however, the horizontal composition of 1-cells is the *normalizing* one, which is why the structure is 2-categories rather than bicategories.[3] Also, their 2-cells are rewriting, while the 2-cells of our bicategory are external equivalences between 1-cells; note that 2-cells in a bicategory cannot interpret rewriting unless the horizontal composition is normalizing since associativity of non-normalizing composition with respect to such 2-cells does not hold.[4] Thus, although their motivations are similar to ours, our *bicategorical* approach seems novel, interpreting an application of terms by a non-normalizing composition, extensional equivalences of terms by 2-cells and rewriting by the hiding operation $\mathcal{H}$ on 1-cells.

[3]I.e., the unit law does not strictly hold if the composition is non-normalizing.

[4]I.e., there is no 'rewriting' between 1-cells $(f; g); h$ and $f; (g; h)$ if the composition is non-normalizing.

Moreover, their frameworks are categorical, while we have instantiated our model by game semantics. Furthermore, neither of the previous work establishes a DCP.

Finally, note that the present work has several implications from theoretical as well as practical viewpoints. From the theoretical perspective, it enables us to study dynamics and intensionality of computation as purely *mathematical* (or *semantic*) concepts, just like any concepts in pure mathematics such as differentiation and integration in calculus, homotopy in topology, etc. Thus, we may rigorously analyze the essence of these concepts ignoring superfluous syntactic details. From the practical point of view, on the other hand, it can be a useful tool for language analysis and design, e.g., our variant of finitary PCF would not exist without the present work.

3.1.5 Chapter Outline

The rest of the chapter proceeds as follows. First, Section 3.2 formulates our target programming language and its bicategorical semantics that satisfies a DCP so that it remains to establish its game-semantic instance. Next, Section 3.3 introduces dynamic games and strategies, and Section 3.4 gives dynamic game semantics of the language. Finally, Section 3.5 draws a conclusion and proposes future work.

3.2 Dynamic Bicategorical Semantics

This section gives a *categorical* description of how dynamic games and strategies capture dynamics and intensionality of computation, respectively, and shows that it is a *refinement* of the standard categorical semantics of type theories [118, 150, 41, 102].

3.2.1 Beta-Categories of Computation

The categorical structure that is essential for our interpretation is *β-categories of computation (BoCs)*, a certain kind of bicategories whose 2-cells are *extensional equivalences* between 1-cells, equipped with an *evaluation* satisfying certain axioms.

Let us first introduce a more general notion of *β-categories*, which are categories *up to an equivalence relation on morphisms*:

Definition 3.2.1 (β-categories)**.** A ***β-category*** is a pair $\mathcal{C} = (\mathcal{C}, \simeq)$ that consists of:

- A class $\mathsf{ob}(\mathcal{C})$ of ***objects***, where we usually write $A \in \mathcal{C}$ for $A \in \mathsf{ob}(\mathcal{C})$;
- A class $\mathcal{C}(A, B)$ of ***β-morphisms*** from A to B for each pair $A, B \in \mathcal{C}$, where we often write $f : A \to B$ for $f \in \mathcal{C}(A, B)$ if $\mathcal{C}$ is obvious from the context;

- A (class) function $\mathcal{C}(A, B) \times \mathcal{C}(B, C) \overset{;_{A,B,C}}{\to} \mathcal{C}(A, C)$, called the ***β-composition*** on β-morphisms from A to B and from B to C, for each triple $A, B, C \in \mathcal{C}$;
- A β-morphism $id_A \in \mathcal{C}(A, A)$, called the ***β-identity*** on A, for each $A \in \mathcal{C}$;
- An equivalence (class) relation $\simeq_{A,B}$ on $\mathcal{C}(A, B)$, called the ***equivalence*** on β-morphisms from A to B, for each pair $A, B \in \mathcal{C}$

where we also write $\mathcal{C}(B, C) \times \mathcal{C}(A, B) \overset{\circ_{A,B,C}}{\to} \mathcal{C}(A, C)$ for the β-composition $;_{A,B,C}$ and often omit the subscripts on $;_{A,B,C}$, $\circ_{A,B,C}$ and $\simeq_{A,B}$, such that it satisfies:

$$\begin{gathered} f \simeq f' \wedge g \simeq g' \Rightarrow f; g \simeq f'; g' \\ (f; g); h \simeq f; (g; h) \\ f; id_B \simeq f \\ id_A; f \simeq f \end{gathered}$$

for any $A, B, C, D \in \mathcal{C}$, $f, f' : A \to B$, $g, g' : B \to C$ and $h : C \to D$. Moreover, it is ***cartesian closed*** iff:

- There is an object $T \in \mathcal{C}$, called a ***β-terminal object***, such that there is a β-morphism $!_A : A \to T$ for each $A \in \mathcal{C}$ that satisfies:

$$!_A \simeq t \text{ for any } t : A \to T;$$

- There is an object $A \times B \in \mathcal{C}$ for each pair $A, B \in \mathcal{C}$, called a ***β-(binary) product*** of A and B, equipped with β-morphisms $\pi_1^{A,B} : A \times B \to A$ and $\pi_2^{A,B} : A \times B \to B$, called the first and the second ***β-projections*** of $A \times B$, respectively, such that given $C \in \mathcal{C}$, $a : C \to A$ and $b : C \to B$ there is a β-morphism $\langle a, b \rangle_{A,B}^C : C \to A \times B$, called the ***β-pairing*** of a and b (with respect to $A \times B$), that satisfies:

$$\begin{gathered} a \simeq a' \wedge b \simeq b' \Rightarrow \langle a, b \rangle \simeq \langle a', b' \rangle \text{ for any } a' : C \to A \text{ and } b' : C \to B \\ \langle a, b \rangle_{A,B}^C; \pi_1^{A,B} \simeq a \\ \langle a, b \rangle_{A,B}^C; \pi_2^{A,B} \simeq b \\ \langle h; \pi_1^{A,B}, h; \pi_2^{A,B} \rangle_{A,B}^C \simeq h \text{ for any } h : C \to A \times B; \end{gathered}$$

- There are an object $C^B \in \mathcal{C}$ and a β-morphism $ev_{B,C} : C^B \times B \to C$, called the ***β-exponential*** and the ***β-evaluation*** of B and C, respectively, for each pair $B, C \in \mathcal{C}$ such that given $A \in \mathcal{C}$ and $k : A \times B \to C$ there is a β-morphism

$\Lambda_{A,B,C}(k) : A \to C^B$ (or written $\Lambda_A(k) : A \to C^B$), called the ***β-currying*** of k (with respect to C^B), that satisfies:

$$k \simeq k' \Rightarrow \Lambda_{A,B,C}(k) \simeq \Lambda_{A,B,C}(k') \text{ for any } k' : A \times B \to C$$
$$\langle \pi_1^{A,B} ; \Lambda_{A,B,C}(k), \pi_2^{A,B} \rangle_{C^B,B}^{A \times B} ; ev_{B,C} \simeq k$$
$$\Lambda_{A,B,C}(\langle \pi_1^{A,B} ; l, \pi_2^{A,B} \rangle_{C^B,B}^{A \times B} ; ev_{B,C}) \simeq l \text{ for any } l : A \to C^B$$

where we often omit the sub/superscripts on $\pi_i^{A,B}$, $\langle \_, \_ \rangle_{A,B}^C$, $ev_{B,C}$ and $\Lambda_{A,B,C}$.

That is, a (resp. cartesian closed) β-category $\mathcal{C} = (\mathcal{C}, \simeq)$ is a (resp. cartesian closed) category *up to* $\simeq$ (i.e., the equation $=$ on morphisms is replaced with $\simeq$), where the prefix 'β-' represents the compromise 'up to $\simeq$'. Alternatively, regarding objects and β-morphisms of $\mathcal{C}$ as 0-cells and 1-cells, respectively, and defining 2-cells by $\mathcal{C}(A,B)(d,c) \stackrel{\text{df.}}{=} \begin{cases} \{\simeq\} & \text{if } d \simeq c; \\ \emptyset & \text{otherwise} \end{cases}$ for any $A, B \in \mathcal{C}$ and $d, c : A \to B$, where $\{\simeq\}$ is any singleton set, we may identify $\mathcal{C}$ with a (resp. cartesian closed [145]) *bicategory* whose 2-cells are only the trivial one.

We are now ready to define *β-categories of computation (BoCs)*:

Definition 3.2.2 (BoCs)**.** A ***β-category of computation (BoC)*** is a β-category $\mathcal{C} = (\mathcal{C}, \simeq)$ equipped with a (class) function $\mathcal{E}$ on β-morphisms of $\mathcal{C}$, called the ***evaluation*** (of computation), that satisfies:

- (Subject Reduction) $\mathcal{E}(f) : A \to B$ and $f \downarrow$ for all $A, B \in \mathcal{C}$ and $f : A \to B$;
- (β-Identities) $\mathcal{E}(id_A) = id_A$ for all $A \in \mathcal{C}$;
- (Evaluation) $f \simeq f' \Leftrightarrow \exists v \in \mathcal{V}_{\mathcal{C}}(A,B).\, f \downarrow v \wedge f' \downarrow v$ for all $A, B \in \mathcal{C}$ and $f, f' : A \to B$

where $\mathcal{V}_{\mathcal{C}}(A,B) \stackrel{\text{df.}}{=} \{ v \in \mathcal{C}(A,B) \mid \mathcal{E}(v) = v \}$, whose elements are called ***values*** from A to B, and we write $f \downarrow$, or specifically $f \downarrow \mathcal{E}^n(f)$, if $\mathcal{E}^n(f) \in \mathcal{V}_{\mathcal{C}}(A,B)$ for some $n \in \mathbb{N}$.[5] It is ***cartesian closed***, which we call a ***cartesian closed BoC (CCBoC)***, iff so is $\mathcal{C}$ as a β-category, all the β-projections and the β-evaluations of $\mathcal{C}$ are values, and all the β-pairing and the β-currying of $\mathcal{C}$ preserve values.

Convention. Since the equivalence $\simeq$ of a BoC $\mathcal{C}$ may be completely recovered from the evaluation $\mathcal{E}$, we usually specify the BoC by a pair $\mathcal{C} = (\mathcal{C}, \mathcal{E})$. If $f \downarrow \mathcal{E}^n(f)$ for some $n \in \mathbb{N}$, then we call $\mathcal{E}^n(f)$ the ***value*** of f and also write $\mathcal{E}^\omega(f)$ for it.

[5]Note that if $\mathcal{E}^{n_1}(f), \mathcal{E}^{n_2}(f) \in \mathcal{V}_{\mathcal{C}}(A,B)$ for any $n_1, n_2 \in \mathbb{N}$, then clearly $\mathcal{E}^{n_1}(f) = \mathcal{E}^{n_2}(f)$, where $\mathcal{E}^n$ denotes the n-times iteration of $\mathcal{E}$ for all $n \in \mathbb{N}$.

Intuition behind Definition 3.2.2 is as follows. In a BoC $\mathcal{C} = (\mathcal{C}, \mathcal{E})$, β-morphisms are (possibly *intensional* but not necessarily 'effective') *computations* with the domain and the codomain (objects) specified, and values are *extensional* computations such as functions (as graphs). The β-composition is 'non-normalizing composition' or *concatenation* of computations, and β-identities are *unit computations* (they are just like identity functions). The *execution* of a computation f is achieved by *evaluating* it into a unique value $\mathcal{E}^{\omega}(f)$, which corresponds to *dynamics* of computation.[6] Also, the equivalence $\simeq$ witnesses *extensional equivalences* between β-morphisms. The three axioms then should make sense from this perspective. In this way, a BoC provides a 'universe' of dynamic, intensional computations.

It is easy to see that a BoC $\mathcal{C} = (\mathcal{C}, \mathcal{E})$ induces the category $\mathcal{V}_{\mathcal{C}}$ given by:

- Objects are those of $\mathcal{C}$;
- Morphisms $A \to B$ are values in $\mathcal{V}_{\mathcal{C}}(A, B)$;
- The composition of morphisms $u : A \to B$ and $v : B \to C$ is $\mathcal{E}^{\omega}(u; v) : A \to C$;
- Identities are β-identities in $\mathcal{C}$.

Regarding the BoC $\mathcal{C}$ as the trivial bicategory as already specified above, and the category $\mathcal{V}_{\mathcal{C}}$ as the trivial 2-category, the evaluation $\mathcal{E}$ induces the 2-functor $\mathcal{E}^{\omega} : \mathcal{C} \to \mathcal{V}_{\mathcal{C}}$ that maps $A \mapsto A$ for 0-cells A, $f \mapsto \mathcal{E}^{\omega}(f)$ for 1-cells f and $\simeq \mapsto =$ for 2-cells $\simeq$. Then clearly, $\mathcal{V}_{\mathcal{C}}$ is cartesian closed if so is $\mathcal{C}$, where projections, evaluations, pairing and currying of $\mathcal{V}_{\mathcal{C}}$ are respectively the corresponding 'β-ones' in $\mathcal{C}$.

The point here is that we may now decompose the standard interpretation $[\![\_]\!]_{\mathcal{S}}$ of functional programming languages in a CCC $\mathcal{V}_{\mathcal{C}}$ [118, 150, 41, 102] as a more intensional interpretation $[\![\_]\!]_{\mathcal{D}}$ in a CCBoC $\mathcal{C} = (\mathcal{C}, \mathcal{E})$ and the full evaluation $\mathcal{E}^{\omega} : \mathcal{C} \to \mathcal{V}_{\mathcal{C}}$, i.e., $[\![\_]\!]_{\mathcal{S}} = \mathcal{E}^{\omega}([\![\_]\!]_{\mathcal{D}})$, and talk about *intensional difference* between computations: Terms M and M' are interpreted to be *intensionally equal* if $[\![\mathsf{M}]\!]_{\mathcal{D}} = [\![\mathsf{M}']\!]_{\mathcal{D}}$ and *extensionally equal* if $[\![\mathsf{M}]\!]_{\mathcal{D}} \simeq [\![\mathsf{M}']\!]_{\mathcal{D}}$. Also, the *one-step* evaluation $\mathcal{E}$ is to capture the small-step operational semantics of the target language, i.e., to satisfy a DCP (see Definition 3.2.15 for the precise definition).

[6]In the present work, every dynamic strategy (or β-morphism), even a non-terminating one, e.g., a *fixed-point strategy* (Definition 4.2.7), becomes a value by a *finite* iteration of the evaluation due to the axiom on labeling functions (Definition 3.3.11), and therefore the axiom Subject Reduction (Definition 3.2.2) makes sense.

3.2.2 Finitary PCF

Next, let us introduce in the present section our target programming language for the first instance of dynamic game semantics.

First, recall that there is a one-to-one correspondence between *PCF Böhm trees* (i.e., terms of PCF in *η-long normal form*) [18] and innocent, well-bracketed strategies [100, 14, 43]; this technical highlight in the literature of game semantics is called *strong definability*. Naturally, we would like to exploit the strong definability result to establish the first instance of dynamic game semantics as the task would be easier.

On the other hand, the programming language *PCF* [163, 151] has the *natural number type* and the *fixed-point combinators*, which make PCF Böhm trees *infinitary* in width and depth, respectively. However, we would like to select, as the first target language for dynamic game semantics, the simplest one possible because then the idea and the mechanism would be most visible. For this reason, let us choose *finitary PCF*, i.e., the fragment of PCF that has only the *boolean type* as the ground type (or equivalently, the *simply-typed λ-calculus* [36, 172] equipped with the boolean type).

We then define a simple small-step operational semantics (or reduction strategy) of finitary PCF whose execution order is obvious from the types of terms and has an immediate counterpart in dynamic game semantics.

Remark. Note that an execution of *linear head reduction (LHR)* [48][7] corresponds in a step-by-step fashion to an 'internal communication' between strategies [47]. Hence, one may wonder if it would be better to employ LHR here; however, note that:

- The correspondence is *not* between terms and strategies;
- LHR is executed by *linear substitution*, which makes the calculus very different from the usual λ-calculus with β-reduction.

By these two points, we have conjectured that it would require significantly more work than the present chapter to establish a game-semantic DCP with respect to LHR, and therefore we leave it as future work.

In the following, we give the precise definition of the resulting target programming language (viz., finitary PCF equipped with a small-step operational semantics).

Remark. It is possible to further simplify the target programming language by replacing the underlying λ-calculus with the *linear* one [1]. However, we need implication $\Rightarrow$, not linear implication $\multimap$, in dynamic game semantics for Chapter 4, and therefore we shall keep the underlying λ-calculus unchanged.

[7] It is rather called *head linear reduction (HLR)* in this paper.

Notation. Henceforth, we employ the following notation:

- Let $\mathscr{V}$ be a countably infinite set of *variables*, written x, y, z, etc., for which we assume the *variable convention* (or *Barendregt's convention*)[8] [85];
- We use sans-serif letters such as $\mathsf{\Gamma}$, A and a for syntactic objects and $\equiv$ for syntactic equality up to α-*equivalence*, i.e., up to renaming of bound variables.

Definition 3.2.3 (FPCF [163, 151, 100, 9, 18])**.** The ***finitary PCF (FPCF)*** is a programming language defined as follows:

- (Types) A ***type*** A is an expression generated by the grammar:

$$\mathsf{A} \overset{\text{df.}}{\equiv} o \mid \mathsf{A}_1 \Rightarrow \mathsf{A}_2$$

  where o is the ***boolean type*** and $\mathsf{A}_1 \Rightarrow \mathsf{A}_2$ is the ***function type*** from A_1 to A_2 ($\Rightarrow$ is right associative). We write $\mathsf{A}, \mathsf{B}, \mathsf{C}$, etc. for types. Note that each type A is uniquely written $\mathsf{A}_1 \Rightarrow \mathsf{A}_2 \Rightarrow \cdots \Rightarrow \mathsf{A}_\mathsf{k} \Rightarrow o$, where $k \in \mathbb{N}$.

- (Raw-Terms) A ***raw-term*** M is an expression generated by the grammar:

$$\mathsf{M} \overset{\text{df.}}{\equiv} \mathsf{x} \mid \mathsf{tt} \mid \mathsf{ff} \mid \mathsf{case}(\mathsf{M})[\mathsf{M}_1; \mathsf{M}_2] \mid \lambda \mathsf{x}^\mathsf{A}.\mathsf{M} \mid \mathsf{M}_1\mathsf{M}_2$$

  where x ranges over variables, and A over types. We call tt, ff, $\lambda\mathsf{x}^\mathsf{A}.\mathsf{M}$ and $\mathsf{M}_1\mathsf{M}_2$ respectively the ***true constant***, the ***false constant***, an ***abstraction*** and an ***application***. We write $\mathsf{M}, \mathsf{P}, \mathsf{Q}, \mathsf{R}$, etc. for raw-terms and often omit A in an abstraction $\lambda\mathsf{x}^\mathsf{A}$; an application is always left-associative, e.g., $\mathsf{M}_1\mathsf{M}_2\mathsf{M}_3$ may be written informally $(\mathsf{M}_1\mathsf{M}_2)\mathsf{M}_3$. The set $\mathscr{FV}(\mathsf{M}) \subseteq \mathscr{V}$ of all ***free variables*** occurring in a raw-term M is defined by the following induction on M:

$$\begin{aligned}
\mathscr{FV}(\mathsf{x}) &\overset{\text{df.}}{=} \{\mathsf{x}\} \\
\mathscr{FV}(\mathsf{tt}) &\overset{\text{df.}}{=} \mathscr{FV}(\mathsf{ff}) \overset{\text{df.}}{=} \emptyset \\
\mathscr{FV}(\mathsf{case}(\mathsf{M})[\mathsf{M}_1; \mathsf{M}_2]) &\overset{\text{df.}}{=} \mathscr{FV}(\mathsf{M}) \cup \mathscr{FV}(\mathsf{M}_1) \cup \mathscr{FV}(\mathsf{M}_2) \\
\mathscr{FV}(\lambda\mathsf{x}.\mathsf{M}) &\overset{\text{df.}}{=} \mathscr{FV}(\mathsf{M}) \setminus \{\mathsf{x}\} \\
\mathscr{FV}(\mathsf{M}_1\mathsf{M}_2) &\overset{\text{df.}}{=} \mathscr{FV}(\mathsf{M}_1) \cup \mathscr{FV}(\mathsf{M}_2).
\end{aligned}$$

- (Contexts) A ***context*** is a finite sequence $\mathsf{x}_1 : \mathsf{A}_1, \mathsf{x}_2 : \mathsf{A}_2, \dots, \mathsf{x}_\mathsf{k} : \mathsf{A}_\mathsf{k}$ of (variable : type)-pairs such that $\mathsf{x}_\mathsf{i} \neq \mathsf{x}_\mathsf{j}$ if $i \neq j$. We write $\mathsf{\Gamma}$, $\mathsf{\Delta}$, $\mathsf{\Theta}$, etc. for contexts.

[8]I.e., we assume that in any term of concern every bound variable is chosen to be different from any free variable occurring in that mathematical context.

- (TERMS) A ***term*** is an expression of the form $\Gamma \vdash \mathsf{M} : \mathsf{B}$, where Γ is a ***context***, M is a raw-term, and B is a type, generated by the following ***typing rules***:

$$(\mathrm{B})\frac{\mathsf{b} \in \{\mathsf{tt}, \mathsf{ff}\}}{\Gamma \vdash \mathsf{b} : o} \quad (\mathrm{C1})\frac{\begin{array}{c}\mathsf{A} \equiv \mathsf{A}_1 \Rightarrow \mathsf{A}_2 \Rightarrow \cdots \Rightarrow \mathsf{A}_k \Rightarrow o \ \ \Gamma \equiv \Delta, \Theta \\ \forall i \in \{1, 2, \dots, k\}. \Gamma \vdash \mathsf{V}_i : \mathsf{A}_i \wedge \sharp(\mathsf{V}_i) = 0 \wedge \mathsf{x} \notin \mathscr{FV}(\mathsf{V}_i) \\ \forall j \in \{1, 2\}. \Gamma \vdash \mathsf{W}_j : o \wedge \sharp(\mathsf{W}_j) = 0 \wedge \mathsf{x} \notin \mathscr{FV}(\mathsf{W}_j)\end{array}}{\Delta, \mathsf{x} : \mathsf{A}, \Theta \vdash \mathsf{case}(\mathsf{x}\mathsf{V}_1\mathsf{V}_2 \dots \mathsf{V}_k)[\mathsf{W}_1; \mathsf{W}_2] : o}$$

$$(\mathrm{C2})\frac{\Gamma \vdash \mathsf{M} : o \ \ \forall j \in \{1, 2\}. \Gamma \vdash \mathsf{P}_j : o}{\Gamma \vdash \mathsf{case}(\mathsf{M})[\mathsf{P}_1; \mathsf{P}_2] : o} \quad (\mathrm{L})\frac{\Gamma, \mathsf{x} : \mathsf{A} \vdash \mathsf{M} : \mathsf{B}}{\Gamma \vdash \lambda \mathsf{x}^{\mathsf{A}}. \mathsf{M} : \mathsf{A} \Rightarrow \mathsf{B}}$$

$$(\mathrm{A})\frac{\Gamma \vdash \mathsf{M}_1 : \mathsf{A} \Rightarrow \mathsf{B} \ \ \Gamma \vdash \mathsf{M}_2 : \mathsf{A}}{\Gamma \vdash \mathsf{M}_1\mathsf{M}_2 : \mathsf{B}}$$

  where $\sharp(\Gamma \vdash \mathsf{M} : \mathsf{B}) \in \mathbb{N}$, often abbreviated as $\sharp(\mathsf{M})$, is the ***execution number*** of each term $\Gamma \vdash \mathsf{M} : \mathsf{B}$ defined by the following induction on $\Gamma \vdash \mathsf{M} : \mathsf{B}$:

  - $\sharp(\mathsf{b}) \overset{\mathrm{df.}}{=} 0$ if $\mathsf{b} \in \{\mathsf{tt}, \mathsf{ff}\}$;
  - $\sharp(\mathsf{case}(\mathsf{x}\mathsf{V}_1\mathsf{V}_2 \dots \mathsf{V}_k)[\mathsf{W}_1; \mathsf{W}_2]) \overset{\mathrm{df.}}{=} 0$;
  - $\sharp(\mathsf{case}(\mathsf{M})[\mathsf{P}_1; \mathsf{P}_2]) \overset{\mathrm{df.}}{=} 0$;
  - $\sharp(\lambda \mathsf{x}^{\mathsf{A}}.\mathsf{M}) \overset{\mathrm{df.}}{=} \sharp(\mathsf{M})$;
  - $\sharp(\mathsf{M}_1\mathsf{M}_2) \overset{\mathrm{df.}}{=} \max(\sharp(\mathsf{M}_1), \sharp(\mathsf{M}_1)) + 1$.

  We identify terms up to *α-equivalence* [85, 172]. We write $\Gamma \vdash \{\mathsf{M}\}_e : \mathsf{B}$ for the term $\Gamma \vdash \mathsf{M} : \mathsf{B}$ such that $\sharp(\mathsf{M}) = e$. Also, we often omit the context and/or the type of a term if it does not bring confusion. A ***program*** (resp. a ***value***) is a term generated by the rules B, C1, L and A (resp. B, C1 and L).[9] A ***subterm*** of a term $\Gamma \vdash \mathsf{M} : \mathsf{B}$ is a term that occurs in the deduction of $\Gamma \vdash \mathsf{M} : \mathsf{B}$.[10]

- ($\beta\vartheta$-REDUCTION) The ***βϑ-reduction*** $\rightarrow_{\beta\vartheta}$ on terms is the *contextual closure*[11] [85] of the union of the following five rules:

$$\begin{aligned}(\lambda \mathsf{x}. \mathsf{M})\mathsf{P} &\rightarrow_{\beta} \mathsf{M}[\mathsf{P}/\mathsf{x}] \\ \mathsf{case}(\mathsf{tt})[\mathsf{M}_1; \mathsf{M}_2] &\rightarrow_{\vartheta_1} \mathsf{M}_1 \\ \mathsf{case}(\mathsf{ff})[\mathsf{M}_1; \mathsf{M}_2] &\rightarrow_{\vartheta_2} \mathsf{M}_2 \\ \mathsf{case}(\mathsf{case}(\mathsf{x}\boldsymbol{\mathsf{V}})[\mathsf{W}_1; \mathsf{W}_2])[\mathsf{M}_1; \mathsf{M}_2] &\rightarrow_{\vartheta_3} \mathsf{case}(\mathsf{x}\boldsymbol{\mathsf{V}})[\mathsf{case}(\mathsf{W}_1)[\mathsf{M}_1; \mathsf{M}_2]; \mathsf{case}(\mathsf{W}_2)[\mathsf{M}_1; \mathsf{M}_2]] \\ \mathsf{case}(\mathsf{case}(\mathsf{M})[\mathsf{P}_1; \mathsf{P}_2])[\mathsf{Q}_1; \mathsf{Q}_2] &\rightarrow_{\vartheta_4} \mathsf{case}(\mathsf{M})[\mathsf{case}(\mathsf{P}_1)[\mathsf{Q}_1; \mathsf{Q}_2]; \mathsf{case}(\mathsf{P}_2)[\mathsf{Q}_1; \mathsf{Q}_2]]\end{aligned}$$

[9]The rules C2 and ϑ_4 are necessary for 'intermediate terms' during an execution of a program.
[10]Note that a *deduction (tree)* of each term of FPCF is *unique*.
[11]I.e., the closure with respect to the typing rules.

where $\mathsf{M}[\mathsf{P}/\mathsf{x}]$ denotes the *capture-free substitution* [85] of P for x in M, and $\mathsf{x}\mathbf{V}$ abbreviates $\mathsf{x}\mathsf{V}_1\mathsf{V}_2\dots\mathsf{V}_\mathsf{k}$ of the rule C1. We write $\mathit{nf}(\mathsf{M})$ for the *normal form* of each term M with respect to $\rightarrow_{\beta\vartheta}$, i.e., $\mathit{nf}(\mathsf{M})$ is a term such that $\mathsf{M} \rightarrow^*_{\beta\vartheta} \mathit{nf}(\mathsf{M})$[12] and $\mathit{nf}(\mathsf{M}) \not\rightarrow_{\beta\vartheta} \mathsf{M}'$ for any term M', which uniquely exists by Theorems 3.2.10 and 3.2.11. The ***parallel $\beta\vartheta$-reduction*** $\rightrightarrows_{\beta\vartheta}$ on terms evaluates each term M in a single-step to its normal form $\mathit{nf}(\mathsf{M})$.

- (Operational Semantics) The ***(small-step) operational semantics*** $\rightarrow$ on programs M is the simultaneous execution of $\rightrightarrows_{\beta\vartheta}$ on all subterms of M with the execution number 1, or more precisely, $\rightarrow$ is defined recursively by:

$$\mathsf{M} \rightarrow \begin{cases} \mathsf{V} & \text{if } \mathsf{M} \equiv \mathsf{M}_1\mathsf{M}_2,\ \sharp(\mathsf{M}_1\mathsf{M}_2) = 1 \text{ and } \mathsf{M}_1\mathsf{M}_2 \rightrightarrows_{\beta\vartheta} \mathsf{V}; \\ \mathsf{M}'_1\mathsf{M}'_2 & \text{if } \mathsf{M} \equiv \mathsf{M}_1\mathsf{M}_2,\ \sharp(\mathsf{M}_1\mathsf{M}_2) \geqslant 2 \text{ and } \mathsf{M}_\mathsf{i} \rightarrow \mathsf{M}'_\mathsf{i} \text{ for } i = 1, 2; \\ \lambda \mathsf{x}^\mathsf{A}.\, \tilde{\mathsf{M}}' & \text{if } \mathsf{M} \equiv \lambda \mathsf{x}^\mathsf{A}.\, \tilde{\mathsf{M}} \text{ and } \tilde{\mathsf{M}} \rightarrow \tilde{\mathsf{M}}'. \end{cases}$$

Eq(FPCF) is the equational theory that consists of judgements $\Gamma \vdash \mathsf{M} = \mathsf{M}' : \mathsf{B}$, where $\Gamma \vdash \mathsf{M} : \mathsf{B}$ and $\Gamma \vdash \mathsf{M}' : \mathsf{B}$ are terms of FPCF such that $\mathit{nf}(\mathsf{M}) \equiv \mathit{nf}(\mathsf{M}')$.

Note that values of FPCF are *PCF Böhm trees* [100, 14, 43, 18] except that the 'bottom term' $\bot$ and the *natural number type* ι are both excluded, and the $\beta\vartheta$-reduction $\rightarrow_{\beta\vartheta}$ is taken from Section 6 of the book [18].

Remark. Let $\mathsf{A} \equiv \mathsf{A}_1 \Rightarrow \mathsf{A}_2 \Rightarrow \cdots \Rightarrow \mathsf{A}_\mathsf{k} \Rightarrow o$ be an arbitrary type of FPCF. Note that an expression of the form $\Delta, \mathsf{x} : \mathsf{A}, \Theta \vdash \mathsf{x} : \mathsf{A}$ is *not* an term of FPCF, but instead there is another $\Delta, \mathsf{x} : \mathsf{A}, \Theta \vdash \underline{\mathsf{x}}^\mathsf{A} : \mathsf{A}$, where $\underline{\mathsf{x}}^\mathsf{A} \stackrel{\text{df.}}{\equiv} \lambda \mathsf{x}_1^{\mathsf{A}_1}\mathsf{x}_2^{\mathsf{A}_2}\dots\mathsf{x}_\mathsf{k}^{\mathsf{A}_\mathsf{k}}.\, \mathsf{case}(\mathsf{x}\underline{\mathsf{x}_1}^{\mathsf{A}_1}\underline{\mathsf{x}_2}^{\mathsf{A}_2}\dots\underline{\mathsf{x}_\mathsf{k}}^{\mathsf{A}_\mathsf{k}})[\mathsf{tt}; \mathsf{ff}]$, which is a term of FPCF. We often write $\underline{\mathsf{x}}$ for $\underline{\mathsf{x}}^\mathsf{A}$ if it does not bring confusion.

Thus, FPCF computes as follows. Given a program $\Gamma \vdash \{\mathsf{M}\}_e : \mathsf{B}$, it produces a *finite* chain of *finitary* rewriting

$$\mathsf{M} \rightarrow \mathsf{M}_1 \rightarrow \mathsf{M}_2 \rightarrow \cdots \rightarrow \mathsf{M}_\mathsf{e} \tag{3.1}$$

where M_e is a value. Note that the program M is constructed from values by a finite number of applications, and the computation (3.1) is executed in the *first-applications-first-evaluated* fashion, e.g., if $\mathsf{M} \equiv (\mathsf{V}_1\mathsf{V}_2)((\mathsf{V}_3\mathsf{V}_4)(\mathsf{V}_5\mathsf{V}_6))$ and $e = 3$, where $\mathsf{V}_1, \mathsf{V}_2, \dots, \mathsf{V}_6$ are values, then the computation (3.1) would be of the form

$$(\mathsf{V}_1\mathsf{V}_2)((\mathsf{V}_3\mathsf{V}_4)(\mathsf{V}_5\mathsf{V}_6)) \rightarrow \mathsf{V}_7(\mathsf{V}_8\mathsf{V}_9) \rightarrow \mathsf{V}_7\mathsf{V}_{10} \rightarrow \mathsf{V}_{11}$$

where $\mathsf{V}_7 \equiv \mathit{nf}(\mathsf{V}_1\mathsf{V}_2)$, $\mathsf{V}_8 \equiv \mathit{nf}(\mathsf{V}_3\mathsf{V}_4)$, $\mathsf{V}_9 \equiv \mathit{nf}(\mathsf{V}_5\mathsf{V}_6)$, $\mathsf{V}_{10} \equiv \mathit{nf}(\mathsf{V}_8\mathsf{V}_9)$ and $\mathsf{V}_{11} \equiv \mathit{nf}(\mathsf{V}_7\mathsf{V}_{10})$.

[12] $\rightarrow^*_{\beta\vartheta}$ is the *reflexive, transitive closure* of $\rightarrow_{\beta\vartheta}$.

The rest of the present section is devoted to showing that the computation (3.1) of FPCF in fact correctly works (Corollary 3.2.13).

First, by the following Proposition 3.2.4 and Theorem 3.2.8, it makes sense that $\rightarrow_{\beta\vartheta}$ is defined *on terms* (not on raw-terms):

Proposition 3.2.4 (Unique typing). *If* $\Gamma \vdash \{\mathsf{M}\}_e : \mathsf{B}$ *and* $\Gamma \vdash \{\mathsf{M}\}_{e'} : \mathsf{B}'$*, then* $e = e'$ *and* $\mathsf{B} \equiv \mathsf{B}'$*.*

Proof. By induction on the construction of $\Gamma \vdash \mathsf{M} : \mathsf{B}$. $\square$

Lemma 3.2.5 (Free variable lemma). *If* $\Gamma \vdash \mathsf{M} : \mathsf{B}$*, and* $\mathsf{x} \in \mathscr{V}$ *occurs free in* M*, then* $\mathsf{x} : \mathsf{A}$ *occurs in* Γ *for some type* A*.*

Proof. By induction on the construction of $\Gamma \vdash \mathsf{M} : \mathsf{B}$. $\square$

Lemma 3.2.6 (EW-lemma). *If* $\mathsf{x}_1 : \mathsf{A}_1, \mathsf{x}_2 : \mathsf{A}_2, \dots, \mathsf{x}_\mathsf{k} : \mathsf{A}_\mathsf{k} \vdash \{\mathsf{M}\}_e : \mathsf{B}$*, then:*

1. $\mathsf{x}_{\sigma(1)} : \mathsf{A}_{\sigma(1)}, \mathsf{x}_{\sigma(2)} : \mathsf{A}_{\sigma(2)}, \dots, \mathsf{x}_{\sigma(\mathsf{k})} : \mathsf{A}_{\sigma(\mathsf{k})} \vdash \{\mathsf{M}\}_e : \mathsf{B}$ *for any permutation* σ *of the set* $\{1, 2, \dots, k\}$*;*

2. $\mathsf{x}_1 : \mathsf{A}_1, \mathsf{x}_2 : \mathsf{A}_2, \dots, \mathsf{x}_\mathsf{k} : \mathsf{A}_\mathsf{k}, \mathsf{x}_{\mathsf{k}+1} : \mathsf{A}_{\mathsf{k}+1} \vdash \{\mathsf{M}\}_e : \mathsf{B}$ *for any variable* $\mathsf{x}_{\mathsf{k}+1} \in \mathscr{V}$ *and type* $\mathsf{A}_{\mathsf{k}+1}$ *such that* $\mathsf{x}_{\mathsf{k}+1} \not\equiv \mathsf{x}_\mathsf{i}$ *for* $i = 1, 2, \dots, k$*.*

Proof. By induction on the construction of $\mathsf{x}_1 : \mathsf{A}_1, \mathsf{x}_2 : \mathsf{A}_2, \dots, \mathsf{x}_\mathsf{k} : \mathsf{A}_\mathsf{k} \vdash \mathsf{M} : \mathsf{B}$. $\square$

Lemma 3.2.7 (Substitution lemma). *If* $\Gamma, \mathsf{x} : \mathsf{A} \vdash \{\mathsf{P}\}_e : \mathsf{B}$ *and* $\Gamma \vdash \mathsf{Q} : \mathsf{A}$*, then* $\Gamma \vdash \{\mathsf{P}[\mathsf{Q}/\mathsf{x}]\}_e : \mathsf{B}$*.*

Proof. By a simple induction on $|\mathsf{P}|$:

- If $\Gamma, \mathsf{x} : \mathsf{A} \vdash \{\mathsf{b}\}_0 : o$, where $\mathsf{b} \in \{\mathsf{tt}, \mathsf{ff}\}$, and $\Gamma \vdash \mathsf{Q} : \mathsf{A}$, then the claim trivially holds as $\Gamma \vdash \{\mathsf{b}\}_0 : o$ by the rule B.

- If $\Gamma, \mathsf{x} : \mathsf{A} \vdash \{\lambda \mathsf{z}^\mathsf{C}.\mathsf{P}\}_d : \mathsf{C} \Rightarrow \mathsf{B}$ and $\Gamma \vdash \mathsf{Q} : \mathsf{A}$, then $\Gamma, \mathsf{x} : \mathsf{A}, \mathsf{z} : \mathsf{C} \vdash \{\mathsf{P}\}_d : \mathsf{B}$; thus, $\Gamma, \mathsf{z} : \mathsf{C}, \mathsf{x} : \mathsf{A} \vdash \{\mathsf{P}\}_d : \mathsf{B}$ by Lemma 3.2.6. Then, $\Gamma, \mathsf{z} : \mathsf{C} \vdash \{\mathsf{P}[\mathsf{Q}/\mathsf{x}]\}_d : \mathsf{B}$ by the induction hypothesis, whence $\Gamma \vdash \{\lambda \mathsf{z}^\mathsf{C}.(\mathsf{P}[\mathsf{Q}/\mathsf{x}]) \equiv (\lambda \mathsf{z}^\mathsf{C}.\mathsf{P})[\mathsf{Q}/\mathsf{x}]\}_d : \mathsf{C} \Rightarrow \mathsf{B}$ by the rule L.

- If $\Gamma, \mathsf{x} : \mathsf{A} \vdash \{\mathsf{case}(\mathsf{y}\mathsf{V}_1\mathsf{V}_2 \dots \mathsf{V}_\mathsf{k})[\mathsf{W}_1; \mathsf{W}_2]\}_0 : o$ and $\Gamma \vdash \mathsf{Q} : \mathsf{A}$ such that $\mathsf{y} \not\equiv \mathsf{x}$, then $\mathsf{y} : \mathsf{C} \in \Gamma$ for some type $\mathsf{C} \equiv \mathsf{C}_1 \Rightarrow \mathsf{C}_2 \Rightarrow \cdots \Rightarrow \mathsf{C}_\mathsf{k} \Rightarrow o$ by Lemma 3.2.5, $\Gamma \setminus \{\mathsf{y} : \mathsf{C}\}, \mathsf{x} : \mathsf{A} \vdash \{\mathsf{V}_\mathsf{i}\}_0 : \mathsf{C}_\mathsf{i}$ for $i = 1, 2, \dots, k$, and $\Gamma \setminus \{\mathsf{y} : \mathsf{C}\}, \mathsf{x} : \mathsf{A} \vdash \{\mathsf{W}_\mathsf{j}\}_0 : o$ for $j = 1, 2$. By the induction hypothesis, $\Gamma \setminus \{\mathsf{y} : \mathsf{C}\} \vdash \{\mathsf{V}_\mathsf{i}[\mathsf{Q}/\mathsf{x}]\}_0 : \mathsf{C}_\mathsf{i}$ for $i = 1, 2, \dots, k$, and $\Gamma \setminus \{\mathsf{y} : \mathsf{C}\} \vdash \{\mathsf{W}_\mathsf{j}[\mathsf{Q}/\mathsf{x}]\}_0 : o$ for $j = 1, 2$. By the rule C1, we get $\Gamma \vdash \{\mathsf{case}(\mathsf{y}(\mathsf{V}_1[\mathsf{Q}/\mathsf{x}])(\mathsf{V}_2[\mathsf{Q}/\mathsf{x}]) \dots (\mathsf{V}_\mathsf{k}[\mathsf{Q}/\mathsf{x}]))[\mathsf{W}_1[\mathsf{Q}/\mathsf{x}]; \mathsf{W}_2[\mathsf{Q}/\mathsf{x}]]\}_0 : o$, which is $\Gamma \vdash \{\mathsf{case}(\mathsf{y}\mathsf{V}_1\mathsf{V}_2 \dots \mathsf{V}_\mathsf{k})[\mathsf{W}_1; \mathsf{W}_2][\mathsf{Q}/\mathsf{x}]\}_0 : o$.

- If $\Gamma, \mathsf{x} : \mathsf{A} \vdash \{\mathsf{case}(\mathsf{x}\mathsf{V}_1\mathsf{V}_2 \ldots \mathsf{V}_\mathsf{k})[\mathsf{W}_1; \mathsf{W}_2]\}_0 : o$ and $\Gamma \vdash \mathsf{Q} : \mathsf{A}$, where $\mathsf{A} \equiv \mathsf{A}_1 \Rightarrow \mathsf{A}_2 \Rightarrow \cdots \Rightarrow \mathsf{A}_\mathsf{k} \Rightarrow o$, then $\Gamma \vdash \{\mathsf{V}_\mathsf{i}\}_0 : \mathsf{A}_\mathsf{i}$ for $i = 1, 2, \ldots, k$, and $\Gamma \vdash \{\mathsf{W}_\mathsf{j}\}_0 : o$ for $j = 1, 2$. By Lemma 3.2.6, $\Gamma, \mathsf{x} : \mathsf{A} \vdash \{\mathsf{V}_\mathsf{i}\}_0 : \mathsf{A}_\mathsf{i}$ for $i = 1, 2, \ldots, k$, and $\Gamma, \mathsf{x} : \mathsf{A} \vdash \{\mathsf{W}_\mathsf{j}\}_0 : o$ for $j = 1, 2$. By the induction hypothesis, $\Gamma \vdash \{\mathsf{V}_\mathsf{i}[\mathsf{Q}/\mathsf{x}]\}_0 : \mathsf{A}_\mathsf{i}$ for $i = 1, 2, \ldots, k$, and $\Gamma \vdash \{\mathsf{W}_\mathsf{j}[\mathsf{Q}/\mathsf{x}]\}_0 : o$ for $j = 1, 2$. Now, by an iterated application of the rule A, we obtain $\Gamma \vdash \mathsf{Q}(\mathsf{V}_1[\mathsf{Q}/\mathsf{x}])(\mathsf{V}_2[\mathsf{Q}/\mathsf{x}]) \ldots (\mathsf{V}_\mathsf{k}[\mathsf{Q}/\mathsf{x}]) : o$. By the rule C2, $\Gamma \vdash \{\mathsf{case}(\mathsf{Q}(\mathsf{V}_1[\mathsf{Q}/\mathsf{x}])(\mathsf{V}_2[\mathsf{Q}/\mathsf{x}]) \ldots (\mathsf{V}_\mathsf{k}[\mathsf{Q}/\mathsf{x}]))[\mathsf{W}_1[\mathsf{Q}/\mathsf{x}]; \mathsf{W}_2[\mathsf{Q}/\mathsf{x}]]\}_0 : o$, i.e., $\Gamma \vdash \{\mathsf{case}(\mathsf{x}\mathsf{V}_1\mathsf{V}_2 \ldots \mathsf{V}_\mathsf{k})[\mathsf{W}_1; \mathsf{W}_2][\mathsf{Q}/\mathsf{x}]\}_0 : o$.

- If $\Gamma, \mathsf{x} : \mathsf{A} \vdash \{\mathsf{MN}\}_{m+1} : \mathsf{B}$ and $\Gamma \vdash \mathsf{Q} : \mathsf{A}$, then $\Gamma, \mathsf{x} : \mathsf{A} \vdash \{\mathsf{M}\}_d : \mathsf{C} \Rightarrow \mathsf{B}$, and $\Gamma, \mathsf{x} : \mathsf{A} \vdash \{\mathsf{N}\}_e : \mathsf{C}$ for some type C such that $\max(d, e) = m$. By the induction hypothesis, we have $\Gamma \vdash \{\mathsf{M}[\mathsf{Q}/\mathsf{x}]\}_d : \mathsf{C} \Rightarrow \mathsf{B}$ and $\Gamma \vdash \{\mathsf{N}[\mathsf{Q}/\mathsf{x}]\}_e : \mathsf{C}$. Thus, by the rule A, we may obtain $\Gamma \vdash \{\mathsf{M}[\mathsf{Q}/\mathsf{x}]\mathsf{N}[\mathsf{Q}/\mathsf{x}]\}_{\max(d,e)+1} : \mathsf{B}$, i.e., $\Gamma \vdash \{\mathsf{MN}[\mathsf{Q}/\mathsf{x}]\}_{m+1} : \mathsf{B}$.

- If $\Gamma, \mathsf{x} : \mathsf{A} \vdash \{\mathsf{case}(\mathsf{M})[\mathsf{M}_1; \mathsf{M}_2]\}_0 : o$ and $\Gamma \vdash \mathsf{Q} : \mathsf{A}$, then $\Gamma, \mathsf{x} : \mathsf{A} \vdash \mathsf{M} : o$, and $\Gamma, \mathsf{x} : \mathsf{A} \vdash \mathsf{M}_\mathsf{j} : o$ for $j = 1, 2$. By the induction hypothesis, $\Gamma \vdash \mathsf{M}[\mathsf{Q}/\mathsf{x}] : o$, and $\Gamma \vdash \mathsf{M}_\mathsf{j}[\mathsf{Q}/\mathsf{x}] : o$ for $j = 1, 2$. Thus, $\Gamma \vdash \{\mathsf{case}(\mathsf{M}[\mathsf{Q}/\mathsf{x}])[\mathsf{M}_1[\mathsf{Q}/\mathsf{x}]; \mathsf{M}_2[\mathsf{Q}/\mathsf{x}]]\}_0 : o$ by the rule C2, i.e., $\Gamma \vdash \{\mathsf{case}(\mathsf{M})[\mathsf{M}_1; \mathsf{M}_2][\mathsf{Q}/\mathsf{x}]\}_0 : o$

which completes the proof. □

Theorem 3.2.8 (Subject reduction). *If* $\Gamma \vdash \mathsf{M} : \mathsf{B}$ *and* $\mathsf{M} \to_{\beta\vartheta} \mathsf{R}$*, then* $\Gamma \vdash \mathsf{R} : \mathsf{B}$.

Proof. By a simple induction on the structure $\mathsf{M} \to_{\beta\vartheta} \mathsf{R}$. In the following, let us write $[\mathsf{P}_1; \mathsf{P}_2]; [\mathsf{Q}_1; \mathsf{Q}_2]$ for $[\mathsf{case}(\mathsf{P}_1)[\mathsf{Q}_1; \mathsf{Q}_2]; \mathsf{case}(\mathsf{P}_2)[\mathsf{Q}_1; \mathsf{Q}_2]]$.

- If $\mathsf{M} \equiv (\lambda \mathsf{x}^\mathsf{A}.\mathsf{P})\mathsf{Q}$ and $\mathsf{R} \equiv \mathsf{P}[\mathsf{Q}/\mathsf{x}]$, then $\Gamma \vdash \mathsf{Q} : \mathsf{A}$ and $\Gamma, \mathsf{x} : \mathsf{A} \vdash \mathsf{P} : \mathsf{B}$. Then, we have, by Lemma 3.2.7, $\Gamma \vdash \mathsf{P}[\mathsf{Q}/\mathsf{x}] : \mathsf{B}$.

- If $\mathsf{M} \equiv \mathsf{case}(\mathsf{tt})[\mathsf{M}_1; \mathsf{M}_2]$ and $\mathsf{R} \equiv \mathsf{M}_1$, then $\mathsf{B} \equiv o$. From $\Gamma \vdash \mathsf{case}(\mathsf{tt})[\mathsf{M}_1; \mathsf{M}_2] : o$, we may conclude that $\Gamma \vdash \mathsf{M}_1 : o$. The case where $\mathsf{M} \equiv \mathsf{case}(\mathsf{ff})[\mathsf{M}_1; \mathsf{M}_2]$ and $\mathsf{R} \equiv \mathsf{M}_2$ is analogous.

- If $\mathsf{M} \equiv \mathsf{case}(\mathsf{case}(\mathsf{P})[\mathsf{P}'_1; \mathsf{P}'_2])[\mathsf{Q}'_1; \mathsf{Q}'_2]$ and $\mathsf{R} \equiv \mathsf{case}(\mathsf{P})[\mathsf{P}'_1; \mathsf{P}'_2]; [\mathsf{Q}'_1; \mathsf{Q}'_2]$, then $\mathsf{B} \equiv o$. Then, we clearly have $\Gamma \vdash \mathsf{case}(\mathsf{P})[\mathsf{P}'_1; \mathsf{P}'_2] : o$, from which we deduce $\Gamma \vdash \mathsf{P} : o$, $\Gamma \vdash \mathsf{P}'_\mathsf{j} : o$ and $\Gamma \vdash \mathsf{Q}'_\mathsf{j} : o$ for $j = 1, 2$. Thus, $\Gamma \vdash \mathsf{case}(\mathsf{P}'_\mathsf{j})[\mathsf{Q}'_1; \mathsf{Q}'_2] : o$ for $j = 1, 2$ by the rule C2. Hence, $\Gamma \vdash \mathsf{case}(\mathsf{P})[\mathsf{P}'_1; \mathsf{P}'_2]; [\mathsf{Q}'_1; \mathsf{Q}'_2] : o$ by the rule C2.

- If $\mathsf{M} \equiv \mathsf{case}(\mathsf{case}(\mathsf{x}\mathbf{V})[\mathsf{W}_1; \mathsf{W}_2])[\mathsf{U}_1; \mathsf{U}_2]$ and $\mathsf{R} \equiv \mathsf{case}(\mathsf{x}\mathbf{V})[\mathsf{W}_1; \mathsf{W}_2]; [\mathsf{U}_1; \mathsf{U}_2]$, then it is handled in a similar manner to the above case.

- If $\mathsf{M} \equiv \lambda \mathsf{x}^{\mathsf{A}}.\mathsf{P}$, $\mathsf{B} \equiv \mathsf{A} \Rightarrow \mathsf{C}$, $\mathsf{R} \equiv \lambda \mathsf{x}^{\mathsf{A}}.\mathsf{Q}$ and $\mathsf{P} \rightarrow_{\beta\vartheta} \mathsf{Q}$, then $\Gamma, \mathsf{x} : \mathsf{A} \vdash \mathsf{P} : \mathsf{C}$. By the induction hypothesis, $\Gamma, \mathsf{x} : \mathsf{A} \vdash \mathsf{Q} : \mathsf{C}$. Hence, $\Gamma \vdash \lambda \mathsf{x}^{\mathsf{A}}.\mathsf{Q} : \mathsf{B}$ by the rule L.
- If $\mathsf{M} \equiv \mathsf{case}(\mathsf{P})[\mathsf{P}'_1; \mathsf{P}'_2]$ and $\mathsf{R} \equiv \mathsf{case}(\mathsf{Q})[\mathsf{Q}'_1; \mathsf{Q}'_2]$ such that just one $\rightarrow_{\beta\vartheta}$ holds in the following conjunction $(\mathsf{P} \rightarrow_{\beta\vartheta} \mathsf{Q} \vee \mathsf{P} \equiv \mathsf{Q}) \wedge (\mathsf{P}'_1 \rightarrow_{\beta\vartheta} \mathsf{Q}'_1 \vee \mathsf{P}'_1 \equiv \mathsf{Q}'_1) \wedge (\mathsf{P}'_2 \rightarrow_{\beta\vartheta} \mathsf{Q}'_2 \vee \mathsf{P}'_2 \equiv \mathsf{Q}'_2)$, then $\mathsf{B} \equiv o$. In either case, it follows from the induction hypothesis and the rule C2 that $\Gamma \vdash \mathsf{case}(\mathsf{Q})[\mathsf{Q}'_1; \mathsf{Q}'_2] : o$.
- If $\mathsf{M} \equiv \mathsf{case}(\mathsf{x}\mathsf{V}_1\mathsf{V}_2 \ldots \mathsf{V}_\mathsf{k})[\mathsf{V}'_1; \mathsf{V}'_2]$ and $\mathsf{R} \equiv \mathsf{case}(\mathsf{x}\mathsf{W}_1\mathsf{W}_2 \ldots \mathsf{W}_\mathsf{k})[\mathsf{W}'_1; \mathsf{W}'_2]$ such that just one $\rightarrow_{\beta\vartheta}$ holds in the following conjunction $(\mathsf{V}_1 \rightarrow_{\beta\vartheta} \mathsf{W}_1 \vee \mathsf{V}_1 \equiv \mathsf{W}_1) \wedge (\mathsf{V}_2 \rightarrow_{\beta\vartheta} \mathsf{W}_2 \vee \mathsf{V}_2 \equiv \mathsf{W}_2) \wedge \cdots \wedge (\mathsf{V}_\mathsf{k} \rightarrow_{\beta\vartheta} \mathsf{W}_\mathsf{k} \vee \mathsf{V}_\mathsf{k} \equiv \mathsf{W}_\mathsf{k}) \wedge (\mathsf{V}'_1 \rightarrow_{\beta\vartheta} \mathsf{W}'_1 \vee \mathsf{V}'_1 \equiv \mathsf{W}'_1) \wedge (\mathsf{V}'_2 \rightarrow_{\beta\vartheta} \mathsf{W}'_2 \vee \mathsf{V}'_2 \equiv \mathsf{W}'_2)$, then $\mathsf{B} \equiv o$. In either case, it immediately follows from the induction hypothesis and the rule C1 that $\Gamma \vdash \mathsf{case}(\mathsf{x}\mathsf{W}_1 \ldots \mathsf{W}_\mathsf{k})[\mathsf{W}'_1; \mathsf{W}'_2] : o$.
- If $\mathsf{M} \equiv \mathsf{PQ}$, $\mathsf{R} \equiv \mathsf{TS}$ and $(\mathsf{P} \rightarrow_{\beta\vartheta} \mathsf{T} \wedge \mathsf{Q} \equiv \mathsf{S}) \vee (\mathsf{Q} \rightarrow_{\beta\vartheta} \mathsf{S} \wedge \mathsf{P} \equiv \mathsf{T})$ with just one conjunct valid, then clearly $\Gamma \vdash \mathsf{P} : \mathsf{A} \Rightarrow \mathsf{B}$ and $\Gamma \vdash \mathsf{Q} : \mathsf{A}$ for some type A. By the induction hypothesis, if $\mathsf{P} \rightarrow_{\beta\vartheta} \mathsf{T}$, then $\Gamma \vdash \mathsf{T} : \mathsf{A} \Rightarrow \mathsf{B}$ and $\Gamma \vdash \mathsf{S} : \mathsf{A}$; thus, $\Gamma \vdash \mathsf{TS} : \mathsf{B}$ by the rule L. The other case is analogous.

We have considered all the cases for $\mathsf{M} \rightarrow_{\beta\vartheta} \mathsf{R}$, establishing the theorem. □

Next, we show that $\rightrightarrows_{\beta\vartheta}$ is well-defined (Theorems 3.2.10 and 3.2.11).

Lemma 3.2.9 (Hidley-Rosen [85]). *Let R_1 and R_2 be any binary relations on the set $\mathcal{T}$ of terms, and let us write $\rightarrow_{R_i}$ for the contextual closure of R_i for $i = 1, 2$. If $\rightarrow_{R_1}$ and $\rightarrow_{R_2}$ are both Church-Rosser, and satisfy $\forall \mathsf{M}, \mathsf{P}, \mathsf{Q} \in \mathcal{T}.\ \mathsf{M} \rightarrow^*_{\mathsf{R}_1} \mathsf{P} \wedge \mathsf{M} \rightarrow^*_{\mathsf{R}_2} \mathsf{Q} \Rightarrow \exists \mathsf{R} \in \mathcal{T}.\ \mathsf{P} \rightarrow^*_{\mathsf{R}_2} \mathsf{R} \wedge \mathsf{Q} \rightarrow^*_{\mathsf{R}_1} \mathsf{R}$, then $\rightarrow_{R_1 \cup R_2}$ is Church-Rosser.*

Proof. By simple 'diagram chases'; see [85] for the detail. □

Theorem 3.2.10 (CR). *The $\beta\vartheta$-reduction $\rightarrow_{\beta\vartheta}$ is Church-Rosser.*

Proof. We may just apply *Mitschke's theorem* [135, 85], but it is insightful to follow its proof here.

First, it is easy to see that the *ϑ-reduction* $\rightarrow_\vartheta \overset{\text{df.}}{=} \bigcup_{i=1}^4 \rightarrow_{\vartheta_i}$ satisfies the *diamond-property*, and thus it is Church-Rosser.

Also, we may show that:

$$\mathsf{M} \rightarrow_\beta \mathsf{P} \wedge \mathsf{M} \rightarrow_\vartheta \mathsf{Q} \Rightarrow \exists \mathsf{R}.\ \mathsf{P} \rightarrow^*_\vartheta \mathsf{R} \wedge \mathsf{Q} \rightarrow_\beta \mathsf{R} \tag{3.2}$$

for all terms M, P and Q, where note the asymmetry of $\rightarrow_\vartheta$ and $\rightarrow_\beta$, by a case analysis on the relation between β- and ϑ-redexes in M:

- If the β-redex is inside the ϑ-redex, then it is easy to see that (3.2) holds;
- If the ϑ-redex is inside the body of the function subterm of the β-redex, then it suffices to show that $\rightarrow_\vartheta$ commutes with substitution, but it is straightforward;
- If ϑ-redex is inside the argument of the β-redex, then it may be duplicated by a finite number n, but whatever the number n is, (3.2) clearly holds;
- If the β- and ϑ-redexes are disjoint, then (3.2) trivially holds.

It then follows from (3.2) that:

$$\mathsf{M} \rightarrow_\beta \mathsf{P} \wedge \mathsf{M} \rightarrow^*_\vartheta \mathsf{Q} \Rightarrow \exists \mathsf{R}.\, \mathsf{P} \rightarrow^*_\vartheta \mathsf{R} \wedge \mathsf{Q} \rightarrow_\beta \mathsf{R} \tag{3.3}$$

which in turn implies that:

$$\mathsf{M} \rightarrow^*_\beta \mathsf{P} \wedge \mathsf{M} \rightarrow^*_\vartheta \mathsf{Q} \Rightarrow \exists \mathsf{R}.\, \mathsf{P} \rightarrow^*_\vartheta \mathsf{R} \wedge \mathsf{Q} \rightarrow^*_\beta \mathsf{R} \tag{3.4}$$

for all terms M, P and Q. Applying Lemma 3.2.9 to (3.4) (or equivalently by the well-known 'diagram chase' argument on $\rightarrow^*_\beta$ and $\rightarrow^*_\vartheta$), we may conclude that the *$\beta\vartheta$-reduction* $\rightarrow_{\beta\vartheta} = \rightarrow_\beta \cup \rightarrow_\vartheta$ is Church-Rosser. □

Finally, we establish SN of $\rightarrow_{\beta\vartheta}$, i.e., there is no infinite chain of $\rightarrow_{\beta\vartheta}$:

Theorem 3.2.11 (Strong normalization). *The $\beta\vartheta$-reduction $\rightarrow_{\beta\vartheta}$ is SN.*

Proof. By a slight modification of the proof of strong normalization of the simply-typed λ-calculus in [85]. □

Thus, together with Theorem 3.2.10, it follows that the normal form $nf(\mathsf{M})$ of each term M of FPCF (with respect to $\rightarrow_{\beta\vartheta}$) uniquely exists. Moreover, we have:

Theorem 3.2.12 (Normal forms are values). *The normal form $nf(\mathsf{M})$ of every program M of FPCF is a value.*

Proof. It has been shown in [18] during the proof to show that PCF Böhm trees are closed under composition. □

Therefore, we have shown that the operational semantics $\rightarrow$ is well-defined:

Corollary 3.2.13 (Correctness of the operational semantics). *If $\Gamma \vdash \{\mathsf{M}\}_e : \mathsf{B}$ is a program of FPCF, and $e > 1$ (resp. $e = 1$), then there exists a unique program (resp. value) $\Gamma \vdash \{\mathsf{M}'\}_{e-1} : \mathsf{B}$ that satisfies $\mathsf{M} \rightarrow \mathsf{M}'$.*

Proof. Immediate from Theorems 3.2.8, 3.2.10, 3.2.11 and 3.2.12. □

3.2.3 Dynamic Semantics of Finitary PCF

Next, we present a general recipe to give semantics of FPCF that satisfies the DCP.

Definition 3.2.14 (Structures for FPCF). A ***structure*** for FPCF in a CCBoC $\mathcal{C} = (\mathcal{C}, \mathcal{E})$ is a tuple $\mathcal{S} = (\mathscr{B}, 1, \times, \pi, \Rightarrow, ev, \underline{tt}, \underline{ff}, \vartheta)$ such that:

- $\mathscr{B} \in \mathcal{C}$;
- 1, $(\times, \pi_1, \pi_2)$ and $(\Rightarrow, ev)$ are respectively a β-terminal object, a β-product (with β-projections) and a β-exponential (with β-evaluations) in $\mathcal{C}$;
- $\underline{tt}, \underline{ff} : 1 \to \mathscr{B}$ and $\vartheta : \mathscr{B} \times (\mathscr{B} \times \mathscr{B}) \to \mathscr{B}$ are values in $\mathcal{C}$.

The ***interpretation*** $[\![\_]\!]_{\mathcal{C}}^{\mathcal{S}}$ of FPCF induced by $\mathcal{S}$ in $\mathcal{C}$ assigns an object $[\![\mathsf{A}]\!]_{\mathcal{C}}^{\mathcal{S}} \in \mathcal{C}$ to each type A, an object $[\![\Gamma]\!]_{\mathcal{C}}^{\mathcal{S}} \in \mathcal{C}$ to each context Γ, and a β-morphism $[\![\mathsf{M}]\!]_{\mathcal{C}}^{\mathcal{S}} : [\![\Gamma]\!]_{\mathcal{C}}^{\mathcal{S}} \to [\![\mathsf{B}]\!]_{\mathcal{C}}^{\mathcal{S}}$ to each term $\Gamma \vdash \mathsf{M} : \mathsf{B}$ as follows:

- (Types) $[\![o]\!]_{\mathcal{C}}^{\mathcal{S}} \overset{\text{df.}}{=} \mathscr{B}$ and $[\![\mathsf{A} \Rightarrow \mathsf{B}]\!]_{\mathcal{C}}^{\mathcal{S}} \overset{\text{df.}}{=} [\![\mathsf{A}]\!]_{\mathcal{C}}^{\mathcal{S}} \Rightarrow [\![\mathsf{B}]\!]_{\mathcal{C}}^{\mathcal{S}}$;
- (Contexts) $[\![\boldsymbol{\epsilon}]\!]_{\mathcal{C}}^{\mathcal{S}} \overset{\text{df.}}{=} 1$ and $[\![\Gamma, \mathsf{x} : \mathsf{A}]\!]_{\mathcal{C}}^{\mathcal{S}} \overset{\text{df.}}{=} [\![\Gamma]\!]_{\mathcal{C}}^{\mathcal{S}} \times [\![\mathsf{A}]\!]_{\mathcal{C}}^{\mathcal{S}}$;
- (Terms)

$$[\![\Gamma \vdash \mathsf{tt} : o]\!]_{\mathcal{C}}^{\mathcal{S}} \overset{\text{df.}}{=} \mathcal{E}^{\omega}(!_{[\![\Gamma]\!]_{\mathcal{C}}^{\mathcal{S}}}; \underline{tt})$$
$$[\![\Gamma \vdash \mathsf{ff} : o]\!]_{\mathcal{C}}^{\mathcal{S}} \overset{\text{df.}}{=} \mathcal{E}^{\omega}(!_{[\![\Gamma]\!]_{\mathcal{C}}^{\mathcal{S}}}; \underline{ff})$$
$$[\![\Gamma \vdash \lambda \mathsf{x}.\mathsf{M} : \mathsf{A} \Rightarrow \mathsf{B}]\!]_{\mathcal{C}}^{\mathcal{S}} \overset{\text{df.}}{=} \Lambda_{[\![\Gamma]\!]_{\mathcal{C}}^{\mathcal{S}}, [\![\mathsf{A}]\!]_{\mathcal{C}}^{\mathcal{S}}, [\![\mathsf{B}]\!]_{\mathcal{C}}^{\mathcal{S}}}([\![\Gamma, \mathsf{x} : \mathsf{A} \vdash \mathsf{M} : \mathsf{B}]\!]_{\mathcal{C}}^{\mathcal{S}})$$
$$[\![\Gamma \vdash \mathsf{MN} : \mathsf{B}]\!]_{\mathcal{C}}^{\mathcal{S}} \overset{\text{df.}}{=} \langle [\![\Gamma \vdash \mathsf{M} : \mathsf{A} \Rightarrow \mathsf{B}]\!]_{\mathcal{C}}^{\mathcal{S}}, [\![\Gamma \vdash \mathsf{N} : \mathsf{A}]\!]_{\mathcal{C}}^{\mathcal{S}} \rangle_{[\![\mathsf{A} \Rightarrow \mathsf{B}]\!]_{\mathcal{C}}^{\mathcal{S}}, [\![\mathsf{A}]\!]_{\mathcal{C}}^{\mathcal{S}}}^{[\![\Gamma]\!]_{\mathcal{C}}^{\mathcal{S}}}; ev_{[\![\mathsf{A}]\!]_{\mathcal{C}}^{\mathcal{S}}, [\![\mathsf{B}]\!]_{\mathcal{C}}^{\mathcal{S}}}$$
$$[\![\Gamma \vdash \mathsf{case}(\mathsf{x}\mathbf{V})[\mathsf{W}_1; \mathsf{W}_2] : o]\!]_{\mathcal{C}}^{\mathcal{S}} \overset{\text{df.}}{=} \mathcal{E}^{\omega}(\langle [\![\Gamma \vdash \mathsf{x}\mathbf{V} : o]\!]_{\mathcal{C}}^{\mathcal{S}}, \langle [\![\Gamma \vdash \mathsf{W}_1 : o]\!]_{\mathcal{C}}^{\mathcal{S}}, [\![\Gamma \vdash \mathsf{W}_2 : o]\!]_{\mathcal{C}}^{\mathcal{S}} \rangle \rangle; \vartheta)$$
$$[\![\Gamma \vdash \mathsf{case}(\mathsf{M})[\mathsf{P}_1; \mathsf{P}_2] : o]\!]_{\mathcal{C}}^{\mathcal{S}} \overset{\text{df.}}{=} \mathcal{E}^{\omega}(\langle [\![\Gamma \vdash \mathsf{M} : o]\!]_{\mathcal{C}}^{\mathcal{S}}, \langle [\![\Gamma \vdash \mathsf{P}_1 : o]\!]_{\mathcal{C}}^{\mathcal{S}}, [\![\Gamma \vdash \mathsf{P}_2 : o]\!]_{\mathcal{C}}^{\mathcal{S}} \rangle \rangle; \vartheta)$$

  where $[\![\Gamma \vdash \mathsf{x} : \mathsf{A}]\!]_{\mathcal{C}}^{\mathcal{S}} : [\![\Gamma]\!]_{\mathcal{C}}^{\mathcal{S}} \to [\![\mathsf{A}]\!]_{\mathcal{C}}^{\mathcal{S}}$ (n.b., $\Gamma \vdash \mathsf{x} : \mathsf{A}$ is not a term of FPCF, but we need it for the application $\mathsf{x}\mathbf{V}$) is the obvious (possibly iterated) β-projection.

Moreover, the structure $\mathcal{S}$ is ***standard*** iff it satisfies the following three axioms:

1. The maps $\Lambda_{A,B,C}$ and $\langle \_, \_ \rangle_{A,B}^{C}$ in $\mathcal{C}$ are bijections for each triple $A, B, C \in \mathcal{C}$;
2. Each β-composition that occurs as the interpretation of a term of FPCF is not a value of $\mathcal{C}$;

3. If $\mathsf{\Gamma \vdash L : A \Rightarrow B}$ and $\mathsf{\Gamma \vdash R : A}$ such that $\mathsf{L \to L' \wedge R \to R'}$, $\mathsf{L \equiv L' \wedge R \to R'}$ or $\mathsf{L \to L' \wedge R \equiv R'}$ in FPCF, then:

$$\langle \llbracket \mathsf{L} \rrbracket^{\mathcal{S}}_{\mathcal{C}}, \llbracket \mathsf{R} \rrbracket^{\mathcal{S}}_{\mathcal{C}} \rangle^{\llbracket \mathsf{\Gamma} \rrbracket^{\mathcal{S}}_{\mathcal{C}}}_{\llbracket \mathsf{A \Rightarrow B} \rrbracket^{\mathcal{S}}_{\mathcal{C}}, \llbracket \mathsf{A} \rrbracket^{\mathcal{S}}_{\mathcal{C}}} ; ev_{\llbracket \mathsf{A} \rrbracket^{\mathcal{S}}_{\mathcal{C}}, \llbracket \mathsf{B} \rrbracket^{\mathcal{S}}_{\mathcal{C}}} \neq \langle \llbracket \mathsf{L'} \rrbracket^{\mathcal{S}}_{\mathcal{C}}, \llbracket \mathsf{R'} \rrbracket^{\mathcal{S}}_{\mathcal{C}} \rangle^{\llbracket \mathsf{\Gamma} \rrbracket^{\mathcal{S}}_{\mathcal{C}}}_{\llbracket \mathsf{A \Rightarrow B} \rrbracket^{\mathcal{S}}_{\mathcal{C}}, \llbracket \mathsf{A} \rrbracket^{\mathcal{S}}_{\mathcal{C}}} ; ev_{\llbracket \mathsf{A} \rrbracket^{\mathcal{S}}_{\mathcal{C}}, \llbracket \mathsf{B} \rrbracket^{\mathcal{S}}_{\mathcal{C}}}. \quad (3.5)$$

The interpretation $\llbracket \_ \rrbracket^{\mathcal{S}}_{\mathcal{C}}$ followed by $\mathcal{E}^{\omega}$, i.e., $\mathcal{E}^{\omega}(\llbracket \_ \rrbracket^{\mathcal{S}}_{\mathcal{C}})$, clearly coincides with the standard categorical interpretation of the theory $\mathsf{Eq(FPCF)}$ in the CCC $\mathcal{V}_{\mathcal{C}}$ [118, 150, 41, 102]. In this sense, we have refined the standard semantics of type theories.

At this point, let us recall the DCP (see Section 1.1.4) specifically for FPCF:

Definition 3.2.15 (DCP for FPCF)**.** The interpretation $\llbracket \_ \rrbracket^{\mathcal{S}}_{\mathcal{C}}$ of FPCF induced by a structure $\mathcal{S}$ for FPCF in a CCBoC $\mathcal{C} = (\mathcal{C}, \mathcal{E})$ satisfies the ***dynamic correspondence property (DCP)*** iff for any programs $\mathsf{M_1}$ and $\mathsf{M_2}$ of FPCF we have:

$$\mathsf{M_1 \to M_2} \Rightarrow \llbracket \mathsf{M_1} \rrbracket^{\mathcal{S}}_{\mathcal{C}} \neq \llbracket \mathsf{M_2} \rrbracket^{\mathcal{S}}_{\mathcal{C}} \wedge \mathcal{E}(\llbracket \mathsf{M_1} \rrbracket^{\mathcal{S}}_{\mathcal{C}}) = \llbracket \mathsf{M_2} \rrbracket^{\mathcal{S}}_{\mathcal{C}}.$$

Now, we reduce the DCP for FPCF to the following:

Definition 3.2.16 (PDCP for FPCF)**.** The interpretation $\llbracket \_ \rrbracket^{\mathcal{S}}_{\mathcal{C}}$ of FPCF induced by a structure $\mathcal{S}$ for FPCF in a CCBoC $\mathcal{C} = (\mathcal{C}, \mathcal{E})$ satisfies the ***pointwise dynamic correspondence property (PDCP)*** iff for each term $\mathsf{\Gamma \vdash \{M\}}_e : \mathsf{B}$ it satisfies:

$$\mathcal{E}(\llbracket \mathsf{M} \rrbracket^{\mathcal{S}}_{\mathcal{C}}) = \begin{cases} \Lambda \circ \mathcal{E}(\llbracket \mathsf{P} \rrbracket^{\mathcal{S}}_{\mathcal{C}}) & \text{if } \mathsf{M \equiv \lambda x.P}; \\ \llbracket \mathsf{W} \rrbracket^{\mathcal{S}}_{\mathcal{C}} \text{ such that } \llbracket \mathsf{W} \rrbracket^{\mathcal{S}}_{\mathcal{C}} \neq \llbracket \mathsf{M} \rrbracket^{\mathcal{S}}_{\mathcal{C}} & \text{if } \mathsf{M \equiv UV},\ e = 1 \text{ and } \mathsf{UV \to W}; \\ \langle \mathcal{E}(\llbracket \mathsf{L} \rrbracket^{\mathcal{S}}_{\mathcal{C}}), \mathcal{E}(\llbracket \mathsf{R} \rrbracket^{\mathcal{S}}_{\mathcal{C}}) \rangle ; ev & \text{if } \mathsf{M \equiv LR} \text{ and } e > 1; \\ \llbracket \mathsf{M} \rrbracket^{\mathcal{S}}_{\mathcal{C}} & \text{otherwise.} \end{cases}$$

Theorem 3.2.17 (Standard semantics of FPCF)**.** *The interpretation $\llbracket \_ \rrbracket^{\mathcal{S}}_{\mathcal{C}}$ of FPCF induced by a standard structure $\mathcal{S}$ for FPCF in a CCBoC $\mathcal{C} = (\mathcal{C}, \mathcal{E})$ satisfies the DCP if it satisfies the PDCP.*

Proof. In the following, we abbreviate $\llbracket \_ \rrbracket^{\mathcal{S}}_{\mathcal{C}}$ as $\llbracket \_ \rrbracket$. Assume that $\llbracket \_ \rrbracket$ satisfies the PDCP. We show $\mathsf{M \to M'} \Rightarrow \llbracket \mathsf{M} \rrbracket \neq \llbracket \mathsf{M'} \rrbracket \wedge \mathcal{E}(\llbracket \mathsf{M} \rrbracket) = \llbracket \mathsf{M'} \rrbracket$ for any programs $\mathsf{\Gamma \vdash \{M\}}_e : \mathsf{B}$ and $\mathsf{\Gamma \vdash \{M'\}}_{e'} : \mathsf{B}$ of FPCF by induction on the construction of M:

- If $\mathsf{M \equiv tt}$, $\mathsf{M \equiv ff}$ or $\mathsf{M \equiv case(xV_1V_2 \dots V_k)[W_1; W_2]}$, then there is no term $\mathsf{M'}$ such that $\mathsf{M \to M'}$.

- If $\Gamma \vdash \mathsf{M} \equiv \lambda \mathsf{x}^{\mathsf{A}}.\mathsf{P} : \mathsf{A} \Rightarrow \mathsf{C}$, then we have:

$$\begin{aligned}
\mathsf{M} \to \mathsf{M}' &\Rightarrow \mathsf{M}' \equiv \lambda \mathsf{x}.\mathsf{P}' \wedge \mathsf{P} \to \mathsf{P}' \text{ for some program } \mathsf{P}' \text{ and variable } \mathsf{x} \\
&\Rightarrow \mathsf{M}' \equiv \lambda \mathsf{x}.\mathsf{P}' \wedge [\![\mathsf{P}]\!] \neq [\![\mathsf{P}']\!] \wedge \mathcal{E}([\![\mathsf{P}]\!]) = [\![\mathsf{P}']\!] \text{ for some } \mathsf{P}' \text{ and } \mathsf{x} \\
&\text{(by the induction hypothesis)} \\
&\Rightarrow [\![\mathsf{P}]\!] \neq \Lambda^{-1}([\![\mathsf{M}']\!]) \wedge \mathcal{E}([\![\mathsf{P}]\!]) = \Lambda^{-1}([\![\mathsf{M}']\!]) \\
&\Rightarrow \Lambda^{-1}([\![\mathsf{M}]\!]) \neq \Lambda^{-1}([\![\mathsf{M}']\!]) \wedge \Lambda^{-1} \circ \mathcal{E}([\![\mathsf{M}]\!]) = \mathcal{E} \circ \Lambda^{-1}([\![\mathsf{M}]\!]) = \Lambda^{-1}([\![\mathsf{M}']\!]) \\
&\Rightarrow [\![\mathsf{M}]\!] \neq [\![\mathsf{M}']\!] \wedge \mathcal{E}([\![\mathsf{M}]\!]) = [\![\mathsf{M}']\!].
\end{aligned}$$

- If $\mathsf{M} \equiv \mathsf{LR}$, $\sharp(\mathsf{L}) \geqslant 1$ and $\sharp(\mathsf{R}) \geqslant 1$, then we have:

$$\begin{aligned}
\mathsf{M} \to \mathsf{M}' &\Rightarrow \mathsf{M}' \equiv \mathsf{L}'\mathsf{R}' \wedge \mathsf{L} \to \mathsf{L}' \wedge \mathsf{R} \to \mathsf{R}' \text{ for some programs } \mathsf{L}' \text{ and } \mathsf{R}' \\
&\Rightarrow \mathsf{M}' \equiv \mathsf{L}'\mathsf{R}' \wedge [\![\mathsf{L}]\!] \neq [\![\mathsf{L}']\!] \wedge \mathcal{E}([\![\mathsf{L}]\!]) = [\![\mathsf{L}']\!] \wedge [\![\mathsf{R}]\!] \neq [\![\mathsf{R}']\!] \wedge \mathcal{E}([\![\mathsf{R}]\!]) = [\![\mathsf{R}']\!] \\
&\text{for some } \mathsf{L}' \text{ and } \mathsf{R}' \text{ (by the induction hypothesis)} \\
&\Rightarrow [\![\mathsf{M}']\!] = \langle \mathcal{E}([\![\mathsf{L}]\!]), \mathcal{E}([\![\mathsf{R}]\!]) \rangle; ev \wedge [\![\mathsf{L}]\!] \neq \mathcal{E}([\![\mathsf{L}]\!]) \wedge [\![\mathsf{R}]\!] \neq \mathcal{E}([\![\mathsf{R}]\!]) \\
&\Rightarrow [\![\mathsf{M}']\!] = \mathcal{E}([\![\mathsf{M}]\!]) \wedge \mathcal{E}([\![\mathsf{M}]\!]) = \langle \mathcal{E}([\![\mathsf{L}]\!]), \mathcal{E}([\![\mathsf{R}]\!]) \rangle; ev \neq \langle [\![\mathsf{L}]\!], [\![\mathsf{R}]\!] \rangle; ev = [\![\mathsf{M}]\!] \\
&\Rightarrow [\![\mathsf{M}]\!] \neq [\![\mathsf{M}']\!] \wedge \mathcal{E}([\![\mathsf{M}]\!]) = [\![\mathsf{M}']\!].
\end{aligned}$$

- If $\mathsf{M} \equiv \mathsf{LR}$, $\sharp(\mathsf{L}) = 0$ and $\sharp(\mathsf{R}) \geqslant 1$, then we have:

$$\begin{aligned}
\mathsf{M} \to \mathsf{M}' &\Rightarrow \mathsf{M}' \equiv \mathsf{L}\mathsf{R}' \wedge \mathsf{R} \to \mathsf{R}' \text{ for some program } \mathsf{R}' \\
&\Rightarrow \mathsf{M}' \equiv \mathsf{L}\mathsf{R}' \wedge [\![\mathsf{R}]\!] \neq [\![\mathsf{R}']\!] \wedge \mathcal{E}([\![\mathsf{R}]\!]) = [\![\mathsf{R}']\!] \text{ for some } \mathsf{R}' \\
&\text{(by the induction hypothesis)} \\
&\Rightarrow [\![\mathsf{M}']\!] = \langle [\![\mathsf{L}]\!], \mathcal{E}([\![\mathsf{R}]\!]) \rangle; ev = \mathcal{E}([\![\mathsf{M}]\!]) \wedge [\![\mathsf{R}]\!] \neq \mathcal{E}([\![\mathsf{R}]\!]) \\
&\Rightarrow [\![\mathsf{M}']\!] = \mathcal{E}([\![\mathsf{M}]\!]) \wedge \mathcal{E}([\![\mathsf{M}]\!]) = \langle [\![\mathsf{L}]\!], \mathcal{E}([\![\mathsf{R}]\!]) \rangle; ev \neq \langle [\![\mathsf{L}]\!], [\![\mathsf{R}]\!] \rangle; ev = [\![\mathsf{M}]\!] \\
&\Rightarrow [\![\mathsf{M}]\!] \neq [\![\mathsf{M}']\!] \wedge \mathcal{E}([\![\mathsf{M}]\!]) = [\![\mathsf{M}']\!].
\end{aligned}$$

- If $\mathsf{M} \equiv \mathsf{LR}$, $\sharp(\mathsf{L}) \geqslant 1$ and $\sharp(\mathsf{R}) = 0$, then it is handled similarly to the above case.
- If $\mathsf{M} \equiv \mathsf{LR}$, $\sharp(\mathsf{L}) = 0$ and $\sharp(\mathsf{R}) = 0$, then, by the PDCP, we have:

$$\mathsf{M} \to \mathsf{M}' \Rightarrow [\![\mathsf{M}]\!] \neq [\![\mathsf{M}']\!] \wedge \mathcal{E}([\![\mathsf{M}]\!]) = [\![\mathsf{M}']\!]$$

which completes the proof. □

To summarize the present section, we have defined bicategorical 'universes' of dynamic, intensional computations, viz., CCBoCs, presented the simple programming language FPCF, and given an interpretation of the latter in the former as well as a sufficient condition, namely, the PDCP, for the interpretation to satisfy the DCP. Hence, our research problem of the present chapter has been reduced to giving a standard structure for FPCF in a game-semantic CCBoC that satisfies the PDCP.

3.3 Dynamic Games and Strategies

The main idea of dynamic games and strategies is to introduce the distinction between *internal* and *external* moves to games and strategies (defined in Chapter 2); internal moves constitute 'internal communication' between dynamic strategies, representing *intensionality* of computation, and they are to be *a posteriori* 'hidden' by the *hiding operation*, capturing *dynamics* of computation. Conceptually, external moves are 'official' ones for the underlying game, while internal moves are supposed to be 'invisible' to Opponent because they represent how Player 'internally' computes the next external move.

The present section introduces dynamic games and strategies.

3.3.1 Dynamic Arenas and Legal Positions

Like games in Chapter 2, dynamic games are based on (the 'dynamic generalizations' of) *arenas* and *legal positions.* Let us first introduce these preliminary concepts.

Definition 3.3.1 (Dynamic arenas)**.** A ***dynamic arena*** is a triple

$$G = (M_G, \lambda_G, \vdash_G)$$

such that:

- M_G is a set, whose elements are called ***moves***;
- λ_G is a function $M_G \to \{\mathsf{O}, \mathsf{P}\} \times \{\mathsf{Q}, \mathsf{A}\} \times \mathbb{N}$, called the ***labeling function***, that satisfies $\mu(G) \stackrel{\text{df.}}{=} \mathsf{Sup}(\{\lambda_G^{\mathbb{N}}(m) \mid m \in M_G\}) \in \mathbb{N}$;
- $\vdash_G$ is a relation $\subseteq (\{\star\} \cup M_G) \times M_G$, where $\star$ is an arbitrary element such that $\star \notin M_G$, called the ***enabling relation***, that satisfies:
  - (E1) If $\star \vdash_G m$, then $\lambda_G(m) = \mathsf{OQ}0$ and $n = \star$ whenever $n \vdash_G m$;
  - (E2) If $m \vdash_G n$ and $\lambda_G^{\mathsf{QA}}(n) = \mathsf{A}$, then $\lambda_G^{\mathsf{QA}}(m) = \mathsf{Q}$ and $\lambda_G^{\mathbb{N}}(m) = \lambda_G^{\mathbb{N}}(n)$;

- (E3) If $m \vdash_G n$ and $m \neq \star$, then $\lambda_G^{\mathsf{OP}}(m) \neq \lambda_G^{\mathsf{OP}}(n)$;
- (E4) If $m \vdash_G n$, $m \neq \star$ and $\lambda_G^{\mathbb{N}}(m) \neq \lambda_G^{\mathbb{N}}(n)$, then $\lambda_G^{\mathsf{OP}}(m) = \mathsf{O}$

in which $\lambda_G^{\mathsf{OP}} \overset{\mathrm{df.}}{=} \pi_1 \circ \lambda_G : M_G \to \{\mathsf{O}, \mathsf{P}\}$, $\lambda_G^{\mathsf{QA}} \overset{\mathrm{df.}}{=} \pi_2 \circ \lambda_G : M_G \to \{\mathsf{Q}, \mathsf{A}\}$ and $\lambda_G^{\mathbb{N}} \overset{\mathrm{df.}}{=} \pi_3 \circ \lambda_G : M_G \to \mathbb{N}$. We adopt the convention and notation employed for arenas (Definition 2.2.1) for dynamic arenas too. In addition, a move $m \in M_G$ is called ***internal***, or more specifically called ***$\lambda_G^{\mathbb{N}}(m)$-internal***, (resp. ***external***) if $\lambda_G^{\mathbb{N}}(m) > 0$ (resp. if $\lambda_G^{\mathbb{N}}(m) = 0$). A finite sequence $\boldsymbol{s} \in M_G^*$ of moves is called ***d-complete*** ($d \in \mathbb{N} \cup \{\omega\}$) if it ends with an external or d'-internal move with $d' > d$, where ω denotes the *least transfinite ordinal number*.

Thus, a dynamic arena is an arena equipped with the ***priority order*** $\lambda_G^{\mathbb{N}}$ on moves that satisfies certain axioms; it is called so because it determines the priority order of moves to be 'hidden' by the hiding operations on dynamic games (Definition 3.3.13) and on dynamic strategies (Definition 3.3.34). We need all natural numbers for $\lambda_G^{\mathbb{N}}$, not only the internal/external (I/E) distinction, to define a *step-by-step* execution of the hiding operation later. Conversely, dynamic arenas are generalized arenas: An arena (Definition 2.2.1) is equivalent to a dynamic arena whose moves are all external.

The additional axioms for dynamic arenas G are intuitively natural ones:

- We require a *finite* upper bound $\mu(G)$ of the priority orders as it is conceptually natural and technically necessary for *concatenation* (Definition 3.3.23) to be well-defined as well as for the *hiding operation* (Definition 3.3.13) to terminate;
- The axiom E1 adds the equation $\lambda_G^{\mathbb{N}}(m_0) = 0$ for all $m_0 \in M_G^{\mathsf{Init}} \overset{\mathrm{df.}}{=} \{m \in M_G \mid \star \vdash m\}$ since Opponent cannot 'see' internal moves;
- The second requirement of the axiom E2 states that the priority orders between a 'QA-pair' must be the same for it is intuitively reasonable;
- The additional axiom E4 states that only Player can make a move for a previous move if they have different priority orders for internal moves are 'invisible' to Opponent (as we shall see, if $\lambda_G^{\mathbb{N}}(m_1) = k_1 < k_2 = \lambda_G^{\mathbb{N}}(m_2)$, then after the k_1-many iteration of the hiding operation, m_1 and m_2 become external and internal, respectively, i.e., the I/E-parity of moves is *relative*, which is why E4 is not only concerned with I/E-parity but more fine-grained priority orders).

Let us define ***justifiers*** and ***j-sequences*** of a dynamic arena exactly in the same manner as those of an arena (Definition 2.2.5), and adopt the same notation.

We now consider justifiers, j-sequences and dynamic arenas from the 'external point of view':

Definition 3.3.2 (External justifiers)**.** Let G be a dynamic arena, and assume $\boldsymbol{s} \in \mathscr{J}_G$ and $d \in \mathbb{N} \cup \{\omega\}$. Each non-initial occurrence n in $\boldsymbol{s}$ has a unique sequence of justifiers $m m_1 m_2 \dots m_k n$ ($k \geqslant 0$), i.e., $\mathcal{J}_{\boldsymbol{s}}(n) = m_k$, $\mathcal{J}_{\boldsymbol{s}}(m_k) = m_{k-1}$, ..., $\mathcal{J}_{\boldsymbol{s}}(m_2) = m_1$ and $\mathcal{J}_{\boldsymbol{s}}(m_1) = m$, such that $\lambda_G^{\mathbb{N}}(m) = 0 \vee \lambda_G^{\mathbb{N}}(m) > d$ and $0 < \lambda_G^{\mathbb{N}}(m_i) \leqslant d$ for $i = 1, 2, \dots, k$. We call m the ***d-external justifier*** of n in $\boldsymbol{s}$.

Notation. We write $\mathcal{J}_{\boldsymbol{s}}^{\ominus d}(n)$ for the d-external justifier of n in a j-sequence $\boldsymbol{s}$.

Note that d-external justifiers are a simple generalization of justifiers: 0-external justifiers coincide with justifiers (as there is no '0-internal' move). More generally, d-external justifiers are intended to be justifiers after the d-times iteration of the hiding operation, as we shall see shortly.

Definition 3.3.3 (External j-subsequences)**.** Let G be a dynamic arena, $\boldsymbol{s} \in \mathscr{J}_G$ and $d \in \mathbb{N} \cup \{\omega\}$. The ***d-external j-subsequence*** $\mathcal{H}_G^d(\boldsymbol{s})$ of $\boldsymbol{s}$ is obtained from $\boldsymbol{s}$ by deleting occurrences of internal moves m such that $0 < \lambda_G^{\mathbb{N}}(m) \leqslant d$ and equipping it with the pointers $\mathcal{J}_{\boldsymbol{s}}^{\ominus d}$ (more precisely, $\mathcal{J}_{\mathcal{H}_G^d(\boldsymbol{s})}$ is a *restriction* of $\mathcal{J}_{\boldsymbol{s}}^{\ominus d}$).

Definition 3.3.4 (External dynamic arenas)**.** Let G be a dynamic arena, and $d \in \mathbb{N} \cup \{\omega\}$. The ***d-external dynamic arena*** $\mathcal{H}^d(G)$ of G is given by:

- $M_{\mathcal{H}^d(G)} \stackrel{\text{df.}}{=} \{m \in M_G \mid \lambda_G^{\mathbb{N}}(m) = 0 \vee \lambda_G^{\mathbb{N}}(m) > d\}$;
- $\lambda_{\mathcal{H}^d(G)} \stackrel{\text{df.}}{=} \lambda_G^{\ominus d} \upharpoonright M_{\mathcal{H}^d(G)}$, where $\lambda_G^{\ominus d} \stackrel{\text{df.}}{=} \langle \lambda_G^{\mathsf{OP}}, \lambda_G^{\mathsf{QA}}, n \mapsto \lambda_G^{\mathbb{N}}(n) \ominus d \rangle$, and $n \ominus d \stackrel{\text{df.}}{=} \begin{cases} n - d & \text{if } n \geqslant d; \\ 0 & \text{otherwise} \end{cases}$ for all $n \in \mathbb{N}$;
- $m \vdash_{\mathcal{H}^d(G)} n \stackrel{\text{df.}}{\Leftrightarrow} \exists k \in \mathbb{N}, m_1, m_2, \dots, m_{2k-1}, m_{2k} \in M_G \setminus M_{\mathcal{H}^d(G)}. m \vdash_G m_1 \wedge m_1 \vdash_G m_2 \wedge \dots \wedge m_{2k-1} \vdash_G m_{2k} \wedge m_{2k} \vdash_G n$ ($\Leftrightarrow m \vdash_G n$ if $k = 0$).

Thus, $\mathcal{H}^d(G)$ is obtained from G by deleting internal moves m such that $0 < \lambda_G^{\mathbb{N}}(m) \leqslant d$, decreasing by d the priority orders of the remaining moves and 'concatenating' the enabling relation to form the 'd-external' one.

Convention. Given $d \in \mathbb{N} \cup \{\omega\}$, we regard $\mathcal{H}^d$ as an operation on dynamic arenas G, and $\mathcal{H}_G^d$ as an operation on j-sequences $\boldsymbol{s} \in \mathscr{J}_G$.

Now, let us establish:

Lemma 3.3.5 (External closure lemma). *If G is a dynamic arena, then, for all $d \in \mathbb{N} \cup \{\omega\}$, so is $\mathcal{H}^d(G)$, and $\mathcal{H}^d_G(\boldsymbol{s}) \in \mathscr{J}_{\mathcal{H}^d(G)}$ for all $\boldsymbol{s} \in \mathscr{J}_G$.*

Proof. The case $d = 0$ is trivial; thus, assume $d > 0$. Clearly, the set $M_{\mathcal{H}^d(G)}$ of moves and the labeling function $\lambda_{\mathcal{H}^d(G)}$ are well-defined. Now, let us verify the axioms for the enabling relation $\vdash_{\mathcal{H}^d(G)}$:

- (E1) Note that $\star \vdash_{\mathcal{H}^d(G)} m \Leftrightarrow \star \vdash_G m$ (because $\Leftarrow$ is immediate, and $\Rightarrow$ holds by E4 on G as initial moves are all external). Thus, if $\star \vdash_{\mathcal{H}^d(G)} m$, then $\lambda_{\mathcal{H}^d(G)}(m) = \lambda^{\ominus d}_G(m) = \mathsf{OQ0}$, and $n \vdash_{\mathcal{H}^d(G)} m \Rightarrow n = \star$.

- (E2) Assume $m \vdash_{\mathcal{H}^d(G)} n$ and $\lambda^{\mathsf{QA}}_{\mathcal{H}^d(G)}(n) = \mathsf{A}$. If $m \vdash_G n$, then $\lambda^{\mathsf{QA}}_{\mathcal{H}^d(G)}(m) = \lambda^{\mathsf{QA}}_G(m) = \mathsf{Q}$ and $\lambda^{\mathbb{N}}_{\mathcal{H}^d(G)}(m) = \lambda^{\mathbb{N}}_G(m) \ominus d = \lambda^{\mathbb{N}}_G(n) \ominus d = \lambda^{\mathbb{N}}_{\mathcal{H}^d(G)}(n)$. Otherwise, i.e., there are some $k \in \mathbb{N}^+$ and $m_1, m_2, \dots, m_{2k} \in M_G \setminus M_{\mathcal{H}^d(G)}$ such that $m \vdash_G m_1 \wedge m_1 \vdash_G m_2 \wedge \cdots \wedge m_{2k-1} \vdash_G m_{2k} \wedge m_{2k} \vdash_G n$, then in particular $m_{2k} \vdash_G n$ with $\lambda^{\mathsf{QA}}_G(n) = \mathsf{A}$, but $\lambda^{\mathbb{N}}_G(m_{2k}) \neq \lambda^{\mathbb{N}}_G(n)$, a contradiction.

- (E3) Assume $m \vdash_{\mathcal{H}^d(G)} n$ and $m \neq \star$. If $m \vdash_G n$, then $\lambda^{\mathsf{OP}}_{\mathcal{H}^d(G)}(m) = \lambda^{\mathsf{OP}}_G(m) \neq \lambda^{\mathsf{OP}}_G(n) = \lambda^{\mathsf{OP}}_{\mathcal{H}^d(G)}(n)$. If $m \vdash_G m_1, m_1 \vdash_G m_2, \dots, m_{2k-1} \vdash_G m_{2k}, m_{2k} \vdash_G n$ for some $k \in \mathbb{N}^+$, $m_1, m_2, \dots, m_{2k} \in M_G \setminus M_{\mathcal{H}^d(G)}$, then $\lambda^{\mathsf{OP}}_{\mathcal{H}^d(G)}(m) = \lambda^{\mathsf{OP}}_G(m) = \lambda^{\mathsf{OP}}_G(m_2) = \lambda^{\mathsf{OP}}_G(m_4) = \cdots = \lambda^{\mathsf{OP}}_G(m_{2k}) \neq \lambda^{\mathsf{OP}}_G(n) = \lambda^{\mathsf{OP}}_{\mathcal{H}^d(G)}(n)$.

- (E4) Assume $m \vdash_{\mathcal{H}^d(G)} n$, $m \neq \star$ and $\lambda^{\mathbb{N}}_{\mathcal{H}^d(G)}(m) \neq \lambda^{\mathbb{N}}_{\mathcal{H}^d(G)}(n)$. Then, we have $\lambda^{\mathbb{N}}_G(m) \neq \lambda^{\mathbb{N}}_G(n)$. If $m \vdash_G n$, then it is trivial; otherwise, i.e., there are some $k \in \mathbb{N}^+$, $m_1, m_2, \dots, m_{2k} \in M_G \setminus M_{\mathcal{H}^d(G)}$ with the same property as in E3 above, $\lambda^{\mathsf{OP}}_{\mathcal{H}^d(G)}(m) = \lambda^{\mathsf{OP}}_G(m) = \mathsf{O}$ by E3 on G since $\lambda^{\mathbb{N}}_G(m) \neq \lambda^{\mathbb{N}}_G(m_1)$.

Hence, we have shown that the structure $\mathcal{H}^d(G)$ forms a well-defined dynamic arena.

Next, let $\boldsymbol{s} \in \mathscr{J}_G$; we have to show $\mathcal{H}^d_G(\boldsymbol{s}) \in \mathscr{J}_{\mathcal{H}^d(G)}$. Assume that m is a non-initial occurrence in $\mathcal{H}^d_G(\boldsymbol{s})$. By the definition, the justifier $\mathcal{J}_{\mathcal{H}^d_G(\boldsymbol{s})}(m) = m_0$ occurs in $\mathcal{H}^d_G(\boldsymbol{s})$. If m is a P-move, then the sequence of justifiers $m_0 \vdash_G m_1 \vdash_G \cdots \vdash_G m_k \vdash m$ satisfies $\mathsf{Even}(k)$ by the axioms E3 and E4 on G, so that $m_0 \vdash_{\mathcal{H}^d(G)} m$ by the definition. If m is an O-move, then its justifier $\mathcal{J}_{\boldsymbol{s}}(m) = m'_0$ satisfies $\lambda^{\mathbb{N}}_G(m'_0) = \lambda^{\mathbb{N}}_G(m)$ by the axiom E4 on G, and so $m'_0 \vdash_{\mathcal{H}^d(G)} m$ by the definition. Since m is arbitrary, we have shown that $\mathcal{H}^d_G(\boldsymbol{s}) \in \mathscr{J}_{\mathcal{H}^d(G)}$, completing the proof. □

Next, let us introduce a useful lemma:

Lemma 3.3.6 (Stepwise hiding on dynamic arenas). *Given a dynamic arena G, $\widetilde{\mathcal{H}}^i(G) = \mathcal{H}^i(G)$ for all $i \in \mathbb{N}$, where $\widetilde{\mathcal{H}}^i$ denotes the i-times iteration of $\mathcal{H}^1$.*

Proof. Let G be a dynamic arena. We establish $\widetilde{\mathcal{H}}^i(G) = \mathcal{H}^i(G)$ for all $i \in \mathbb{N}$ by induction on i. The base case $i = 0$ is trivial. For the inductive step $i+1$, note that $\widetilde{\mathcal{H}}^{i+1}(G) = \mathcal{H}^1(\widetilde{\mathcal{H}}^i(G)) = \mathcal{H}^1(\mathcal{H}^i(G))$ by the induction hypothesis; thus, it suffices to show $\mathcal{H}^{i+1}(G) = \mathcal{H}^1(\mathcal{H}^i(G))$. For the sets of moves, we clearly have:

$$\begin{aligned} M_{\mathcal{H}^{i+1}(G)} &= \{ m \in M_G \mid \lambda_G^{\mathbb{N}}(m) = 0 \vee \lambda_G^{\mathbb{N}}(m) > i+1 \} \\ &= \{ m \in M_{\mathcal{H}^i(G)} \mid \lambda_G^{\mathbb{N}}(m) = 0 \vee \lambda_G^{\mathbb{N}}(m) > i+1 \} \\ &= \{ m \in M_{\mathcal{H}^i(G)} \mid \lambda_{\mathcal{H}^i(G)}^{\mathbb{N}}(m) = 0 \vee \lambda_{\mathcal{H}^i(G)}^{\mathbb{N}}(m) > 1 \} \\ &= M_{\mathcal{H}^1(\mathcal{H}^i(G))}. \end{aligned}$$

Next, the labeling functions clearly coincide:

$$\begin{aligned} \lambda_{\mathcal{H}^{i+1}(G)} &= \lambda_G^{\ominus(i+1)} \upharpoonright M_{\mathcal{H}^{i+1}(G)} \\ &= (\lambda_G^{\ominus i} \upharpoonright M_{\mathcal{H}^i(G)})^{\ominus 1} \upharpoonright M_{\mathcal{H}^1(\mathcal{H}^i(G))} \\ &= \lambda_{\mathcal{H}^i(G)}^{\ominus 1} \upharpoonright M_{\mathcal{H}^1(\mathcal{H}^i(G))} \\ &= \lambda_{\mathcal{H}^1(\mathcal{H}^i(G))}. \end{aligned}$$

Finally, for the enabling relations between m and n, if $m = \star$, then it is trivial: $\star \vdash_{\mathcal{H}^{i+1}(G)} n \Leftrightarrow \star \vdash_G n \Leftrightarrow \star \vdash_{\mathcal{H}^i(G)} n \Leftrightarrow \star \vdash_{\mathcal{H}^1(\mathcal{H}^i(G))} n$; thus, assume $m \neq \star$. Then,

$$\begin{aligned} & m \vdash_{\mathcal{H}^{i+1}(G)} n \\ \Leftrightarrow\ & \exists k \in \mathbb{N}, m_1, m_2, \dots, m_{2k} \in M_G \setminus M_{\mathcal{H}^{i+1}(G)}.\, m \vdash_G m_1 \wedge m_1 \vdash_G m_2 \wedge \dots \\ & \wedge m_{2k-1} \vdash_G m_{2k} \wedge m_{2k} \vdash_G n \\ \Leftrightarrow\ & (m \vdash_{\mathcal{H}^i(G)} n) \vee \exists k, l \in \mathbb{N}^+.\, l \leqslant k \wedge \exists m_1, m_2, \dots, m_{2k} \in M_G \setminus M_{\mathcal{H}^{i+1}(G)}, \\ & m_{2j_1-1}, m_{2j_1}, m_{2j_2-1}, m_{2j_2}, \dots, m_{2j_l-1}, m_{2j_l} \in M_{\mathcal{H}^i(G)} \setminus M_{\mathcal{H}^{i+1}(G)}.\, m \vdash_G m_1 \\ & \wedge m_1 \vdash_G m_2 \wedge \dots \wedge m_{2k-1} \vdash_G m_{2k} \wedge m_{2k} \vdash_G n \\ \Leftrightarrow\ & (m \vdash_{\mathcal{H}^i(G)} n) \vee \exists l \in \mathbb{N}^+, m'_1, m'_2, \dots, m'_{2l} \in M_{\mathcal{H}^i(G)} \setminus M_{\mathcal{H}^1(\mathcal{H}^i(G))}.\, m \vdash_{\mathcal{H}^i(G)} m'_1 \\ & \wedge m'_1 \vdash_{\mathcal{H}^i(G)} m'_2 \wedge \dots \wedge m'_{2l-1} \vdash_{\mathcal{H}^i(G)} m'_{2l} \wedge m'_{2l} \vdash_{\mathcal{H}^i(G)} n \\ \Leftrightarrow\ & m \vdash_{\mathcal{H}^1(\mathcal{H}^i(G))} n \end{aligned}$$

where $\mathbb{N}^+ \overset{\mathrm{df.}}{=} \{ n \in \mathbb{N} \mid n > 0 \}$, which completes the proof. □

Thus, we may just focus on $\mathcal{H}^1$: Henceforth, we write $\mathcal{H}$ for $\mathcal{H}^1$ and call it the ***hiding operation*** on arenas; $\mathcal{H}^i$ for each $i \in \mathbb{N}$ denotes the i-times iteration of $\mathcal{H}$.

We may establish a similar inductive property for j-sequences:

Lemma 3.3.7 (Stepwise hiding on j-sequences)**.** *Given a j-sequence $\boldsymbol{s} \in \mathscr{J}_G$ of a dynamic arena G, $\mathcal{H}_G^{i+1}(\boldsymbol{s}) = \mathcal{H}^1_{\mathcal{H}^i(G)}(\mathcal{H}_G^i(\boldsymbol{s}))$ for all $i \in \mathbb{N}$.*

Proof. Note that $\mathcal{H}_G^{i+1}(\boldsymbol{s}), \mathcal{H}_{\mathcal{H}^i(G)}^1(\mathcal{H}_G^i(\boldsymbol{s})) \in \mathscr{J}_{\mathcal{H}^{i+1}(G)}$ by Lemmata 3.3.5 and 3.3.6. We show the equation by induction on $i \in \mathbb{N}$. The base case $i = 0$ is trivial.

Consider the inductive step $i+1$. Recall that $\mathcal{H}_G^{i+1}(\boldsymbol{s})$ is obtained from $\boldsymbol{s}$ by deleting occurrences m with $1 \leqslant \lambda_G^{\mathbb{N}}(m) \leqslant i+1$, equipped with the pointers $\mathcal{J}_{\boldsymbol{s}}^{\ominus(i+1)}$. On the other hand, $\mathcal{H}_{\mathcal{H}^i(G)}^1(\mathcal{H}_G^i(\boldsymbol{s}))$ is obtained from $\mathcal{H}_G^i(\boldsymbol{s})$ by deleting occurrences m with $\lambda_{\mathcal{H}^i(G)}^{\mathbb{N}}(m) = 1$, equipped with the pointers $\mathcal{J}_{\mathcal{H}_G^i(\boldsymbol{s})}^{\ominus 1} = (\mathcal{J}_{\boldsymbol{s}}^{\ominus i})^{\ominus 1} = \mathcal{J}_{\boldsymbol{s}}^{\ominus(i+1)}$. Since $\lambda_{\mathcal{H}^i(G)}^{\mathbb{N}}(m) = 1 \Leftrightarrow \lambda_G^{\mathbb{N}}(m) = i+1$ and $\mathcal{H}_G^i(\boldsymbol{s})$ is obtained from $\boldsymbol{s}$ by deleting occurrences m with $1 \leqslant \lambda_G^{\mathbb{N}}(m) \leqslant i$, they are in fact the same j-sequence of $\mathcal{H}^{i+1}(G)$. □

Lemma 3.3.7 implies that the equation

$$\mathcal{H}_G^i(\boldsymbol{s}) = \mathcal{H}_{\mathcal{H}^{i-1}(G)}^1 \circ \mathcal{H}_{\mathcal{H}^{i-2}(G)}^1 \circ \dots \circ \mathcal{H}_{\mathcal{H}^1(G)}^1 \circ \mathcal{H}_G^1(\boldsymbol{s}) \tag{3.6}$$

holds for any dynamic arena G, $\boldsymbol{s} \in \mathscr{J}_G$ and $i \in \mathbb{N}$ (n.b., the equation (3.6) means $\boldsymbol{s} = \boldsymbol{s}$ if $i = 0$). Thus, we may focus on the operation $\mathcal{H}_G^1$ on j-sequences of G (n.b., we do not need $\mathcal{H}_G^\omega$ as j-sequences are *finite*). Henceforth, we write $\mathcal{H}_G$ for $\mathcal{H}_G^1$ and call it the ***hiding operation*** on j-sequences of G; $\mathcal{H}_G^i$ for each $i \in \mathbb{N}$ denotes the operation on the right-hand side of (3.6).

However, to deal with external j-subsequences in a rigorous manner, we need to extend the hiding operation on j-sequences to j-subsequences (Definition 2.2.6):

Definition 3.3.8 (Point-wise hiding on j-sequences)**.** Let $\boldsymbol{s} \in \mathscr{J}_G$ be a j-sequence of a dynamic arena G. The ***point-wise hiding operation*** $\widehat{\mathcal{H}}_G^{\boldsymbol{s}}$ on each occurrence m and pointers to m in $\boldsymbol{s}$ is defined by:

$$\widehat{\mathcal{H}}_G^{\boldsymbol{s}}(m) \stackrel{\text{df.}}{=} \begin{cases} \boldsymbol{\epsilon} \text{ with pointers to } m \text{ changed to pointing } \mathcal{J}_{\boldsymbol{s}}(m) & \text{if } m \text{ is 1-internal;} \\ m \text{ with pointers to } m \text{ unchanged} & \text{otherwise.} \end{cases}$$

Given a j-subsequence $\boldsymbol{t} = m_1 m_2 \dots m_k$ of $\boldsymbol{s}$, $\widehat{\mathcal{H}}_G^{\boldsymbol{s}}(\boldsymbol{t})$ is defined to be the result of applying $\widehat{\mathcal{H}}_G^{\boldsymbol{s}}$ to m_i for $i = 1, 2, \dots, k$.

Note that the point-wise hiding operation $\widehat{\mathcal{H}}_G^{\boldsymbol{s}}$ makes sense only in the context of $\boldsymbol{s}$; it affects some part of $\boldsymbol{s}$. The point here is that the hiding operation on j-sequences can be executed in the 'point-wise' fashion (in any order):

Lemma 3.3.9 (Point-wise lemma for hiding on j-sequences)**.** *Given a j-sequence $\boldsymbol{s} \in \mathscr{J}_G$ of a dynamic arena G, $\mathcal{H}_G(\boldsymbol{s}) = \widehat{\mathcal{H}}_G^{\boldsymbol{s}}(\boldsymbol{s})$.*

Proof. It suffices to establish, for each $\boldsymbol{s} = m_1 m_2 \dots m_k \in \mathscr{J}_G$, the equation

$$\mathcal{H}_G(\boldsymbol{s}) = \widehat{\mathcal{H}}_G^{\boldsymbol{s}}(m_1)\widehat{\mathcal{H}}_G^{\boldsymbol{s}}(m_2)\dots\widehat{\mathcal{H}}_G^{\boldsymbol{s}}(m_k).$$

First, it is clear by the definition that $\mathcal{H}_G(\boldsymbol{s})$ and $\widehat{\mathcal{H}}_G^{\boldsymbol{s}}(m_1)\widehat{\mathcal{H}}_G^{\boldsymbol{s}}(m_2)\dots\widehat{\mathcal{H}}_G^{\boldsymbol{s}}(m_k)$ are both the subsequence of $\boldsymbol{s}$ obtained from $\boldsymbol{s}$ by deleting 1-internal moves. Thus, it suffices to show that each move m in $\widehat{\mathcal{H}}_G^{\boldsymbol{s}}(m_1)\widehat{\mathcal{H}}_G^{\boldsymbol{s}}(m_2)\dots\widehat{\mathcal{H}}_G^{\boldsymbol{s}}(m_k)$ points to $\mathcal{J}_{\boldsymbol{s}}^{\ominus 1}(m)$. Now, let m be any non-1-internal move in $\boldsymbol{s}$. For the pointer from m in $\widehat{\mathcal{H}}_G^{\boldsymbol{s}}(m_1)\widehat{\mathcal{H}}_G^{\boldsymbol{s}}(m_2)\dots\widehat{\mathcal{H}}_G^{\boldsymbol{s}}(m_k)$, it suffices to consider the subsequence $nn_1n_2\dots n_lm$ of $\boldsymbol{s}$, where $n_1, n_2, \dots, n_l$ are 1-internal but n is not, satisfying $\mathcal{J}_{\boldsymbol{s}}(m) = n_l, \mathcal{J}_{\boldsymbol{s}}(n_l) = n_{l-1}, \dots, \mathcal{J}_{\boldsymbol{s}}(n_2) = n_1, \mathcal{J}_{\boldsymbol{s}}(n_1) = n$ since the operation on the other moves will not affect the pointer from m. Applying $\widehat{\mathcal{H}}_G^{\boldsymbol{s}}$ to $n_1, n_2, \dots, n_l$ in any order, the resulting pointer from m clearly points to n, which is $\mathcal{J}_{\boldsymbol{s}}^{\ominus 1}(m)$. □

By virtue of Lemma 3.3.9, we may identify the operations $\mathcal{H}_G$ and $\widehat{\mathcal{H}}_G^{s}$; thus, from now on, we shall not notationally distinguish them, and use only the former. As a result, what we have established is the 'point-wise' procedure to execute the hiding operation on j-sequences, in which the order of moves to apply the 'point-wise' operation is irrelevant. In particular, $\mathcal{H}_G^d(\boldsymbol{st}) = \mathcal{H}_G^d(\boldsymbol{s})\mathcal{H}_G^d(\boldsymbol{t})$ for any dynamic arena G, $d \in \mathbb{N} \cup \{\omega\}$ and $\boldsymbol{st} \in \mathscr{J}_G$, which will be useful in the rest of the chapter.

We are now ready to introduce a 'dynamic generalization' of legal positions:

Definition 3.3.10 (Dynamic legal positions)**.** Given a dynamic arena G, a ***dynamic legal position*** of G is a j-sequence $\boldsymbol{s} \in \mathscr{J}_G$ that satisfies:

- (Alternation) If $\boldsymbol{s} = \boldsymbol{s_1} mn \boldsymbol{s_2}$, then $\lambda_G^{\mathsf{OP}}(m) \neq \lambda_G^{\mathsf{OP}}(n)$;
- (Generalized Visibility) If $\boldsymbol{s} = \boldsymbol{t} m \boldsymbol{u}$ with m non-initial, and $d \in \mathbb{N} \cup \{\omega\}$ satisfy $\lambda_G^{\mathbb{N}}(m) = 0 \vee \lambda_G^{\mathbb{N}}(m) > d$, then $\mathcal{J}_{\boldsymbol{s}}^{\ominus d}(m)$ occurs in $\lceil \mathcal{H}_G^d(\boldsymbol{t}) \rceil_{\mathcal{H}^d(G)}$ if m is a P-move, and it occurs in $\lfloor \mathcal{H}_G^d(\boldsymbol{t}) \rfloor_{\mathcal{H}^d(G)}$ if m is an O-move;
- (IE-Switch) If $\boldsymbol{s} = \boldsymbol{s_1} mn \boldsymbol{s_2}$ with $\lambda_G^{\mathbb{N}}(m) \neq \lambda_G^{\mathbb{N}}(n)$, then m is an O-move.

Notation. $\mathscr{L}_G$ denotes the set of all dynamic legal positions of a dynamic arena G.

The additional axioms on dynamic legal positions are conceptually natural ones:

- Generalized visibility is a generalization of visibility; it requires that visibility holds after any iteration of the hiding operation on j-sequences;

- IE-switch states that only Player can change a priority order during a play as internal moves are 'invisible' to Opponent, where same remark as in the axiom E4 is applied for the finer distinction of priority orders than the I/E-parity.

A dynamic legal position of an *arena* (Definition 2.2.1), seen as a dynamic arena whose moves are all external, is clearly a legal position. Hence, dynamic legal positions are a generalization of arenas.

3.3.2 Dynamic Games

We are now ready to define the central notion of *dynamic games*:

Definition 3.3.11 (Dynamic games)**.** A ***dynamic game*** is a quintuple

$$G = (M_G, \lambda_G, \vdash_G, P_G, \simeq_G)$$

such that:

- The triple $(M_G, \lambda_G, \vdash_G)$ forms a dynamic arena (Definition 3.3.1);
- P_G is a subset of $\mathscr{L}_G$, whose elements are called ***(valid) positions*** of G, that satisfies:
  - (P1) P_G is non-empty and prefix-closed;
  - (DP2) If $\boldsymbol{s}mn \in P_G^{\mathsf{Even}}$ and $\lambda_G^{\mathbb{N}}(n) > 0$, then $\exists r \in M_G.\, \boldsymbol{s}mnr \in P_G$;
  - (DP3) Given $\boldsymbol{t}r, \boldsymbol{t'}r' \in P_G^{\mathsf{Odd}}$ and $i \in \mathbb{N}$ such that $i < \lambda_G^{\mathbb{N}}(r) = \lambda_G^{\mathbb{N}}(r')$, if $\mathcal{H}_G^i(\boldsymbol{t}) = \mathcal{H}_G^i(\boldsymbol{t'})$, then $\mathcal{H}_G^i(\boldsymbol{t}r) = \mathcal{H}_G^i(\boldsymbol{t'}r')$;
- $\simeq_G$ is an equivalence relation on P_G, called the ***identification of (valid) positions***, that satisfies:
  - (I1) $\boldsymbol{s} \simeq_G \boldsymbol{t} \Rightarrow |\boldsymbol{s}| = |\boldsymbol{t}|$;
  - (I2) $\boldsymbol{s}m \simeq_G \boldsymbol{t}n \Rightarrow \boldsymbol{s} \simeq_G \boldsymbol{t} \wedge \lambda_G(m) = \lambda_G(n) \wedge (m, n \in M_G^{\mathsf{Init}} \vee (\exists i \in \{1, 2, \dots, |\boldsymbol{s}|\}.\, \mathcal{J}_{\boldsymbol{s}m}(m) = s_i \wedge \mathcal{J}_{\boldsymbol{t}n}(n) = t_i))$;
  - (DI3) $\forall d \in \mathbb{N} \cup \{\omega\}.\, \boldsymbol{s} \simeq_G^d \boldsymbol{t} \wedge \boldsymbol{s}m \in P_G \Rightarrow \exists \boldsymbol{t}n \in P_G.\, \boldsymbol{s}m \simeq_G^d \boldsymbol{t}n$, where $\boldsymbol{u} \simeq_G^d \boldsymbol{v} \overset{\mathrm{df.}}{\Leftrightarrow} \exists \boldsymbol{u'}, \boldsymbol{v'} \in P_G.\, \boldsymbol{u'} \simeq_G \boldsymbol{v'} \wedge \mathcal{H}_G^d(\boldsymbol{u'}) = \mathcal{H}_G^d(\boldsymbol{u}) \wedge \mathcal{H}_G^d(\boldsymbol{v'}) = \mathcal{H}_G^d(\boldsymbol{v})$ for all $\boldsymbol{u}, \boldsymbol{v} \in P_G$.

A ***play*** of G is an finitely or infinitely increasing sequence of positions $\boldsymbol{\epsilon}, m_1, m_1 m_2, \dots$ of G. A dynamic game whose moves are all external is said to be ***normalized***.

Thus, a dynamic game may be seen as a game (Definition 2.2.10) that satisfies the additional axioms DP2, DP3 and DI3, where DI3 is clearly a generalization of I3 (n.b., what should be thought of as P2 (resp. P3) is 'vacant', and thus it is not required on games). Conversely, games are equivalent to normalized dynamic games.

The axioms DP2 and DP3 are in order to enable Player to 'play alone', i.e., Opponent does not have to choose odd-length positions, for the internal part of a play since conceptually Opponent cannot 'see' internal moves; technically, the axiom DP2 is to preserve totality of dynamic strategies under the hiding operation (Corollary 3.3.40), and the axiom DP3 is for *external consistency* of dynamic strategies: A dynamic strategy behaves always in the same manner from the viewpoint of Opponent, i.e., the external part of a play by a dynamic strategy does not depend on the internal part (Theorem 3.3.33). Note that the axiom DP2 is slightly involved to be preserved under the hiding operation (Theorem 3.3.14); it is necessary to generalize the axiom I3 to the axiom DI3 for the same reason.

It is clear that we may just apply the definitions of ***economy***, ***well-openness*** and ***well-foundedness*** of games to dynamic games.

Convention. Henceforth, ***dynamic games*** refer to *economical* ones by default.

Also, it is natural to define:

Definition 3.3.12 (Dynamic subgames)**.** Given dynamic games G and H, we say that H is a ***dynamic subgame*** of G, written $H \trianglelefteq G$, iff $M_H \subseteq M_G$, $\lambda_H = \lambda_G \upharpoonright M_H$, $\vdash_H \subseteq \vdash_G \cap ((\{\star\} \cup M_H) \times M_H)$, $P_H \subseteq P_G$, $\forall d \in \mathbb{N} \cup \{\omega\}.\ \simeq_H^d = \simeq_G^d \cap (P_H \times P_H)$ and $\mu(H) = \mu(G)$.

For $H \trianglelefteq G$, the condition on the identifications of positions is required *for all* $d \in \mathbb{N} \cup \{\omega\}$ so that the dynamic subgame relation $\trianglelefteq$ is preserved under the hiding operation (Theorem 3.3.14); the last condition $\mu(H) = \mu(G)$ is to preserve the relation $\trianglelefteq$ under *concatenation* of dynamic games (Definition 3.3.23).

Now, let us define the *hiding operation* on dynamic games:

Definition 3.3.13 (Hiding operation on dynamic games)**.** For each $d \in \mathbb{N} \cup \{\omega\}$, the ***d-hiding operation (on dynamic games)*** maps each dynamic game G to the ***d-external dynamic game*** $\mathcal{H}^d(G)$ of G given by:

- The triple $(M_{\mathcal{H}^d(G)}, \lambda_{\mathcal{H}^d(G)}, \vdash_{\mathcal{H}^d(G)})$ is the d-external dynamic arena $\mathcal{H}^d(G)$ of G (Definition 3.3.4);
- $P_{\mathcal{H}^d(G)} \overset{\text{df.}}{=} \{\mathcal{H}^d_G(\boldsymbol{s}) \mid \boldsymbol{s} \in P_G\}$;

- $\mathcal{H}^d_G(\boldsymbol{s}) \simeq_{\mathcal{H}^d(G)} \mathcal{H}^d_G(\boldsymbol{t}) \overset{\text{df.}}{\Leftrightarrow} \boldsymbol{s} \simeq^d_G \boldsymbol{t}$.

Now, we give the first main theorem of the present chapter:

Theorem 3.3.14 (External closure of dynamic games). *Given $d \in \mathbb{N} \cup \{\omega\}$, dynamic games are closed under the operation $\mathcal{H}^d$, and $H \trianglelefteqslant G$ implies $\mathcal{H}^d(H) \trianglelefteqslant \mathcal{H}^d(G)$.*

Proof. Let G be a dynamic game, and assume $d \in \mathbb{N} \cup \{\omega\}$; we have to show that $\mathcal{H}^d(G)$ is a dynamic game. By Lemma 3.3.5, it suffices to show that j-sequences in $P_{\mathcal{H}^d(G)}$ are dynamic legal positions of the arena $\mathcal{H}^d(G)$, the set $P_{\mathcal{H}^d(G)}$ satisfies the axioms P1, DP2 and DP3, and the relation $\simeq_{\mathcal{H}^d(G)}$ is an equivalence relation on $P_{\mathcal{H}^d(G)}$ that satisfies the axioms I1, I2 and DI3. Since $\mu(G) \in \mathbb{N}$, we may assume $d \in \mathbb{N}$.

For alternation, assume $\boldsymbol{s_1} mn \boldsymbol{s_2} \in P_{\mathcal{H}^d(G)}$; we have to show $\lambda^{\mathsf{OP}}_{\mathcal{H}^d(G)}(m) \neq \lambda^{\mathsf{OP}}_{\mathcal{H}^d(G)}(n)$. We have $\mathcal{H}^d_G(\boldsymbol{t}_1 m m_1 m_2 \dots m_k n \boldsymbol{t}_2) = \boldsymbol{s_1} mn \boldsymbol{s_2}$ for some $\boldsymbol{t}_1 m m_1 m_2 \dots m_k n \boldsymbol{t_2} \in P_G$, where $\mathcal{H}^d_G(\boldsymbol{t_1}) = \boldsymbol{s_1}$, $\mathcal{H}^d_G(\boldsymbol{t_2}) = \boldsymbol{s_2}$ and $\mathcal{H}^d_G(m_1 m_2 \dots m_k) = \boldsymbol{\epsilon}$. Note that $(\lambda^{\mathbb{N}}_G(m) = 0 \vee \lambda^{\mathbb{N}}_G(m) > d) \wedge (\lambda^{\mathbb{N}}_G(n) = 0 \vee \lambda^{\mathbb{N}}_G(n) > d)$ and $0 < \lambda^{\mathbb{N}}_G(m_i) \leqslant d$ for $i = 1, 2, \dots, k$. By E3 and E4 on G, k must be an even number, and thus $\lambda^{\mathsf{OP}}_{\mathcal{H}^d(G)}(m) = \lambda^{\mathsf{OP}}_G(m) = \lambda^{\mathsf{OP}}_G(m_2) = \lambda^{\mathsf{OP}}_G(m_4) = \dots = \lambda^{\mathsf{OP}}_G(m_k) \neq \lambda^{\mathsf{OP}}_G(n) = \lambda^{\mathsf{OP}}_{\mathcal{H}^d(G)}(n)$.

For generalized visibility, let $\boldsymbol{t} m \boldsymbol{u} \in P_{\mathcal{H}^d(G)}$ with m non-initial. We have to show, for each $e \in \mathbb{N} \cup \{\omega\}$, that if $\boldsymbol{t}m$ is e-complete, then:

- if m is a P-move, then the justifier $(\mathcal{J}^{\ominus d}_{\boldsymbol{s}})^{\ominus e}(m)$ occurs in $\lceil \mathcal{H}^e_{\mathcal{H}^d(G)}(\boldsymbol{t}) \rceil_{\mathcal{H}^e(\mathcal{H}^d(G))}$;
- if m is an O-move, then the justifier $(\mathcal{J}^{\ominus d}_{\boldsymbol{s}})^{\ominus e}(m)$ occurs in $\lfloor \mathcal{H}^e_{\mathcal{H}^d(G)}(\boldsymbol{t}) \rfloor_{\mathcal{H}^e(\mathcal{H}^d(G))}$.

Again, for $\mu(G) \in \mathbb{N}$, we may assume without loss of generality that $e \in \mathbb{N}$. Note that the condition is then equivalent to:

- if m is a P-move, then the justifier $\mathcal{J}^{\ominus(d+e)}_{\boldsymbol{s}}(m)$ occurs in $\lceil \mathcal{H}^{d+e}_G(\boldsymbol{t'}) \rceil_{\mathcal{H}^{d+e}(G)}$;
- if m is an O-move, then the justifier $\mathcal{J}^{\ominus(d+e)}_{\boldsymbol{s}}(m)$ occurs in $\lfloor \mathcal{H}^{d+e}_G(\boldsymbol{t'}) \rfloor_{\mathcal{H}^{d+e}(G)}$

where $\boldsymbol{t'} m \in P_G$ such that $\mathcal{H}^d_G(\boldsymbol{t'} m) = \boldsymbol{t} m$. It holds by generalized visibility on G.

For IE-switch, let $\boldsymbol{s_1} mn \boldsymbol{s_2} \in P_{\mathcal{H}^d(G)}$ such that $\lambda^{\mathbb{N}}_{\mathcal{H}^d(G)}(m) \neq \lambda^{\mathbb{N}}_{\mathcal{H}^d(G)}(n)$. Then, there is some $\boldsymbol{t_1} m \boldsymbol{u} n \boldsymbol{t_2} \in P_G$ such that $\mathcal{H}^d_G(\boldsymbol{t_1} m \boldsymbol{u} n \boldsymbol{t_2}) = \boldsymbol{s_1} mn \boldsymbol{s_2}$, where note that $\lambda^{\mathbb{N}}_G(m) \neq \lambda^{\mathbb{N}}_G(n)$. Therefore, if $\boldsymbol{u} = \boldsymbol{\epsilon}$, then we clearly have $\lambda^{\mathsf{OP}}_{\mathcal{H}^d(G)}(m) = \mathsf{O}$ by IE-switch on G; otherwise, i.e., $\boldsymbol{u} = l\boldsymbol{u'}$, then we have the same conclusion as $\lambda^{\mathbb{N}}_G(m) \neq \lambda^{\mathbb{N}}_G(l)$.

We have established $P_{\mathcal{H}^d(G)} \subseteq \mathscr{L}_{\mathcal{H}^d(G)}$. Next, we verify P1, DP2 and DP3:

- (P1) Because $\boldsymbol{\epsilon} \in P_G$, we have $\boldsymbol{\epsilon} = \mathcal{H}^d_G(\boldsymbol{\epsilon}) \in P_{\mathcal{H}^d(G)}$; thus, $P_{\mathcal{H}^d(G)}$ is non-empty. For prefix-closure, let $\boldsymbol{s} m \in P_{\mathcal{H}^d(G)}$; we have to show $\boldsymbol{s} \in P_{\mathcal{H}^d(G)}$. There must be some $\boldsymbol{t} m \in P_G$ such that $\boldsymbol{s} m = \mathcal{H}^d_G(\boldsymbol{t} m) = \mathcal{H}^d_G(\boldsymbol{t}) m$. Thus, $\boldsymbol{s} = \mathcal{H}^d_G(\boldsymbol{t}) \in P_{\mathcal{H}^d(G)}$.

- (DP2) If $\boldsymbol{s}mn \in P^{\mathsf{Even}}_{\mathcal{H}^d(G)}$ and $\lambda^{\mathbb{N}}_{\mathcal{H}^d(G)}(n) > 0$, then there is some $\boldsymbol{t}m\boldsymbol{u}n \in P^{\mathsf{Even}}_G$ such that $\mathcal{H}^d_G(\boldsymbol{t}m\boldsymbol{u}n) = \boldsymbol{s}mn$ and $\lambda^{\mathbb{N}}_G(n) > d > 0$. Hence, by DP2 on G, there is some $\boldsymbol{t}m\boldsymbol{u}nr \in P_G$ such that $\lambda^{\mathbb{N}}_G(r) = \lambda^{\mathbb{N}}_G(n) > d$ by IE-switch on G. Therefore, we have found $\boldsymbol{s}mnr = \mathcal{H}^d(\boldsymbol{t}m\boldsymbol{u}nr) \in P_{\mathcal{H}^d(G)}$, establishing DP2 on $\mathcal{H}^d(G)$.

- (DP3) Assume $\boldsymbol{t}r, \boldsymbol{t'}r' \in P^{\mathsf{Odd}}_{\mathcal{H}^d(G)}$ and $i \in \mathbb{N}$ such that $i < \lambda^{\mathbb{N}}_{\mathcal{H}^d(G)}(r) = \lambda^{\mathbb{N}}_{\mathcal{H}^d(G)}(r')$ and $\mathcal{H}^i_{\mathcal{H}^d(G)}(\boldsymbol{t}) = \mathcal{H}^i_{\mathcal{H}^d(G)}(\boldsymbol{t'})$. We have some $\boldsymbol{u}r, \boldsymbol{u'}r' \in P_G$ with $\mathcal{H}^d_G(\boldsymbol{u}) = \boldsymbol{t}$ and $\mathcal{H}^d_G(\boldsymbol{u'}) = \boldsymbol{t'}$. Then, $\mathcal{H}^{d+i}_G(\boldsymbol{u}) = \mathcal{H}^i_{\mathcal{H}^d(G)}(\mathcal{H}^d_G(\boldsymbol{u})) = \mathcal{H}^i_{\mathcal{H}^d(G)}(\boldsymbol{t}) = \mathcal{H}^i_{\mathcal{H}^d(G)}(\boldsymbol{t'}) = \mathcal{H}^i_{\mathcal{H}^d(G)}(\mathcal{H}^d_G(\boldsymbol{u'})) = \mathcal{H}^{d+i}_G(\boldsymbol{u'})$. Hence, by DP3 on G, $r = r'$ and $\mathcal{J}^{\ominus i}_{\boldsymbol{t}r}(r) = \mathcal{J}^{\ominus(d+i)}_{\boldsymbol{u}r}(r) = \mathcal{J}^{\ominus(d+i)}_{\boldsymbol{u'}r'}(r') = \mathcal{J}^{\ominus i}_{\boldsymbol{t'}r'}(r')$, establishing DP3 on $\mathcal{H}^d(G)$.

Next, $\simeq_{\mathcal{H}^d(G)}$ is a well-defined relation on $P_{\mathcal{H}^d(G)}$ since $\mathcal{H}^d_G(\boldsymbol{s}) \simeq_{\mathcal{H}^d(G)} \mathcal{H}^d_G(\boldsymbol{t})$ does not depend on the choice of representatives $\boldsymbol{s}, \boldsymbol{t} \in P_G$. Also, it is straightforward to see that $\simeq_{\mathcal{H}^d(G)}$ is an equivalence relation. Now, we show that $\simeq_{\mathcal{H}^d(G)}$ satisfies I1, I2 and DI3. Note that I1 and I2 on $\simeq_{\mathcal{H}^d(G)}$ immediately follow from those on $\simeq_G$. For DI3 on $\simeq_{\mathcal{H}^d(G)}$, if $\mathcal{H}^d_G(\boldsymbol{s}) \simeq^e_{\mathcal{H}^d(G)} \mathcal{H}^d_G(\boldsymbol{t})$, and $\mathcal{H}^d_G(\boldsymbol{s}).m \in P_{\mathcal{H}^d(G)}$, where we may assume $e \neq \omega$, then $\exists \boldsymbol{s'}m \in P_G. \mathcal{H}^d_G(\boldsymbol{s'}m) = \mathcal{H}^d_G(\boldsymbol{s}).m$, and so $\mathcal{H}^{d+e}_G(\boldsymbol{s'}) = \mathcal{H}^{d+e}_G(\boldsymbol{s}) \simeq_{\mathcal{H}^{d+e}(G)} \mathcal{H}^{d+e}_G(\boldsymbol{t})$. By DI3 on $\simeq_G$, we may conclude that $\exists \boldsymbol{t}n \in P_G. \boldsymbol{s'}m \simeq^{d+e}_G \boldsymbol{t}n$, whence we obtain $\mathcal{H}^d_G(\boldsymbol{t}).n \in P_{\mathcal{H}^d(G)}$ such that $\mathcal{H}^d_G(\boldsymbol{s}).m = \mathcal{H}^d_G(\boldsymbol{s'}m) \simeq^e_{\mathcal{H}^d(G)} \mathcal{H}^d_G(\boldsymbol{t}n) = \mathcal{H}^d_G(\boldsymbol{t}).n$.

Finally, the preservation of the dynamic subgame relation $\trianglelefteq$ under the operation $\mathcal{H}^d$ is clear from the definition, completing the proof. □

Corollary 3.3.15 (Stepwise hiding on dynamic games)**.** *For any dynamic game G, $\mathcal{H}^1(\mathcal{H}^i(G)) = \mathcal{H}^{i+1}(G)$ for all $i \in \mathbb{N}$.*

Proof. By Lemmata 3.3.6 and 3.3.7, it suffices to show $\simeq_{\mathcal{H}^1(\mathcal{H}^i(G))} = \simeq_{\mathcal{H}^{i+1}(G)}$. Then, given $\boldsymbol{s}, \boldsymbol{t} \in P_G$, we have:

$$\begin{aligned}
&\Leftrightarrow \mathcal{H}^1_{\mathcal{H}^i(G)}(\mathcal{H}^i_G(\boldsymbol{s})) \simeq_{\mathcal{H}^1(\mathcal{H}^i(G))} \mathcal{H}^1_{\mathcal{H}^i(G)}(\mathcal{H}^i_G(\boldsymbol{t})) \\
&\Leftrightarrow \exists \mathcal{H}^i(\boldsymbol{s'}), \mathcal{H}^i(\boldsymbol{t'}) \in P_{\mathcal{H}^i(G)}. \mathcal{H}^i(\boldsymbol{s'}) \simeq_{\mathcal{H}^i(G)} \mathcal{H}^i(\boldsymbol{t'}) \wedge \mathcal{H}^1_{\mathcal{H}^i(G)}(\mathcal{H}^i(\boldsymbol{s'})) = \mathcal{H}^1_{\mathcal{H}^i(G)}(\mathcal{H}^i(\boldsymbol{s})) \\
&\qquad \wedge \mathcal{H}^1_{\mathcal{H}^i(G)}(\mathcal{H}^i(\boldsymbol{t'})) = \mathcal{H}^1_{\mathcal{H}^i(G)}(\mathcal{H}^i(\boldsymbol{t})) \\
&\Leftrightarrow \exists \boldsymbol{s''}, \boldsymbol{t''} \in P_G. \boldsymbol{s''} \simeq_G \boldsymbol{t''} \wedge \mathcal{H}^{i+1}_G(\boldsymbol{s''}) = \mathcal{H}^{i+1}_G(\boldsymbol{s}) \wedge \mathcal{H}^{i+1}_G(\boldsymbol{t''}) = \mathcal{H}^{i+1}_G(\boldsymbol{t}) \\
&\Leftrightarrow \mathcal{H}^{i+1}_G(\boldsymbol{s}) \simeq_{\mathcal{H}^{i+1}(G)} \mathcal{H}^{i+1}_G(\boldsymbol{t})
\end{aligned}$$

completing the proof. □

By the corollary, we may just focus on $\mathcal{H}^1$:

Convention. We write $\mathcal{H}$ for $\mathcal{H}^1$ and call it the ***hiding operation (on dynamic games)***; $\mathcal{H}^i$ denotes the i-times iteration of $\mathcal{H}$ for all $i \in \mathbb{N}$.

Corollary 3.3.16 (Hiding operation on dynamic legal positions)**.** *Given a dynamic arena G and a number $d \in \mathbb{N} \cup \{\omega\}$, we have $\{\mathcal{H}^d_G(\boldsymbol{s}) \mid \boldsymbol{s} \in \mathscr{L}_G\} = \mathscr{L}_{\mathcal{H}^d(G)}$.*

Proof. Since there is an upper bound $\mu(G) \in \mathbb{N}$, it suffices to consider the case $d \in \mathbb{N}$. Then, by Lemmata 3.3.6 and 3.3.7, we may just focus on the case $d = 1$.

The inclusion $\{\mathcal{H}_G(\boldsymbol{s}) \mid \boldsymbol{s} \in \mathscr{L}_G\} \subseteq \mathscr{L}_{\mathcal{H}(G)}$ is immediate by Theorem 3.3.14. For the other inclusion, let $\boldsymbol{t} \in \mathscr{L}_{\mathcal{H}(G)}$; we shall find some $\boldsymbol{s} \in \mathscr{L}_G$ such that

1. $\mathcal{H}_G(\boldsymbol{s}) = \boldsymbol{t}$;
2. 1-internal moves in $\boldsymbol{s}$ occur as even-length consecutive segments $m_1 m_2 \dots m_{2k}$, where m_i justifies m_{i+1} for $i = 1, 2, \dots, 2k-1$;
3. $\boldsymbol{s}$ is 1-complete.

We proceed by induction on the length $|\boldsymbol{t}|$ of $\boldsymbol{t}$. The base case $\boldsymbol{t} = \boldsymbol{\epsilon}$ is trivial. For the inductive step, let $\boldsymbol{t}m \in \mathscr{L}_{\mathcal{H}(G)}$. Then, $\boldsymbol{t} \in \mathscr{L}_{\mathcal{H}(G)}$, and by the induction hypothesis there is some $\boldsymbol{s} \in \mathscr{L}_G$ that satisfies the three conditions (n.b., the first one is for $\boldsymbol{t}$).

If m is initial, then $\boldsymbol{s}m \in \mathscr{L}_G$, and $\boldsymbol{s}m$ satisfies the three conditions. Thus, assume that m is non-initial; we may write $\boldsymbol{t}m = \boldsymbol{t_1}n\boldsymbol{t_2}m$, where m is justified by n.

We then need a case analysis:

- Assume $n \vdash_G m$. We take $\boldsymbol{s}m$, where m points to n. Then, $\boldsymbol{s}m \in \mathscr{L}_G$ since:
  - (Justification) It is immediate because $n \vdash_G m$.
  - (Alternation) By the condition 3 on $\boldsymbol{s}$, the last moves of $\boldsymbol{s}$ and $\boldsymbol{t}$ just coincide. Thus, the alternation condition holds for $\boldsymbol{s}m$.
  - (Generalized Visibility) It suffices to establish the visibility on $\boldsymbol{s}m$, as the other cases are included as the generalized visibility on $\boldsymbol{t}m$. It is straightforward to see that, by the condition 2 on $\boldsymbol{s}$, if the view of $\boldsymbol{t}$ contains n, then so does the view of $\boldsymbol{s}$. And since $\boldsymbol{t}m \in \mathscr{L}_{\mathcal{H}(G)}$, the view of $\boldsymbol{t}$ contains n. Hence, the view of $\boldsymbol{s}$ contains n as well.
  - (IE-Switch) Again, the last moves of $\boldsymbol{s}$ and $\boldsymbol{t}$ coincide by the condition 3 on $\boldsymbol{s}$; thus, IE-switch on $\boldsymbol{t}m$ can be directly applied.

  Also, it is easy to see that $\boldsymbol{s}m$ satisfies the three conditions.

- Assume $n \neq \star$ and $\exists k \in \mathbb{N}^+, m_1, m_2, \dots, m_{2k} \in M_G \setminus M_{\mathcal{H}(G)}$ such that

  $$n \vdash_G m_1 \wedge m_1 \vdash_G m_2 \wedge \dots \wedge m_{2k-1} \vdash_G m_{2k} \wedge m_{2k} \vdash_G m.$$

  We then take $\boldsymbol{s}m_1m_2\dots m_{2k}m$, in which m_1 points to n, m_i points to m_{i-1} for $i = 2, 3, \dots, 2k$, and m points to m_{2k}. Then, $\boldsymbol{s}m_1m_2\dots m_{2k}m \in \mathscr{L}_G$ because:

  – (Justification) Obvious.

  – (Alternation) By the condition 3 on $\boldsymbol{s}$, the last moves of $\boldsymbol{s}$ and $\boldsymbol{t}$ just coincide. Thus, the alternation condition holds for $\boldsymbol{s}m_1m_2\dots m_{2k}m$.

  – (Generalized Visibility) By the same argument as the above case.

  – (IE-Switch) It clearly holds by the axiom E4.

  Finally, it is easy to see that $\boldsymbol{s}m_1m_2\dots m_{2k}m$ satisfies the three conditions, which completes the proof.

□

3.3.3 Constructions on Dynamic Games

Next, let us prove that dynamic games accommodate all the constructions on games in Chapter 2, i.e., they preserve the additional axioms on dynamic games, where note that the constructions may be adopted for dynamic games without any problem. This result implies that the definition of dynamic games is in some sense 'correct'.

Let us begin with *tensor (product)* $\otimes$ (Definition 2.2.19):

Theorem 3.3.17 (Well-defined tensor on dynamic games)**.** *Dynamic games are closed under tensor* $\otimes$*.*

Proof. It suffices to show that tensor $\otimes$ preserves the condition on labeling function and the axioms E1, E2, E4, DP2, DP3 and DI3. However, non-trivial ones are just DP3 and DI3; thus, we just focus on these two. Let A and B be any dynamic games.

To verify DP3 on $A \otimes B$, let $\boldsymbol{s}lmn, \boldsymbol{s'}l'm'n' \in P^{\mathsf{Odd}}_{A\otimes B}$ and $i \in \mathbb{N}$ such that $\mathcal{H}^i_{A\otimes B}(\boldsymbol{s}lm) = \mathcal{H}^i_{A\otimes B}(\boldsymbol{s'}l'm')$ and $i < \lambda^{\mathbb{N}}_{A\otimes B}(n) = \lambda^{\mathbb{N}}_{A\otimes B}(n')$. Note that $\lambda^{\mathbb{N}}_{A\otimes B}(m) = \lambda^{\mathbb{N}}_{A\otimes B}(n) = \lambda^{\mathbb{N}}_{A\otimes B}(n') = \lambda^{\mathbb{N}}_{A\otimes B}(m')$ by IE-switch. At a first glance, it seems that $A \otimes B$ does not satisfy DP3 as Opponent may choose to play in A or B at will. It is, however, not the case for internal moves for $\boldsymbol{s}lmn \in P^{\mathsf{Odd}}_{A\otimes B}$ with m internal implies $m, n \in M_A$ or $m, n \in M_B$. This property immediately follows from Table 3.1 which shows the possible transitions of OP- and IE-parities for a play of $A\otimes B$, where a state

$$\begin{tikzcd}
(\mathsf{P}^{\mathsf{I}}, \mathsf{O}^{\mathsf{E}}) \arrow[d, leftrightarrow, "A"] & & (\mathsf{O}^{\mathsf{E}}, \mathsf{O}^{\mathsf{E}}) \arrow[ll, "A"'] \arrow[rr, "B"] \arrow[dl, leftrightarrow, "A"'] \arrow[dr, leftrightarrow, "B"] & & (\mathsf{O}^{\mathsf{E}}, \mathsf{P}^{\mathsf{I}}) \arrow[d, leftrightarrow, "B"'] \\
(\mathsf{O}^{\mathsf{I}}, \mathsf{O}^{\mathsf{E}}) \arrow[r, "A"] & (\mathsf{P}^{\mathsf{E}}, \mathsf{O}^{\mathsf{E}}) & & (\mathsf{O}^{\mathsf{E}}, \mathsf{P}^{\mathsf{E}}) & (\mathsf{O}^{\mathsf{E}}, \mathsf{O}^{\mathsf{I}}) \arrow[l, "B"']
\end{tikzcd}$$

Table 3.1: The double parity diagram

(X^Y, Z^W) indicates that the next move of A (resp. B) has the OP-parity X (resp. Z) and the IE-parity Y (resp. W). Note that $m = m'$ and $\mathcal{J}^{\ominus i}_{\boldsymbol{slm}}(m) = \mathcal{J}^{\ominus i}_{\boldsymbol{s'}l'm'}(m')$ as $\mathcal{H}^i_{A\otimes B}(\boldsymbol{s}l).m = \mathcal{H}^i_{A\otimes B}(\boldsymbol{s}lm) = \mathcal{H}^i_{A\otimes B}(\boldsymbol{s'}l'm') = \mathcal{H}^i_{A\otimes B}(\boldsymbol{s'}l').m'$. Thus, m, n, m' and n' belong to the same component game. If $m, n, m', n' \in M_A$, then $(\boldsymbol{s}l \upharpoonright A).mn, (\boldsymbol{s'}l' \upharpoonright A).m'n' \in P_A^{\mathsf{Odd}}$, $\mathcal{H}^i_A((\boldsymbol{s}l \upharpoonright A).m) = \mathcal{H}^i_{A\otimes B}(\boldsymbol{s}lm) \upharpoonright \mathcal{H}^i(A) = \mathcal{H}^i_{A\otimes B}(\boldsymbol{s'}l'm') \upharpoonright \mathcal{H}^i(A) = \mathcal{H}^i_A((\boldsymbol{s'}l' \upharpoonright A).m')$ and $i < \lambda^{\mathbb{N}}_A(n) = \lambda^{\mathbb{N}}_A(n')$; thus by DP3 on A, we conclude that $n = n'$ and $\mathcal{J}^{\ominus i}_{\boldsymbol{s}lmn}(n) = \mathcal{J}^{\ominus i}_{(\boldsymbol{s}l\upharpoonright A).mn}(n) = \mathcal{J}^{\ominus i}_{(\boldsymbol{s'}l'\upharpoonright A).m'n'}(n') = \mathcal{J}^{\ominus i}_{\boldsymbol{s'}l'm'n'}(n')$. The other case is completely similar, showing that $A \otimes B$ satisfies DP3.

Finally, to show that $A \otimes B$ satisfies DI3, assume that $\boldsymbol{s} \simeq^d_{A\otimes B} \boldsymbol{t}$ and $\boldsymbol{s}m \in P_{A\otimes B}$ for some $d \in \mathbb{N}$; we have to find some $\boldsymbol{t}n \in P_{A\otimes B}$ such that $\boldsymbol{s}m \simeq^d_{A\otimes B} \boldsymbol{t}n$. Let us assume $m \in M_A$ for the other case is completely symmetric. Since $\boldsymbol{s} \simeq^d_{A\otimes B} \boldsymbol{t}$, we have some $\boldsymbol{s'} \simeq_{A\otimes B} \boldsymbol{t'}$ such that $\mathcal{H}^d_{A\otimes B}(\boldsymbol{s'}) = \mathcal{H}^d_{A\otimes B}(\boldsymbol{s})$ and $\mathcal{H}^d_{A\otimes B}(\boldsymbol{t'}) = \mathcal{H}^d_{A\otimes B}(\boldsymbol{t})$. Thus, $\boldsymbol{s'} \upharpoonright A \simeq_A \boldsymbol{t'} \upharpoonright A$, $\mathcal{H}^d_A(\boldsymbol{s} \upharpoonright A) = \mathcal{H}^d_{A\otimes B}(\boldsymbol{s}) \upharpoonright \mathcal{H}^d(A) = \mathcal{H}^d_{A\otimes B}(\boldsymbol{s'}) \upharpoonright \mathcal{H}^d(A) = \mathcal{H}^d_A(\boldsymbol{s'} \upharpoonright A)$ and $\mathcal{H}^d_A(\boldsymbol{t} \upharpoonright A) = \mathcal{H}^d_{A\otimes B}(\boldsymbol{t}) \upharpoonright \mathcal{H}^d(A) = \mathcal{H}^d_{A\otimes B}(\boldsymbol{t'}) \upharpoonright \mathcal{H}^d(A) = \mathcal{H}^d_A(\boldsymbol{t'} \upharpoonright A)$, whence $\boldsymbol{s} \upharpoonright A \simeq^d_A \boldsymbol{t} \upharpoonright A$. Similarly, we have $\boldsymbol{s} \upharpoonright B \simeq^d_B \boldsymbol{t} \upharpoonright B$ with $\boldsymbol{s'} \upharpoonright B \simeq_B \boldsymbol{t'} \upharpoonright B$, $\mathcal{H}^d_B(\boldsymbol{s} \upharpoonright B) = \mathcal{H}^d_B(\boldsymbol{s'} \upharpoonright B)$ and $\mathcal{H}^d_B(\boldsymbol{t} \upharpoonright B) = \mathcal{H}^d_B(\boldsymbol{t'} \upharpoonright B)$. Now, since $(\boldsymbol{s} \upharpoonright A).m = \boldsymbol{s}m \upharpoonright A \in P_A$, we have some $(\boldsymbol{t} \upharpoonright A).n \in P_A$ such that $(\boldsymbol{s} \upharpoonright A).m \simeq^d_A (\boldsymbol{t} \upharpoonright A).n$, i.e., some $\boldsymbol{u} \simeq_A \boldsymbol{v}$ such that $\mathcal{H}^d_A(\boldsymbol{u}) = \mathcal{H}^d_A((\boldsymbol{s} \upharpoonright A).m)$ and $\mathcal{H}^d_A(\boldsymbol{v}) = \mathcal{H}^d_A((\boldsymbol{t} \upharpoonright A).n)$. By Table 3.1, we may clearly obtain a unique $\tilde{\boldsymbol{s}} \in P_{A\otimes B}$ from $\boldsymbol{u}$ and $\boldsymbol{s'} \upharpoonright B$ as well as a unique $\tilde{\boldsymbol{t}} \in P_{A\otimes B}$ from $\boldsymbol{v}$ and $\boldsymbol{t'} \upharpoonright B$ such that $\tilde{\boldsymbol{s}} \simeq_{A\otimes B} \tilde{\boldsymbol{t}}$, $\mathcal{H}^d_{A\otimes B}(\tilde{\boldsymbol{s}}) = \mathcal{H}^d_{A\otimes B}(\boldsymbol{s}m)$ and $\mathcal{H}^d_{A\otimes B}(\tilde{\boldsymbol{t}}) = \mathcal{H}^d_{A\otimes B}(\boldsymbol{t}n)$, establishing $\boldsymbol{s}m \simeq^d_{A\otimes B} \boldsymbol{t}n$. □

Next, let us verify closure of dynamic games under *linear implication* $\multimap$ (Definition 2.2.21), for which we need to apply the ω-hiding operation $\mathcal{H}^\omega$ on the domain:

Theorem 3.3.18 (Well-defined linear implication on dynamic games)**.** *The linear implication $\mathcal{H}^\omega(A) \multimap B$ is a dynamic game for any dynamic games A and B.*

Proof. Again, it suffices to show the preservation property of the additional conditions on the labeling function and the axioms E1, E2, E4, DP2, DP3 and DI3. For brevity, assume that A is normalized and consider $A \multimap B$. Again, non-trivial conditions are just DP3 and DI3, but DI3 may be shown in a way similar to the case of tensor.

To verify DP3, let $i \in \mathbb{N}$ and $\boldsymbol{slmn}, \boldsymbol{s'}l'm'n' \in P^{\mathsf{Odd}}_{A \multimap B}$ such that $\mathcal{H}^i_{A \multimap B}(\boldsymbol{s}lm) = \mathcal{H}^i_{A \multimap B}(\boldsymbol{s'}l'm')$ and $i < \lambda^{\mathbb{N}}_{A \multimap B}(n) = \lambda^{\mathbb{N}}_{A \multimap B}(n')$. Again, m and m' are both internal, and so m, n, m' and n' all belong to B. Thus, $(\boldsymbol{s}l \upharpoonright B).mn, (\boldsymbol{s'}l' \upharpoonright B).m'n' \in P^{\mathsf{Odd}}_B$ such that $\mathcal{H}^i_B((\boldsymbol{s}l \upharpoonright B).m) = \mathcal{H}^i_{A \multimap B}(\boldsymbol{s}lm) \upharpoonright \mathcal{H}^i(B) = \mathcal{H}^i_{A \multimap B}(\boldsymbol{s'}l'm') \upharpoonright \mathcal{H}^i(B) = \mathcal{H}^i_B((\boldsymbol{s'}l' \upharpoonright B).m')$ and $i < \lambda^{\mathbb{N}}_B(n) = \lambda^{\mathbb{N}}_B(n')$; thus, by DP2 on B, we may conclude that $n = n'$ and $\mathcal{J}^{\ominus i}_{\boldsymbol{s}lmn}(n) = \mathcal{J}^{\ominus i}_{(\boldsymbol{s}l \upharpoonright B).mn}(n) = \mathcal{J}^{\ominus i}_{(\boldsymbol{s}l \upharpoonright B).mn}(n') = \mathcal{J}^{\ominus i}_{\boldsymbol{s}lmn}(n')$. □

Remark. We need to apply the ω-hiding operation $\mathcal{H}^\omega$ on the domain A for a linear implication $\mathcal{H}^\omega(A) \multimap B$ since otherwise the linear implication may not satisfy the axiom DP2 or DP3. It conceptually makes sense too for the roles of Player and Opponent in the domain A are exchanged, and thus Player cannot 'see' internal moves of A. Notationally, we shall usually omit the operation $\mathcal{H}^\omega$ if A is normalized.

Now, we make a straightforward generalization of *product* & (Definition 2.2.23):

Notation. Given a function $f : X \to Y$ and a subset $Z \subseteq X$, we write $f \downharpoonright Z : X \setminus Z \to Y$ for the restrictions of f to the subset $X \setminus Z \subseteq X$.

Definition 3.3.19 (Pairing of dynamic games)**.** The ***pairing*** $\langle L, R \rangle$ of dynamic games L and R such that $\mathcal{H}^\omega(L) \trianglelefteq C \multimap A$ and $\mathcal{H}^\omega(R) \trianglelefteq C \multimap B$ for some normalized dynamic games A, B and C is defined by:

- $M_{\langle L,R \rangle} \stackrel{\text{df.}}{=} M_C + (M_L \setminus M_C) + (M_R \setminus M_C)$, where 'tags' for the disjoint union is chosen in such a way that $\langle \mathcal{H}^\omega(L), \mathcal{H}^\omega(R) \rangle \trianglelefteq C \multimap A \& B$ holds (Theorem 3.3.20); see Definition 4.3.18 for a possible concrete implementation of the 'tags';
- $\lambda_{\langle L,R \rangle} \stackrel{\text{df.}}{=} [\overline{\lambda_C}, \lambda_L \downharpoonright M_C, \lambda_R \downharpoonright M_C]$;
- $m \vdash_{\langle L,R \rangle} n \stackrel{\text{df.}}{\Leftrightarrow} m \vdash_L n \vee m \vdash_R n$;
- $P_{\langle L,R \rangle} \stackrel{\text{df.}}{=} \{ \boldsymbol{s} \in \mathscr{L}_{L \& R} \mid (\boldsymbol{s} \upharpoonright L \in P_L \wedge \boldsymbol{s} \upharpoonright R = \boldsymbol{\epsilon}) \vee (\boldsymbol{s} \upharpoonright L = \boldsymbol{\epsilon} \wedge \boldsymbol{s} \upharpoonright R \in P_R) \}$;
- $\boldsymbol{s} \simeq_{\langle L,R \rangle} \boldsymbol{t} \stackrel{\text{df.}}{\Leftrightarrow} (\boldsymbol{s} \upharpoonright L = \boldsymbol{\epsilon} \Leftrightarrow \boldsymbol{t} \upharpoonright L = \boldsymbol{\epsilon}) \wedge \boldsymbol{s} \upharpoonright L \simeq_L \boldsymbol{t} \upharpoonright L \wedge \boldsymbol{s} \upharpoonright R \simeq_R \boldsymbol{t} \upharpoonright R$.

Theorem 3.3.20 (Well-defined pairing on dynamic games)**.** *If dynamic games L and R satisfy $\mathcal{H}^\omega(L) \trianglelefteq C \multimap A$ and $\mathcal{H}^\omega(R) \trianglelefteq C \multimap B$ for normalized dynamic games A, B and C, then we have $\langle \mathcal{H}^\omega(L), \mathcal{H}^\omega(R) \rangle \trianglelefteq C \multimap A \& B$.*

Proof. Similar to and simpler than the case of tensor $\otimes$. □

Pairing $\langle \_, \_ \rangle$ is certainly a generalization of product & since $\langle T \multimap A, T \multimap B \rangle = T \multimap A \& B \cong A \& B$ for any normalized dynamic games A and B.

Next, we generalize *exponential* ! (Definition 2.2.25):

Definition 3.3.21 (Promotion of dynamic games)**.** Given a dynamic game G such that $\mathcal{H}^\omega(G) \trianglelefteq {!A} \multimap B$ for some normalized dynamic games A and B, its ***promotion*** $G^\dagger$ is defined by:

- $M_{G^\dagger} \overset{\text{df.}}{=} ((M_G \setminus M_{!A}) \times \mathbb{N}) + M_{!A}$, where 'tags' for the disjoint union is chosen in such a way that $\mathcal{H}^\omega(G)^\dagger \trianglelefteq {!A} \multimap {!B}$ holds (Theorem 3.3.22); see Definition 4.3.22 for a possible concrete implementation of the 'tags';
- $\lambda_{G^\dagger} : ((m,i) \in (M_G \setminus M_{!A}) \times \mathbb{N}) \mapsto \lambda_G(m), ((a,j) \in M_{!A}) \mapsto \lambda_G(a,j)$;
- $\star \vdash_{G^\dagger} (m,i) \overset{\text{df.}}{\Leftrightarrow} \star \vdash_G m$ for all $i \in \mathbb{N}$;
- $(m,i) \vdash_{G^\dagger} (n,j) \overset{\text{df.}}{\Leftrightarrow} (i = j \wedge m, n \in M_G \setminus M_{!A} \wedge m \vdash_G n)$
  $\vee (i = j \wedge m \vdash_A n) \vee (m \in M_G \setminus M_{!A} \wedge (n,j) \in M_{!A} \wedge m \vdash_G (n,j))$ for all $i, j \in \mathbb{N}$;
- $P_{G^\dagger} \overset{\text{df.}}{=} \{ \boldsymbol{s} \in \mathscr{L}_{G^\dagger} \mid \forall i \in \mathbb{N}.\, \boldsymbol{s} \upharpoonright i \in P_G \}$, where $\boldsymbol{s} \upharpoonright i$ is the j-subsequence of $\boldsymbol{s}$ that consists of moves of the form (m,i) with $m \in M_G \setminus M_{!A}$ or $(a, \langle i, j \rangle)$ with $a \in M_A$ and $j \in \mathbb{N}$ but changed into m or (a,j), respectively;
- $\boldsymbol{s} \simeq_{G^\dagger} \boldsymbol{t} \overset{\text{df.}}{\Leftrightarrow} \exists \varphi \in \mathcal{P}(\mathbb{N}).\, \forall i \in \mathbb{N}.\, \boldsymbol{s} \upharpoonright \varphi(i) \simeq_G \boldsymbol{t} \upharpoonright i \wedge \pi_2^*(\boldsymbol{s}) = (\varphi \circ \pi_2)^*(\boldsymbol{t})$.

Theorem 3.3.22 (Well-defined promotion on dynamic games)**.** *If a dynamic game G satisfies $\mathcal{H}^\omega(G) \trianglelefteq {!A} \multimap B$ for some normalized dynamic games A and B, then $\mathcal{H}^\omega(G)^\dagger \trianglelefteq {!A} \multimap {!B}$.*

Proof. Straightforward. □

Again, promotion $(\_)^\dagger$ is clearly a generalization of exponential ! for $({!T} \multimap A)^\dagger = {!T} \multimap {!A} \cong {!A}$ for any normalized dynamic game A.

Now, let us introduce a new construction on dynamic games:

Definition 3.3.23 (Concatenation of dynamic games)**.** Given dynamic games J and K such that $\mathcal{H}^\omega(J) \trianglelefteq A \multimap B$ and $\mathcal{H}^\omega(K) \trianglelefteq B \multimap C$ for some normalized dynamic games A, B and C, the ***concatenation*** $J \ddagger K$ of J and K is defined by:

- $M_{J \ddagger K} \overset{\text{df.}}{=} M_J + M_K$, where 'tags' for the disjoint union is chosen in such a way that $\mathcal{H}^\omega((A \multimap B) \ddagger (B \multimap C)) = A \multimap C$ holds (Lemma 3.3.30); see Definition 4.3.23 for a possible concrete implementation of the 'tags';

- $\lambda_{J \ddagger K} \overset{\text{df.}}{=} [\lambda_J \downharpoonright M_{B_{[1]}}, \lambda_J^{+\mu} \upharpoonright M_{B_{[1]}}, \lambda_K^{+\mu} \upharpoonright M_{B_{[2]}}, \lambda_K \downharpoonright M_{B_{[2]}}]$, where $\lambda_G^{+\mu} \overset{\text{df.}}{=} \langle \lambda_G^{\mathsf{OP}}, \lambda_G^{\mathsf{QA}}, n \mapsto \lambda_G^{\mathbb{N}}(n) + \mu \rangle$ (G is J or K), and $\mu \overset{\text{df.}}{=} \max(\mu(J), \mu(K)) + 1$, where $\max(n_1, n_2) \overset{\text{df.}}{=} \begin{cases} n_1 & \text{if } n_1 \geqslant n_2 \\ n_2 & \text{otherwise} \end{cases}$ for all $n_1, n_2 \in \mathbb{N}$;
- $\star \vdash_{J \ddagger K} m \overset{\text{df.}}{\Leftrightarrow} \star \vdash_K m$;
- $m \vdash_{J \ddagger K} n \ (m \neq \star) \overset{\text{df.}}{\Leftrightarrow} m \vdash_J n \vee m \vdash_K n \vee (\star \vdash_{B_{[2]}} m \wedge \star \vdash_{B_{[1]}} n)$;
- $P_{J \ddagger K} \overset{\text{df.}}{=} \{ \boldsymbol{s} \in \mathscr{J}_{J \ddagger K} \mid \boldsymbol{s} \upharpoonright J \in P_J, \boldsymbol{s} \upharpoonright K \in P_K, \boldsymbol{s} \upharpoonright B_{[1]}, B_{[2]} \in pr_B \}$;
- $\boldsymbol{s} \simeq_{J \ddagger K} \boldsymbol{t} \overset{\text{df.}}{\Leftrightarrow} (\forall i \in \mathbb{N}.\, s_i \in M_J \Leftrightarrow t_i \in M_J) \wedge \boldsymbol{s} \upharpoonright J \simeq_J \boldsymbol{t} \upharpoonright J \wedge \boldsymbol{s} \upharpoonright K \simeq_K \boldsymbol{t} \upharpoonright K$.

We shall see later that the 'non-hiding composition' or *concatenation* of dynamic strategies $\sigma : J$ and $\tau : K$ forms a dynamic strategy on the concatenation $J \ddagger K$.

Definition 3.3.24 (Composition of dynamic games)**.** Given the concatenation $J \ddagger K$ of dynamic games J and K, their ***composition*** $J; K$ (or $K \circ J$) is given by:

$$J; K \overset{\text{df.}}{=} \mathcal{H}^{\omega}(J \ddagger K).$$

Theorem 3.3.25 (Well-defined concatenation and composition on dynamic games)**.** *Dynamic games are closed under concatenation and composition.*

Proof. By Theorem 3.3.14, it suffices to focus on concatenation. We first show that the dynamic arena $J \ddagger K$ is well-defined. The set $M_{J \ddagger K}$ and the function $\lambda_{J \ddagger K}$ are clearly well-defined, where the *finite* upper bounds $\mu(J)$ and $\mu(K)$ are crucial. For the enabling relation, the axioms E1 and E3 clearly hold. For the axiom E2, if $m \vdash_{J \ddagger K} n$ and $\lambda_{J \ddagger K}^{\mathsf{QA}}(n) = \mathsf{A}$, then $m, n \in M_K \setminus M_{B_{[2]}}$, $m, n \in M_{B_{[2]}}$, $m, n \in M_{B_{[1]}}$ or $m, n \in M_J \setminus M_{B_{[1]}}$. In either case, $\lambda_{J \ddagger K}^{\mathsf{QA}}(m) = \mathsf{Q}$ and $\lambda_{J \ddagger K}^{\mathbb{N}}(m) = \lambda_{J \ddagger K}^{\mathbb{N}}(n)$.

For the axiom E4, let $m \vdash_{J \ddagger K} n$, $m \neq \star$ and $\lambda_{J \ddagger K}^{\mathbb{N}}(m) \neq \lambda_{J \ddagger K}^{\mathbb{N}}(n)$. We proceed by a case analysis. If $(m \vdash_K n) \wedge (m, n \in M_K \setminus M_{B_{[2]}} \vee m, n \in M_{B_{[2]}})$, then we may just apply E4 on K. It is similar if $(m \vdash_J n) \wedge (m, n \in M_J \setminus M_{B_{[1]}} \vee m, n \in M_{B_{[1]}})$. Note that the case $\star \vdash_{B_{[2]}} m \wedge \star \vdash_{B_{[1]}} n$ cannot happen. Now, consider the case $m \vdash_K n \wedge m \in M_K \setminus M_{B_{[2]}} \wedge n \in M_{B_{[2]}}$. If m is external, then $m \in M_C$, and so E4 on $J \ddagger K$ is satisfied by the definition of $B \multimap C$; if m is internal, then we may apply E4 on K. The case $m \vdash_K n \wedge n \in M_K \setminus M_{B_{[2]}} \wedge m \in M_{B_{[2]}}$ is simpler as m must be internal. The remaining cases $m \vdash_J n \wedge m \in M_J \setminus M_{B_{[1]}} \wedge n \in M_{B_{[1]}}$ and $m \vdash_J n \wedge n \in M_J \setminus M_{B_{[1]}} \wedge m \in M_{B_{[1]}}$ are similar. Thus, the arena $J \ddagger K$ is well-defined.

$$\begin{tikzcd}
 & (\mathsf{O^E}, \mathsf{O^E}) \arrow[r, "C"] \arrow[d, shift right, "C"'] & (\mathsf{O^E}, \mathsf{P^I}) \arrow[d, shift right, "K"'] \\
(\mathsf{P^I}, \mathsf{O^E}) \arrow[d, shift right, "J"'] & (\mathsf{O^E}, \mathsf{P^E}) \arrow[l, dotted, "B_{[1]}B_{[2]}"'] \arrow[u, shift right, "C"'] \arrow[d, dotted, shift right, "B_{[2]} B_{[1]}"'] & (\mathsf{O^E}, \mathsf{O^I}) \arrow[l, "K"'] \arrow[u, shift right, "K"'] \arrow[d, shift right, "K"'] \\
(\mathsf{O^I}, \mathsf{O^E}) \arrow[u, shift right, "J"'] \arrow[r, "J"] \arrow[d, shift right, "J"'] & (\mathsf{P^E}, \mathsf{O^E}) \arrow[u, dotted, shift right, "B_{[2]} B_{[1]}"'] \arrow[r, dotted, "B_{[1]}B_{[2]}"] \arrow[d, shift right, "A"'] & (\mathsf{O^E}, \mathsf{P^I}) \arrow[u, shift right, "K"'] \\
(\mathsf{P^I}, \mathsf{O^E}) \arrow[u, shift right, "J"'] & (\mathsf{O^E}, \mathsf{O^E}) \arrow[l, "A"'] \arrow[u, shift right, "A"'] &
\end{tikzcd}$$

Table 3.2: The concatenation double parity diagram

Next, we show that $P_{J \ddagger K} \subseteq \mathscr{L}_{J \ddagger K}$. For justification, let $\boldsymbol{s}m \in P_{J \ddagger K}$ with m non-initial. The non-trivial case is when m is initial in J. But in this case m is initial in $B_{[1]}$, and so it has a justifier in $B_{[2]}$. For alternation and IE-switch, similarly to Table 3.1 for tensor $\otimes$, we have Table 3.2 for $J \ddagger K$, in which the first (resp. second) component of each state is about the OP- and IE-parities of the next move of J (resp. K). For readability some states are written twice, and the dotted arrow indicates two consecutive moves in B. Then, alternation and IE-switch on $J \ddagger K$ immediately follows from this diagram and the corresponding conditions on J and K.

For generalized visibility, let $\boldsymbol{s}m \in P_{J \ddagger K}$ with m non-initial and $d \in \mathbb{N} \cup \{\omega\}$ such that $\boldsymbol{s}m$ is d-complete. Without loss of generality, we may assume $d \in \mathbb{N}$ as $\boldsymbol{s}$ is finite. It is not hard to see that $\mathcal{H}^d_{J \ddagger K}(\boldsymbol{s}m) \in P_{\mathcal{H}^d(J) \ddagger \mathcal{H}^d(K)}$ if $\mathcal{H}^d(J \ddagger K)$ is not normalized; thus, this case is reduced to the (usual) visibility on $\mathcal{H}^d(J) \ddagger \mathcal{H}^d(K)$. Otherwise, it is no harm to select the *least* $d \in \mathbb{N}^+$ such that $\mathcal{H}^d(J \ddagger K)$ is normalized; then $\mathcal{H}^{d-1}_{J \ddagger K}(\boldsymbol{s}m) \in P_{(A \multimap B_{[1]}) \ddagger (B_{[2]} \multimap C)}$, and thus the visibility of $\mathcal{H}^d_{J \ddagger K}(\boldsymbol{s}m) = \mathcal{H}_{\mathcal{H}^{d-1}(J \ddagger K)}(\mathcal{H}^{d-1}_{J \ddagger K}(\boldsymbol{s}m))$ can be shown completely in the same way as the proof that shows the composition of strategies is well-defined (in particular it satisfies visibility) [129, 14]. Consequently, it suffices to consider the case $d = 0$, i.e., to show the (usual) visibility.

For this, we need the following:

Lemma 3.3.26 (Visibility lemma). *Assume that $\boldsymbol{t} \in P_{J \ddagger K}$ and $\boldsymbol{t} \neq \boldsymbol{\epsilon}$. Then we have:*

1. *If the last move of $\boldsymbol{t}$ is in $M_J \setminus M_{B_{[1]}}$, then $\lceil \boldsymbol{t} \upharpoonright J \rceil_J \preceq \lceil \boldsymbol{t} \rceil_{J \ddagger K} \upharpoonright J$ and $\lfloor \boldsymbol{t} \upharpoonright J \rfloor_J \preceq \lfloor \boldsymbol{t} \rfloor_{J \ddagger K} \upharpoonright J$;*

2. If the last move of $\boldsymbol{t}$ is in $M_K \setminus M_{B_{[2]}}$, then $\lceil \boldsymbol{t} \upharpoonright K \rceil_K \preceq \lceil \boldsymbol{t} \rceil_{J \ddagger K} \upharpoonright K$ and $\lfloor \boldsymbol{t} \upharpoonright K \rfloor_K \preceq \lfloor \boldsymbol{t} \rfloor_{J \ddagger K} \upharpoonright K$;

3. If the last move of $\boldsymbol{t}$ is an O-move in $M_{B_{[1]}} \cup M_{B_{[2]}}$, then $\lceil \boldsymbol{t} \upharpoonright B_{[1]}, B_{[2]} \rceil_{B_{[1]} \multimap B_{[2]}} \preceq \lfloor \boldsymbol{t} \rfloor_{J \ddagger K} \upharpoonright B_{[1]}, B_{[2]}$ and $\lfloor \boldsymbol{t} \upharpoonright B_{[1]}, B_{[2]} \rfloor_{B_{[1]} \multimap B_{[2]}} \preceq \lceil \boldsymbol{t} \rceil_{J \ddagger K} \upharpoonright B_{[1]}, B_{[2]}$.

Proof of the lemma. By induction on $|\boldsymbol{t}|$ with case analysis on the last move of $\boldsymbol{t}$. □

Note that we may write $\boldsymbol{s}m = \boldsymbol{s_1}n\boldsymbol{s_2}m$, where n justifies m. If $\boldsymbol{s_2} = \boldsymbol{\epsilon}$, then it is trivial; so assume $\boldsymbol{s_2} = \boldsymbol{s_2'}r$. We then proceed by a case analysis on m:

- Assume $m \in M_J \setminus M_{B_{[1]}}$. Then we have $n \in M_J$ and $r \in M_J$ by Table 3.2. By Lemma 3.3.26, $\lceil \boldsymbol{s} \upharpoonright J \rceil \preceq \lceil \boldsymbol{s} \rceil \upharpoonright J$ and $\lfloor \boldsymbol{s} \upharpoonright J \rfloor \preceq \lfloor \boldsymbol{s} \rfloor \upharpoonright J$. Also, since $(\boldsymbol{s} \upharpoonright J).m \in P_J$, visibility on J implies:

$$\begin{aligned} &n \text{ occurs in } \lceil \boldsymbol{s} \upharpoonright J \rceil \text{ if } m \text{ is a P-move;} \\ &n \text{ occurs in } \lfloor \boldsymbol{s} \upharpoonright J \rfloor \text{ if } m \text{ is an O-move.} \end{aligned}$$

  Hence we may conclude that n occurs in $\lceil \boldsymbol{s} \rceil$ (resp. $\lfloor \boldsymbol{s} \rfloor$) if m is a P-move (resp. an O-move).

- Assume $m \in M_K \setminus M_{B_{[2]}}$. This case can be handled in a completely analogous way to the above case.

- Assume $m \in M_{B_{[1]}}$. If m is a P-move, then $n, r \in M_J$ and so it can be handled in the same way as the case $m \in M_J \setminus M_{B_{[1]}}$; thus, assume that m is an O-move. Then, $r \in M_{B_{[2]}}$ and it is a 'copy' of m. Since r is an O-move of $B_{[1]} \multimap B_{[2]}$, by Lemma 3.3.26, $\lceil \boldsymbol{s} \upharpoonright B_{[1]}, B_{[2]} \rceil \preceq \lfloor \boldsymbol{s} \rfloor \upharpoonright B_{[1]}, B_{[2]}$. Note that n is a move of $B_{[1]}$ or an initial move of $B_{[2]}$. In either case, we have $(\boldsymbol{s} \upharpoonright B_{[1]}, B_{[2]}).m \in P_{B_{[1]} \multimap B_{[2]}}$; thus, n occurs in $\lceil \boldsymbol{s} \upharpoonright B_{[1]}, B_{[2]} \rceil$. Hence we may conclude that n occurs in $\lfloor \boldsymbol{s} \rfloor$.

- Assume $m \in M_{B_{[2]}}$. If m is a P-move, then $n, r \in M_K$; so it can be dealt with in the same way as the case $m \in M_K \setminus M_{B_{[2]}}$. Thus assume m is an O-move. By Table 3.2, $r \in M_{B_{[1]}}$ and it is an O-move of $B_{[1]} \multimap B_{[2]}$. Thus by Lemma 3.3.26, $\lceil \boldsymbol{s} \upharpoonright B_{[1]}, B_{[2]} \rceil \preceq \lfloor \boldsymbol{s} \rfloor \upharpoonright B_{[1]}, B_{[2]}$. Then again, $(\boldsymbol{s} \upharpoonright B_{[1]}, B_{[2]}).m \in P_{B_{[1]} \multimap B_{[2]}}$; thus, n occurs in $\lceil \boldsymbol{s} \upharpoonright B_{[1]}, B_{[2]} \rceil$, and so n occurs in $\lfloor \boldsymbol{s} \rfloor$.

Next, we verify the axioms P1, DP2 and DP3. For the axiom P1, it is clear that $\boldsymbol{\epsilon} \in P_{J \ddagger K}$. For the prefix-closure, let $\boldsymbol{s}m \in P_{J \ddagger K}$. If $m \in M_J \setminus M_{B_{[1]}}$, then $(\boldsymbol{s} \upharpoonright J).m = \boldsymbol{s}m \upharpoonright J \in P_J$; thus $\boldsymbol{s} \upharpoonright J \in P_J$, $\boldsymbol{s} \upharpoonright K = \boldsymbol{s}m \upharpoonright K \in P_K$ and

$\boldsymbol{s} \upharpoonright B_{[1]}, B_{[2]} = \boldsymbol{s}m \upharpoonright B_{[1]}, B_{[2]} \in pr_B$, whence $\boldsymbol{s} \in P_{J \ddagger K}$. The other cases may be handled similarly. For the axiom DP2, assume that $\boldsymbol{s}mn \in P_{J \ddagger K}^{\mathsf{Even}}$ and $\lambda_{J \ddagger K}^{\mathbb{N}}(n) > 0$. If $n \notin M_{B_{[1]}} \cup M_{B_{[2]}}$, then we may just apply DP2 on J or K; and the remaining case is trivial by the definition of $J \ddagger K$.

For the axiom DP3, let $i \in \mathbb{N}$ and $\boldsymbol{s}m, \boldsymbol{s'}m' \in P_{J \ddagger K}^{\mathsf{Odd}}$ such that $i < \lambda_{J \ddagger K}^{\mathbb{N}}(m) = \lambda_{J \ddagger K}^{\mathbb{N}}(m')$ and $\mathcal{H}_{J \ddagger K}^{i}(\boldsymbol{s}) = \mathcal{H}_{J \ddagger K}^{i}(\boldsymbol{s'})$. Without loss of generality, we may assume $i = 0$ and $\lambda_{J \ddagger K}^{\mathbb{N}}(m) = 1 = \lambda_{J \ddagger K}^{\mathbb{N}}(m')$ because if $\lambda_{J \ddagger K}^{\mathbb{N}}(m) = \lambda_{J \ddagger K}^{\mathbb{N}}(m') = j > 1$, then we may consider $\mathcal{H}_{J \ddagger K}^{j-1}(\boldsymbol{s}m), \mathcal{H}_{J \ddagger K}^{j-1}(\boldsymbol{s'}m') \in P_{\mathcal{H}^{j-1}(J) \ddagger \mathcal{H}^{j-1}(K)}$ (n.b., the justifiers of m and m' have the same priority order). Thus, $\boldsymbol{s} = \boldsymbol{s'}$ and $m, m' \in M_J \vee m, m' \in M_K$. If $m, m' \in M_J$ (resp. $m, m' \in M_K$), then $(\boldsymbol{s} \upharpoonright J).m, (\boldsymbol{s'} \upharpoonright J).m' \in P_J^{\mathsf{Odd}}$ (resp. $(\boldsymbol{s} \upharpoonright K).m, (\boldsymbol{s'} \upharpoonright K).m' \in P_K^{\mathsf{Odd}}$), and so we may just apply DP3 on J (resp. K).

Finally, the axioms I1, I2 and DI3 on $\simeq_{J \ddagger K}$ can be verified similarly to the case of tensor $\otimes$, completing the proof. $\square$

For completeness, let us explicitly define the rather trivial *currying*:

Definition 3.3.27 (Currying of dynamic games). Given a dynamic game G such that $\mathcal{H}^{\omega}(G) \trianglelefteq A \otimes B \multimap C$ for some normalized dynamic games A, B and C, the ***currying*** $\Lambda(G)$ of G is defined to be G up to 'tags' such that the 'tags' on moves from A, B and C are adjusted in the way that satisfies $\mathcal{H}^{\omega}(\Lambda(G)) \trianglelefteq A \multimap (B \multimap C)$.

Proposition 3.3.28 (Well-defined currying on dynamic games). *Dynamic games are closed under currying.*

Proof. Obvious. $\square$

Next, we show that these constructions as well as the hiding operation preserve the dynamic subgame relation $\trianglelefteq$ (Definition 3.3.12):

Notation. We write $\clubsuit_{i \in I}$, where I is $\{1\}$ or $\{1, 2\}$, for a construction on dynamic games, i.e., $\clubsuit_{i \in I}$ is either $\otimes$, $\multimap$, $\langle \_, \_ \rangle$, $(\_)^{\dagger}$, $\ddagger$ or Λ.

Lemma 3.3.29 (Preservation of dynamic subgames). *Let $\clubsuit_{i \in I}$ be a construction on dynamic games, and assume $H_i \trianglelefteq G_i$ for all $i \in I$. Then, $\clubsuit_{i \in I} H_i \trianglelefteq \clubsuit_{i \in I} G_i$.*

Proof. Let us first consider tensor $\otimes$. It is trivial to check the conditions on the sets of moves and the labeling functions, and so we omit them. For the enabling relations:

$$\begin{aligned}\vdash_{H_1 \otimes H_2} &= \vdash_{H_1} + \vdash_{H_2} \\ &\subseteq (\vdash_{G_1} \cap ((\{\star\} \cup M_{H_1}) \times M_{H_1})) + (\vdash_{G_2} \cap ((\{\star\} \cup M_{H_2}) \times M_{H_2})) \\ &= (\vdash_{G_1} \cap ((\{\star\} \cup M_{H_1 \otimes H_2}) \times M_{H_1 \otimes H_2})) + (\vdash_{G_2} \cap ((\{\star\} \cup M_{H_1 \otimes H_2}) \times M_{H_1 \otimes H_2})) \\ &= (\vdash_{G_1} + \vdash_{G_2}) \cap ((\{\star\} \cup M_{H_1 \otimes H_2}) \times M_{H_1 \otimes H_2}) \\ &= \vdash_{G_1 \otimes G_2} \cap ((\{\star\} \cup M_{H_1 \otimes H_2}) \times M_{H_1 \otimes H_2}).\end{aligned}$$

For the positions, we have:

$$\begin{aligned}P_{H_1 \otimes H_2} &= \{\boldsymbol{s} \in \mathscr{L}_{H_1 \otimes H_2} \mid \forall i \in \{1,2\}. \, \boldsymbol{s} \upharpoonright H_i \in P_{H_i}\} \\ &\subseteq \{\boldsymbol{s} \in \mathscr{L}_{G_1 \otimes G_2} \mid \forall i \in \{1,2\}. \, \boldsymbol{s} \upharpoonright G_i \in P_{G_i}\} \\ &= P_{G_1 \otimes G_2}.\end{aligned}$$

For the identifications of positions, given $d \in \mathbb{N} \cup \{\omega\}$, we have:

$$\begin{aligned}&\boldsymbol{s} \simeq^d_{H_1 \otimes H_2} \boldsymbol{t} \\ \Leftrightarrow\ &\exists \boldsymbol{s'}, \boldsymbol{t'} \in P_{H_1 \otimes H_2}. \, \boldsymbol{s'} \simeq_{H_1 \otimes H_2} \boldsymbol{t'} \wedge \mathcal{H}^d_{H_1 \otimes H_2}(\boldsymbol{s'}) = \mathcal{H}^d_{H_1 \otimes H_2}(\boldsymbol{s}) \\ &\wedge \mathcal{H}^d_{H_1 \otimes H_2}(\boldsymbol{t'}) = \mathcal{H}^d_{H_1 \otimes H_2}(\boldsymbol{t}) \\ \Leftrightarrow\ &\forall j \in \{1,2\}. \, \exists \boldsymbol{s'_j}, \boldsymbol{t'_j} \in P_{H_j}. \, \boldsymbol{s'_j} \simeq_{H_j} \boldsymbol{t'_j} \wedge \mathcal{H}^d_{H_j}(\boldsymbol{s'_j}) = \mathcal{H}^d_{H_j}(\boldsymbol{s} \upharpoonright H_j) \\ &\wedge \mathcal{H}^d_{H_j}(\boldsymbol{t'_j}) = \mathcal{H}^d_{H_j}(\boldsymbol{t} \upharpoonright H_j) \wedge \forall k \in \mathbb{N}. \, s_k \in M_{H_1} \Leftrightarrow t_k \in M_{H_1} \\ \Leftrightarrow\ &\forall j \in \{1,2\}. \, \boldsymbol{s} \upharpoonright H_j \simeq^d_{H_j} \boldsymbol{t} \upharpoonright H_j \wedge \forall k \in \mathbb{N}. \, s_k \in M_{H_1} \Leftrightarrow t_k \in M_{H_1} \\ \Leftrightarrow\ &\forall j \in \{1,2\}. \, \boldsymbol{s} \upharpoonright G_j, \boldsymbol{t} \upharpoonright G_j \in P_{H_j} \wedge \boldsymbol{s} \upharpoonright G_j \simeq^d_{G_j} \boldsymbol{t} \upharpoonright G_j \\ &\wedge \forall k \in \mathbb{N}. \, s_k \in M_{G_1} \Leftrightarrow t_k \in M_{G_1} \\ \Leftrightarrow\ &\boldsymbol{s}, \boldsymbol{t} \in P_{H_1 \otimes H_2} \wedge \boldsymbol{s} \simeq^d_{G_1 \otimes G_2} \boldsymbol{t}.\end{aligned}$$

Finally, $\mu(H_1 \otimes H_2) = \max(\mu(H_1), \mu(H_2)) = \max(\mu(G_1), \mu(G_2)) = \mu(G_1 \otimes G_2)$. Therefore, we have shown that $H_1 \otimes H_2 \trianglelefteq G_1 \otimes G_2$.

Linear implication and promotion are similar, and pairing is even simpler; thus we omit them. Next, let us consider concatenation. Assume that $\mathcal{H}^\omega(H_1) \trianglelefteq A \multimap B$, $\mathcal{H}^\omega(H_2) \trianglelefteq B \multimap C$, $\mathcal{H}^\omega(G_1) \trianglelefteq D \multimap E$, $\mathcal{H}^\omega(G_2) \trianglelefteq E \multimap F$ for some normalized dynamic games A, B, C, D, E and F; without loss of generality, we assume that these normalized dynamic games are the least ones with respect to the dynamic subgame relation $\trianglelefteq$. By Theorem 3.3.14, $\mathcal{H}^\omega(H_1) \trianglelefteq \mathcal{H}^\omega(G_1) \trianglelefteq D \multimap E$ and $\mathcal{H}^\omega(H_2) \trianglelefteq \mathcal{H}^\omega(G_2) \trianglelefteq E \multimap F$, which in turn implies $A \trianglelefteq D$, $B \trianglelefteq E$ and $C \trianglelefteq F$. First, we clearly have $M_{H_1 \ddagger H_2} \subseteq M_{G_1 \ddagger G_2}$ and $\lambda_{G_1 \ddagger G_2} \upharpoonright M_{H_1 \ddagger H_2} = \lambda_{H_1 \ddagger H_2}$, where $\mu(H_i) = \mu(G_i)$ for $i = 1, 2$ ensures that the priority orders of moves of B coincide.

Next, for the enabling relations, we have:

$$\star \vdash_{H_1 \ddagger H_2} m \Leftrightarrow \star \vdash_{H_2} m \Leftrightarrow \star \vdash_C m \Rightarrow \star \vdash_F m \Leftrightarrow \star \vdash_{G_1 \ddagger G_2} m$$

as well as:

$$\begin{aligned} m \vdash_{H_1 \ddagger H_2} n &\Leftrightarrow m \vdash_{H_1} n \vee m \vdash_{H_2} n \vee (\star \vdash_{B_{[2]}} m \wedge \star \vdash_{B_{[1]}} n) \\ &\Rightarrow m \vdash_{G_1} n \vee m \vdash_{G_2} n \vee (\star \vdash_{E_{[2]}} m \wedge \star \vdash_{E_{[1]}} n) \\ &\Leftrightarrow m \vdash_{G_1 \ddagger G_2} n \end{aligned}$$

for any $m, n \in M_{H_1 \ddagger H_2}$. For the positions, we have:

$$\begin{aligned} P_{H_1 \ddagger H_2} &= \{ \boldsymbol{s} \in \mathscr{J}_{H_1 \ddagger H_2} \mid \boldsymbol{s} \upharpoonright H_1 \in P_{H_1}, \boldsymbol{s} \upharpoonright H_2 \in P_{H_2}, \boldsymbol{s} \upharpoonright B_{[1]}, B_{[2]} \in pr_B \} \\ &\subseteq \{ \boldsymbol{s} \in \mathscr{J}_{G_1 \ddagger G_2} \mid \boldsymbol{s} \upharpoonright G_1 \in P_{G_1}, \boldsymbol{s} \upharpoonright G_2 \in P_{G_2}, \boldsymbol{s} \upharpoonright E_{[1]}, E_{[2]} \in pr_E \} \\ &= P_{G_1 \ddagger G_2}. \end{aligned}$$

Finally, we may show, in the same manner as in the case of tensor, the required condition on the identifications of positions, completing the proof. □

At the end of the present section, we establish the following useful lemma:

Lemma 3.3.30 (Hiding lemma on dynamic games)**.** *Let $\clubsuit_{i \in I}$ be a construction on dynamic games and G_i a dynamic game for all $i \in I$. For all $d \in \mathbb{N} \cup \{\omega\}$, we have:*

1. *$\mathcal{H}^d(\clubsuit_{i \in I} G_i) \trianglelefteq \clubsuit_{i \in I} \mathcal{H}^d(G_i)$ if $\clubsuit \neq \ddagger$, where $\trianglelefteq$ becomes $=$ if $\clubsuit$ is $\langle \_, \_ \rangle$ or Λ;*
2. *$\mathcal{H}^d((G_1) \ddagger (G_2)) \trianglelefteq A \multimap C$ if $\mathcal{H}^d(G_1 \ddagger G_2)$ is normalized, where A, B and C are normalized dynamic games such that $\mathcal{H}^\omega(G_1) \trianglelefteq A \multimap B$ and $\mathcal{H}^\omega(G_2) \trianglelefteq B \multimap C$, and $(A \multimap B); (B \multimap C) = A \multimap C$;*
3. *$\mathcal{H}^d(G_1 \ddagger G_2) = \mathcal{H}^d(G_1) \ddagger \mathcal{H}^d(G_2)$ otherwise.*

Proof. Since there is an upper bound of the priority orders of each dynamic game, it suffices to consider $d \in \mathbb{N}$. But then, as $\mathcal{H}^{i+1} = \mathcal{H} \circ \mathcal{H}^i$ for all $i \in \mathbb{N}$, we may focus on $d = 1$. We focus on tensor $\otimes$ as the other constructions may be handled similarly.

We have to show $\mathcal{H}(G_1 \otimes G_2) \trianglelefteq \mathcal{H}(G_1) \otimes \mathcal{H}(G_2)$. Their sets of moves and labeling functions clearly coincide. For the enabling relations, we have:

$$\begin{aligned} \star \vdash_{\mathcal{H}(G_1 \otimes G_2)} m \Leftrightarrow \star \vdash_{G_1 \otimes G_2} m &\Leftrightarrow \star \vdash_{G_1} m \vee \star \vdash_{G_2} m \\ &\Leftrightarrow \star \vdash_{\mathcal{H}(G_1)} m \vee \star \vdash_{\mathcal{H}(G_2)} m \\ &\Leftrightarrow \star \vdash_{\mathcal{H}(G_1) \otimes \mathcal{H}(G_2)} m \end{aligned}$$

as well as:

$$
\begin{aligned}
& m \vdash_{\mathcal{H}(G_1 \otimes G_2)} n \ (m \neq \star) \\
\Leftrightarrow\ & (m \vdash_{G_1 \otimes G_2} n) \vee \exists k \in \mathbb{N}^+, m_1, m_2, \dots, m_{2k} \in M_{G_1 \otimes G_2} \setminus M_{\mathcal{H}(G_1 \otimes G_2)} .\, m \vdash_{G_1 \otimes G_2} m_1 \\
& \wedge m_1 \vdash_{G_1 \otimes G_2} m_2 \wedge \cdots \wedge\ m_{2k-1} \vdash_{G_1 \otimes G_2} m_{2k} \wedge m_{2k} \vdash_{G_1 \otimes G_2} n \\
\Leftrightarrow\ & (m \vdash_{G_1} n \vee m \vdash_{G_2} n) \vee \exists i \in \{1,2\}, k \in \mathbb{N}^+, m_1, m_2, \dots, m_{2k} \in M_{G_i} \setminus M_{\mathcal{H}(G_i)} . \\
& m \vdash_{G_i} m_1 \wedge\ m_1 \vdash_{G_i} m_2 \wedge \cdots \wedge m_{2k-1} \vdash_{G_i} m_{2k} \wedge m_{2k} \vdash_{G_i} n \\
\Leftrightarrow\ & \exists i \in \{1,2\} .\, m \vdash_{G_i} n \vee \exists k \in \mathbb{N}^+, m_1, m_2, \dots, m_{2k} \in M_{G_i} \setminus M_{\mathcal{H}(G_i)} .\, m \vdash_{G_i} m_1 \\
& \wedge m_1 \vdash_{G_i} m_2 \wedge \cdots \wedge m_{2k-1} \vdash_{G_i} m_{2k} \wedge m_{2k} \vdash_{G_i} n \\
\Leftrightarrow\ & m \vdash_{\mathcal{H}(G_1) \otimes \mathcal{H}(G_2)} n.
\end{aligned}
$$

Thus, the dynamic arenas $\mathcal{H}(G_1 \otimes G_2)$ and $\mathcal{H}(G_1) \otimes \mathcal{H}(G_2)$ coincide. For the positions, we have:

$$
\begin{aligned}
\boldsymbol{s} \in P_{\mathcal{H}(G_1 \otimes G_2)} & \Rightarrow \exists \boldsymbol{t} \in \mathscr{L}_{G_1 \otimes G_2} .\, \mathcal{H}_{G_1 \otimes G_2}(\boldsymbol{t}) = \boldsymbol{s} \wedge \forall i \in \{1,2\} .\, \boldsymbol{t} \upharpoonright G_i \in P_{G_i} \\
& \Rightarrow \exists \boldsymbol{t} \in \mathscr{L}_{G_1 \otimes G_2} .\, \mathcal{H}_{G_1 \otimes G_2}(\boldsymbol{t}) = \boldsymbol{s} \wedge \forall i \in \{1,2\} .\, \mathcal{H}_{G_i}(\boldsymbol{t} \upharpoonright G_i) \in P_{\mathcal{H}(G_i)} \\
& \Rightarrow \exists \boldsymbol{t} \in \mathscr{L}_{G_1 \otimes G_2} .\, \mathcal{H}_{G_1 \otimes G_2}(\boldsymbol{t}) = \boldsymbol{s} \wedge \forall i \in \{1,2\} .\, \mathcal{H}_{G_1 \otimes G_2}(\boldsymbol{t}) \upharpoonright \mathcal{H}(G_i) \in P_{\mathcal{H}(G_i)} \\
& \Rightarrow \boldsymbol{s} \in \mathscr{L}_{\mathcal{H}(G_1 \otimes G_2)} = \mathscr{L}_{\mathcal{H}(G_1) \otimes \mathcal{H}(G_2)} \wedge \forall i \in \{1,2\} .\, \boldsymbol{s} \upharpoonright \mathcal{H}(G_i) \in P_{\mathcal{H}(G_i)} \\
& \Rightarrow \boldsymbol{s} \in P_{\mathcal{H}(G_1) \otimes \mathcal{H}(G_2)} .
\end{aligned}
$$

Finally, for the identifications of positions, given $d \in \mathbb{N} \cup \{\omega\}$, we have:

$$
\begin{aligned}
& \mathcal{H}_{G_1 \otimes G_2}(\boldsymbol{s}) \simeq^{d}_{\mathcal{H}(G_1 \otimes G_2)} \mathcal{H}_{G_1 \otimes G_2}(\boldsymbol{t}) \\
\Leftrightarrow\ & \exists \boldsymbol{s'}, \boldsymbol{t'} \in P_{G_1 \otimes G_2} .\, \mathcal{H}_{G_1 \otimes G_2}(\boldsymbol{s'}) \simeq_{\mathcal{H}(G_1 \otimes G_2)} \mathcal{H}_{G_1 \otimes G_2}(\boldsymbol{t'}) \wedge \mathcal{H}^{d+1}_{G_1 \otimes G_2}(\boldsymbol{s'}) = \mathcal{H}^{d+1}_{G_1 \otimes G_2}(\boldsymbol{s}) \\
& \wedge \mathcal{H}^{d+1}_{G_1 \otimes G_2}(\boldsymbol{t'}) = \mathcal{H}^{d+1}_{G_1 \otimes G_2}(\boldsymbol{t}) \\
\Leftrightarrow\ & \forall j \in \{1,2\} .\, \exists \boldsymbol{s'_j}, \boldsymbol{t'_j} \in P_{G_j} .\, \mathcal{H}_{G_j}(\boldsymbol{s'_j}) \simeq_{\mathcal{H}(G_j)} \mathcal{H}_{G_j}(\boldsymbol{t'_j}) \wedge \mathcal{H}^{d+1}_{G_j}(\boldsymbol{s'_j}) = \mathcal{H}^{d+1}_{G_j}(\boldsymbol{s} \upharpoonright G_j) \\
& \wedge \mathcal{H}^{d+1}_{G_j}(\boldsymbol{t'_j}) = \mathcal{H}^{d+1}_{G_j}(\boldsymbol{t} \upharpoonright G_j) \wedge \forall k \in \mathbb{N} .\, \mathcal{H}^{d+1}_{G_1 \otimes G_2}(s_k) \in M_{\mathcal{H}^{d+1}(G_1)} \Leftrightarrow \mathcal{H}^{d+1}_{G_1 \otimes G_2}(t_k) \in M_{\mathcal{H}^{d+1}(G_1)} \\
\Leftrightarrow\ & \forall j \in \{1,2\} .\, \boldsymbol{s} \upharpoonright G_j \simeq^{d+1}_{G_j} \boldsymbol{t} \upharpoonright G_j \wedge \forall k \in \mathbb{N} .\, \mathcal{H}^{d+1}_{G_1 \otimes G_2}(s_k) \in M_{\mathcal{H}^{d+1}(G_1)} \Leftrightarrow \mathcal{H}^{d+1}_{G_1 \otimes G_2}(t_k) \in M_{\mathcal{H}^{d+1}(G_1)} \\
\Leftrightarrow\ & \mathcal{H}_{G_1 \otimes G_2}(\boldsymbol{s}) \simeq^{d}_{\mathcal{H}(G_1) \otimes \mathcal{H}(G_2)} \mathcal{H}_{G_1 \otimes G_2}(\boldsymbol{t}) \wedge \mathcal{H}_{G_1 \otimes G_2}(\boldsymbol{s}), \mathcal{H}_{G_1 \otimes G_2}(\boldsymbol{t}) \in P_{\mathcal{H}(G_1 \otimes G_2)}
\end{aligned}
$$

which completes the proof. □

3.3.4 Dynamic Strategies

To define the notion of *dynamic strategies*, we just apply the definition of strategies (Definition 2.3.1) in the context of dynamic games:

Definition 3.3.31 (Dynamic strategies)**.** A ***dynamic strategy*** on a dynamic game G is a subset $\sigma \subseteq P_G^{\mathsf{Even}}$, written $\sigma : G$, that satisfies:

- (S1) It is non-empty and even-prefix-closed;
- (S2) It is deterministic (on even-length positions).

A dynamic strategy $\sigma : G$ is said to be ***normalized*** if it contains external moves only, i.e., $\forall \boldsymbol{s} \in \sigma, \forall i \in \{1, 2, \dots, |\boldsymbol{s}|\}. \lambda_G^{\mathbb{N}}(s_i) = 0$.

Since internal moves are conceptually 'invisible' to Opponent, a dynamic strategy $\sigma : G$ must be *externally consistent*: If $\boldsymbol{s}mn, \boldsymbol{s'}m'n' \in \sigma$, $\lambda_G^{\mathbb{N}}(n) = \lambda_G^{\mathbb{N}}(n') = 0$ and $\mathcal{H}_G^{\omega}(\boldsymbol{s}m) = \mathcal{H}_G^{\omega}(\boldsymbol{s'}m')$, then $n = n'$ and $\mathcal{J}_{\boldsymbol{s}mn}^{\ominus\omega}(n) = \mathcal{J}_{\boldsymbol{s'}m'n'}^{\ominus\omega}(n')$. In fact, a stronger property holds (see Theorem 3.3.33 below).

Lemma 3.3.32 (O-determinacy)**.** *Let $\sigma : G$ be a dynamic strategy, and assume that $\boldsymbol{s}, \boldsymbol{s'} \in \sigma$, $d \in \mathbb{N} \cup \{\omega\}$, and $\boldsymbol{s}m, \boldsymbol{s'}m' \in P_G$ are both d-complete. Then, if $\mathcal{H}_G^d(\boldsymbol{s}m) = \mathcal{H}_G^d(\boldsymbol{s'}m')$, then $\boldsymbol{s}m = \boldsymbol{s'}m'$.*

Proof. By induction on $|\boldsymbol{s}|$. The base case $\boldsymbol{s} = \boldsymbol{\epsilon}$ is trivial: For any $d \in \mathbb{N} \cup \{\omega\}$, if $\mathcal{H}_G^d(\boldsymbol{s}m) = \mathcal{H}_G^d(\boldsymbol{s'}m')$, then $\mathcal{H}_G^d(\boldsymbol{s'}m') = \mathcal{H}_G^d(\boldsymbol{s}m) = m$, and so $\boldsymbol{s'}m' = m = \boldsymbol{s}m$.

For the induction step, let $d \in \mathbb{N} \cup \{\omega\}$ be fixed, and assume $\mathcal{H}_G^d(\boldsymbol{s}m) = \mathcal{H}_G^d(\boldsymbol{s'}m')$. We may suppose that $\boldsymbol{s}m = \boldsymbol{t}l\boldsymbol{r}m$, where l is the rightmost O-move in $\boldsymbol{s}$ such that $\lambda_G^{\mathbb{N}}(l) = 0 \vee \lambda_G^{\mathbb{N}}(l) > d$. Then $\mathcal{H}_G^d(\boldsymbol{s'}m') = \mathcal{H}_G^d(\boldsymbol{s}m) = \mathcal{H}_G^d(\boldsymbol{t}).l.\mathcal{H}_G^d(\boldsymbol{r}m)$, and so we may write $\boldsymbol{s'}m' = \boldsymbol{t'_1}.l.\boldsymbol{t'_2}.m'$. Now, $\boldsymbol{t}, \boldsymbol{t'_1} \in \sigma$, $\boldsymbol{t}l, \boldsymbol{t'_1}l \in P_G$, $\mathcal{H}_G^d(\boldsymbol{t}l) = \mathcal{H}_G^d(\boldsymbol{t'_1}l)$, and $\boldsymbol{t}l$ and $\boldsymbol{t'}l'$ are both d-complete; thus, by the induction hypothesis, $\boldsymbol{t}l = \boldsymbol{t'_1}l$. Thus, $\mathcal{H}_G^d(\boldsymbol{t}).l.\mathcal{H}_G^d(\boldsymbol{t'_2}m') = \mathcal{H}_G^d(\boldsymbol{s'}m') = \mathcal{H}_G^d(\boldsymbol{s}m) = \mathcal{H}_G^d(\boldsymbol{t}).l.\mathcal{H}_G^d(\boldsymbol{r}m)$, whence $\boldsymbol{t'_2}$ is of the form $r\boldsymbol{t''_2}$ by the determinacy of σ. Hence, $\boldsymbol{s}m = \boldsymbol{t}l\boldsymbol{r}m$ and $\boldsymbol{s'}m' = \boldsymbol{t}lr\boldsymbol{t''_2}m'$. Finally, if r is external, then so is m by IE-switch, and so $\boldsymbol{s'}m' = \boldsymbol{s}m$; if r is j-internal ($j > d$), then so is m, and so we may apply the axiom DP2 for $i = j-1$ to $\boldsymbol{s}$ and $\boldsymbol{s'}$, concluding that $\boldsymbol{s}m = \boldsymbol{s'}m'$. □

Theorem 3.3.33 (External consistency)**.** *Let σ be a dynamic strategy on a dynamic game G, and assume that $\boldsymbol{s}mn, \boldsymbol{s'}m'n' \in \sigma$ and $d \in \mathbb{N} \cup \{\omega\}$. If $\boldsymbol{s}mn, \boldsymbol{s'}m'n'$ are both d-complete and $\mathcal{H}_G^d(\boldsymbol{s}m) = \mathcal{H}_G^d(\boldsymbol{s'}m')$, then $n = n'$ and $\mathcal{J}_{\boldsymbol{s}mn}^{\ominus d}(n) = \mathcal{J}_{\boldsymbol{s'}m'n'}^{\ominus d}(n')$.*

Proof. Let $\sigma : G$ be a dynamic strategy, $\boldsymbol{s}mn, \boldsymbol{s'}m'n' \in \sigma$ and $d \in \mathbb{N} \cup \{\omega\}$, and assume that $\boldsymbol{s}mn, \boldsymbol{s'}m'n'$ are both d-complete and $\mathcal{H}_G^d(\boldsymbol{s}m) = \mathcal{H}_G^d(\boldsymbol{s'}m')$. By Lemma 3.3.32, we have $\boldsymbol{s}m = \boldsymbol{s'}m'$; thus, by the axiom S2 on σ, we have $n = n'$ and $\mathcal{J}_{\boldsymbol{s}mn}(n) = \mathcal{J}_{\boldsymbol{s'}m'n'}(n')$, whence $\mathcal{J}_{\boldsymbol{s}mn}^{\ominus d}(n) = \mathcal{J}_{\boldsymbol{s'}m'n'}^{\ominus d}(n')$. □

Let us note that even-length positions are not necessarily preserved under the hiding operation on j-sequences (Definition 3.3.3). For instance, let $\boldsymbol{smnt}$ be an even-length position of a dynamic game G such that $\boldsymbol{s}m$ (resp. $n\boldsymbol{t}$) consists of external (resp. internal) moves only. By IE-switch on G, m is an O-move, and so $\mathcal{H}^{\omega}_G(\boldsymbol{s}mn\boldsymbol{t}) = \boldsymbol{s}m$ is of odd-length.

Taking into account this fact, we define:

Definition 3.3.34 (Hiding operation on dynamic strategies)**.** Let G be a dynamic game and $d \in \mathbb{N} \cup \{\omega\}$. Given $\boldsymbol{s} \in P_G$, we define:

$$\boldsymbol{s} \natural \mathcal{H}^d_G \stackrel{\text{df.}}{=} \begin{cases} \mathcal{H}^d_G(\boldsymbol{s}) & \text{if } \boldsymbol{s} \text{ is } d\text{-complete (Definition 3.3.1);} \\ \boldsymbol{t} & \text{otherwise, where } \mathcal{H}^d_G(\boldsymbol{s}) = \boldsymbol{t}m. \end{cases}$$

The ***d-hiding operation*** $\mathcal{H}^d$ ***(on dynamic strategies)*** is then defined by:

$$\mathcal{H}^d : (\sigma : G) \mapsto \{\boldsymbol{s} \natural \mathcal{H}^d_G \mid \boldsymbol{s} \in \sigma\}.$$

Next, we shall show a beautiful fact: $\sigma : G \Rightarrow \mathcal{H}^d(\sigma) : \mathcal{H}^d(G)$ for all $d \in \mathbb{N} \cup \{\omega\}$. For this task, we need the following lemma:

Lemma 3.3.35 (Asymmetry lemma)**.** *Let $\sigma : G$ be a dynamic strategy, and $d \in \mathbb{N} \cup \{\omega\}$. Assume that $\boldsymbol{s}mn \in \mathcal{H}^d(\sigma)$, where $\boldsymbol{s}mn = \boldsymbol{t}m\boldsymbol{u}n\boldsymbol{v} \natural \mathcal{H}^d_G$ with $\boldsymbol{t}m\boldsymbol{u}n\boldsymbol{v} \in \sigma$ not d-complete. Then, we have $\boldsymbol{s}mn = \mathcal{H}^d(\boldsymbol{t}m\boldsymbol{u}n) = \mathcal{H}^d(\boldsymbol{t})mn$.*

Proof. Since $\boldsymbol{t}m\boldsymbol{u}n\boldsymbol{v} \in \sigma$ is not d-complete, we may write $\boldsymbol{v} = \boldsymbol{v}_1 l \boldsymbol{v}_2 r$ with $\lambda^{\mathbb{N}}_G(l) = 0 \vee \lambda^{\mathbb{N}}_G(l) > d$, $0 < \lambda^{\mathbb{N}}_G(r) \leqslant d$ and $0 < \lambda^{\mathbb{N}}_G(x) \leqslant d$ for all moves x in $\boldsymbol{v}_1$ or $\boldsymbol{v}_2$. Then, we have $\boldsymbol{s}mn = \boldsymbol{t}m\boldsymbol{u}n\boldsymbol{v}_1 l \boldsymbol{v}_2 r \natural \mathcal{H}^d_G = \mathcal{H}^d_G(\boldsymbol{t})m\mathcal{H}^d_G(\boldsymbol{u})n = \mathcal{H}^d_G(\boldsymbol{t})mn$. □

We are now ready to establish:

Theorem 3.3.36 (Hiding theorem)**.** *If $\sigma : G$, then $\mathcal{H}^d(\sigma) : \mathcal{H}^d(G)$ for all $d \in \mathbb{N} \cup \{\omega\}$.*

Proof. We first show $\mathcal{H}^d(\sigma) \subseteq P^{\mathsf{Even}}_{\mathcal{H}^d(G)}$. Let $\boldsymbol{s} \in \mathcal{H}^d(\sigma)$, i.e., $\boldsymbol{s} = \boldsymbol{t} \natural \mathcal{H}^d_G$ for some $\boldsymbol{t} \in \sigma$. Let us write $\boldsymbol{t} = \boldsymbol{t'}m$ as the case $\boldsymbol{t} = \boldsymbol{\epsilon}$ is trivial.

- If $\boldsymbol{t}$ is d-complete, then $\boldsymbol{s} = \boldsymbol{t} \natural \mathcal{H}^d_G = \mathcal{H}^d_G(\boldsymbol{t}) \in P_{\mathcal{H}^d(G)}$. Also, since $\boldsymbol{s} = \mathcal{H}^d_G(\boldsymbol{t'})m$ and m is a P-move, $\boldsymbol{s}$ must be of even-length by alternation on $\mathcal{H}^d(G)$.
- If $\boldsymbol{t}$ is not d-complete, then we may write $\boldsymbol{t} = \boldsymbol{t''}m_0 m_1 \dots m_k$, where $m_k = m$, $\boldsymbol{t''}m_0$ is d-complete, and $0 < \lambda^{\mathbb{N}}_G(m_i) \leqslant d$ for $i = 1, 2, \dots, k$. By IE-switch, m_0 is an O-move, and thus $\boldsymbol{s} = \mathcal{H}^d_G(\boldsymbol{t''}) \in P_{\mathcal{H}^d(G)}$ is of even-length.

It remains to verify the axioms S1 and S2. For the axiom S1, $\mathcal{H}^d(\sigma)$ is clearly non-empty as $\boldsymbol{\epsilon} \in \mathcal{H}^d(\sigma)$. For the even-prefix-closure, let $\boldsymbol{smn} \in \mathcal{H}^d(\sigma)$; we have to show $\boldsymbol{s} \in \mathcal{H}^d(\sigma)$. We have some $\boldsymbol{tmunv} \in \sigma$ such that $\boldsymbol{tmunv} \natural \mathcal{H}^d_G = \boldsymbol{smn}$. By Lemma 3.3.35, $\boldsymbol{smn} = \mathcal{H}^d_G(\boldsymbol{t})mn$, whence $\boldsymbol{s} = \mathcal{H}^d_G(\boldsymbol{t})$. For $\boldsymbol{t}m$ is d-complete, so is $\boldsymbol{t}$ by IE-switch. Therefore, $\boldsymbol{s} = \mathcal{H}^d_G(\boldsymbol{t}) = \boldsymbol{t} \natural \mathcal{H}^d_G \in \mathcal{H}^d(\sigma)$.

Finally for the axiom S2, let $\boldsymbol{smn}, \boldsymbol{smn'} \in \mathcal{H}^d(\sigma)$; we have to show $n = n'$ and $\mathcal{J}^{\ominus d}_{\boldsymbol{sm}}(n) = \mathcal{J}^{\ominus d}_{\boldsymbol{sm}}(n')$. By the definition, $\boldsymbol{smn} = \boldsymbol{tmunv} \natural \mathcal{H}^d_G$, $\boldsymbol{smn'} = \boldsymbol{t'mu'}n'\boldsymbol{v'} \natural \mathcal{H}^d_G$ for some $\boldsymbol{tmunv}, \boldsymbol{t'mu'}n'\boldsymbol{v'} \in \sigma$. Then, by Lemma 3.3.35, $\boldsymbol{smn} = \mathcal{H}^d_G(\boldsymbol{tmu})n$ and $\boldsymbol{smn'} = \mathcal{H}^d_G(\boldsymbol{t'mu'})n'$. Therefore, by Theorem 3.3.33, we may conclude that $n = n'$ and $\mathcal{J}^{\ominus d}_{\boldsymbol{smn}}(n) = \mathcal{J}^{\ominus d}_{\boldsymbol{smn'}}(n')$, completing the proof. $\square$

Next, let us just apply the definitions of ***totality***, ***innocence***, ***well-bracketing***, ***noetherianity*** and ***winning*** on strategies in Chapter 2 to dynamic strategies. We shall see shortly that these constraints except totality are preserved under the hiding operation (Corollary 3.3.40). On the other hand, let us generalize *validity* of strategies (Definition 2.3.9) in order to preserve it under the hiding operation:

Definition 3.3.37 (Validity of dynamic strategies)**.** A dynamic strategy $\sigma : G$ is ***valid*** iff $\forall d \in \mathbb{N} \cup \{\omega\}, \boldsymbol{s}, \boldsymbol{t} \in \sigma, \boldsymbol{sm}, \boldsymbol{tl} \in P_G.\, \boldsymbol{sm} \simeq^d_G \boldsymbol{tl} \Rightarrow \forall \boldsymbol{smn} \in \sigma.\, \exists \boldsymbol{tlr} \in \sigma.\, \boldsymbol{smn} \simeq^d_G \boldsymbol{tlr}$.

Definition 3.3.37 suggests that we should identify 'essentially the same' dynamic strategies as follows:

Definition 3.3.38 (Identification of dynamic strategies)**.** The ***identification of dynamic strategies*** on a dynamic game G, written $\simeq_G$, is the relation between dynamic strategies on G defined for any $\sigma, \tau : G$ by:

$$\sigma \simeq_G \tau \overset{\text{df.}}{\Leftrightarrow} \forall d \in \mathbb{N} \cup \{\omega\}, \boldsymbol{s} \in \sigma, \boldsymbol{t} \in \tau.\, \boldsymbol{sm} \simeq^d_G \boldsymbol{tl} \Rightarrow \forall \boldsymbol{smn} \in \sigma.\, \exists \boldsymbol{tlr} \in \tau.\, \boldsymbol{smn} \simeq^d_G \boldsymbol{tlr} \\ \wedge \forall \boldsymbol{tlr} \in \tau.\, \exists \boldsymbol{smn} \in \sigma.\, \boldsymbol{tlr} \simeq^d_G \boldsymbol{smn}.$$

Clearly, a dynamic strategy $\sigma : G$ is valid iff it is identified with itself, i.e., $\sigma \simeq_G \sigma$. Recall that normalized dynamic strategies are equivalent to strategies (in Chapter 2), and validity and identification of normalized dynamic strategies coincide respectively with validity and identification of strategies (Definitions 2.3.6 and 2.3.3); thus, we have generalized the existing concepts on strategies in a conservative manner.

Moreover, as in the case of games and strategies (in Chapter 2), we may show that the identification $\simeq_G$ of dynamic strategies on any dynamic game G is a PER:

Proposition 3.3.39 (PERs on dynamic strategies)**.** *Given a dynamic game G, the identification $\simeq_G$ of dynamic strategies on G is a PER.*

Proof. Similar to the proof of Corollary 2.3.5. □

Now, let us establish:

Corollary 3.3.40 (Preservation of constraints on dynamic strategies under hiding)**.** *If a dynamic strategy $\sigma : G$ is valid, innocent, well-bracketed or noetherian, then so is $\mathcal{H}^d(\sigma) : \mathcal{H}^d(G)$ for all $d \in \mathbb{N} \cup \{\omega\}$.*

Proof. Theorem 3.3.36 implies that:

- Preservation of validity follows from Lemma 3.3.32 and the axiom DI3 on G;
- Preservation of innocence and noetherianity holds because $\lceil \mathcal{H}^d_G(\boldsymbol{s}m) \rceil_{\mathcal{H}^d(G)}$ is a j-subsequence of $\mathcal{H}^d_G(\lceil \boldsymbol{s}m \rceil_G)$ for any $\boldsymbol{s}m \in P_G^{\mathsf{Odd}}$;
- Well-bracketing is preserved under $\mathcal{H}^d$ because both of the question and the answer of each 'QA-pair' are either deleted or retained

which completes the proof. □

Remark. Totality of dynamic strategies is *not* preserved under the hiding operation $\mathcal{H}$ on dynamic strategies. For instance, consider any total dynamic strategy that always performs a 1-internal P-move, which is no longer total when $\mathcal{H}$ is applied.

At the end of the present section, we establish an inductive property of the d-hiding operation on dynamic strategies for each $d \in \mathbb{N} \cup \{\omega\}$:

Notation. Given a dynamic strategy $\sigma : G$ and a number $d \in \mathbb{N} \cup \{\omega\}$, we define $\sigma^d_\downarrow \overset{\text{df.}}{=} \{ \boldsymbol{s} \in \sigma \mid \boldsymbol{s} \text{ is } d\text{-complete} \}$ and $\sigma^d_\uparrow \overset{\text{df.}}{=} \sigma \setminus \sigma^d_\downarrow$.

Lemma 3.3.41 (Hiding and complete positions)**.** *Let $\sigma : G$ be a dynamic strategy. Given $i, d \in \mathbb{N}$ such that $i \geqslant d$, $\mathcal{H}^i(\sigma) = \mathcal{H}^i(\sigma^d_\downarrow) \overset{\text{df.}}{=} \{ \boldsymbol{s} \natural \mathcal{H}^i_G \mid \boldsymbol{s} \in \sigma^d_\downarrow \}$.*

Proof. The inclusion $\mathcal{H}^i(\sigma^d_\downarrow) \subseteq \mathcal{H}^i(\sigma)$ is obvious. For the opposite inclusion, let $\boldsymbol{s} \in \mathcal{H}^i(\sigma)$, i.e., $\boldsymbol{s} = \boldsymbol{t} \natural \mathcal{H}^i_G$ for some $\boldsymbol{t} \in \sigma$; we have to show $\boldsymbol{s} \in \mathcal{H}^i(\sigma^d_\downarrow)$. If $\boldsymbol{t} \in \sigma^d_\downarrow$, then we are done; so assume otherwise. Also, if there is no external or j-internal move with $j > i$ other than the first move m_0 in $\boldsymbol{t}$, then $\boldsymbol{s} = \boldsymbol{\epsilon} \in \mathcal{H}^i(\sigma^d_\downarrow)$; so assume otherwise. As a consequence, we may write $\boldsymbol{t} = m_0 \boldsymbol{t_1} m n \boldsymbol{t_2} r$, where $\boldsymbol{t_2} r$ consists of only j-internal moves with $0 < j \leqslant i$, and m and n are P- and O-moves, respectively, such that $\lambda^{\mathbb{N}}_G(m) = \lambda^{\mathbb{N}}_G(n) = 0 \vee \lambda^{\mathbb{N}}_G(m) = \lambda^{\mathbb{N}}_G(n) > i$. Now, we take $m_0 \boldsymbol{t_1} m \in \sigma^d_\downarrow$ that satisfies $m_0 \boldsymbol{t_1} m \natural \mathcal{H}^i_G = m_0 \mathcal{H}^i_G(\boldsymbol{t_1}) m = \boldsymbol{t} \natural \mathcal{H}^i_G = \boldsymbol{s}$, whence $\boldsymbol{s} \in \mathcal{H}^i(\sigma^d_\downarrow)$. □

We are now ready to show:

Lemma 3.3.42 (Stepwise hiding on dynamic strategies)**.** *Let $\sigma : G$ be a dynamic strategy. Then, $\mathcal{H}^{i+1}(\sigma) = \mathcal{H}^1(\mathcal{H}^i(\sigma))$ for all $i \in \mathbb{N}$.*

Proof. We first show the inclusion $\mathcal{H}^{i+1}(\sigma) \subseteq \mathcal{H}^1(\mathcal{H}^i(\sigma))$. By Lemma 3.3.41, we may write any element of the set $\mathcal{H}^{i+1}(\sigma)$ as $\boldsymbol{s} \natural \mathcal{H}_G^{i+1}$ for some $\boldsymbol{s} \in \sigma_{\downarrow}^{i+1}$. Then observe that:

$$\boldsymbol{s} \natural \mathcal{H}_G^{i+1} = \mathcal{H}_G^{i+1}(\boldsymbol{s}) = \mathcal{H}_{\mathcal{H}^i(G)}(\mathcal{H}_G^i(\boldsymbol{s})) = (\boldsymbol{s} \natural \mathcal{H}_G^i) \natural \mathcal{H}_{\mathcal{H}^i(G)}^1 \in \mathcal{H}^1(\mathcal{H}^i(\sigma)).$$

For the opposite inclusion $\mathcal{H}^1(\mathcal{H}^i(\sigma)) \subseteq \mathcal{H}^{i+1}(\sigma)$, again by Lemma 3.3.41, we may write any element of $\mathcal{H}^1(\mathcal{H}^i(\sigma))$ as $(\boldsymbol{s} \natural \mathcal{H}_G^i) \natural \mathcal{H}_{\mathcal{H}^i(G)}^1$ for some $\boldsymbol{s} \in \sigma_{\downarrow}^i$. We have to show that $(\boldsymbol{s} \natural \mathcal{H}_G^i) \natural \mathcal{H}_{\mathcal{H}^i(G)}^1 \in \mathcal{H}^{i+1}(\sigma)$. If $\boldsymbol{s} \in \sigma_{\downarrow}^{i+1}$, then it is completely analogous to the above argument; so assume otherwise. Also, if an external or j-internal move with $j > i+1$ in $\boldsymbol{s}$ is only the first move m_0, then $(\boldsymbol{s} \natural \mathcal{H}_G^i) \natural \mathcal{H}_{\mathcal{H}^i(G)}^1 = \boldsymbol{\epsilon} \in \mathcal{H}^{i+1}(\sigma)$; thus assume othewise. Now, we may write:

$$\boldsymbol{s} = \boldsymbol{s'} m n m_1 m_2 \dots m_{2k} r$$

where $\lambda_G^{\mathbb{N}}(r) = i+1$, $m_1, m_2, \dots, m_{2k}$ are j-internal with $0 < j \leqslant i+1$, and m and n are external or j-internal P- and O-moves with $j > i+1$, respectively. Then,

$$\begin{aligned}(\boldsymbol{s} \natural \mathcal{H}_G^i) \natural \mathcal{H}_{\mathcal{H}^i(G)}^1 &= \mathcal{H}_G^i(\boldsymbol{s}) \natural \mathcal{H}_{\mathcal{H}^i(G)}^1 \\ &= \mathcal{H}_{\mathcal{H}^i(G)}(\mathcal{H}_G^i(\boldsymbol{s'})).m \\ &= \mathcal{H}_G^{i+1}(\boldsymbol{s'}).m \text{ (by Lemma 3.3.7)} \\ &= \boldsymbol{s} \natural \mathcal{H}_G^{i+1} \in \mathcal{H}^{i+1}(\sigma)\end{aligned}$$

which completes the proof. □

Thus, as in the case of dynamic games, we may focus on the 1-hiding operation $\mathcal{H}^1$ on dynamic strategies.

Convention. Henceforth, we write $\mathcal{H}$ for $\mathcal{H}^1$ and call it the ***hiding operation*** on dynamic strategies; $\mathcal{H}^i$ ($i \in \mathbb{N}$) denotes the i-times iteration of $\mathcal{H}$.

3.3.5 Constructions on Dynamic Strategies

Next, let us consider constructions on dynamic strategies. However, since dynamic strategies are just 'strategies on dynamic games', they are clearly closed under all the constructions on strategies introduced in Chapter 2.

Nevertheless, to give a categorical structure, specifically a CCBoC, of dynamic games and strategies in Section 3.4, we need to generalize *pairing* (Definition 2.3.20) and *promotion* (Definition 2.3.23) of strategies; in fact, we have generalized product (Definition 2.2.23) and exponential (Definition 2.2.25) of games respectively to *pairing* (Definition 3.3.19) and *promotion* (Definition 3.3.21) of dynamic games for this aim.

Definition 3.3.43 (Pairing of dynamic strategies)**.** Given dynamic strategies $\phi : L$ and $\psi : R$ such that $\mathcal{H}^\omega(L) \trianglelefteq C \multimap A$ and $\mathcal{H}^\omega(R) \trianglelefteq C \multimap B$ for some normalized dynamic games A, B and C, their ***pairing*** $\langle \phi, \psi \rangle$ is defined by:

$$\langle \phi, \psi \rangle \stackrel{\mathrm{df.}}{=} \{ \boldsymbol{s} \in \mathscr{L}_{\langle L, R \rangle} \mid (\boldsymbol{s} \upharpoonright L \in \phi \wedge \boldsymbol{s} \upharpoonright R = \boldsymbol{\epsilon}) \vee (\boldsymbol{s} \upharpoonright R \in \psi \wedge \boldsymbol{s} \upharpoonright L = \boldsymbol{\epsilon}) \}.$$

Theorem 3.3.44 (Well-defined pairing of dynamic strategies)**.** *Given dynamic strategies $\phi : L$ and $\psi : R$ with $\mathcal{H}^\omega(L) \trianglelefteq C \multimap A$ and $\mathcal{H}^\omega(R) \trianglelefteq C \multimap B$ for some normalized dynamic games A, B and C, $\langle \phi, \psi \rangle$ is a dynamic strategy on $\langle L, R \rangle$. If ϕ and ψ are innocent (resp. wb, total, noetherian), then so is $\langle \phi, \psi \rangle$. Given $\phi' : L$ and $\psi' : R$ with $\phi \simeq_L \phi'$ and $\psi \simeq_R \psi'$, we have $\langle \phi, \psi \rangle \simeq_{\langle L, R \rangle} \langle \phi', \psi' \rangle$.*

Proof. Straightforward. □

Definition 3.3.45 (Promotion of dynamic strategies)**.** Given a dynamic strategy $\varphi : G$ such that $\mathcal{H}^\omega(G) \trianglelefteq \, !A \multimap B$ for some normalized dynamic games A and B, its ***promotion*** $\varphi^\dagger$ is defined by:

$$\varphi^\dagger \stackrel{\mathrm{df.}}{=} \{ \boldsymbol{s} \in \mathscr{L}_{G^\dagger} \mid \forall i \in \mathbb{N}. \, \boldsymbol{s} \upharpoonright i \in \varphi \}.$$

Theorem 3.3.46 (Well-defined promotion on dynamic strategies)**.** *Given a dynamic strategy $\varphi : G$ such that $\mathcal{H}^\omega(G) \trianglelefteq \, !A \multimap B$ for some normalized dynamic games A and B, $\varphi^\dagger$ is a dynamic strategy on $G^\dagger$. If φ is innocent (resp. wb, total, noetherian), then so is $\varphi^\dagger$. Given $\varphi' : G$ with $\varphi \simeq_G \varphi'$, we have $\varphi^\dagger \simeq_{G^\dagger} \varphi'^\dagger$.*

Proof. Straightforward. □

Next, let us introduce a new construction on dynamic strategies:

Definition 3.3.47 (Concatenation on dynamic strategies)**.** Let $\sigma : J$ and $\tau : K$ be dynamic strategies such that $\mathcal{H}^\omega(J) \trianglelefteq A \multimap B$ and $\mathcal{H}^\omega(K) \trianglelefteq B \multimap C$ for some normalized dynamic games A, B and C. Their ***concatenation*** $\sigma \ddagger \tau$ is defined by:

$$\sigma \ddagger \tau \stackrel{\mathrm{df.}}{=} \{ \boldsymbol{s} \in \mathscr{J}_{J \ddagger K} \mid \boldsymbol{s} \upharpoonright J \in \sigma, \boldsymbol{s} \upharpoonright K \in \tau, \boldsymbol{s} \upharpoonright B_{[1]}, B_{[2]} \in pr_B \}.$$

Theorem 3.3.48 (Well-defined concatenation on dynamic strategies)**.** *Let $\sigma : J$ and $\tau : K$ be dynamic strategies such that $\mathcal{H}^\omega(J) \trianglelefteq A \multimap B$ and $\mathcal{H}^\omega(K) \trianglelefteq B \multimap C$, where A, B and C are normalized dynamic games. Then, $\sigma \ddagger \tau : J \ddagger K$ and $\sigma ; \tau = \mathcal{H}^\omega(\sigma \ddagger \tau) : A \multimap C$. If σ and τ are innocent (resp. wb, noetherian, winning), then so is $\sigma \ddagger \tau$. Given $\sigma' : J$ and $\tau' : K$ with $\sigma \simeq_J \sigma'$ and $\tau \simeq_K \tau'$, we have $\sigma \ddagger \tau \simeq_{J \ddagger K} \sigma' \ddagger \tau'$.*

Proof. We just show the first statement as the other ones are straightforward. It then suffices to prove $\sigma \ddagger \tau : J \ddagger K$ and $\mathcal{H}^\omega(\sigma \ddagger \tau) = \sigma ; \tau$ since it implies $\sigma ; \tau = \mathcal{H}^\omega(\sigma \ddagger \tau) : \mathcal{H}^\omega(J \ddagger K) \trianglelefteq A \multimap C$ by Lemmata 3.3.30 and 3.3.36. However, $\mathcal{H}^\omega(\sigma \ddagger \tau) = \sigma ; \tau$ is immediate from the definition of concatenation; thus, we focus on $\sigma \ddagger \tau : J \ddagger K$.

First, we have $\sigma \ddagger \tau \subseteq P_{J \ddagger K}$ as any $\boldsymbol{s} \in \sigma \ddagger \tau$ satisfies $\boldsymbol{s} \in \mathscr{J}_{J \ddagger K}$, $\boldsymbol{s} \upharpoonright J \in \sigma \subseteq P_J$, $\boldsymbol{s} \upharpoonright K \in \tau \subseteq P_K$ and $\boldsymbol{s} \upharpoonright B_{[1]}, B_{[2]} \in pr_B$. It is also immediate that such $\boldsymbol{s}$ is of even-length. It remains to verify the axioms S1 and S2. For this, we need:

($\diamondsuit$) Each $\boldsymbol{s} \in \sigma \ddagger \tau$ consists of adjacent pairs mn such that $m, n \in M_J$ or $m, n \in M_K$.

Proof of the claim $\diamondsuit$. By induction on $|\boldsymbol{s}|$. The base case is trivial. For the inductive step, let $\boldsymbol{s}mn \in \sigma \ddagger \tau$. If $m \in M_J$, then $(\boldsymbol{s} \upharpoonright J).m.(n \upharpoonright J) \in \sigma$, where $\boldsymbol{s} \upharpoonright J$ is of even-length by the induction hypothesis. Thus, we must have $n \in M_J$. If $m \in M_K$, then $n \in M_K$ by the same argument. □

- (S1) Clearly, $\boldsymbol{\epsilon} \in \sigma \ddagger \tau$, so $\sigma \ddagger \tau$ is non-empty. For even-prefix-closure, assume $\boldsymbol{s}mn \in \sigma \ddagger \tau$. Then, by the claim $\diamondsuit$, either $m, n \in M_J$ or $m, n \in M_K$. In either case, it is straightforward to see that $\boldsymbol{s} \in P_{J \ddagger K}$, $\boldsymbol{s} \upharpoonright J \in \sigma$, $\boldsymbol{s} \upharpoonright K \in \tau$ and $\boldsymbol{s} \upharpoonright B_{[1]}, B_{[2]} \in pr_B$, i.e., $\boldsymbol{s} \in \sigma \ddagger \tau$.

- (S2) Assume $\boldsymbol{s}mn, \boldsymbol{s}mn' \in \sigma \ddagger \tau$. By the claim $\diamondsuit$, either $m, n, n' \in M_J$ or $m, n, n' \in M_K$. In the former case, $(\boldsymbol{s} \upharpoonright J).mn, (\boldsymbol{s} \upharpoonright J).mn' \in \sigma$. Thus, $n = n'$ and $\mathcal{J}_{\boldsymbol{s}mn}(n) = \mathcal{J}_{(\boldsymbol{s} \upharpoonright J).mn}(n) = \mathcal{J}_{(\boldsymbol{s} \upharpoonright J).mn'}(n') = \mathcal{J}_{\boldsymbol{s}mn'}(n')$ by S2 on σ, where note that n and n' are both P-moves and thus non-initial in J. The latter case may be handled similarly.

Therefore, we have shown that $\sigma \ddagger \tau : J \ddagger K$. □

Note that totality of (dynamic) strategies is *not* preserved under composition, but it is preserved under concatenation. This phenomenon is essentially because totality is not preserved under the hiding operation as already remarked above.

Now, for completeness, let us explicitly define the rather trivial *currying*:

Definition 3.3.49 (Currying of dynamic strategies)**.** Given a dynamic strategy $\sigma : G$ such that $\mathcal{H}^{\omega}(G) \trianglelefteq A \otimes B \multimap C$ for some normalized dynamic games A, B and C, the ***currying*** $\Lambda(\sigma) : \Lambda(G)$ of σ is defined to be σ up to 'tags'.

Proposition 3.3.50 (Well-defined currying on dynamic strategies)**.** *Dynamic strategies are closed under currying, and currying preserves innocence, well-bracketing, noetherianity and identification of dynamic strategies.*

Proof. Obvious. □

At the end of the present section, as in the case of games, we establish the *hiding lemma* on strategies. We first need the following:

Lemma 3.3.51 (Hiding on legal positions in the second form)**.** *For any dynamic arena G and number $d \in \mathbb{N} \cup \{\omega\}$, we have $L_{\mathcal{H}^d(G)} = \{ \boldsymbol{s} \natural \mathcal{H}^d_G \mid \boldsymbol{s} \in L_G \}$.*

Proof. Observe that:

$$\begin{aligned}
\{ \boldsymbol{s} \natural \mathcal{H}^d_G \mid \boldsymbol{s} \in L_G \} &= \{ \boldsymbol{s} \natural \mathcal{H}^d_G \mid \boldsymbol{s} \in L_G, \boldsymbol{s} \text{ is } d\text{-complete} \} \\
&= \{ \mathcal{H}^d_G(\boldsymbol{s}) \mid \boldsymbol{s} \in L_G, \boldsymbol{s} \text{ is } d\text{-complete} \} \\
&= \{ \mathcal{H}^d_G(\boldsymbol{s}) \mid \boldsymbol{s} \in L_G \} \text{ (by the same argument as above)} \\
&= L_{\mathcal{H}^d(G)} \text{ (by Corollary 3.3.16)}
\end{aligned}$$

completing the proof. □

Notation. We write $\spadesuit_{i \in I}$, where I is $\{1\}$ or $\{1, 2\}$, for a construction on dynamic strategies, i.e., $\spadesuit_{i \in I}$ is either $\otimes$, ;, $(\_)^{\dagger}$, $\langle \_, \_ \rangle$, $\ddagger$ or Λ.

Lemma 3.3.52 (Hiding lemma on dynamic strategies)**.** *Let $\spadesuit_{i \in I}$ be a construction on dynamic strategies, and $\sigma_i : G_i$ for each $i \in I$. Then, for all $d \in \mathbb{N} \cup \{\omega\}$, we have:*

1. $\mathcal{H}^d(\spadesuit_{i \in I} \sigma_i) = \spadesuit_{i \in I} \mathcal{H}^d(\sigma_i)$ *if* $\spadesuit_{i \in I}$ *is* $\otimes$, ;, $(\_)^{\dagger}$, $\langle \_, \_ \rangle$ *or* Λ;
2. $\mathcal{H}^d(\sigma_1 \ddagger \sigma_2) = \mathcal{H}^d(\sigma_1) \ddagger \mathcal{H}^d(\sigma_2)$ *if* $\mathcal{H}^d(\sigma_1 \ddagger \sigma_2)$ *is not normalized;*
3. $\mathcal{H}^d(\sigma_1 \ddagger \sigma_2) = \mathcal{H}^d(\sigma_1); \mathcal{H}^d(\sigma_2)$ *otherwise.*

Proof. As in the case of dynamic games, it suffices to assume $d = 1$. Here, we just focus on pairing since the other constructions may be handled analogously.

Let $\sigma_i : G_i$, $i = 1, 2$, be dynamic strategies such that $\mathcal{H}^\omega(G_1) \trianglelefteqslant C \multimap A$, $\mathcal{H}^\omega(G_2) \trianglelefteqslant C \multimap B$ for some normalized dynamic games A, B and C. For $\mathcal{H}(\langle \sigma_1, \sigma_2 \rangle) \subseteq \langle \mathcal{H}(\sigma_1), \mathcal{H}(\sigma_2) \rangle$, observe:

$$\begin{aligned}
&\boldsymbol{s} \in \mathcal{H}(\langle \sigma_1, \sigma_2 \rangle) \\
\Rightarrow\ & \exists \boldsymbol{t} \in \langle \sigma_1, \sigma_2 \rangle .\, \boldsymbol{t} \natural \mathcal{H}^1_{\langle G_1, G_2 \rangle} = \boldsymbol{s} \\
\Rightarrow\ & \exists \boldsymbol{t} \in \mathscr{L}_{\langle G_1, G_2 \rangle} .\, \boldsymbol{t} \natural \mathcal{H}^1_{\langle G_1, G_2 \rangle} = \boldsymbol{s} \wedge ((\boldsymbol{t} \upharpoonright G_1 \in \sigma_1 \wedge \boldsymbol{t} \upharpoonright G_2 = \boldsymbol{\epsilon}) \vee (\boldsymbol{t} \upharpoonright G_2 \in \sigma_2 \wedge \boldsymbol{t} \upharpoonright G_1 = \boldsymbol{\epsilon})) \\
\Rightarrow\ & \boldsymbol{s} \in \mathscr{L}_{\mathcal{H}(\langle G_1, G_2 \rangle)} \wedge (\boldsymbol{s} \upharpoonright \mathcal{H}(G_1) \in \mathcal{H}(\sigma_1) \wedge \boldsymbol{s} \upharpoonright \mathcal{H}(G_2) = \boldsymbol{\epsilon}) \\
& \vee (\boldsymbol{s} \upharpoonright \mathcal{H}(G_2) \in \mathcal{H}(\sigma_2) \wedge \boldsymbol{s} \upharpoonright \mathcal{H}(G_1) = \boldsymbol{\epsilon})) \text{ (by Lemma 3.3.51)} \\
\Rightarrow\ & \boldsymbol{s} \in \langle \mathcal{H}(\sigma_1), \mathcal{H}(\sigma_2) \rangle .
\end{aligned}$$

Next, we show the converse:

$$\begin{aligned}
&\boldsymbol{s} \in \langle \mathcal{H}(\sigma_1), \mathcal{H}(\sigma_2) \rangle \\
\Rightarrow\ & \boldsymbol{s} \in \mathscr{L}_{\mathcal{H}(\langle G_1, G_2 \rangle)} \wedge (\boldsymbol{s} \upharpoonright \mathcal{H}(G_1) \in \mathcal{H}(\sigma_1) \wedge \boldsymbol{s} \upharpoonright \mathcal{H}(G_2) = \boldsymbol{\epsilon}) \\
& \vee (\boldsymbol{s} \upharpoonright \mathcal{H}(G_2) \in \mathcal{H}(\sigma_2) \wedge \boldsymbol{s} \upharpoonright \mathcal{H}(G_1) = \boldsymbol{\epsilon})) \\
\Rightarrow\ & (\exists \boldsymbol{u} \in \sigma_1 .\, \boldsymbol{u} \natural \mathcal{H}^1_{G_1} = \boldsymbol{s} \upharpoonright \mathcal{H}(G_1) \wedge \boldsymbol{u} \upharpoonright G_2 = \boldsymbol{\epsilon}) \\
& \vee (\exists \boldsymbol{v} \in \sigma_2 .\, \boldsymbol{v} \natural \mathcal{H}^1_{G_2} = \boldsymbol{s} \upharpoonright \mathcal{H}(G_2) \wedge \boldsymbol{v} \upharpoonright \mathcal{H}(G_1) = \boldsymbol{\epsilon}) \\
\Rightarrow\ & \exists \boldsymbol{w} \in \langle \sigma_1, \sigma_2 \rangle .\, \boldsymbol{w} \natural \mathcal{H}^1_{\langle G_1, G_2 \rangle} = \boldsymbol{s} \\
\Rightarrow\ & \boldsymbol{s} \in \mathcal{H}(\langle \sigma_1, \sigma_2 \rangle)
\end{aligned}$$

which completes the proof. □

Finally, as a technical preparation for the next section, let us define:

Definition 3.3.53 (Dereliction games)**.** The ***dereliction game*** on a dynamic game G is the dynamic subgame $\varXi_G \trianglelefteqslant G \Rightarrow G$ given by $M_{\varXi_G} \overset{\text{df.}}{=} M_{G \Rightarrow G}$, $\lambda_{\varXi_G} \overset{\text{df.}}{=} \lambda_{G \Rightarrow G}$, $\vdash_{\varXi_G} \overset{\text{df.}}{=} \vdash_{G \Rightarrow G}$ (n.b., we have to take the obvious subset of $\vdash_{G \Rightarrow G}$ as $\vdash_{\varXi_G}$ if we would like to make $\varXi_G$ economical), $P_{\varXi_G} \overset{\text{df.}}{=} \{ \boldsymbol{s} \in P_{G \Rightarrow G} \mid \forall \boldsymbol{t} \preceq \boldsymbol{s} .\, \mathsf{Even}(\boldsymbol{t}) \Rightarrow \boldsymbol{t} \upharpoonright G_{[1]} = \boldsymbol{t} \upharpoonright G_{[2]} \}$, and $\simeq_{\varXi_G} \overset{\text{df.}}{=} \simeq_{G \Rightarrow G} \upharpoonright P_{\varXi_G} \times P_{\varXi_G}$. Given normalized dynamic games A and B, we define:

- $\varPi_1^{A,B} \trianglelefteqslant A \& B \Rightarrow A$ to be $\varXi_A$ up to 'tags', where we often abbreviate it as $\varPi_1$;
- $\varPi_2^{A,B} \trianglelefteqslant A \& B \Rightarrow B$ to be $\varXi_B$ up to 'tags', where we often abbreviate it as $\varPi_2$;
- $\varUpsilon_{A,B} \trianglelefteqslant B^A \& A \Rightarrow B$ to be $\varXi_{A \Rightarrow B}$ up to 'tags', where we often omit A, B.

That is, the dereliction game $\varXi_G$ on a dynamic game G is the subgame of $G \Rightarrow G$, in which only the plays by the dereliction der_G are possible.

Lemma 3.3.54 (D-lemma). *Given normalized dynamic games A, B, C, $L \trianglelefteqslant C \Rightarrow A$, $R \trianglelefteqslant C \Rightarrow B$, $P \trianglelefteqslant C \Rightarrow A\&B$, $U \trianglelefteqslant A\&B \Rightarrow C$ and $V \trianglelefteqslant A \Rightarrow C^B$, we have:*

$$\begin{aligned}
\langle L, R \rangle^\dagger ; \mathit{\Pi}_1^{A,B} &= L \\
\langle L, R \rangle^\dagger ; \mathit{\Pi}_2^{A,B} &= R \\
\langle P^\dagger ; \mathit{\Pi}_1^{A,B}, P^\dagger ; \mathit{\Pi}_2^{A,B} \rangle &= P \\
\langle (\mathit{\Pi}_1^{A,B})^\dagger ; \Lambda(U), \mathit{\Pi}_2^{A,B} \rangle^\dagger ; \mathit{\Upsilon}_{B,C} &= U \\
\Lambda(\langle (\mathit{\Pi}_1^{A,B})^\dagger ; V, \mathit{\Pi}_2^{A,B} \rangle^\dagger ; \mathit{\Upsilon}_{B,C}) &= V.
\end{aligned}$$

Proof. Straightforward. □

3.4 Dynamic Game Semantics of Finitary PCF

This section is the climax of the present chapter: It first establishes a game-semantic instance of a CCBoC $\mathcal{LDG}$ (Definition 3.4.1) and a standard structure $\mathcal{S}_\mathcal{G}$ for FPCF in $\mathcal{LDG}$ (Definition 3.4.3), and then shows that the induced interpretation $[\![\_]\!]_{\mathcal{LDG}}^{\mathcal{S}_\mathcal{G}}$ satisfies the PDCP (Theorem 3.4.4), and thus the DCP by Theorem 3.2.17, giving the first instance of dynamic game semantics.

3.4.1 Dynamic Game Semantics of Finitary PCF

Let us first give the CCBoC $\mathcal{LDG}$ of dynamic games and strategies:

Definition 3.4.1 (The CCBoC $\mathcal{LDG}$). The CCBoC $\mathcal{LDG} = (\mathcal{LDG}, \mathcal{H})$ is defined by:

- Objects are normalized dynamic games;
- A β-morphisms $A \to B$ is a pair $(J, [\phi])$ of a dynamic game J and the equivalence class $[\phi] \overset{\mathrm{df.}}{=} \{\psi : J \mid \psi \text{ is winning}, \psi \simeq_J \phi\}$ of a valid, winning dynamic strategy $\phi : J$ with respect to the identification $\simeq_J$ of dynamic strategies on J such that $\mathcal{H}^\omega(J) \trianglelefteqslant A \Rightarrow B$ and $\mathcal{H}^\omega(J^\dagger) = \mathcal{H}^\omega(J)^\dagger$;
- The β-composition $A \overset{(J,[\phi])}{\to} B \overset{(K,[\psi])}{\to} C$ is the pair $(J^\dagger \ddagger K, [\phi^\dagger \ddagger \psi])$;
- The β-identity $id_A : A \to A$ on each object A is the pair $(A \Rightarrow A, [der_A])$;
- The evaluation $\mathcal{H}$ maps morphisms $(J, [\phi]) : A \to B$ to $\mathcal{H}(J, [\phi]) \overset{\mathrm{df.}}{=} (\mathcal{H}(J), [\mathcal{H}(\phi)])$;
- The β-terminal object is the terminal game T (Example 2.2.11) modified to a normalized dynamic game in the obvious manner;

- β-product and β-exponential are respectively defined by $A \times B \stackrel{\text{df.}}{=} A \& B$ and $B^A \stackrel{\text{df.}}{=} A \Rightarrow B = {!A} \multimap B$ for any objects A and B;

- β-pairing is defined by $\langle (L, [\alpha]), (R, [\beta]) \rangle \stackrel{\text{df.}}{=} (\langle L, R \rangle, [\langle \alpha, \beta \rangle]) : C \to A \& B$ for any objects A, B and C, and morphisms $(L, [\alpha]) : C \to A$ and $(R, [\beta]) : C \to B$;

- The β-projections $\pi_1 : A \& B \to A$ and $\pi_2 : A \& B \to B$ are respectively the pairs $(\mathit{\Pi}_1^{A,B}, [\varpi_1^{A,B}])$ and $(\mathit{\Pi}_2^{A,B}, [\varpi_2^{A,B}])$ for any objects A and B, where $\varpi_1^{A,B} : \mathit{\Pi}_1^{A,B}$ and $\varpi_2^{A,B} : \mathit{\Pi}_2^{A,B}$ are respectively the derelictions der_A and der_B up to 'tags';

- β-currying is defined by $\Lambda(G, [\varphi]) \stackrel{\text{df.}}{=} (\Lambda(G), [\Lambda(\varphi)]) : A \to (B \Rightarrow C)$ for any objects A, B and C, and morphism $(G, [\varphi]) : A \& B \to C$;

- The β-evaluation $ev_{B,C} : C^B \& B \to C$ for any objects B and C is the pair $(\mathit{\Upsilon}_{B,C}, [\upsilon_{B,C}])$, where $\upsilon_{B,C} : \mathit{\Upsilon}_{B,C}$ is the dereliction $der_{B \Rightarrow C}$ up to 'tags'.

Theorem 3.4.2 (Well-defined $\mathcal{LDG}$)**.** *The structure $\mathcal{LDG}$ forms a CCBoC.*

Proof. First, it is obvious that the β-identity $id_A = (A \Rightarrow A, [der_A])$ on each object A is a β-morphism $A \to A$ in $\mathcal{LDG}$.

Next, for β-composition, let $A, B, C \in \mathcal{LDG}$, and $(J, [\phi]) : A \to B$ and $(K, [\psi]) : B \to C$ in $\mathcal{LDG}$. Then, $\phi^\dagger : J^\dagger$ by Theorem 3.3.46, and $\mathcal{H}^\omega(J^\dagger) \trianglelefteq \mathcal{H}^\omega(J)^\dagger \trianglelefteq {!A} \multimap {!B}$ by Lemma 3.3.30 and Theorem 3.3.22; thus, we may form $\phi^\dagger \ddagger \psi : J^\dagger \ddagger K$ such that $\mathcal{H}^\omega(J^\dagger \ddagger K) \trianglelefteq \mathcal{H}^\omega(J)^\dagger ; \mathcal{H}^\omega(K) \trianglelefteq A \Rightarrow C$ by Theorem 3.3.48 and Lemma 3.3.30, and clearly $\mathcal{H}^\omega((J^\dagger \ddagger K)^\dagger) = \mathcal{H}^\omega(J^\dagger \ddagger K^\dagger) = \mathcal{H}^\omega(J^\dagger) ; \mathcal{H}^\omega(K^\dagger) = \mathcal{H}^\omega(J)^\dagger ; \mathcal{H}^\omega(K)^\dagger = (\mathcal{H}^\omega(J^\dagger) ; \mathcal{H}^\omega(K))^\dagger = \mathcal{H}^\omega(J^\dagger \ddagger K)^\dagger$. Also, promotion and concatenation both preserve validity and winning of dynamic strategies (by Theorems 3.3.46 and 3.3.48). Hence, the pair $(J^\dagger \ddagger K, [\phi^\dagger \ddagger \psi])$ is a β-morphism $A \to C$ in $\mathcal{LDG}$. Note that the composition does not depend on the choice of representatives ϕ and ψ for $[\phi]$ and $[\psi]$, respectively.

Then clearly, associativity of β-composition up to $\simeq$ holds: Given $D \in \mathcal{LDG}$, and $(G, [\varphi]) : C \to D$ in $\mathcal{LDG}$, by Lemma 3.3.30 we have:

$$\begin{aligned}
\mathcal{H}^\omega((J^\dagger \ddagger K)^\dagger \ddagger G) &= (\mathcal{H}^\omega(J^\dagger) ; \mathcal{H}^\omega(K))^\dagger ; \mathcal{H}^\omega(G) \\
&= (\mathcal{H}^\omega(J)^\dagger ; \mathcal{H}^\omega(K)^\dagger) ; \mathcal{H}^\omega(G) \\
&= \mathcal{H}^\omega(J)^\dagger ; (\mathcal{H}^\omega(K)^\dagger ; \mathcal{H}^\omega(G)) \\
&= \mathcal{H}^\omega(J^\dagger) ; (\mathcal{H}^\omega(K^\dagger) ; \mathcal{H}^\omega(G)) \\
&= \mathcal{H}^\omega(J^\dagger \ddagger (K^\dagger \ddagger G))
\end{aligned}$$

as well as by Lemma 3.3.52:

$$\begin{aligned}\mathcal{H}^\omega([(\phi^\dagger \ddagger \psi)^\dagger \ddagger \varphi]) &= [\mathcal{H}^\omega((\phi^\dagger \ddagger \psi)^\dagger \ddagger \varphi)] \\ &= [(\mathcal{H}^\omega(\phi)^\dagger ; \mathcal{H}^\omega(\psi))^\dagger ; \mathcal{H}^\omega(\varphi)] \\ &= [(\mathcal{H}^\omega(\phi)^\dagger ; \mathcal{H}^\omega(\psi)^\dagger) ; \mathcal{H}^\omega(\varphi)] \\ &= [(\mathcal{H}^\omega(\phi)^\dagger ; (\mathcal{H}^\omega(\psi)^\dagger ; \mathcal{H}^\omega(\varphi))] \\ &= [\mathcal{H}^\omega(\phi^\dagger \ddagger (\psi^\dagger \ddagger \varphi))] \\ &= \mathcal{H}^\omega([\phi^\dagger \ddagger (\psi^\dagger \ddagger \varphi)])\end{aligned}$$

whence $((J, [\phi]); (K, [\psi])); (G, [\varphi]) \simeq (J, [\phi]); ((K, [\psi]); (G, [\varphi]))$.

Similarly, unit law up to $\simeq$ holds: Given a β-morphism $(J, [\phi]) : A \to B$, by Lemma 3.3.30 we have:

$$\begin{aligned}\mathcal{H}^\omega((A \Rightarrow A)^\dagger \ddagger J) &= (A \Rightarrow A)^\dagger ; \mathcal{H}^\omega(J) \\ &= \mathcal{H}^\omega(J)\end{aligned}$$

as well as:

$$\mathcal{H}^\omega([der_A^\dagger \ddagger \phi]) = \mathcal{H}^\omega([\phi]).$$

Similarly, $\mathcal{H}^\omega(J^\dagger \ddagger (B \Rightarrow B)) = \mathcal{H}^\omega(J)$ and $\mathcal{H}^\omega([\phi^\dagger \ddagger der_B]) = \mathcal{H}^\omega([\phi])$ hold, whence

$$(A \Rightarrow A, [der_A]); (J, [\phi]) \simeq (J, [\phi]) \simeq (J, [\phi]); (B \Rightarrow B, [der_B]).$$

Moreover, β-composition preserves $\simeq$: For any $A, B, C \in \mathcal{LDG}$, $(J, [\iota]), (\tilde{J}, [\tilde{\iota}]) : A \to B$ and $(K, [\kappa]), (\tilde{K}, [\tilde{\kappa}]) : B \to C$ in $\mathcal{LDG}$, if $\mathcal{H}^\omega(J, [\iota]) = \mathcal{H}^\omega(\tilde{J}, [\tilde{\iota}])$ and $\mathcal{H}^\omega(K, [\kappa]) = \mathcal{H}^\omega(\tilde{K}, [\tilde{\kappa}])$, i.e., $\mathcal{H}^\omega(J) = \mathcal{H}^\omega(\tilde{J})$, $\mathcal{H}^\omega(K) = \mathcal{H}^\omega(\tilde{K})$, $\mathcal{H}^\omega(\iota) \simeq_{A \Rightarrow B} \mathcal{H}^\omega(\tilde{\iota})$ and $\mathcal{H}^\omega(\kappa) \simeq_{B \Rightarrow C} \mathcal{H}^\omega(\tilde{\kappa})$, then we have $\mathcal{H}^\omega(J^\dagger \ddagger K) = \mathcal{H}^\omega(J^\dagger); \mathcal{H}^\omega(K) = \mathcal{H}^\omega(J)^\dagger; \mathcal{H}^\omega(K) = \mathcal{H}^\omega(\tilde{J})^\dagger; \mathcal{H}^\omega(\tilde{K}) = \mathcal{H}^\omega(\tilde{J}^\dagger); \mathcal{H}^\omega(\tilde{K}) = \mathcal{H}^\omega(\tilde{J}^\dagger \ddagger \tilde{K})$ by Lemma 3.3.30, and $\mathcal{H}^\omega(\iota^\dagger \ddagger \kappa) \simeq_{\mathcal{H}^\omega(J^\dagger \ddagger K)} \mathcal{H}^\omega(\tilde{\iota}^\dagger \ddagger \tilde{\kappa})$ by Lemma 3.3.52, Theorems 3.3.46 and 3.3.48, and Corollary 3.3.40, whence $\mathcal{H}^\omega(J^\dagger \ddagger K, [\iota^\dagger \ddagger \kappa]) = \mathcal{H}^\omega(\tilde{J}^\dagger \ddagger \tilde{K}, [\tilde{\iota}^\dagger \ddagger \tilde{\kappa}])$.

Also, $\mathcal{H}$ clearly satisfies the three axioms of BoC (Definition 3.2.2), having shown that $\mathcal{LDG}$ is a BoC. It remains to verify its cartesian closed structure up to $\simeq$.

The universal property of the β-terminal dynamic game T up to $\simeq$ is obvious, where we define $!_A \stackrel{\text{df.}}{=} (A \Rightarrow T, [\{\boldsymbol{\epsilon}\}]) : A \to T$ for each $A \in \mathcal{LDG}$. The β-projections are clearly well-defined values in $\mathcal{LDG}$. Given β-morphisms $(L, [\alpha]) : C \to A$ and $(R, [\beta]) : C \to B$ in $\mathcal{LDG}$, i.e., $\alpha : L$, $\beta : R$, $\mathcal{H}^\omega(L) \trianglelefteq C \Rightarrow A$, $\mathcal{H}^\omega(L^\dagger) = \mathcal{H}^\omega(L)^\dagger$,

$\mathcal{H}^\omega(R) \trianglelefteqslant C \Rightarrow B$ and $\mathcal{H}^\omega(R^\dagger) = \mathcal{H}^\omega(R)^\dagger$, we may obtain the valid, winning pairing $\langle \alpha, \beta \rangle : \langle L, R \rangle$ such that:

$$\begin{aligned}\mathcal{H}^\omega(\langle L, R\rangle) &\trianglelefteqslant \langle \mathcal{H}^\omega(L), \mathcal{H}^\omega(R)\rangle \\ &\trianglelefteqslant C \Rightarrow A\&B\end{aligned}$$

again by Lemmata 3.3.30 and 3.3.29, and Theorem 3.3.20, as well as:

$$\begin{aligned}\mathcal{H}^\omega(\langle L, R\rangle^\dagger) &= \mathcal{H}^\omega(\langle L^\dagger, R^\dagger\rangle) \\ &= \langle \mathcal{H}^\omega(L^\dagger), \mathcal{H}^\omega(R^\dagger)\rangle \\ &= \langle \mathcal{H}^\omega(L)^\dagger, \mathcal{H}^\omega(R)^\dagger\rangle \\ &= \langle \mathcal{H}^\omega(L), \mathcal{H}^\omega(R)\rangle^\dagger \\ &= \mathcal{H}^\omega(\langle L, R\rangle)^\dagger.\end{aligned}$$

Hence, the pair $(\langle L, R\rangle, [\langle \alpha, \beta\rangle])$ is a β-morphism $C \to A\&B$ in $\mathcal{LDG}$, which does not depend on the choice of the representatives α and β. Note also that the β-pairing clearly preserves values in $\mathcal{LDG}$.

Also, we have by Lemmata 3.3.30 and 3.3.54:

$$\mathcal{H}^\omega(\langle L, R\rangle^\dagger \ddagger \mathit{\Pi}_1^{A,B}) = \langle \mathcal{H}^\omega(L), \mathcal{H}^\omega(R)\rangle^\dagger ; \mathit{\Pi}_1^{A,B} = \mathcal{H}^\omega(L)$$

as well as by Lemma 3.3.52:

$$\begin{aligned}\mathcal{H}^\omega([\langle \alpha, \beta\rangle^\dagger \ddagger \varpi_1^{A,B}]) &= [\mathcal{H}^\omega(\langle \alpha^\dagger, \beta^\dagger\rangle \ddagger \varpi_1^{A,B})] \\ &= [\langle \mathcal{H}^\omega(\alpha)^\dagger, \mathcal{H}^\omega(\beta)^\dagger\rangle ; \varpi_1^{A,B}] \\ &= [\mathcal{H}^\omega(\alpha)] \\ &= \mathcal{H}^\omega([\alpha]).\end{aligned}$$

Similarly, $\mathcal{H}^\omega(\langle L, R\rangle^\dagger \ddagger \mathit{\Pi}_2^{A,B}) = \mathcal{H}^\omega(R)$ and $\mathcal{H}^\omega([\langle \alpha, \beta\rangle]; [\varpi_2^{A,B}]) = \mathcal{H}^\omega([\beta])$. Hence, $\langle (L, [\alpha]), (R, [\beta])\rangle ; \pi_1 \simeq (L, [\alpha])$ and $\langle (L, [\alpha]), (R, [\beta])\rangle ; \pi_2 = (R, [\beta])$ hold in $\mathcal{LDG}$.

Next, given any β-morphism $(P, [\rho]) : C \to A\&B$ in $\mathcal{LDG}$, we have:

$$\begin{aligned}\mathcal{H}^\omega(\langle P^\dagger \ddagger \mathit{\Pi}_1^{A,B}, P^\dagger \ddagger \mathit{\Pi}_2^{A,B}\rangle) &= \langle \mathcal{H}^\omega(P)^\dagger ; \mathit{\Pi}_1^{A,B}, \mathcal{H}^\omega(P)^\dagger ; \mathit{\Pi}_2^{A,B}\rangle \\ &= \mathcal{H}^\omega(P)\end{aligned}$$

again by Lemmata 3.3.30 and 3.3.54, as well as by Lemma 3.3.52:

$$\begin{aligned}\mathcal{H}^\omega([\langle \rho^\dagger \ddagger \varpi_1^{A,B}, \rho^\dagger \ddagger \varpi_2^{A,B}\rangle]) &= [\mathcal{H}^\omega(\langle \rho^\dagger \ddagger \varpi_1^{A,B}, \rho^\dagger \ddagger \varpi_2^{A,B}\rangle)] \\ &= [\langle \mathcal{H}^\omega(\rho)^\dagger ; \varpi_1^{A,B}, \mathcal{H}^\omega(\rho)^\dagger ; \varpi_2^{A,B}\rangle] \\ &= [\mathcal{H}^\omega(\rho)] \\ &= \mathcal{H}^\omega([\rho]).\end{aligned}$$

Hence, $\langle (P, [\rho]); \pi_1, (P, [\rho]); \pi_2 \rangle \simeq (P, [\rho])$ holds in $\mathcal{LDG}$.

It is also straightforward to check that β-pairing in $\mathcal{LDG}$ preserves $\simeq$: Given any β-morphisms $(L, [\alpha]), (\tilde{L}, [\tilde{\alpha}]) : C \to A$ and $(R, [\beta]), (\tilde{R}, [\tilde{\beta}]) : C \to B$ in $\mathcal{LDG}$ such that $\mathcal{H}^\omega(L, [\alpha]) = \mathcal{H}^\omega(\tilde{L}, [\tilde{\alpha}])$ and $\mathcal{H}^\omega(R, [\beta]) = \mathcal{H}^\omega(\tilde{R}, [\tilde{\beta}])$, we have:

$$\begin{aligned}
\mathcal{H}^\omega(\langle (L, [\alpha]), (R, [\beta]) \rangle) &= (\mathcal{H}^\omega(\langle L, R \rangle), \mathcal{H}^\omega([\langle \alpha, \beta \rangle])) \\
&= (\langle \mathcal{H}^\omega(L), \mathcal{H}^\omega(R) \rangle, [\langle \mathcal{H}^\omega(\alpha), \mathcal{H}^\omega(\beta) \rangle]) \\
&= (\langle \mathcal{H}^\omega(\tilde{L}), \mathcal{H}^\omega(\tilde{R}) \rangle, [\langle \mathcal{H}^\omega(\tilde{\alpha}), \mathcal{H}^\omega(\tilde{\beta}) \rangle]) \\
&= (\mathcal{H}^\omega(\langle \tilde{L}, \tilde{R} \rangle), \mathcal{H}^\omega(\langle [\tilde{\alpha}], [\tilde{\beta}] \rangle)) \\
&= \mathcal{H}^\omega(\langle (\tilde{L}, [\tilde{\alpha}]), (\tilde{R}, [\tilde{\beta}]) \rangle).
\end{aligned}$$

The β-evaluation $ev_{B,C} = (\varUpsilon_{B,C}, [\upsilon_{B,C}])$ for any $B, C \in \mathcal{LDG}$ is clearly a value $ev_{B,C} : C^B \& B \to C$ in $\mathcal{LDG}$ that satisfies by Lemmata 3.3.30 and 3.3.54:

$$\begin{aligned}
\mathcal{H}^\omega(\langle (\varPi_1^{A,B})^\dagger \ddagger \Lambda(U), \varPi_2^{A,B} \rangle^\dagger \ddagger \varUpsilon_{B,C}) &= \langle (\varPi_1^{A,B})^\dagger ; \Lambda \circ \mathcal{H}^\omega(U), \varPi_2^{A,B} \rangle^\dagger ; \varUpsilon_{B,C} \\
&= \mathcal{H}^\omega(U)
\end{aligned}$$

as well as by Lemma 3.3.52:

$$\begin{aligned}
\mathcal{H}^\omega([\langle (\varpi_1^{A,B})^\dagger \ddagger \Lambda(\mu), (\varpi_2^{A,B})^\dagger \ddagger der_B \rangle^\dagger \ddagger \upsilon_{B,C}]) &= [\langle (\varpi_1^{A,B})^\dagger ; \Lambda \circ \mathcal{H}^\omega(\mu), (\varpi_2^{A,B})^\dagger ; der_B \rangle^\dagger ; \upsilon_{B,C}] \\
&= [\mathcal{H}^\omega(\mu)] \\
&= \mathcal{H}^\omega([\mu])
\end{aligned}$$

which establishes $(\Lambda(U, [\mu]) \times id_B); ev_{B,C} \simeq (U, [\mu])$ for any β-morphism $(U, [\mu]) : A \& B \to C$ in $\mathcal{LDG}$. Note also that β-currying in $\mathcal{LDG}$ does not depend on the choice of representatives, and it preserves values in $\mathcal{LDG}$.

Similarly, we have again by Lemmata 3.3.30 and 3.3.54:

$$\begin{aligned}
\mathcal{H}^\omega(\Lambda(\langle (\varPi_1^{A,B})^\dagger \ddagger V, \varPi_2^{A,B} \rangle^\dagger \ddagger \varUpsilon_{B,C})) &= \Lambda(\langle (\varPi_1^{A,B})^\dagger ; \mathcal{H}^\omega(V), \varPi_2^{A,B} \rangle^\dagger ; \varUpsilon_{B,C}) \\
&= \mathcal{H}^\omega(V)
\end{aligned}$$

as well as by Lemma 3.3.52:

$$\begin{aligned}
\mathcal{H}^\omega[\Lambda(\langle (\varpi_1^{A,B})^\dagger \ddagger \nu, (\varpi_2^{A,B})^\dagger \ddagger der_B \rangle^\dagger \ddagger \upsilon_{B,C})] &= [\Lambda(\langle (\varpi_1^{A,B})^\dagger ; \mathcal{H}^\omega(\nu), (\varpi_2^{A,B})^\dagger ; der_B \rangle^\dagger ; \upsilon_{B,C})] \\
&= [\mathcal{H}^\omega(\nu)] \\
&= \mathcal{H}^\omega([\nu])
\end{aligned}$$

which establishes $\Lambda(((V, [\nu]) \times id_B); ev_{B,C}) \simeq (V, [\nu])$ for any β-morphism $(V, [\nu]) : A \to (B \Rightarrow C)$ in $\mathcal{LDG}$.

Finally, it is easy to see that β-currying in $\mathcal{LDG}$ preserves the equivalence relation $\simeq$ by Lemma 3.3.52 and Proposition 3.3.50, completing the proof. □

We proceed to give a standard structure (Definition 3.2.14) for FPCF in $\mathcal{LDG}$:

Definition 3.4.3 (Standard structure in $\mathcal{LDG}$)**.** The standard structure

$$\mathcal{S}_{\mathcal{G}} = (\mathbf{2}, T, \&, \pi, \Rightarrow, ev, \underline{tt}, \underline{\mathit{ff}}, \vartheta)$$

of dynamic games and strategies for FPCF in $\mathcal{LDG}$ is given by:

- $\mathbf{2}$ and T are the game of booleans (Example 2.2.12) and the terminal game (Example 2.2.11), respectively, both modified to normalized dynamic games in the obvious manner;
- $\&$ is product of dynamic games, $\pi_1^{A,B} \overset{\text{df.}}{=} (\varPi_1^{A,B}, [\varpi_1^{A,B}])$ and $\pi_2^{A,B} \overset{\text{df.}}{=} (\varPi_2^{A,B}, [\varpi_2^{A,B}])$ for any normalized dynamic games A and B (see Definition 3.4.1);
- $\Rightarrow$ is function space (or implication) of dynamic games, and $ev_{A,B} \overset{\text{df.}}{=} (\varUpsilon_{A,B}, [\upsilon_{A,B}])$ for any normalized dynamic games A and B (see Definition 3.4.1);
- $\underline{tt} \overset{\text{df.}}{=} (T \Rightarrow \mathbf{2}, [\mathsf{Pref}(\{q.tt\})^{\mathsf{Even}}]), \underline{\mathit{ff}} \overset{\text{df.}}{=} (T \Rightarrow \mathbf{2}, [\mathsf{Pref}(\{q.\mathit{ff}\})^{\mathsf{Even}}]) : T \to \mathbf{2}$;
- $\vartheta \overset{\text{df.}}{=} (\mathbf{2}\&(\mathbf{2}\&\mathbf{2}) \Rightarrow \mathbf{2}, [case]) : \mathbf{2}\&(\mathbf{2}\&\mathbf{2}) \to \mathbf{2}$, where $case : \mathbf{2}\&(\mathbf{2}\&\mathbf{2}) \Rightarrow \mathbf{2}$, is the standard interpretation of the case-construction in the language PCF [100, 14] modified to a normalized dynamic strategy in the obvious manner.

It is clear that the structure $\mathcal{S}_{\mathcal{G}}$ is *standard* (Definition 3.2.14), where the inequality (3.5) is satisfied by the underlying dynamic game of each morphism in $\mathcal{LDG}$.

3.4.2 Dynamic Correspondence Property for FPCF

At last, we are now ready to prove that our game semantics satisfies a DCP:

Theorem 3.4.4 (PDCP-theorem)**.** *The interpretation* $[\![\_]\!]_{\mathcal{LDG}}^{\mathcal{S}_{\mathcal{G}}}$ *of FPCF (Definitions 3.2.14 and 3.4.1) satisfies the PDCP (Definition 3.2.16).*

Proof. For proving that the interpretation satisfies the PDCP, the non-trivial case is only to show for any $(\lambda \mathsf{x}^{\mathsf{A}}.\, \mathsf{V})\mathsf{W} \to \mathsf{U}$ of FPCF, where V, W and U are values, $\mathcal{H}([\![(\lambda \mathsf{x}^{\mathsf{A}}.\, \mathsf{V})\mathsf{W}]\!]_{\mathcal{LDG}}^{\mathcal{S}_{\mathcal{G}}}) = [\![\mathsf{U}]\!]_{\mathcal{LDG}}^{\mathcal{S}_{\mathcal{G}}}$ (n.b., $[\![(\lambda \mathsf{x}^{\mathsf{A}}.\, \mathsf{V})\mathsf{W}]\!]_{\mathcal{LDG}}^{\mathcal{S}_{\mathcal{G}}} \neq [\![\mathsf{U}]\!]_{\mathcal{LDG}}^{\mathcal{S}_{\mathcal{G}}}$ is immediate from the first component of each β-morphism in $\mathcal{LDG}$ and the second axiom on standardness of $\mathcal{S}_{\mathcal{G}}$); the other conditions follow from Lemma 3.3.30. For this, we define the ***height*** $Ht(\mathsf{B}) \in \mathbb{N}$ of each type B by $Ht(o) \overset{\text{df.}}{=} 0$ and $Ht(\mathsf{B}_1 \Rightarrow \mathsf{B}_2) \overset{\text{df.}}{=} \max(Ht(\mathsf{B}_1)+1, Ht(\mathsf{B}_2))$. Then, the required equation is shown by induction on the height of the type A of W.

Below, given β-morphisms $(H, [\tau]) : C \to (A \Rightarrow B)$ and $(G, [\sigma]) : C \to A$ in $\mathcal{LDG}$, we define the β-morphism $(H, [\tau])\lfloor (G, [\sigma]) \rfloor \overset{\text{df.}}{=} (\langle G, H \rangle^{\dagger} \ddagger \varUpsilon_{A,B}, [\langle \tau, \sigma \rangle^{\dagger} \ddagger \upsilon_{A,B}]) : C \to B$ in $\mathcal{LDG}$. If $(H, [\tau]) : C \to (A_1 \Rightarrow A_2 \Rightarrow \cdots \Rightarrow A_k \Rightarrow B)$ and $(G_i, [\sigma_i]) : C \to A_i$ for $i = 1, 2, \dots, k$, then we write $(H, [\tau])\lfloor (G_1, [\sigma_1]), (G_2, [\sigma_2]), \dots, (G_k, [\sigma_k]) \rfloor$ for $(H, [\tau])\lfloor (G_1, [\sigma_1]) \rfloor \lfloor (G_2, [\sigma_2]) \rfloor \dots \lfloor (G_k, [\sigma_k]) \rfloor : C \to B$. We abbreviate in this proof the interpretation $[\![\_]\!]^{\mathcal{S}_{\mathcal{G}}}_{\mathcal{LDG}}$ as $[\![\_]\!]$. Let $\mathsf{\Gamma}$ be the context of $(\lambda \mathsf{x}^{\mathsf{A}}.\, \mathsf{V})\mathsf{W}$ (as well as U). In the following, we abbreviate each β-morphism $(G, [\sigma])$ in $\mathcal{LDG}$ as $[\sigma]$ for brevity, and focus on the second components; the corresponding equations on the first components may be obtained, thanks to Lemmata 3.3.30 and 3.3.54, similarly to or simpler than the ways for the first components shown below.

For the base case, assume $Ht(\mathsf{A}) = 0$, i.e., $\mathsf{A} \equiv o$. By induction on $|\mathsf{V}|$, we have:

- If $\mathsf{V} \equiv \mathsf{tt}$, then $(\lambda \mathsf{x}^{\mathsf{A}}.\, \mathsf{tt})\mathsf{W} \to \mathsf{tt}$, and clearly $\mathcal{H}([\![(\lambda \mathsf{x}^{\mathsf{A}}.\, \mathsf{tt})\mathsf{W}]\!]) = \langle \Lambda([\![\mathsf{tt}]\!]), [\![\mathsf{W}]\!] \rangle^{\dagger}; [\upsilon] = [\![\mathsf{tt}]\!]$. The case where $\mathsf{V} \equiv \mathsf{ff}$ is analogous.

- If $\mathsf{V} \equiv \lambda \mathsf{y}^{\mathsf{C}}.\, \mathsf{V}'$, then $(\lambda \mathsf{x}^{\mathsf{A}} \mathsf{y}^{\mathsf{C}}.\, \mathsf{V}')\mathsf{W} \to \lambda \mathsf{y}^{\mathsf{C}}.\mathsf{U}'$ such that $(\lambda \mathsf{x}^{\mathsf{A}}.\, \mathsf{V}')\mathsf{W} \to \mathsf{U}'$ (because $\mathit{nf}((\lambda \mathsf{x}^{\mathsf{A}} \mathsf{y}^{\mathsf{C}}.\, \mathsf{V}')\mathsf{W}) \equiv \mathit{nf}(\lambda \mathsf{y}^{\mathsf{C}}.\, \mathsf{V}'[\mathsf{W}/\mathsf{x}]) \equiv \lambda \mathsf{y}^{\mathsf{C}}.\mathit{nf}(\mathsf{V}'[\mathsf{W}/\mathsf{x}]) \equiv \lambda \mathsf{y}^{\mathsf{C}}.\, \mathit{nf}((\lambda \mathsf{x}^{\mathsf{A}}.\mathsf{V}')\mathsf{W})$). By the induction hypothesis, we have $\mathcal{H}([\![(\lambda \mathsf{x}^{\mathsf{A}}.\, \mathsf{V}')\mathsf{W}]\!]) = [\![\mathsf{U}']\!]$. Hence,

$$\begin{aligned}
\mathcal{H}([\![\mathsf{VW}]\!]) &= \mathcal{H}(\langle \Lambda_{[\![\mathsf{A}]\!]}(\Lambda_{[\![\mathsf{C}]\!]}([\![\mathsf{V}']\!])), [\![\mathsf{W}]\!] \rangle^{\dagger} \ddagger [\upsilon]) \\
&= \langle \Lambda_{[\![\mathsf{A}]\!]}(\Lambda_{[\![\mathsf{C}]\!]}([\![\mathsf{V}']\!])), [\![\mathsf{W}]\!] \rangle^{\dagger}; [\upsilon] \text{ (by Lemma 3.3.52)} \\
&= \Lambda_{[\![\mathsf{C}]\!]}(\langle \Lambda_{[\![\mathsf{A}]\!]}([\![\mathsf{V}']\!]), [\![\mathsf{W}]\!] \rangle^{\dagger}; [\upsilon]) \\
&= \Lambda_{[\![\mathsf{C}]\!]}(\mathcal{H}(\langle \Lambda_{[\![\mathsf{A}]\!]}([\![\mathsf{V}']\!]), [\![\mathsf{W}]\!] \rangle^{\dagger} \ddagger [\upsilon])) \\
&= \Lambda_{[\![\mathsf{C}]\!]}(\mathcal{H}([\![(\lambda \mathsf{x}^{\mathsf{A}}.\mathsf{V}')\mathsf{W}]\!])) \\
&= \Lambda_{[\![\mathsf{C}]\!]}([\![\mathsf{U}']\!]) \\
&= [\![\lambda \mathsf{y}^{\mathsf{C}}.\mathsf{U}']\!].
\end{aligned}$$

- If $\mathsf{V} \equiv \mathsf{case}(\mathsf{y}\mathsf{V}_1 \dots \mathsf{V}_\mathsf{k})[\tilde{\mathsf{V}}_1; \tilde{\mathsf{V}}_2]$ with $\mathsf{x} \neq \mathsf{y}$, then $(\lambda \mathsf{x}^{\mathsf{A}}.\mathsf{V})\mathsf{W} \to \mathsf{U}$, where

$$\mathsf{U} \equiv \mathsf{case}(\mathsf{y}\mathit{nf}(\mathsf{V}_1[\mathsf{W}/\mathsf{x}]) \dots \mathit{nf}(\mathsf{V}_\mathsf{k}[\mathsf{W}/\mathsf{x}]))[\mathit{nf}(\tilde{\mathsf{V}}_1[\mathsf{W}/\mathsf{x}]); \mathit{nf}(\tilde{\mathsf{V}}_1[\mathsf{W}/\mathsf{x}])].$$

By the induction hypothesis and the interpretation of the variable y, we have:

$$
\begin{aligned}
&[\![(\lambda \mathsf{x}^{\mathsf{A}}.\mathsf{V})\mathsf{W}]\!]\\
&= \mathcal{H}^{\omega}(\Lambda_{[\![\mathsf{A}]\!]}(\langle [\![\mathsf{y}]\!] \lfloor [\![\mathsf{V}_1]\!], \ldots, [\![\mathsf{V}_k]\!] \rfloor, \langle [\![\tilde{\mathsf{V}}_1]\!], [\![\tilde{\mathsf{V}}_2]\!] \rangle \rangle^{\dagger} \ddagger [case]) \lfloor [\![\mathsf{W}]\!] \rfloor)\\
&= \mathcal{H}^{\omega}(\langle \Lambda_{[\![\mathsf{A}]\!]}([\![\mathsf{y}]\!]) \lfloor [\![\mathsf{W}]\!] \rfloor \lfloor \Lambda_{[\![\mathsf{A}]\!]}([\![\mathsf{V}_1]\!]) \lfloor [\![\mathsf{W}]\!] \rfloor, \ldots, \Lambda_{[\![\mathsf{A}]\!]}([\![\mathsf{V}_k]\!]) \lfloor [\![\mathsf{W}]\!] \rfloor \rfloor, \langle \Lambda_{[\![\mathsf{A}]\!]}([\![\tilde{\mathsf{V}}_1]\!]) \lfloor [\![\mathsf{W}]\!] \rfloor,\\
&\quad \Lambda_{[\![\mathsf{A}]\!]}([\![\tilde{\mathsf{V}}_2]\!]) \lfloor [\![\mathsf{W}]\!] \rfloor \rangle \rangle^{\dagger} \ddagger [case])\\
&= \mathcal{H}^{\omega}(\langle [\![(\lambda \mathsf{x}.\, \mathsf{y})\mathsf{W}]\!] \lfloor [\![(\lambda \mathsf{x}.\, \mathsf{V}_1)\mathsf{W}]\!], \ldots, [\![(\lambda \mathsf{x}.\, \mathsf{V}_k)\mathsf{W}]\!] \rfloor, \langle [\![(\lambda \mathsf{x}.\, \tilde{\mathsf{V}}_1)\mathsf{W}]\!], [\![(\lambda \mathsf{x}.\, \tilde{\mathsf{V}}_2)\mathsf{W}]\!] \rangle \rangle^{\dagger} \ddagger [case])\\
&= \mathcal{H}^{\omega}(\langle [\![\mathsf{y}]\!] \lfloor [\![nf(\mathsf{V}_1[\mathsf{W}/\mathsf{x}])]\!], \ldots, [\![nf(\mathsf{V}_k[\mathsf{W}/\mathsf{x}])]\!] \rfloor, \langle [\![nf(\tilde{\mathsf{V}}_1[\mathsf{W}/\mathsf{x}])]\!], [\![nf(\tilde{\mathsf{V}}_2[\mathsf{W}/\mathsf{x}])]\!] \rangle \rangle^{\dagger} \ddagger [case])\\
&= [\![\mathsf{U}]\!].
\end{aligned}
$$

- If $\mathsf{V} \equiv \mathsf{case}(\mathsf{x})[\tilde{\mathsf{V}}_1; \tilde{\mathsf{V}}_2]$, then $(\lambda \mathsf{x}^{\mathsf{A}}.\mathsf{V})\mathsf{W} \to \mathsf{U}$, where

$$\mathsf{U} \equiv \mathsf{case}(\mathsf{W})[nf(\tilde{\mathsf{V}}_1[\mathsf{W}/\mathsf{x}]); nf(\tilde{\mathsf{V}}_2[\mathsf{W}/\mathsf{x}])].$$

By the same reasoning as the above case, we have $\mathcal{H}([\![(\lambda \mathsf{x}^{\mathsf{A}}.\mathsf{V})\mathsf{W}]\!]) = [\![\mathsf{U}]\!]$.

Next, for the inductive step, assume $Ht(\mathsf{A}) = h + 1$. We may proceed in the same way as the base case, i.e., by induction on $|\mathsf{V}|$, except that the last case is generalized to $\mathsf{V} \equiv \mathsf{case}(\mathsf{x}\mathsf{V}_1 \ldots \mathsf{V}_k)[\tilde{\mathsf{V}}_1; \tilde{\mathsf{V}}_2]$, where $\mathsf{A} \equiv \mathsf{A}_1 \Rightarrow \mathsf{A}_2 \Rightarrow \cdots \Rightarrow \mathsf{A}_k \Rightarrow o$ $(k \geqslant 0)$. We have to consider the cases for $k \geqslant 1$; then we have $(\lambda \mathsf{x}^{\mathsf{A}}.\, \mathsf{V})\mathsf{W} \to \mathsf{U}$, where

$$\mathsf{U} \equiv \mathsf{case}(nf(\mathsf{W}(\mathsf{V}_1[\mathsf{W}/\mathsf{x}]) \ldots (\mathsf{V}_k[\mathsf{W}/\mathsf{x}])))[nf(\tilde{\mathsf{V}}_1[\mathsf{W}/\mathsf{x}]); nf(\tilde{\mathsf{V}}_2[\mathsf{W}/\mathsf{x}])].$$

We then have the following chain of equations:

$$
\begin{aligned}
&\mathcal{H}[\![(\lambda \mathsf{x}.\mathsf{V})\mathsf{W}]\!]\\
&= \mathcal{H}(\Lambda(\mathcal{H}^{\omega}(\langle [\![\mathsf{x}]\!] \lfloor [\![\mathsf{V}_1]\!], \ldots, [\![\mathsf{V}_k]\!] \rfloor, \langle [\![\tilde{\mathsf{V}}_1]\!], [\![\tilde{\mathsf{V}}_2]\!] \rangle \rangle^{\dagger} \ddagger [case])) \lfloor [\![\mathsf{W}]\!] \rfloor)\\
&= \mathcal{H}^{\omega}(\langle \Lambda([\![\mathsf{x}]\!]) \lfloor [\![\mathsf{W}]\!] \rfloor \lfloor \Lambda([\![\mathsf{V}_1]\!]) \lfloor [\![\mathsf{W}]\!] \rfloor, \ldots, \Lambda([\![\mathsf{V}_k]\!]) \lfloor [\![\mathsf{W}]\!] \rfloor \rfloor, \langle \Lambda([\![\tilde{\mathsf{V}}_1]\!]) \lfloor [\![\mathsf{W}]\!] \rfloor, \Lambda([\![\tilde{\mathsf{V}}_2]\!]) \lfloor [\![\mathsf{W}]\!] \rfloor \rangle \rangle^{\dagger} \ddagger [case])\\
&= \mathcal{H}^{\omega}(\langle [\![(\lambda \mathsf{x}.\, \mathsf{x})\mathsf{W}]\!] \lfloor [\![(\lambda \mathsf{x}.\, \mathsf{V}_1)\mathsf{W}]\!], \ldots, [\![(\lambda \mathsf{x}.\, \mathsf{V}_k)\mathsf{W}]\!] \rfloor, \langle [\![(\lambda \mathsf{x}.\, \tilde{\mathsf{V}}_1)\mathsf{W}]\!], [\![(\lambda \mathsf{x}.\, \tilde{\mathsf{V}}_2)\mathsf{W}]\!] \rangle \rangle^{\dagger} \ddagger [case])\\
&= \mathcal{H}^{\omega}(\langle [\![\mathsf{W}]\!] \lfloor [\![nf(\mathsf{V}_1[\mathsf{W}/\mathsf{x}])]\!], \ldots, [\![nf(\mathsf{V}_k[\mathsf{W}/\mathsf{x}])]\!] \rfloor, \langle [\![nf(\tilde{\mathsf{V}}_1[\mathsf{W}/\mathsf{x}])]\!], [\![nf(\tilde{\mathsf{V}}_2[\mathsf{W}/\mathsf{x}])]\!] \rangle \rangle^{\dagger} \ddagger [case])\\
&\quad \text{(by the induction hypothesis with respect to } |\mathsf{V}|)\\
&= \mathcal{H}^{\omega}(\langle [\![nf(\mathsf{W}(\mathsf{V}_1[\mathsf{W}/\mathsf{x}]) \ldots (\mathsf{V}_k[\mathsf{W}/\mathsf{x}]))]\!], \langle [\![nf(\tilde{\mathsf{V}}_1[\mathsf{W}/\mathsf{x}])]\!], [\![nf(\tilde{\mathsf{V}}_2[\mathsf{W}/\mathsf{x}])]\!] \rangle \rangle^{\dagger} \ddagger [case])\\
&\quad \text{(by the induction hypothesis (applied } k\text{-times) with respect to the hight of types } \mathsf{A})\\
&= [\![\mathsf{case}(nf(\mathsf{W}(\mathsf{V}_1[\mathsf{W}/\mathsf{x}]) \ldots (\mathsf{V}_k[\mathsf{W}/\mathsf{x}])))[nf(\tilde{\mathsf{V}}_1[\mathsf{W}/\mathsf{x}]); nf(\tilde{\mathsf{V}}_2[\mathsf{W}/\mathsf{x}])]]\!]\\
&= [\![\mathsf{U}]\!]
\end{aligned}
$$

which completes the proof. □

Corollary 3.4.5 (Dynamic game semantics of FPCF)**.** *The interpretation* $[\![\_]\!]^{\mathcal{S}g}_{\mathcal{LDG}}$ *of FPCF and the hiding operation* $\mathcal{H}$ *satisfy the DCP in the sense of Definition 3.2.15.*

Proof. By Theorems 3.2.17 and 3.4.4. □

The relation between the syntax and the semantics is in fact much tighter than Corollary 3.4.5: Exploiting the strong definability result [18, 100], FPCF can be seen as a *formal calculus* for the game-semantic computation in the CCBoC $\mathcal{LDG}$.

Finally, let us establish a dynamic variant of full completeness [44]: Any dynamic strategy on a dynamic game that interprets a type of FPCF is the denotation of some term of FPCF:

Corollary 3.4.6 (Dynamic full completeness)**.** *Let G be a dynamic game such that for some dynamic strategy $\sigma : G$ the pair $(G, [\sigma])$ is the interpretation* $[\![\mathsf{\Gamma} \vdash \mathsf{M} : \mathsf{B}]\!]^{\mathcal{S}g}_{\mathcal{LDG}}$ *of a term* $\mathsf{\Gamma} \vdash \mathsf{M} : \mathsf{B}$ *of FPCF. Then, for any dynamic strategy $\tilde{\sigma} : G$ there is a term* $\mathsf{\Gamma} \vdash \tilde{\mathsf{M}} : \mathsf{B}$ *of FPCF such that* $[\![\mathsf{\Gamma} \vdash \tilde{\mathsf{M}} : \mathsf{B}]\!]^{\mathcal{S}g}_{\mathcal{LDG}} = (G, [\tilde{\sigma}])$.

Proof. Note that the dynamic game G is constructed along with the construction of type B of FPCF. We proceed by induction on the construction of G (or B).

First, since values of FPCF are PCF Böhm trees except that the natural number type ι is replaced with the boolean type o, and the bottom term $\perp$ is deleted, the conventional full completeness or the strong definability holds for values of FPCF in the same way as that of the conventional game semantics of PCF, where totality excludes the denotation of the bottom term $\perp$; see [14, 43] for the detail.

It remains to consider the rule A for applications, i.e., the case where G is of the form $\langle U, V \rangle^{\dagger} \ddagger \varUpsilon$. But then, note that only the canonical play is possible in $\varUpsilon$ (Definition 3.3.53), and therefore we may just apply the induction hypothesis. □

Finally, let us make a brief remark on *dynamics* of computation. Note that we have introduced internal moves and called them *intensionality* of computation, but one may say that they also contain dynamics for dynamic strategies play along with the passage of time, and internal moves represent step-by-step processes of the play. This claim sounds reasonable, but dynamics of computation in the present thesis refers only to what corresponds to dynamics of syntax, namely, *reduction* of programs. Conceptually, reduction is the process of *carving out* the extensional input-output behaviors from algorithms, which makes sense because often such extensional behaviors are not directly obtainable (e.g., functions on natural numbers), i.e., one first needs to execute an algorithm, and then extract its input-output behavior. That is, we call this very extraction process dynamics of computation in the present thesis.

Let us also remark that our result does not contradict the standard result [47], i.e., the correspondence between the execution of linear head reduction (LHR) and the step-by-step 'internal communication' between conventional strategies. In fact, LHR is a finer reduction strategy than the operational semantics of FPCF (Definition 3.2.3), and the work implies that LHR corresponds in conventional game semantics what should be called a 'move-wise' execution of the hiding operation. On the other hand, our operational semantics is executed in a much coarser, 'type-wise' fashion, and thus it may be seen as executing at a time a certain 'chunk' of LHR in a specific order. Our dynamic game semantics captures such a coarser dynamics of computation, and therefore it does not contradict the work [47]. Of course, it would be highly interesting to refine the present work to capture LHR by dynamic game semantics, which we leave as future work (see the next section too).

3.5 Conclusion and Future Work of the Chapter

We have presented a *mathematical* (and *syntax-independent*) formulation of dynamics and intensionality of computation in terms of games and strategies in the present chapter. From the opposite angle, we have developed a game-semantic framework for dynamic, intensional computational processes with a convenient formal calculus.

The most immediate future work is to apply the framework of dynamic game semantics to various logics and computations as in the case of static (usual) game semantics. Moreover, since the hiding operation can be further refined into the 'move-wise' fashion, the present work may be applicable for finer calculi such as *explicit substitution* [158] and the *differential λ-calculus* [55]. We conjecture that the well-known correspondence between 'internal communications' between strategies and executions of LHR [47] can be exploited for establishing such finer dynamic game semantics. Also, it would be interesting to see how accurately our game-semantic approach can measure the computational complexity of (higher-order) programming.

Finally, the notion of (CC)BoCs can be a concept of interest in its own right. For instance, it might be fruitful to develop it further to accommodate various models of computations in the same spirit of [122] but on *computation*, not computability. Also, it might be interesting to consider their relation with *computations as monads* in the sense introduced by Eugenio Moggi [136].

Chapter 4

Game-Semantic Model of Higher-Order Computation

4.1 Introduction to the Chapter

In this second main chapter, we shall develop the mathematical structure defined in the previous chapter further to address the problem described in Section 1.2.4.

4.1.1 Towards Game-Semantic Model of Computation

First, let us emphasize that conventional game semantics has never been formulated as a mathematical model of computation in the sense of TMs; rather, a primary focus of the field has been *full completeness* and *full abstraction* [44]. In other words, game semantics has not been concerned that much with *step-by-step processes* in computation or their 'effective computability', and it has been identifying programs with the same value [18, 188, 82]. As seen in the previous chapter, this point is due to *hiding internal moves* when composing strategies so that strategies are always in normal form. For instance, strategies on the game $N_{[0]} \multimap N_{[1]}$ typically play by $q_{[1]}q_{[0]}n_{[0]}m_{[1]}$, where $n, m \in \mathbb{N}$, as described in Section 2.1, and so they are essentially (linear) functions $n \mapsto m$; in particular, it is not formulated at all how they calculate the fourth move $m_{[1]}$ from the third one $n_{[0]}$. Consequently, 'effective computability' in game semantics has been *extrinsic*: A strategy has been defined to be 'effective' or *recursive* if it is representable by a partial recursive function [9, 100, 60].

Thus, to achieve what should be called a *game-semantic model of computation*, perhaps we first need to refine the category $\mathcal{CMG}$ (Definition 2.4.7) so that it may accommodate step-by-step processes in computation, and then define their 'effective computability' in terms of atomic steps of the processes like the framework of TMs so that it would be *intrinsic* (in the sense explained in Section 1.2), non-inductive and non-axiomatic. Fortunately, there is already the bicategory $\mathcal{LDG}$ of dynamic games and strategies (Definition 3.4.1), which has addressed the first point.

Hence, the remaining problem is to define 'effective' dynamic strategies in terms of their atomic steps (or by another method that is intrinsic, non-inductive and non-axiomatic). To go beyond classical computation (see Section 1.2.2), such 'effective' dynamic strategies should be at least as powerful as the higher-order programming language *PCF* [151, 163] or ***PCF-complete***. This sets up, in addition to the conceptual quest in Section 1.2.4, an intriguing mathematical question in its own right:

> Is there any intrinsic, non-inductive, non-axiomatic notion of 'effective computability' of dynamic strategies that is PCF-complete?

Our answer to this question is *positive*, providing a novel approach on low-level computational processes (see Section 1.2.3) in this chapter *solely in terms of games and strategies*. We explain the main idea of the solution in the next section.

4.1.2 Viable Strategies

Let us explain the idea of the purely game-semantic approach (n.b., the concepts introduced below make sense for conventional (or *static*) games and strategies as well, but they are not PCF-complete as composition of conventional strategies does not preserve 'effectivity' or *viability* of strategies defined below). First, a key observation is that every dynamic strategy modeling PCF only needs as an input the *last three moves* of each P-view and the *state* (Definition 4.2.11) of the P-view (Lemma 4.2.12); thus, it can be presented as a partial function $(\boldsymbol{s}, m_3, m_2, m_1) \mapsto m$, where m_1 is the last move, m_2 is the second last move, m_3 is the third last move of the current P-view, $\boldsymbol{s}$ is the state of the P-view, and m is the next P-move (n.b., m_2 or m_3 may be $\Box$, representing 'no move', if it does not exist in the P-view). Hence, assuming that m_1, m_2, m_3 and $\boldsymbol{s}$ are all 'effectively' obtainable (which is reasonable as we shall see), it suffices to achieve the computation $(\boldsymbol{s}, m_3, m_2, m_1) \mapsto m$ by a means that is clearly 'effective'. For this point, we give a concrete formalization of 'tags' for disjoint union of sets of moves in order to rigorously formalize 'effectivity' of dynamic strategies.

We then define a strategy to be *finitary* if its representation by a partial function [129, 100] that assigns the next P-move to a partial history of previous moves (which is 'effectively' obtainable from the entire history), called its *table*, is finite. Unfortunately, however, finitary dynamic strategies are *not* PCF-complete for they cannot handle unboundedly many 'tags' for exponential !, where such manipulations of 'tags' are particularly vital for promotion $(\_)^\dagger$ and *fixed-points* [101, 100]. Our solution is then to define a strategy to be *viable* if its table is 'describable' by another finitary strategy (Definition 4.4.7). Viability gives a reasonable notion of 'effectivity' of strategies as finitary strategies are clearly 'computable', and thus their 'descriptions' can be 'effectively read off' by 'executing' them. The main idea here is to achieve the computation $(\boldsymbol{s}, m_3, m_2, m_1) \mapsto m$ by a method that is clearly 'effective' yet more powerful than finite tables, viz., finitary strategies, arriving at viability of strategies.

As a main technical result, we show that dynamic strategies definable by PCF are all viable (Theorem 4.4.18), establishing 'effectivity' of dynamic strategies that is PCF-complete. Note that viability is defined solely in terms of games and strategies without any axiom or induction. Also, as immediate corollaries, some of the well-known theorems in recursion theory such as the *smn-theorem* and the *first recursion theorem (FRT)* [46, 157] are generalized (Corollaries 4.4.20 and 4.4.21).

As a more conceptual point, the game-semantic approach solves the problem raised in Section 1.2.4 in the following manner. On the one hand, dynamic games and strategies capture high-level computational processes of PCF in a conceptually natural, mathematically precise manner, where note that they are abstract, syntax-independent concepts, e.g., the *lazy natural number game* $\mathcal{N}$ (Definition 4.2.1) defines natural numbers (not their representation) as 'counting processes' in an abstract, syntax-independent fashion.[1] On the other hand, strategies that 'describe' viable strategies (i.e., high-level computational processes) correspond to low-level computational processes of the viable strategies. In this way, we have developed a single mathematical framework for both high-level and low-level computational processes as well as 'effectivity' of the former in terms of the latter, achieving *mathematics of computational processes* in the sense defined in Section 1.2.4.

4.1.3 Related Work and Contribution of the Chapter

The first achievement of the chapter is to define an *intrinsic*, *non-inductive*, *non-axiomatic* notion of 'effective computability' solely in terms of games and strategies, namely, viable dynamic strategies, and show that they are *PCF-complete.* As game semantics was not employed for theory of computation before, the work is novel as a mathematical model of computation; also, it raises further questions to explore; see Section 4.5. From a more methodological viewpoint, it indicates a high potential of the game-semantic approach in the context of theory of computation and recursion

[1]The game $\mathcal{N}$ itself can be defined as a *static* game, but we need the framework of *dynamic* game semantics for compositions of static games and strategies *with hiding* cannot capture step-by-step processes of computation as already pointed out.

theory. In addition, some of the well-known theorems in recursion theory such as the smn-theorem and the first recursion theorem are also generalized to non-classical computation.

Another, more conceptual contribution of the chapter is to establish a single mathematical framework for both high-level and low-level computational processes, where the former defines *what* computation does, while the latter describes *how* to execute the former. In comparison with existing models of computation, our game-semantic approach has some novel features. First, as opposed to computation by TMs or programming languages, plays of games are a more general, abstract concept; in particular they are not necessarily symbol manipulations, which is why they are suitable for abstract, high-level computational processes. Second, computation in a game proceeds as an interaction between the participants of the game, which may be seen as a generalization of computation by TMs (n.b., just one interaction occurs for a TM, i.e., Opponent gives an input on the infinite tape, and then Player returns an output on the tape); this means that Opponent's computation is part of the formalization, which is why a game-semantic approach in general captures higher-order computation in a natural, systematic manner. The present work inherits the interactive nature of game semantics. Last but not least, games are a semantic counterpart of *types* in computation, for which note that types do not a priori exist in TMs, and types in programming languages are a syntactic concept. Hence, our approach would provide both conceptual and mathematical investigations of types, particularly in the context of theory of computation and recursion theory.

Moreover, by exploiting the flexibility of game semantics, our approach would be applicable to a wide range of computations beyond functional one though it is left as future work. Also, game semantics has interpreted various logics as well [7, 101, 10, 189], and therefore it would be possible to employ our framework for a *realizability interpretation* of constructive logic [179, 186]; note that viable dynamic strategies would be more suitable for *realizers* than existing strategies, e.g., [28], since the former contains more 'computational contents' and makes more sense as a model of computation than the latter. The present work would serve as a technical basis towards these extensions.

In the literature, there have been several attempts to provide a mathematical foundation of computation beyond classical or symbolic one. We do not claim at all our game-semantic approach is best or canonical in comparison with previous work; however, our approach certainly has some advantages. For instance, Robin Gandy proposed in the famous paper [63] a notion of 'mechanical devices', now known as

Gandy machines (GMs), which are more general than TMs, and showed that TMs are actually as powerful as GMs. However, GMs are an *axiomatic* approach to define a general class of 'mechanical devices' that are 'effectively executable', and they do not give a distinction between high-level and low-level computational processes, where GMs formulate the latter. More recent *abstract state machines (ASMs)* [83] developed by Yuri Gurevich employ an idea similar to GMs for 'effective computability', namely, by requiring an upper bound on the number of elements that may change in a single step of computation, based on *structures* in the sense of mathematical logic [168]. Notably, ASMs define a very general notion of computation, namely, computation as *structure transition*. However, it seems that this framework is in some sense too general; for instance, it is possible that an ASM computes a real number in a single-step, but then its 'effectivity' is questionable. In general, an appropriate notion of 'effective computability' of ASMs has been missing. Also, the way of computing a function by an ASM is to update input/output-pairs in the *element-wise* fashion, but it does not seem to be a common or natural computational processes in practice. Yiannis Moschovakis also considered a mathematical foundation of algorithms [137] in which, similarly to us, he proposed that algorithms and their 'implementations' should be distinguished, where by algorithms he refers to what we call high-level computational processes. However, his framework, called *recursors*, is also based on structures, and his notion of algorithms is *relative* to atomic operations given in each structure; thus, it does not give a foundational analysis on the notion of 'effective computability'. To summarize, although these previous attempts capture broader notions of computation, our approach has advantages in achieving both of the distinction between high-level and low-level computational processes, and the primitive notion of 'effective computability'. Also, the interactive and typed natures of game semantics stand in sharp contrast to previous work.

At this point, let us mention *computability logic* [103] developed by Giorgi Japaridze since his idea is similar to ours; he defines 'effective computability' via computing machines playing in games. Nevertheless, there are notable differences between computability logic and the present work. First, the computing machines of computability logic are a variant of TMs, and so they are less novel than our approach as a model of computation; in fact, the notion of 'effectivity' of computability logic can be seen more or less as a consequence of just spelling out the standard notion of recursive strategies [9, 100, 60]. Second, our framework inherits the categorical structure of game semantics, providing a *compositional* formulation of logic and computation, i.e., a compound proof or program is constructed from its components, while there has been no known categorical structure of computability logic.

Finally, let us mention some of the precursors of game semantics. To clarify the notion of higher-order computability, Stephen Cole Kleene considered a model of higher-order computation based on 'dialogues' between computational oracles in a series of papers [110, 111, 112], which can be seen as the first attempt to define a mathematical notion of algorithms in a higher-order setting [122]. Moreover, Gandy and his student Giovanni Pani refined the work by Kleene to obtain a model of PCF that satisfies *universality* [44] (though this work was eventually not published). They are direct ancestors of game semantics (in particular of HO-games [100] by Martin Hyland and Luke Ong). As another line of research (motivated by the problem of full abstraction for PCF [151]), Pierre-Louis Curien and Gerard Berry proposed *sequential algorithms* [22], which was the first attempt to go beyond (extensional) functions to capture sequentiality of PCF [18]. Again, sequential algorithms preceded and became highly influential to the development of game semantics. In fact, sequential algorithms are presented in the style of game semantics in [122], and it is shown in [31] that the oracle computation given by Kleene can be represented by sequential algorithms (but the converse does not hold). Nevertheless, the point here is that neither of previous work defines 'effectivity' in a similar manner to the present work; our definition of 'effectivity' has an advantage in its intrinsic, non-inductive, non-axiomatic nature.

4.1.4 Chapter Outline

The rest of the present chapter is structured as follows. In Section 4.2, we present dynamic strategies that model PCF (Definition 4.2.9) and establish a key lemma (Lemma 4.2.12) which states that the dynamic strategies for PCF only need as an input at most the *last three moves* of each P-view and the *state* (Definition 4.2.11) of the P-view. Then, formalizing 'tags' for disjoint union of sets in Section 4.3, we give in Section 4.4 a solely game-semantic formulation of 'effectivity' or *viability* of strategies (Definition 4.4.7) by introducing game-semantic low-level computational processes underlying the high-level ones (i.e., dynamic strategies), and prove as a main theorem that viable dynamic strategies are PCF-complete (Theorem 4.4.18). This section also generalizes the smn- and the first recursion theorems (Corollaries 4.4.20 and 4.4.21). Finally, we draw a conclusion and propose future work of the chapter in Section 4.5.

4.2 Game-Semantic PCF-Computation

In this section, we formulate our game-semantic high-level computational processes for PCF. This is achieved by simply applying the framework of dynamic game semantics in Chapter 3 to PCF, where we replace the usual flat game N of natural numbers with the *lazy* one $\mathcal{N}$ (Definition 4.2.1), which gives a unary representation of natural numbers, for representing *step-by-step processes* in computation.

Then, we prove a key fact (Lemmma 4.2.12) for these high-level computational processes, which states that each dynamic strategy modeling a term of PCF only needs as an input the *last three moves* of each P-view and the *state* (Definition 4.2.11) of the P-view. By this fact, the problem of game-semantic PCF-completeness is reduced to formulating low-level computational processes underlying these dynamic strategies that calculate the next P-move from such three moves and states.

4.2.1 The Lazy Natural Number Game

Let us first point out that the flat game N (Example 2.2.12) has almost the same mathematical structure as that of the set $\mathbb{N}$ of all natural numbers. This point is unsatisfactory for at least the following two reasons:

1. It is not clear how to define an intrinsic, non-inductive, non-axiomatic notion of 'effective computability' of (dynamic) strategies on games generated from N via $\Rightarrow$ since there is no intensional or low-level structure in N;
2. The game N contributes almost nothing to foundations of mathematics.

Motivated by these points, we adopt the following 'lazy' variant:

Definition 4.2.1 (The lazy natural number game)**.** The ***lazy natural number game*** is the dynamic game $\mathcal{N}$ defined by:

- $M_{\mathcal{N}} \stackrel{\text{df.}}{=} \{\hat{q}, q, yes, no\}$;
- $\lambda_{\mathcal{N}} : \hat{q} \mapsto \mathsf{OQ}0, q \mapsto \mathsf{OQ}0, yes \mapsto \mathsf{PA}0, no \mapsto \mathsf{PA}0$;
- $\vdash_{\mathcal{N}} \stackrel{\text{df.}}{=} \{(\star, \hat{q}), (\hat{q}, no), (\hat{q}, yes), (q, no), (q, yes), (yes, q)\}$;
- $P_{\mathcal{N}} \stackrel{\text{df.}}{=} \mathsf{Pref}(\{\hat{q}.(yes.q)^n.no \mid n \in \mathbb{N}\})$, where each non-initial occurrence is justified by the last occurrence;
- $\boldsymbol{s} \simeq_{\mathcal{N}} \boldsymbol{t} \stackrel{\text{df.}}{\Leftrightarrow} \boldsymbol{s} = \boldsymbol{t}$.

We need both $\hat{q}$ and q for the axiom E1 (Definition 3.3.1). A play of $\mathcal{N}$ proceeds as an iteration of: Opponent asks by a question $\hat{q}$ or q if Player would like to 'count one more', and Player replies it by *yes* if she would like to, and by *no* otherwise. Thus, for each $n \in \mathbb{N}$, the position $\hat{q}.(yes.q)^n.no \in P_{\mathcal{N}}$ represents the number n. Let us define $\underline{n} \stackrel{\mathrm{df.}}{=} \mathsf{Pref}(\{\hat{q}.(yes.q)^n.no\})^{\mathsf{Even}} : \mathcal{N}$ for each $n \in \mathbb{N}$.

Note that $\mathcal{N}$ defines natural numbers in a primitive, natural way, viz., as 'counting processes', where the notation for moves is inessential, i.e., $\mathcal{N}$ is syntax-independent. Also, we may define it without recourse to the set $\mathbb{N}$ by specifying its maximal positions inductively: $\hat{q}.no \in P_{\mathcal{N}} \wedge (\hat{q}.\boldsymbol{s}.no \in P_{\mathcal{N}} \Rightarrow \hat{q}.\boldsymbol{s}.yes.q.no \in P_{\mathcal{N}})$. Thus, we may *define* (rather than *represent*) natural numbers to be the dynamic strategies $\underline{n} : \mathcal{N}$ though we shall not investigate its foundational consequences in the present thesis.

Also, this step-by-step process of natural numbers allows us to define 'effectivity' of strategies in an intrinsic, non-inductive, non-axiomatic manner in Section 4.4.1.

4.2.2 Dynamic Games and Strategies for PCF-Computation

Next, let us recall games and strategies that model PCF [14, 101], where we replace the flat natural number game N with the lazy one $\mathcal{N}$ (Definition 4.2.1), and more generally conventional or static games and strategies with dynamic ones (defined in Chapter 3). Note that we do not recall PCF itself or its relation with the static game semantics (e.g., full abstraction), leaving these points to [14, 100, 9], and just focus on dynamic games and strategies modeling PCF (without referring to the syntax).

We shall see in particular that each of the dynamic strategies introduced below is representable by a finite partial function that maps the last three moves of each P-view to the next P-move. Note that *states* (Definition 4.2.11) of P-views are not necessary until we form a pairing (Definition 3.3.43) of dynamic strategies; see Section 4.2.3.

Notation. We shall represent dynamic strategies given below as partial functions $(m_3, m_2, m_1) \mapsto m$, where m_1 is the last move, m_2 is the second last move, and m_3 is the third last move of the P-view of a given odd-length position, and m is the next P-move. If there is no such m_2 or m_3 in the P-view, then we replace it with $\Box$, representing 'no move'. For these dynamic strategies, the justifier of m is always m_1, m_3 or the justifier of m_2 as we shall see; we omit it in the partial map representation as it is always clear. We henceforth write $(\_)^{[i]}$ as another notation for $(\_)_{[i]}$.

Let us first recall the simplest dynamic strategy that represents $0 \in \mathbb{N}$:

Definition 4.2.2 (Zero strategies [14, 101])**.** Given a normalized dynamic game A, the ***zero strategy*** on A is the dynamic strategy $zero_A : A^{[0]} \Rightarrow \mathcal{N}^{[1]}$ given by:

$$zero_A \overset{\text{df.}}{=} \mathsf{Pref}(\hat{q}^{[1]}.no^{[1]})^{\mathsf{Even}}.$$

The canonical play by $zero_A$ can be described as Figure 4.1 below. Clearly, $zero_A$

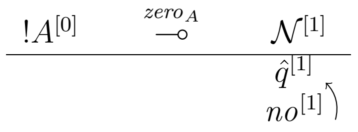


Figure 4.1: The computation of the zero strategy $zero_A : A \Rightarrow \mathcal{N}$.

may be represented by the following constant function:

$$(\Box, \Box, \hat{q}^{[1]}) \mapsto no^{[1]}.$$

Next, let us recall the *successor strategy*:

Definition 4.2.3 (Successor strategy [14, 101])**.** The ***succesor strategy*** is the dynamic strategy $succ : \mathcal{N}^{[0]} \Rightarrow \mathcal{N}^{[1]}$ defined by:

$$succ \overset{\text{df.}}{=} \mathsf{Pref}(\{\hat{q}^{[1]}.(\hat{q},0)^{[0]}.((yes,0)^{[0]}.yes^{[1]}.q^{[1]}.(q,0)^{[0]})^i.(no,0)^{[0]}.yes^{[1]}.q^{[1]}.no^{[1]} \mid i \in \mathbb{N}\})^{\mathsf{Even}}$$

where pointers occurring in $succ$ are as described in Figure 4.2.

Roughly, $succ$ copies a given input on the domain $!\mathcal{N}^{[0]}$ and repeats it as an output on the codomain $\mathcal{N}^{[1]}$ in the 'move-wise' manner, like the dereliction $der_{\mathcal{N}} : \mathcal{N} \Rightarrow \mathcal{N}$, except that $succ$ adds one more $yes^{[1]}$ before $no^{[1]}$; see Figure 4.2. Clearly, $succ \circ \underline{n}^{\dagger} = \underline{n+1}$ for all $n \in \mathbb{N}$, and thus $succ$ indeed computes the successor function.

It is then easy to observe that $succ$ only needs as an input at most the last three moves of each P-view to compute the next P-move. More concretely, $succ$ can be represented by the following finite list of the input-output pairs:

$$\begin{aligned}
&(\Box, \Box, \hat{q}^{[1]}) \mapsto (\hat{q},0)^{[0]} \mid ((no,0)^{[0]}, yes^{[1]}, q^{[1]}) \mapsto no^{[1]} \mid \\
&(q^{[1]}, (q,0)^{[0]}, (yes,0)^{[0]}) \mapsto yes^{[1]} \mid (q^{[1]}, (q,0)^{[0]}, (no,0)^{[0]}) \mapsto yes^{[1]} \mid \\
&(\hat{q}^{[1]}, (\hat{q},0)^{[0]}, (yes,0)^{[0]}) \mapsto yes^{[1]} \mid (\hat{q}^{[1]}, (\hat{q},0)^{[0]}, (no,0)^{[0]}) \mapsto yes^{[1]} \mid \\
&((yes,0)^{[0]}, yes^{[1]}, q^{[1]}) \mapsto (q,0)^{[0]}
\end{aligned}$$

Next, let us recall a left inverse of the successor strategy:

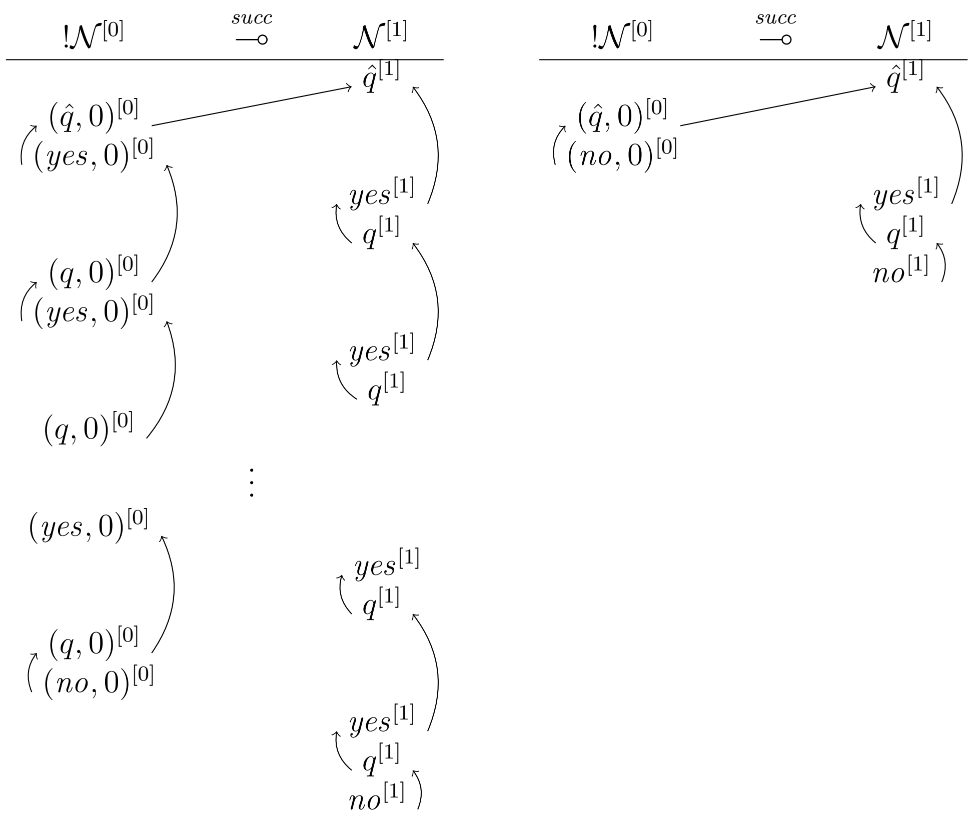


Figure 4.2: The computation of the successor strategy $succ : \mathcal{N} \Rightarrow \mathcal{N}$.

Definition 4.2.4 (Predecessor strategy [14, 101])**.** The ***predecessor strategy*** is the dynamic strategy $pred : \mathcal{N}^{[0]} \Rightarrow \mathcal{N}^{[1]}$ defined by:

$$\begin{aligned} pred \overset{\text{df.}}{=} \mathsf{Pref}(&\{\hat{q}^{[1]}.(\hat{q},0)^{[0]}.(yes,0)^{[0]}.(q,0)^{[0]}.((yes,0)^{[0]}.yes^{[1]}.q^{[1]}.(q,0)^{[0]})^i.(no,0)^{[0]}.no^{[1]} \mid i \in \mathbb{N}\} \\ &\cup \{\hat{q}^{[1]}.(\hat{q},0)^{[0]}.(no,0)^{[0]}.no^{[1]}\})^{\mathsf{Even}} \end{aligned}$$

where pointers occurring in *pred* are as described in Figure 4.3.

Somewhat similarly to *succ*, *pred* computes like $der_{\mathcal{N}}$ except that if a given input on the domain $!\mathcal{N}^{[0]}$ is non-zero, then it does not copy the first *yes*. The computation of *pred* is described in Figure 4.3. It is then easy to see that $pred \circ \underline{n+1}^\dagger = \underline{n}$ for all $n \in \mathbb{N}$, and $pred \circ \underline{0}^\dagger = \underline{0}$; thus, *pred* in fact computes the predecessor function. We also have $pred \circ succ^\dagger = der_{\mathcal{N}}$ as mentioned above.

Also, *pred* only needs at most the last three moves in each P-view to compute the

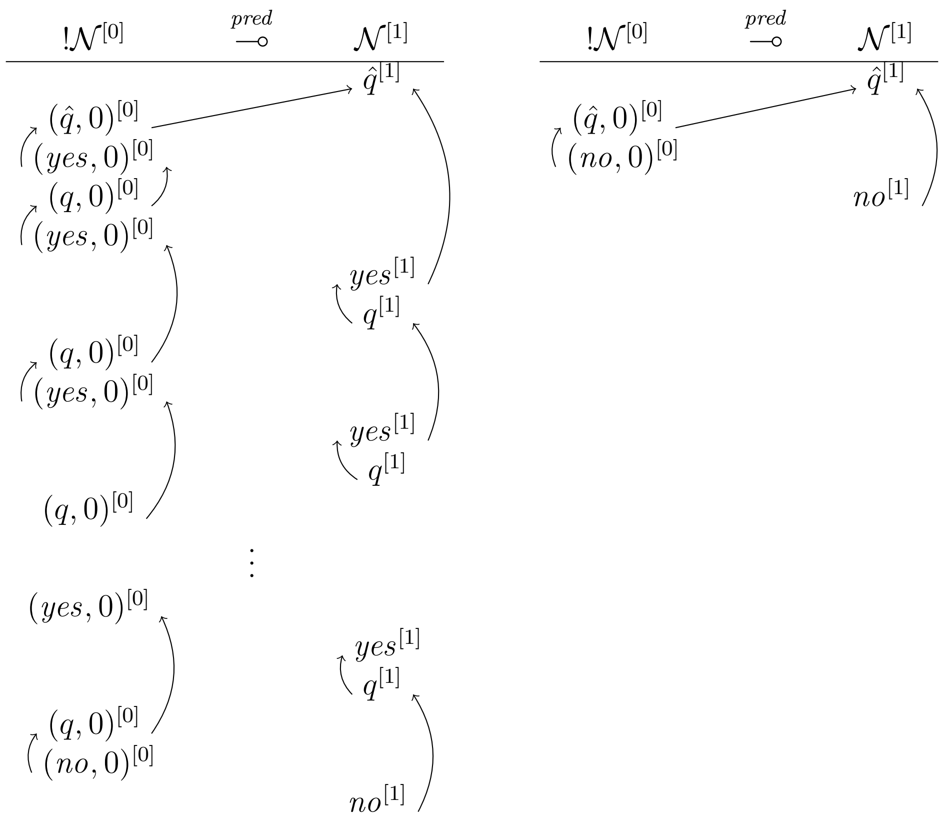


Figure 4.3: The computation of the predecessor strategy $pred : \mathcal{N} \Rightarrow \mathcal{N}$.

next P-move as it can be represented by:

$$(\Box, \Box, \hat{q}^{[1]}) \mapsto (\hat{q}, 0)^{[0]} \mid (\hat{q}^{[1]}, (\hat{q}, 0)^{[0]}, (yes, 0)^{[0]}) \mapsto (q, 0)^{[0]} \mid$$
$$((yes, 0)^{[0]}, (q, 0)^{[0]}, (yes, 0)^{[0]}) \mapsto yes^{[1]} \mid ((yes, 0)^{[0]}, yes^{[1]}, q^{[1]}) \mapsto (q, 0)^{[0]} \mid$$
$$(q^{[1]}, (q, 0)^{[0]}, (yes, 0)^{[0]}) \mapsto yes^{[1]} \mid (\hat{q}^{[1]}, (\hat{q}, 0)^{[0]}, (no, 0)^{[0]}) \mapsto no^{[1]} \mid$$
$$(q^{[1]}, (q, 0)^{[0]}, (no, 0)^{[0]}) \mapsto no^{[1]}$$

Convention. In the present chapter, the ***boolean game*** refers to the corresponding normalized dynamic game $\mathbf{2}$, where we replace the question q with $\hat{q}$, by default.

Notation. We define dynamic strategies $\underline{tt} \stackrel{\text{df.}}{=} \{\boldsymbol{\epsilon}, \hat{q}.tt\} : \mathbf{2}$ and $\underline{ff} \stackrel{\text{df.}}{=} \{\boldsymbol{\epsilon}, \hat{q}.ff\} : \mathbf{2}$.

Then, let us recall a dynamic strategy for the conditional 'if...then...else...':

Definition 4.2.5 (Case strategies [14, 101])**.** The ***case strategy*** on a normalized dynamic game A is the dynamic strategy $case_A : A^{[0]} \& A^{[1]} \& \mathbf{2}^{[2]} \Rightarrow A^{[3]}$ defined by:

$$case_A \stackrel{\text{df.}}{=} \mathsf{Pref}(\{a^{[3]}.(\hat{q}, 0)^{[2]}.(tt, 0)^{[2]}.(a, 0)^{[0]}.\boldsymbol{s} \mid a^{[1]}.(a, 0)^{[0]}.\boldsymbol{s}^{[3 \mapsto 1]} \in der_A\}$$
$$\cup \{a^{[3]}.(\hat{q}, 0)^{[2]}.(ff, 0)^{[2]}.(a, 0)^{[1]}.\boldsymbol{s} \mid a^{[1]}.(a, 0)^{[0]}.\boldsymbol{s}^{[1 \mapsto 0, 3 \mapsto 1]} \in der_A\})^{\mathsf{Even}}$$

where $\boldsymbol{s}^{[3 \mapsto 1]}$ (resp. $\boldsymbol{s}^{[1 \mapsto 0, 3 \mapsto 1]}$) is obtained from $\boldsymbol{s}$ by replacing the 'tag' $(\_)^{[3]}$ with $(\_)^{[1]}$ (resp. $(\_)^{[1]}$ with $(\_)^{[0]}$, and $(\_)^{[3]}$ with $(\_)^{[1]}$), and pointers occurring in $case_A$ are as illustrated in Figure 4.4 (n.b., $!(A \& A \& \mathbf{2})$ and $!A \otimes !A \otimes !\mathbf{2}$ coincide up to 'tags').

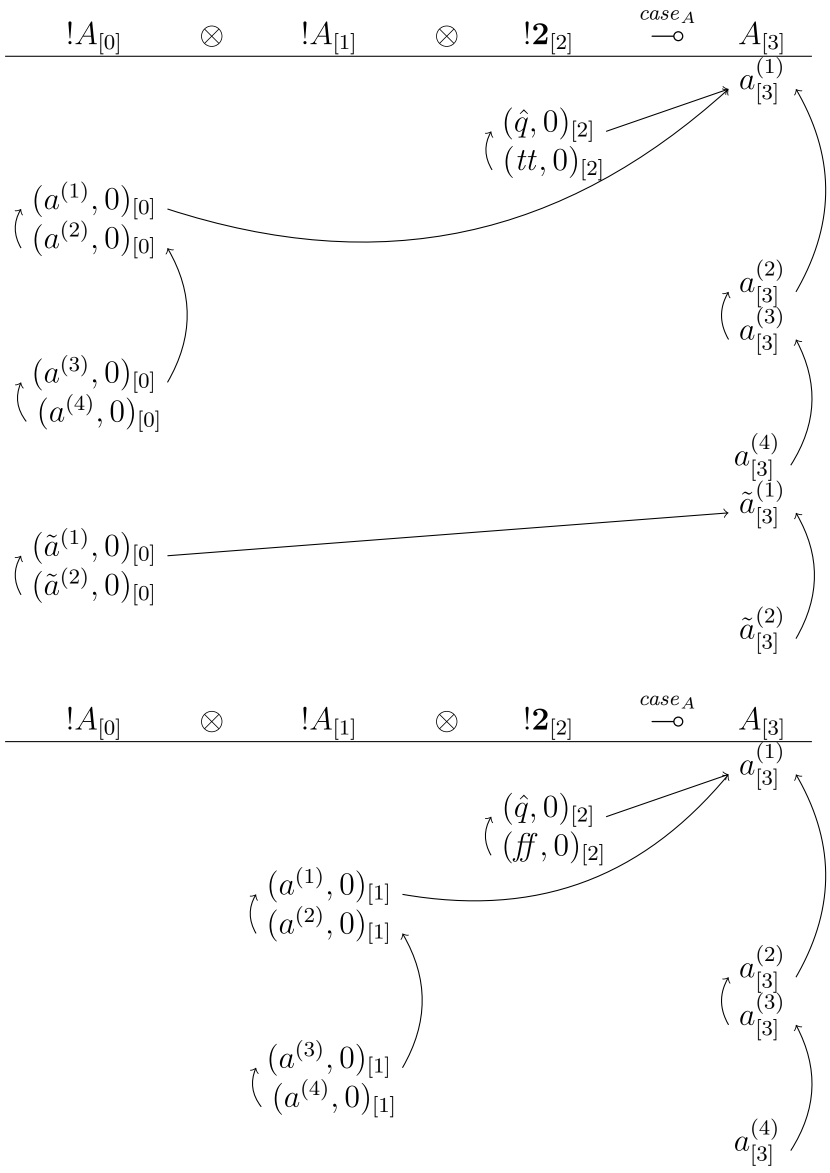


Figure 4.4: A computation of the case strategy $case_A : A \& A \& \mathbf{2} \Rightarrow A$.

That is, $case_A$ first investigates a given input on $\mathbf{2}^{[2]}$ and then plays as der_A between $!A^{[0]}$ and $A^{[3]}$ if the input is $\underline{tt}$, and between $!A^{[1]}$ and $A^{[3]}$ otherwise. Typical plays by $case_A$ are illustrated in Figure 4.4. Clearly, given any $\alpha_1, \alpha_2 : A$, we have $case_A \circ \langle \alpha_1, \alpha_2, \underline{tt} \rangle^{\dagger} = \alpha_1$ and $case_A \circ \langle \alpha_1, \alpha_2, \underline{ff} \rangle^{\dagger} = \alpha_2$, and thus $case_A$ indeed computes the conditional 'if...then...else...'. At this point, it should be clear how $case_A$ may be represented as a finite table, where it only needs at most the last three moves of each P-view; thus, we leave it to the reader.

Next, let us recall a dynamic strategy that sees if a given input is zero:

Definition 4.2.6 (Ifzero strategy [14, 101])**.** The ***ifzero strategy*** is the dynamic strategy $\mathit{zero?} : \mathcal{N}^{[0]} \Rightarrow \mathbf{2}^{[1]}$ defined by:

$$\mathit{zero?} \overset{\mathrm{df.}}{=} \mathsf{Pref}(\{\hat{q}^{[1]}.(\hat{q},0)^{[0]}.(no,0)^{[0]}.tt^{[1]}, \hat{q}^{[1]}.(\hat{q},0)^{[0]}.(yes,0)^{[0]}.\mathit{ff}^{[1]}\})^{\mathsf{Even}}$$

where pointers occurring in $\mathit{zero?}$ are described in Figure 4.5.

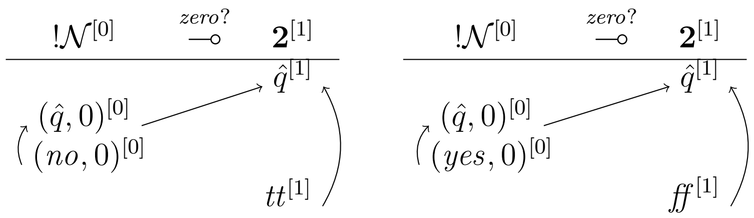


Figure 4.5: The computation of the ifzero strategy $\mathit{zero?} : \mathcal{N} \Rightarrow \mathbf{2}$.

That is, $\mathit{zero?}$ investigates an input on the domain $!\mathcal{N}^{[0]}$ by seeing the first digit and outputs an answer on the codomain $\mathbf{2}^{[1]}$ accordingly. The computation of $\mathit{zero?}$ is depicted in Figure 4.5. Clearly, $\mathit{zero?} \circ \underline{n+1}^{\dagger} = \underline{tt}$ for all $n \in \mathbb{N}$, and $\mathit{zero?} \circ \underline{0}^{\dagger} = \underline{\mathit{ff}}$; thus, $\mathit{zero?}$ correctly decides if an input is zero. Also, $\mathit{zero?}$ only needs as an input at most the last three moves of each P-view; again, we omit the detail.

Finally, let us recall dynamic strategies modeling *fixed-point combinators* [151, 18]. Since their formal definition is rather involved, we give an informal description:

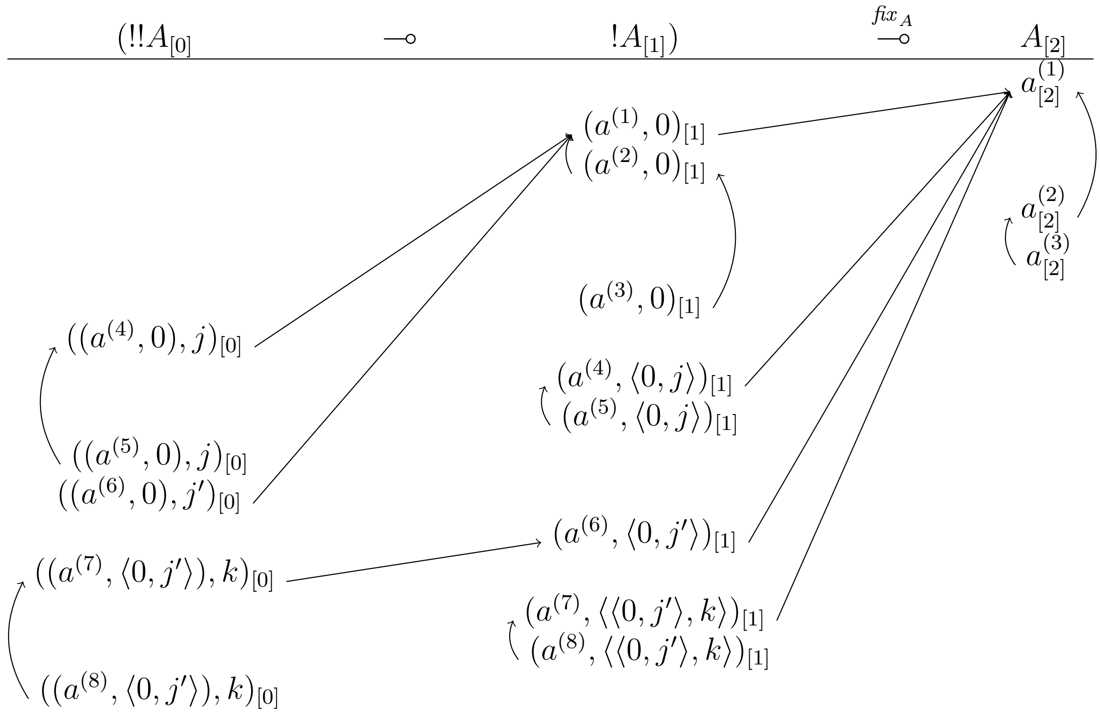
Definition 4.2.7 (Fixed-point strategies [101])**.** Given a normalized dynamic game A, the ***fixed-point strategy*** $\mathit{fix}_A : (A_{[0]} \Rightarrow A_{[1]}) \Rightarrow A_{[2]}$ on A computes as follows:

- After an initial occurrence $a_{[2]}$, fix_A copies it and performs the move $(a,0)_{[1]}$ with the pointer towards the initial occurrence $a_{[2]}$;
- If Opponent initiates a new *thread*[2] in the inner implication by $((a',i),j)_{[0]}$, then fix_A copies it and launches a new thread in the outer implication by performing the move $(a',\langle i,j\rangle)_{[1]}$ with the pointer towards the justifier of the justifier of $((a',i),j)_{[0]}$, i.e., the justifier of the second last move of the current P-view (n.b., recall that $\langle \_,\_ \rangle$ is any fixed bijection $\mathbb{N} \times \mathbb{N} \xrightarrow{\sim} \mathbb{N}$ [46, 181]);

[2]A *thread* in a j-sequence $\boldsymbol{s}$ is a j-subsequence of $\boldsymbol{s}$ that consists of moves *hereditarily justified* by the same initial occurrence in $\boldsymbol{s}$; see [14, 129] for its precise definition.

- If Opponent performs a move $((a'', i), j)_{[0]}$ (resp. $(a'', 0)_{[1]}$, $(a'', \langle i, j \rangle)_{[1]}$, $a''_{[2]}$) in an existing thread, then fix_A copies it and performs the move $(a'', \langle i, j \rangle)_{[1]}$ (resp. $a''_{[2]}$, $((a'', i), j)_{[0]}$, $(a'', 0)_{[1]}$) in the *dual thread*, i.e., the thread to which the third last move of the P-view belongs, with the pointer towards the third last move.

A typical play by fix_A can be depicted as in Figure 4.6. Note that we cannot give



$(!!A_{[0]}$
$\multimap$
$!A_{[1]})$
$\overset{\mathit{fix}_A}{\multimap}$
$A_{[2]}$
$a^{(1)}_{[2]}$
$(a^{(1)}, 0)_{[1]}$
$(a^{(2)}, 0)_{[1]}$
$a^{(2)}_{[2]}$
$a^{(3)}_{[2]}$
$(a^{(3)}, 0)_{[1]}$
$((a^{(4)}, 0), j)_{[0]}$
$(a^{(4)}, \langle 0, j \rangle)_{[1]}$
$(a^{(5)}, \langle 0, j \rangle)_{[1]}$
$((a^{(5)}, 0), j)_{[0]}$
$((a^{(6)}, 0), j')_{[0]}$
$(a^{(6)}, \langle 0, j' \rangle)_{[1]}$
$((a^{(7)}, \langle 0, j' \rangle), k)_{[0]}$
$(a^{(7)}, \langle \langle 0, j' \rangle, k \rangle)_{[1]}$
$(a^{(8)}, \langle \langle 0, j' \rangle, k \rangle)_{[1]}$
$((a^{(8)}, \langle 0, j' \rangle), k)_{[0]}$

Figure 4.6: A computation of the fixed-point strategy $\mathit{fix}_A : (A \Rightarrow A) \Rightarrow A$.

a finite table for fix_A, i.e., fix_A is not finitary, because there are infinitely many 'tags' for exponential which fix_A has to handle. Nevertheless, since fix_A behaves essentially in the same way as the dereliction der_A (up to 'tags' and pointers), it is not hard to see that fix_A only needs to refer to as an input at most the last three moves of each P-view. More concretely, fix_A can be represented by the following *infinite* table:

$$(x, y, a_{[2]}) \mapsto (a, 0)_{[1]} \mid (x, y, (a, 0)_{[1]}) \mapsto a_{[2]} \mid (x, y, a_{[2]}) \mapsto (a, 0)_{[1]} \mid$$
$$(x, y, (a, (i, j))_{[0]}) \mapsto (a, \langle i, j \rangle)_{[1]} \mid (x, y, (a, \langle i, j \rangle)_{[1]}) \mapsto (a, (i, j))_{[0]}$$

where $i, j \in \mathbb{N}$, $a \in M_A$, and x and y denote any moves or $\square$.

We have given all the 'atomic' dynamic strategies for PCF-computation. Hence, we may now define an enumeration of all the dynamic strategies modeling PCF. For clarity, however, let us first recall the corresponding *normalized* dynamic strategies, where we regard normalized dynamic games and strategies as conventional (or static) games and strategies, respectively (see Chapter 3 for the detail):

Definition 4.2.8 (Static strategies for PCF [100, 9, 101, 14])**.** Let $\mathcal{PCF}$ be the least set of normalized dynamic strategies $\sigma : S_\sigma$ that satisfies:

1. $\sigma : S_\sigma \in \mathcal{PCF}$ if $\sigma : S_\sigma$ is ***PCF-atomic***, i.e., $der_A : A \Rightarrow A$, $zero_A : A \Rightarrow \mathcal{N}$, $succ : \mathcal{N} \Rightarrow \mathcal{N}$, $pred : \mathcal{N} \Rightarrow \mathcal{N}$, $zero? : \mathcal{N} \Rightarrow \mathbf{2}$, $case_A : A \& A \& \mathbf{2} \Rightarrow A$ or $fix_A : (A \Rightarrow A) \Rightarrow A$, where A is generated from $\mathcal{N}$, $\mathbf{2}$ and/or T, by & and/or $\Rightarrow$ (n.b., the construction of A is 'orthogonal' to that of $\sigma : S_\sigma$);
2. $\Lambda(\sigma) : A \Rightarrow (B \Rightarrow C) \in \mathcal{PCF}$ if $\sigma : A \& B \Rightarrow C \in \mathcal{PCF}$ for some normalized dynamic games A, B and C;
3. $\langle \varphi, \psi \rangle : C \Rightarrow A \& B \in \mathcal{PCF}$ if $\varphi : C \Rightarrow A, \psi : C \Rightarrow B \in \mathcal{PCF}$ for some normalized dynamic games A, B and C;
4. $\iota^\dagger ; \kappa : A \Rightarrow C \in \mathcal{PCF}$ if $\iota : A \Rightarrow B, \kappa : B \Rightarrow C \in \mathcal{PCF}$ for some normalized dynamic games A, B and C.

It is easy to see that $\mathcal{PCF}$ contains all the normalized dynamic strategies $\sigma : S_\sigma$ that model PCF [14, 101], where note that *projections* and *evaluations* are derelictions up to 'tags', and thus we count them as PCF-atomic ones as well. Note that the dynamic strategy $\underline{n}^T : T \Rightarrow \mathcal{N}$, which models the n^{th}-numeral, may be obtained by $zero_T^\dagger ; \underbrace{succ^\dagger ; succ^\dagger \cdots ; succ}_{n}$, and thus $\underline{n}^T : T \Rightarrow \mathcal{N} \in \mathcal{PCF}$, for each $n \in \mathbb{N}$. Hence, the set $\mathcal{PCF}$ contains all the conventional game semantics of terms of PCF [14, 101] (except that the dynamic game N is replaced with $\mathcal{N}$).

Now, by replacing composition $(\_)^\dagger ; (\_)$ in the inductive construction of the set $\mathcal{PCF}$ with concatenation $(\_)^\dagger \ddagger (\_)$ (Definition 3.3.47), and accordingly static currying and pairing with the generalized ones (Definitions 3.3.49 and 3.3.43), respectively, we obtain an enumeration of all the *dynamic* game semantics of terms of PCF:

Definition 4.2.9 (Dynamic strategies for PCF)**.** Let $\phi_\sigma : D_\sigma$ be the dynamic strategy assigned to each element $\sigma : S_\sigma \in \mathcal{PCF}$ by the following induction:

1. $\phi_\sigma : D_\sigma \overset{\text{df.}}{=} \sigma : S_\sigma$ if $\sigma : S_\sigma \in \mathcal{PCF}$, and it is PCF-atomic (Definition 4.2.8), where we call ϕ_σ ***PCF-atomic*** as well;
2. $\phi_{\Lambda(\sigma)} \overset{\text{df.}}{=} \Lambda(\phi_\sigma)$ and $D_{\Lambda(\sigma)} \overset{\text{df.}}{=} \Lambda(D_\sigma)$ if $\Lambda(\sigma) : A \Rightarrow (B \Rightarrow C) \in \mathcal{PCF}$ is obtained from $\sigma : A \& B \Rightarrow C$;

3. $\phi_{\langle \varphi, \psi \rangle} \overset{\mathrm{df.}}{=} \langle \phi_\varphi, \phi_\psi \rangle$ and $D_{\langle \varphi, \psi \rangle} \overset{\mathrm{df.}}{=} \langle D_\varphi, D_\psi \rangle$ if $\langle \varphi, \psi \rangle : C \Rightarrow A \& B \in \mathcal{PCF}$ is obtained from $\varphi : C \Rightarrow A, \psi : C \Rightarrow B \in \mathcal{PCF}$;

4. $\phi_{\iota^\dagger ; \kappa} \overset{\mathrm{df.}}{=} \phi_\iota^\dagger \ddagger \phi_\kappa$ and $D_{\iota^\dagger ; \kappa} \overset{\mathrm{df.}}{=} D_\iota^\dagger \ddagger D_\kappa$ if $\iota^\dagger ; \kappa : A \Rightarrow C \in \mathcal{PCF}$ is obtained from $\iota : A \Rightarrow B, \kappa : B \Rightarrow C \in \mathcal{PCF}$.

The set $\mathcal{DPCF}$ is then defined by $\mathcal{DPCF} \overset{\mathrm{df.}}{=} \{ \phi_\sigma : D_\sigma \mid \sigma : S_\sigma \in \mathcal{PCF} \}$.

In fact, the set $\mathcal{DPCF}$ contains all the dynamic strategies modeling PCF as the following lemma implies:

Lemma 4.2.10 (DPCF-lemma)**.** *For each $\sigma : S_\sigma \in \mathcal{PCF}$, ϕ_σ is a dynamic strategy on a dynamic game D_σ that satisfies $\mathcal{H}^\omega(\phi_\sigma) = \sigma$ and $\mathcal{H}^\omega(D_\sigma) \trianglelefteq S_\sigma$.*

Proof. By induction on the construction of $\sigma : S_\sigma \in \mathcal{PCF}$:

1. Assume that $\sigma : S_\sigma \in \mathcal{PCF}$ is PCF-atomic (Definition 4.2.8). Then, $\phi_\sigma = \sigma : S_\sigma = D_\sigma$, $\mathcal{H}^\omega(\phi_\sigma) = \mathcal{H}^\omega(\sigma) = \sigma$ and $\mathcal{H}^\omega(D_\sigma) = \mathcal{H}^\omega(S_\sigma) = S_\sigma \trianglelefteq S_\sigma$.

2. Assume that $\Lambda(\sigma) : A \Rightarrow (B \Rightarrow C) \in \mathcal{PCF}$ is constructed from $\sigma : A \& B \Rightarrow C \in \mathcal{PCF}$. Since $\sigma : A \& B \Rightarrow C \in \mathcal{PCF}$, the induction hypothesis implies that $\phi_\sigma : D_\sigma$, $\mathcal{H}^\omega(\phi_\sigma) = \sigma$ and $\mathcal{H}^\omega(D_\sigma) \trianglelefteq S_\sigma = A \& B \Rightarrow C$. Thus, $\Lambda(D_\sigma)$ is a dynamic game such that $\mathcal{H}^\omega(\Lambda(D_\sigma)) \trianglelefteq A \Rightarrow (B \Rightarrow C)$. By Lemmata 3.3.30, 3.3.52 and 3.3.29, $\phi_{\Lambda(\sigma)} = \Lambda(\phi_\sigma) : \Lambda(D_\sigma) = D_{\Lambda(\sigma)}$, $\mathcal{H}^\omega(\phi_{\Lambda(\sigma)}) = \mathcal{H}^\omega(\Lambda(\phi_\sigma)) = \Lambda(\mathcal{H}^\omega(\phi_\sigma)) = \Lambda(\sigma)$ and $\mathcal{H}^\omega(D_{\Lambda(\sigma)}) = \mathcal{H}^\omega(\Lambda(D_\sigma)) \trianglelefteq \Lambda(\mathcal{H}^\omega(D_\sigma)) \trianglelefteq \Lambda(S_\sigma)$.

3. Assume that $\langle \varphi, \psi \rangle : C \Rightarrow A \& B \in \mathcal{PCF}$ is constructed from $\varphi : C \Rightarrow A, \psi : C \Rightarrow B \in \mathcal{PCF}$. Since $\varphi : C \Rightarrow A, \psi : C \Rightarrow B \in \mathcal{PCF}$, the induction hypothesis implies that $\phi_\varphi : D_\varphi$, $\mathcal{H}^\omega(\phi_\varphi) = \varphi$, $\mathcal{H}^\omega(D_\varphi) \trianglelefteq S_\varphi = C \Rightarrow A$, $\phi_\psi : D_\psi$, $\mathcal{H}^\omega(\phi_\psi) = \psi$ and $\mathcal{H}^\omega(D_\psi) \trianglelefteq S_\psi = C \Rightarrow B$. Thus, $\langle D_\varphi, D_\psi \rangle$ is a dynamic game such that $\mathcal{H}^\omega(\langle D_\varphi, D_\psi \rangle) \trianglelefteq C \Rightarrow A \& B$. By Lemmata 3.3.30, 3.3.52 and 3.3.29, $\phi_{\langle \varphi, \psi \rangle} = \langle \phi_\varphi, \phi_\psi \rangle : \langle D_\varphi, D_\psi \rangle = D_{\langle \varphi, \psi \rangle}$, $\mathcal{H}^\omega(\phi_{\langle \varphi, \psi \rangle}) = \mathcal{H}^\omega(\langle \phi_\varphi, \phi_\psi \rangle) = \langle \mathcal{H}^\omega(\phi_\varphi), \mathcal{H}^\omega(\phi_\psi) \rangle = \langle \varphi, \psi \rangle$ and $\mathcal{H}^\omega(D_{\langle \varphi, \psi \rangle}) = \mathcal{H}^\omega(\langle D_\varphi, D_\psi \rangle) \trianglelefteq \langle \mathcal{H}^\omega(D_\varphi), \mathcal{H}^\omega(D_\psi) \rangle \trianglelefteq \langle S_\varphi, S_\psi \rangle = S_{\langle \varphi, \psi \rangle}$.

4. Assume that $\iota^\dagger ; \kappa : A \Rightarrow C \in \mathcal{PCF}$ is constructed from $\iota : A \Rightarrow B, \kappa : B \Rightarrow C \in \mathcal{PCF}$. The induction hypothesis implies that $\phi_\iota : D_\iota$, $\mathcal{H}^\omega(\phi_\iota) = \iota$, $\mathcal{H}^\omega(D_\iota) \trianglelefteq S_\iota = A \Rightarrow B$, $\phi_\kappa : D_\kappa$, $\mathcal{H}^\omega(\phi_\kappa) = \kappa$ and $\mathcal{H}^\omega(D_\kappa) \trianglelefteq S_\kappa = B \Rightarrow C$. Thus, $\mathcal{H}^\omega(D_\iota^\dagger) \trianglelefteq \mathcal{H}^\omega(D_\iota)^\dagger \trianglelefteq S_\iota^\dagger \trianglelefteq A \Rightarrow {!B}$ by Lemmata 3.3.30 and 3.3.29, and therefore $D_\iota^\dagger \ddagger D_\kappa$ is a dynamic game such that $\mathcal{H}^\omega(D_\iota^\dagger \ddagger D_\kappa) \trianglelefteq A \Rightarrow C$ by

Lemma 3.3.30. Finally, by Lemmata 3.3.30, 3.3.52 and 3.3.29, $\phi_{\iota^\dagger ; \kappa} = \phi_\iota^\dagger \ddagger \phi_\kappa : D_\iota^\dagger \ddagger D_\kappa = D_{\iota^\dagger;\kappa}$, $\mathcal{H}^\omega(\phi_{\iota^\dagger;\kappa}) = \mathcal{H}^\omega(\phi_\iota^\dagger \ddagger \phi_\kappa) = \mathcal{H}^\omega(\phi_\iota)^\dagger ; \mathcal{H}^\omega(\phi_\kappa) = \iota^\dagger ; \kappa$ and $\mathcal{H}^\omega(D_{\iota^\dagger;\kappa}) = \mathcal{H}^\omega(D_\iota^\dagger \ddagger D_\kappa) \trianglelefteqslant A \Rightarrow C = S_{\iota^\dagger;\kappa}$

which completes the proof. $\square$

Hence, the main problem of the present chapter (see Section 4.1.1) has been reduced to defining an intrinsic, non-inductive, non-axiomatic notion of 'effective computability' of dynamic strategies that holds for all the elements of the set $\mathcal{DPCF}$.

4.2.3 The Last-Three-Move Lemma

We have seen in the previous section that every PCF-atomic dynamic strategy in the set $\mathcal{DPCF}$ (Definition 4.2.9) refers to as an input at most the last three moves of each P-view. Also, the second and fourth constructions of $\mathcal{DPCF}$, i.e., currying Λ and concatenation (and promotion) $(\_)^\dagger \ddagger (\_)$, clearly preserve this property, where note that concatenation preserves the property because a P-view of a concatenation $J \ddagger K$ is either that of J or K, which lies at the heart of our approach to theory of computation, n.b., it does not hold for composition $(\_)^\dagger ; (\_)$ in the set $\mathcal{PCF}$ (Definition 4.2.8).

However, the third construction, i.e., pairing $\langle \_, \_ \rangle$, does not preserve the property. To see why it is the case, consider an arbitrary pairing $\langle \varphi, \psi \rangle : \langle L, R \rangle \in \mathcal{DPCF}$, where $\mathcal{H}^\omega(L) \trianglelefteqslant C \Rightarrow A$ and $\mathcal{H}^\omega(R) \trianglelefteqslant C \Rightarrow B$ for some normalized dynamic games A, B and C. The point is that if the last three moves of a P-view of the dynamic game $\langle L, R \rangle$ all come from C, then the pairing $\langle \varphi, \psi \rangle$ cannot make a choice between φ and ψ to employ for computing the next P-move.

To overcome this problem, it seems a reasonable idea to allow the pairing $\langle \varphi, \psi \rangle$ to refer to the *first move* of the P-view as well, which must come from A or B. Note, however, that such a move becomes no longer the first move of a P-view as soon as the pairing is *post-concatenated*, i.e., it is not the first move of a P-view of the concatenation $\langle \varphi, \psi \rangle^\dagger \ddagger \phi : \langle L, R \rangle \ddagger D$ with some $\phi : D \in \mathcal{DPCF}$ such that $\mathcal{H}^\omega(D) \trianglelefteqslant A \& B \Rightarrow E$ for some normalized dynamic game E; hence, it does not suffice to trace the first move (in addition to the last three moves) of each P-view. For this point, we distinguish the relevant moves of each P-view as the *state* of the P-view[3]:

Definition 4.2.11 (States of P-views). Let $\phi_\sigma : D_\sigma \in \mathcal{DPCF}$. The ***state*** $\varsigma_{D_\sigma}(\lceil \boldsymbol{s} \rceil_{D_\sigma})$ of each P-view $\lceil \boldsymbol{s} \rceil_{D_\sigma} \in \lceil P_{D_\sigma} \rceil_{D_\sigma}$ such that $\boldsymbol{s} \neq \boldsymbol{\epsilon}$ is the subsequence of $\lceil \boldsymbol{s} \rceil_{D_\sigma}$ defined by the following induction on the construction of $\sigma : S_\sigma \in \mathcal{PCF}$:

[3]From Definition 4.2.11, it is clear that each state consists of initial or internal moves only.

1. $\varsigma_{D_\sigma}(\lceil \boldsymbol{s} \rceil_{D_\sigma}) \stackrel{\mathrm{df.}}{=} \lceil \boldsymbol{s} \rceil_{D_\sigma}(1)$, i.e., the first move of $\lceil \boldsymbol{s} \rceil_{D_\sigma}$, if $\sigma : S_\sigma \in \mathcal{PCF}$ is PCF-atomic (Definition 4.2.8);

2. $\varsigma_{D_{\Lambda(\sigma)}}(\lceil \boldsymbol{s} \rceil_{D_{\Lambda(\sigma)}}) \stackrel{\mathrm{df.}}{=} \varsigma_{D_\sigma}(\lceil \boldsymbol{s} \rceil_{D_\sigma})$ (up to 'tags') if $\Lambda(\sigma) : A \Rightarrow (B \Rightarrow C) \in \mathcal{PCF}$ is constructed from $\sigma : A \& B \Rightarrow C \in \mathcal{PCF}$;

3. $\varsigma_{D_{\langle \varphi, \psi \rangle}}(\lceil \boldsymbol{s} \rceil_{D_{\langle \varphi, \psi \rangle}}) \stackrel{\mathrm{df.}}{=} \begin{cases} \varsigma_{D_\varphi}(\lceil \boldsymbol{s} \upharpoonright D_\varphi \rceil_{D_\varphi}) & \text{if } \boldsymbol{s} \upharpoonright A \neq \boldsymbol{\epsilon} \\ \varsigma_{D_\psi}(\lceil \boldsymbol{s} \upharpoonright D_\psi \rceil_{D_\psi}) & \text{otherwise} \end{cases}$ (up to 'tags') if $\langle \varphi, \psi \rangle : C \Rightarrow A \& B \in \mathcal{PCF}$ is constructed from $\varphi : C \Rightarrow A, \psi : C \Rightarrow B \in \mathcal{PCF}$;

4. $\varsigma_{D_{\iota^\dagger ; \kappa}}(\lceil \boldsymbol{s} \rceil_{D_{\iota^\dagger ; \kappa}}) \stackrel{\mathrm{df.}}{=} \varsigma_{D_\iota}(\lceil \boldsymbol{s} \upharpoonright D_{\iota^\dagger} \rceil_{D_{\iota^\dagger}}).\varsigma_{D_\kappa}(\lceil \boldsymbol{s} \upharpoonright D_\kappa \rceil_{D_\kappa})$ (up to 'tags') if $\iota^\dagger ; \kappa : A \Rightarrow C \in \mathcal{PCF}$ is constructed from $\iota : A \Rightarrow B, \kappa : B \Rightarrow C \in \mathcal{PCF}$ (n.b., $\lceil \boldsymbol{s} \upharpoonright D_{\iota^\dagger} \rceil_{D_{\iota^\dagger}}$ forms a P-view of D_ι up to 'tags').

The provision *up to 'tags'* will be made precise in Section 4.4, particularly in Definition 4.4.4, but the informal treatment given above should be clear enough and appropriate for now. Observe that states of P-views contain enough information for the case analysis on pairing, and therefore they in fact solve the problem raised above.

We are now ready to prove the main result of the present section:

Lemma 4.2.12 (Last-three-move lemma)**.** *Each dynamic strategy $\phi_\sigma : D_\sigma \in \mathcal{DPCF}$ is representable by a partial function $(\boldsymbol{s}, m_3, m_2, m_1) \mapsto m$ that assigns the next P-move m to the quadruple $(\boldsymbol{s}, m_3, m_2, m_1)$ of the last three moves m_1, m_2 and m_3 of the P-view of a given odd-length position of D_σ and the state $\boldsymbol{s}$ of the P-view.*

Proof. By induction on $\sigma : S_\sigma \in \mathcal{PCF}$. The base case has been already handled in Section 4.2, and the three induction steps are just straightforward. □

4.3 On 'Tags' for Disjoint Union of Sets

In the previous section, we have given our game-semantic formulation of high-level processes of PCF-computation (Definition 4.2.9) and proved that they all satisfy the 'last-three-move property' (Lemma 4.2.12). Based on the property, we shall define low-level computational processes that define 'effectivity' of the high-level ones.

In the present section, we formalize, as a preparation, 'tags' for disjoint union of sets of moves that can be manipulated 'effectively' (Definitions 4.3.5 and 4.3.6). We also restrict each internal O-move occurring in a position of a game to a 'dummy' of the previous occurrence of an internal P-move so that the calculation of next internal O-moves *trivial* (Definition 4.3.9). This restriction is due to the computability-theoretic motivation of the present chapter: We have to make the calculation of next internal O-moves 'effective' for Player because they are conceptually 'invisible' to Opponent.

Let us first formalize 'tags' for exponential ! (Definition 2.2.25), which should be 'effectively' manipulable. Recall that they are usually natural numbers, where note that exponential may be applied in an *iterated* manner, e.g., $!!A$. Hence, a natural idea is to employ a binary representation of finite sequences of natural numbers:

Definition 4.3.1 (Effective tags)**.** An ***effective tag*** is a finite sequence over the two-element set $\Sigma = \{\ell, \hbar\}$, where ℓ and $\hbar$ are arbitrarily fixed, distinct elements.

Notation. We often abbreviate the sequence $\underbrace{\ell\ell\dots\ell}_{i}$ by $\underline{i}$ for each $i \in \mathbb{N}$; it would not be confused with the dynamic strategies $\underline{n} : \mathcal{N}$ $(n \in \mathbb{N})$ in practice.

Definition 4.3.2 (Decoding and encoding)**.** The ***decoding function*** $de : \Sigma^* \to \mathbb{N}^*$ and the ***encoding function*** $en : \mathbb{N}^* \to \Sigma^*$ are given by:

$$de(\boldsymbol{\gamma}) \overset{\text{df.}}{=} (i_1, i_2, \dots, i_k)$$
$$en(j_1, j_2, \dots, j_l) \overset{\text{df.}}{=} \underline{j_1}\hbar\underline{j_2}\hbar\dots\underline{j_{l-1}}\hbar\underline{j_l}$$

for all $\boldsymbol{\gamma} \in \Sigma^*$ and $(j_1, j_2, \dots, j_l) \in \mathbb{N}^*$, where $\boldsymbol{\gamma} = \underline{i_1}\hbar\underline{i_2}\hbar\dots\underline{i_{k-1}}\hbar\underline{i_k}$.

Clearly, the functions $de : \Sigma^* \leftrightarrows \mathbb{N}^* : en$ are mutually inverses (n.b., they both map the empty sequence $\boldsymbol{\epsilon}$ to itself). In fact, each effective tag $\boldsymbol{\gamma} \in \Sigma^*$ is intended to be a binary representation of the finite sequence $de(\boldsymbol{\gamma}) \in \mathbb{N}^*$ of natural numbers.

However, effective tags are not sufficient for our purpose: For iterated exponentials occurring in promotions (Definition 3.3.45) or fixed-point strategies (Definition 4.2.7), we need to *'effectively'* associate a natural number to each pair of natural numbers in an *'effectively'* invertible manner. Of course it is possible as there is a recursive bijection $\mathbb{N} \times \mathbb{N} \xrightarrow{\sim} \mathbb{N}$ whose inverse is recursive too, which is an elementary fact in computability theory [46, 157], but we cannot rely on it for we are aiming at developing an *autonomous* foundation of 'effective computability'.

On the other hand, such a bijection is necessary only for manipulating effective tags, and so we would like to avoid an involved mechanism to achieve it. Then, our solution for this problem is to simply introduce elements to *denote* the bijection:

Definition 4.3.3 (2e-tags)**.** An ***extended effective (2e-) tags*** is an expression $\boldsymbol{e} \in (\Sigma \cup \{\wr, \int\})^*$, where $\wr$ and $\int$ are arbitrarily fixed elements such that $\wr \neq \int$ and $\Sigma \cap \{\wr, \int\} = \emptyset$, generated by the grammar $\boldsymbol{e} \overset{\text{df.}}{\equiv} \boldsymbol{\gamma} \mid \boldsymbol{e}_1\hbar\boldsymbol{e}_2 \mid \wr\boldsymbol{e}\int$, where $\boldsymbol{\gamma} \in \Sigma^*$.

Definition 4.3.4 (Extended decoding)**.** The ***extended decoding function*** $ede : \Sigma^\star \to \mathbb{N}^*$, where $\Sigma^\star$ is the set of all 2e-tags, is recursively defined by:

$$\begin{aligned} ede(\boldsymbol{\gamma}) &\overset{\text{df.}}{=} de(\boldsymbol{\gamma}) \\ ede(\boldsymbol{e}_1 \hbar\, \boldsymbol{e}_2) &\overset{\text{df.}}{=} ede(\boldsymbol{e}_1).ede(\boldsymbol{e}_2) \\ ede(\wr \boldsymbol{e} \int) &\overset{\text{df.}}{=} (\wp \circ ede(\boldsymbol{e})) \end{aligned}$$

where $\wp$ is any recursive bijection $\mathbb{N}^* \overset{\sim}{\to} \mathbb{N}$ fixed throughout the present chapter such that $\wp(i_1, i_2, \dots, i_k) \neq \wp(j_1, j_2, \dots, j_l)$ whenever $k \neq l$ (see, e.g., [46]).

Of course, we lose the bijectivity between Σ^* and $\mathbb{N}^*$ for 2e-tags (since if $ede(\wr \boldsymbol{e} \int) = (i)$, then $ede(\underline{i}) = (i)$ but $\wr \boldsymbol{e} \int \neq \underline{i}$), but in return, we may 'effectively execute' the bijection $\wp : \mathbb{N}^* \overset{\sim}{\to} \mathbb{N}$ by just inserting the elements $\wr, \int$.[4]

Definition 4.3.5 (Outer tags)**.** In each 2e-tag $\boldsymbol{e}$, pairing each occurrence of $\int$ with the most recent yet unpaired occurrence of $\wr$, one component of such a pair is called the ***mate*** of the other. The ***depth*** of an occurrence of $\wr$ in $\boldsymbol{e}$ is the number of previous occurrences of $\wr$ in $\boldsymbol{e}$ whose mate does not occur before that occurrence, and the ***depth*** of an occurrence of $\int$ in $\boldsymbol{e}$ is the depth of its mate in $\boldsymbol{e}$. An ***outer tag*** is an expression $\mathscr{O}(\boldsymbol{e})$ that is obtained from a 2e-tag $\boldsymbol{e}$ by replacing each occurrence of $\wr$ (resp. $\int$) with $\wr. \lhd .\underline{d}. \rhd$ (resp. $\int. \lhd .\underline{d}. \rhd$), where $d \in \mathbb{N}$ is the depth of the occurrence, and $\lhd$ and $\rhd$ are arbitrarily fixed, distinct elements such that $\{\lhd, \rhd\} \cap \{\ell, \hbar, \wr, \int\} = \emptyset$.

Notation. We write $\mathcal{T}$ for the set of all outer tags. We regard the operation $\mathscr{O}$ as the obvious bijection $\Sigma^\star \overset{\sim}{\to} \mathcal{T}$ and define the ***extended decoding function on outer tags*** to be the composition $\mathcal{T} \overset{\mathscr{O}^{-1}}{\to} \Sigma^\star \overset{ede}{\to} \mathbb{N}^*$, which we also write $ede : \mathcal{T} \to \mathbb{N}^*$.

Remark. We have replaced each occurrence of $\wr$ (resp. $\int$) in 2e-tags with $\wr. \lhd .\underline{d}. \rhd$ (resp. $\int. \lhd .\underline{d}. \rhd$) only for the computation of *m-views* (Definition 4.4.3).

On the other hand, a finite number (specifically, just four) of elements suffice as 'tags' for constructions on dynamic games other than exponential:

Definition 4.3.6 (Inner tags)**.** Let $\mathscr{W}$, $\mathscr{E}$, $\mathscr{N}$ and $\mathscr{S}$ be arbitrarily fixed, pairwise distinct elements. A finite sequence $\boldsymbol{s} \in \{\mathscr{W}, \mathscr{E}, \mathscr{N}, \mathscr{S}\}^*$ is called an ***inner tag***.

Convention. A ***tag*** refers to an outer or inner tag.

Using tags, let us focus on dynamic games whose moves are all *tagged elements*:

[4]We employ a bijection $\mathbb{N}^* \overset{\sim}{\to} \mathbb{N}$, not $\mathbb{N} \times \mathbb{N} \overset{\sim}{\to} \mathbb{N}$, for a simple definition of 2e-tags (i.e., so that the pair $\wr, \int$ may be inserted anywhere in a 2e-tag as long as it is in the 'well-bracketing' manner).

Definition 4.3.7 (Inner elements)**.** An ***inner element*** is a finitely nested pair $(\dots((m, t_1), t_2), \dots, t_k)$, usually written $m_{t_1 t_2 \dots t_k}$, such that m is a distinguished element, called the ***substance*** of $m_{t_1 t_2 \dots t_k}$, and $t_1 t_2 \dots t_k$ is an inner tag.

Definition 4.3.8 (Tagged elements)**.** A ***tagged element*** is any pair $(m_{t_1 t_2 \dots t_k}, \boldsymbol{e})$, usually written $[m]_{\boldsymbol{e}}$, of an inner element $m_{t_1 t_2 \dots t_k}$ and an outer tag $\boldsymbol{e}$.

Notation. We often abbreviate an inner element $m_{t_1 t_2 \dots t_k}$ as m when the inner tag $t_1 t_2 \dots t_k$ is not very important.

Definition 4.3.9 (Dynamic games revisited)**.** In the present chapter, a ***dynamic game*** refers to a dynamic game G in the sense defined in Definition 3.3.11 such that:

- Its moves are all *tagged elements*;
- The set $\pi_1(M_G)$ of all the inner elements is *finite*;
- It is equipped with a bijection $\Delta_G : M_G^{\mathsf{PI}} \xrightarrow{\sim} M_G^{\mathsf{OI}}$, called the ***dummy function***, such that for some *finite* partial function δ_G on inner tags if $[m_{\boldsymbol{t}}]_{\boldsymbol{e}} \in M_G^{\mathsf{PI}}$, $[n_{\boldsymbol{u}}]_{\boldsymbol{f}} \in M_G^{\mathsf{OI}}$ and $\Delta_G([m_{\boldsymbol{t}}]_{\boldsymbol{e}}) = [n_{\boldsymbol{u}}]_{\boldsymbol{f}}$, then $m = n$, $\boldsymbol{e} = \boldsymbol{f}$, and $\boldsymbol{u} = \delta_G(\boldsymbol{t})$;
- (D) If $\boldsymbol{s} \in P_G$, $\boldsymbol{s} = \boldsymbol{t}.p.o'.\boldsymbol{u}.p'.o$ (resp. $\boldsymbol{s} = \boldsymbol{t}.o'.\boldsymbol{u}.p'.o$), $o \in M_G^{\mathsf{OI}}$, $o' \in M_G^{\mathsf{OI}}$ (resp. $o' \in M_G^{\mathsf{OE}}$), and $o' = \mathcal{J}_{\boldsymbol{s}}(p')$, then $o = \Delta_G(p')$ and $\mathcal{J}_{\boldsymbol{s}}(o) = p$ (resp. $\mathcal{J}_{\boldsymbol{s}}(o) = p'$)

where M_G^{OI} (resp. M_G^{PI}, M_G^{OE}, M_G^{PE}) is the set of all internal O-moves (resp. internal P-moves, external O-moves, external P-moves) of G.

The set $\pi_1(M_G)$ is required to be finite so that each move is distinguishable (for the computability-theoretic motivation). The dummy function Δ_G is to define the 'dummy' $\Delta_G([m_{\boldsymbol{t}}]_{\boldsymbol{e}}) \in M_G^{\mathsf{OI}}$ of each $[m_{\boldsymbol{t}}]_{\boldsymbol{e}} \in M_G^{\mathsf{PI}}$ such that they differ only in inner tags, and the inner tag of the former is obtainable from that of the latter by a finitary computation δ_G (so that the computation $[m_{\boldsymbol{t}}]_{\boldsymbol{e}} \mapsto \Delta_G([m_{\boldsymbol{t}}]_{\boldsymbol{e}})$ is trivial). Then, the axiom D requires that each internal O-move occurring in a position of G must be the 'dummy' of the last internal P-move, where the involved pointers capture the phenomenon of concatenation of dynamic games.

Remark. It is straightforward to see that the axiom D is stronger than the axiom DP3 (Definition 3.3.11), and the content of Chapter 3 would not change in essence if we replace the latter with the former in Definition 3.3.11. It conceptually makes sense too: The internal part of a play of a dynamic game consists essentially of Player's calculations only. On the other hand, dynamic games as defined in Chapter 3 are simpler and more general than those in the present chapter.

Accordingly, we need to adjust the dynamic games $\mathcal{N}$, $\mathbf{2}$ and T as follows:

Definition 4.3.10 (Lazy natural number game revisited)**.** In the rest of the chapter, the ***lazy natural number game*** refers to the dynamic game $\mathcal{N}$ given by:

- $M_{\mathcal{N}} \stackrel{\mathrm{df.}}{=} \{[\hat{q}], [q], [yes], [no]\}$;
- $\lambda_{\mathcal{N}} : [\hat{q}] \mapsto \mathsf{OQ0}, [q] \mapsto \mathsf{OQ0}, [yes] \mapsto \mathsf{PA0}, [no] \mapsto \mathsf{PA0}$;
- $\vdash_{\mathcal{N}} \stackrel{\mathrm{df.}}{=} \{(\star, [\hat{q}]), ([\hat{q}], [no]), ([\hat{q}], [yes]), ([q], [no]), ([q], [yes]), ([yes], [q])\}$;
- $P_{\mathcal{N}} \stackrel{\mathrm{df.}}{=} \mathsf{Pref}(\{[\hat{q}].([yes].[q])^n.[no] \mid n \in \mathbb{N}\})$, where each non-initial occurrence is justified by the previous occurrence;
- $\boldsymbol{s} \simeq_{\mathcal{N}} \boldsymbol{t} \stackrel{\mathrm{df.}}{\Leftrightarrow} \boldsymbol{s} = \boldsymbol{t}$;
- $\Delta_{\mathcal{N}} \stackrel{\mathrm{df.}}{=} \emptyset$.

Definition 4.3.11 (Boolean game revisited)**.** In the rest of the chapter, the ***boolean game*** refers to the dynamic game $\mathbf{2}$ given by:

- $M_{\mathbf{2}} \stackrel{\mathrm{df.}}{=} \{[\hat{q}], [tt], [\mathit{ff}]\}$;
- $\lambda_{\mathbf{2}} : [\hat{q}] \mapsto \mathsf{OQ0}, [tt] \mapsto \mathsf{PA0}, [\mathit{ff}] \mapsto \mathsf{PA0}$;
- $\vdash_{\mathbf{2}} \stackrel{\mathrm{df.}}{=} \{(\star, [\hat{q}]), ([\hat{q}], [tt]), ([\hat{q}], [\mathit{ff}])\}$;
- $P_{\mathbf{2}} \stackrel{\mathrm{df.}}{=} \mathsf{Pref}(\{[\hat{q}].[tt], [\hat{q}].[\mathit{ff}]\})$, where $[\hat{q}]$ justifies each non-initial occurrence;
- $\boldsymbol{s} \simeq_{\mathbf{2}} \boldsymbol{t} \stackrel{\mathrm{df.}}{\Leftrightarrow} \boldsymbol{s} = \boldsymbol{t}$;
- $\Delta_{\mathbf{2}} \stackrel{\mathrm{df.}}{=} \emptyset$.

Definition 4.3.12 (Terminal game revisited)**.** In the rest of the chapter, the ***terminal game*** refers to the dynamic game $T \stackrel{\mathrm{df.}}{=} (\emptyset, \emptyset, \emptyset, \{\boldsymbol{\epsilon}\}, \{(\boldsymbol{\epsilon}, \boldsymbol{\epsilon})\}, \emptyset)$.

4.3.1 Constructions on Dynamic Games Revisited

Now, let us implement 'tags' for disjoint union of sets of moves for constructions on dynamic games defined in Section 3.3.3. It is straightforward to see that the additional axioms on dynamic games imposed in Definition 4.3.9 are all preserved under the constructions, and therefore we leave the proof to the reader.

Let us begin with *tensor* $\otimes$ (Definition 2.2.19):

Definition 4.3.13 (Tensor of dynamic games revisited)**.** In the rest of the chapter, the ***tensor (product)*** $A \otimes B$ of dynamic games A and B is given by:

- $M_{A \otimes B} \stackrel{\text{df.}}{=} \{[(a, \mathscr{W})]_{\boldsymbol{e}} \mid [a]_{\boldsymbol{e}} \in M_A\} \cup \{[(b, \mathscr{E})]_{\boldsymbol{f}} \mid [b]_{\boldsymbol{f}} \in M_B\}$;
- $\lambda_{A \otimes B}([(m, X)]_{\boldsymbol{e}}) \stackrel{\text{df.}}{=} \begin{cases} \lambda_A([m]_{\boldsymbol{e}}) & \text{if } X = \mathscr{W}; \\ \lambda_B([m]_{\boldsymbol{e}}) & \text{if } X = \mathscr{E}; \end{cases}$
- $\star \vdash_{A \otimes B} [(m, X)]_{\boldsymbol{e}} \stackrel{\text{df.}}{\Leftrightarrow} (X = \mathscr{W} \wedge \star \vdash_A [m]_{\boldsymbol{e}}) \vee (X = \mathscr{E} \wedge \star \vdash_B [m]_{\boldsymbol{e}})$;
- $[(m, X)]_{\boldsymbol{e}} \vdash_{A \otimes B} [(n, Y)]_{\boldsymbol{f}} \stackrel{\text{df.}}{\Leftrightarrow} (X = \mathscr{W} = Y \wedge [m]_{\boldsymbol{e}} \vdash_A [n]_{\boldsymbol{f}}) \vee (X = \mathscr{E} = Y \wedge [m]_{\boldsymbol{e}} \vdash_B [n]_{\boldsymbol{f}})$;
- $P_{A \otimes B} \stackrel{\text{df.}}{=} \{\boldsymbol{s} \in \mathscr{L}_{A \otimes B} \mid \boldsymbol{s} \upharpoonright \mathscr{W} \in P_A, \boldsymbol{s} \upharpoonright \mathscr{E} \in P_B\}$, where $\boldsymbol{s} \upharpoonright X$ (with $X \in \{\mathscr{W}, \mathscr{E}\}$) is the j-subsequence of $\boldsymbol{s}$ that consists of moves of the form $[(m, X)]_{\boldsymbol{e}}$ changed into $[m]_{\boldsymbol{e}}$;
- $\boldsymbol{s} \simeq_{A \otimes B} \boldsymbol{t} \stackrel{\text{df.}}{\Leftrightarrow} (\pi_2 \circ \pi_1)^*(\boldsymbol{s}) = (\pi_2 \circ \pi_1)^*(\boldsymbol{t}) \wedge \boldsymbol{s} \upharpoonright \mathscr{W} \simeq_A \boldsymbol{t} \upharpoonright \mathscr{W} \wedge \boldsymbol{s} \upharpoonright \mathscr{E} \simeq_B \boldsymbol{t} \upharpoonright \mathscr{E}$;
- $\Delta_{A \otimes B}([(m, X)]_{\boldsymbol{e}}) \stackrel{\text{df.}}{=} \begin{cases} [(m', \mathscr{W})]_{\boldsymbol{e}} & \text{if } X = \mathscr{W}, \text{ where } \Delta_A([m]_{\boldsymbol{e}}) = [m']_{\boldsymbol{e}}; \\ [(m'', \mathscr{E})]_{\boldsymbol{e}} & \text{if } X = \mathscr{E}, \text{ where } \Delta_B([m]_{\boldsymbol{e}}) = [m'']_{\boldsymbol{e}}. \end{cases}$

It is easy to see that this is just a possible formalization of Definition 2.2.19 (n.b., the same remark holds for other constructions given below, which we shall omit).

Example 4.3.14. Typical plays of the tensor $\mathcal{N} \otimes \mathcal{N}$ are described in Figure 4.7.

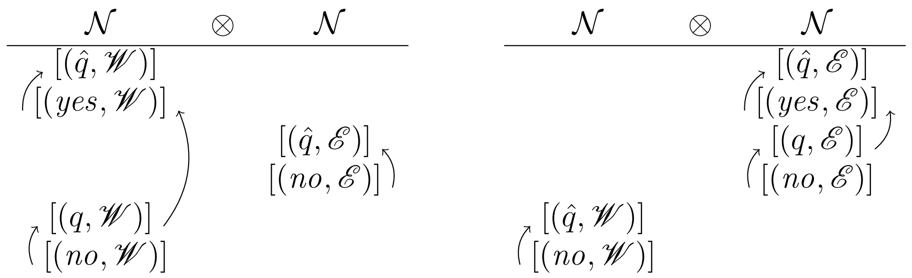

$\mathcal{N} \otimes \mathcal{N}$
$[(\hat{q}, \mathscr{W})]$
$[(yes, \mathscr{W})]$
$[(\hat{q}, \mathscr{E})]$
$[(no, \mathscr{E})]$
$[(q, \mathscr{W})]$
$[(no, \mathscr{W})]$
$\mathcal{N} \otimes \mathcal{N}$
$[(\hat{q}, \mathscr{E})]$
$[(yes, \mathscr{E})]$
$[(q, \mathscr{E})]$
$[(no, \mathscr{E})]$
$[(\hat{q}, \mathscr{W})]$
$[(no, \mathscr{W})]$

Figure 4.7: Typical plays of the tensor $\mathcal{N} \otimes \mathcal{N}$.

We proceed to implement 'tags' for *linear implication* $\multimap$ (Definition 2.2.21):

Definition 4.3.15 (Linear implication between dynamic games revisited)**.** In the rest of the chapter, the ***linear implication*** $A \multimap B$ from a dynamic game A to another B is given by:

- $M_{A \multimap B} \stackrel{\mathrm{df.}}{=} \{[(a, \mathscr{W})]_{\boldsymbol{e}} \mid [a]_{\boldsymbol{e}} \in M_{\mathcal{H}^\omega(A)}\} \cup \{[(b, \mathscr{E})]_{\boldsymbol{f}} \mid [b]_{\boldsymbol{f}} \in M_B\}$;
- $\lambda_{A \multimap B}([(m, X)]_{\boldsymbol{e}}) \stackrel{\mathrm{df.}}{=} \begin{cases} \overline{\lambda_{\mathcal{H}^\omega(A)}}([m]_{\boldsymbol{e}}) & \text{if } X = \mathscr{W}; \\ \lambda_B([m]_{\boldsymbol{e}}) & \text{if } X = \mathscr{E}; \end{cases}$
- $\star \vdash_{A \multimap B} [(m, X)]_{\boldsymbol{e}} \stackrel{\mathrm{df.}}{\Leftrightarrow} X = \mathscr{E} \wedge \star \vdash_B [m]_{\boldsymbol{e}}$;
- $[(m, X)]_{\boldsymbol{e}} \vdash_{A \multimap B} [(n, Y)]_{\boldsymbol{f}} \stackrel{\mathrm{df.}}{\Leftrightarrow} \begin{cases} (X = \mathscr{W} = Y \wedge [m]_{\boldsymbol{e}} \vdash_{\mathcal{H}^\omega(A)} [n]_{\boldsymbol{f}}) \\ \vee (X = \mathscr{E} = Y \wedge [m]_{\boldsymbol{e}} \vdash_B [n]_{\boldsymbol{f}}) \\ \vee (X = \mathscr{E} \wedge Y = \mathscr{W} \wedge \star \vdash_B [m]_{\boldsymbol{e}} \wedge \star \vdash_{\mathcal{H}^\omega(A)} [n]_{\boldsymbol{f}}); \end{cases}$
- $P_{A \multimap B} \stackrel{\mathrm{df.}}{=} \{\boldsymbol{s} \in \mathscr{L}_{\mathcal{H}^\omega(A) \multimap B} \mid \boldsymbol{s} \upharpoonright \mathscr{W} \in P_{\mathcal{H}^\omega(A)}, \boldsymbol{s} \upharpoonright \mathscr{E} \in P_B\}$;
- $\boldsymbol{s} \simeq_{A \multimap B} \boldsymbol{t} \stackrel{\mathrm{df.}}{\Leftrightarrow} (\pi_2 \circ \pi_1)^*(\boldsymbol{s}) = (\pi_2 \circ \pi_1)^*(\boldsymbol{t}) \wedge \boldsymbol{s} \upharpoonright \mathscr{W} \simeq_{\mathcal{H}^\omega(A)} \boldsymbol{t} \upharpoonright \mathscr{W} \wedge \boldsymbol{s} \upharpoonright \mathscr{E} \simeq_B \boldsymbol{t} \upharpoonright \mathscr{E}$;
- $\Delta_{A \multimap B}([(b, \mathscr{E})]_{\boldsymbol{f}}) \stackrel{\mathrm{df.}}{=} [(b', \mathscr{E})]_{\boldsymbol{f}}$, where $\Delta_B([b]_{\boldsymbol{f}}) = [b']_{\boldsymbol{f}}$.

Example 4.3.16. Some typical plays of the linear implication $\mathbf{2} \multimap \mathbf{2}$ are described in Figure 4.8.

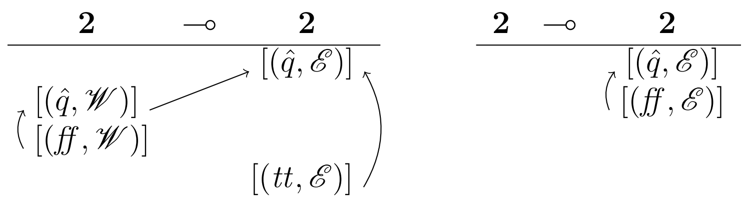

$\mathbf{2}$
$\multimap$
$\mathbf{2}$
$[(\hat{q}, \mathscr{E})]$
$[(\hat{q}, \mathscr{W})]$
$[(\mathit{ff}, \mathscr{W})]$
$[(tt, \mathscr{E})]$
$\mathbf{2}$
$\multimap$
$\mathbf{2}$
$[(\hat{q}, \mathscr{E})]$
$[(\mathit{ff}, \mathscr{E})]$

Figure 4.8: Typical plays of the linear implication $\mathbf{2} \multimap \mathbf{2}$.

Similarly, let us formalize ‘tags’ for *product* & (Definition 2.2.23):

Definition 4.3.17 (Product of dynamic games revisited)**.** In the rest of the chapter, the ***product*** $A \& B$ of dynamic games A and B is given by:

- $M_{A \& B} \stackrel{\mathrm{df.}}{=} \{[(a, \mathscr{W})]_{\boldsymbol{e}} \mid [a]_{\boldsymbol{e}} \in M_A\} \cup \{[(b, \mathscr{E})]_{\boldsymbol{f}} \mid [b]_{\boldsymbol{f}} \in M_B\}$;
- $\lambda_{A \& B}([(m, X)]_{\boldsymbol{e}}) \stackrel{\mathrm{df.}}{=} \begin{cases} \lambda_A([m]_{\boldsymbol{e}}) & \text{if } X = \mathscr{W}; \\ \lambda_B([m]_{\boldsymbol{e}}) & \text{if } X = \mathscr{E}; \end{cases}$
- $\star \vdash_{A \& B} [(m, X)]_{\boldsymbol{e}} \stackrel{\mathrm{df.}}{\Leftrightarrow} (X = \mathscr{W} \wedge \star \vdash_A [m]_{\boldsymbol{e}}) \vee (X = \mathscr{E} \wedge \star \vdash_B [m]_{\boldsymbol{e}})$;
- $[(m, X)]_{\boldsymbol{e}} \vdash_{A \& B} [(n, Y)]_{\boldsymbol{f}} \stackrel{\mathrm{df.}}{\Leftrightarrow} (X = \mathscr{W} = Y \wedge [m]_{\boldsymbol{e}} \vdash_A [n]_{\boldsymbol{f}}) \vee (X = \mathscr{E} = Y \wedge [m]_{\boldsymbol{e}} \vdash_B [n]_{\boldsymbol{f}})$;

- $P_{A\&B} \overset{\text{df.}}{=} \{ \boldsymbol{s} \in \mathscr{L}_{A\&B} \mid (\boldsymbol{s} \upharpoonright \mathscr{W} \in P_A \wedge \boldsymbol{s} \upharpoonright \mathscr{E} = \boldsymbol{\epsilon}) \vee (\boldsymbol{s} \upharpoonright \mathscr{W} = \boldsymbol{\epsilon} \wedge \boldsymbol{s} \upharpoonright \mathscr{E} \in P_B) \}$;
- $\boldsymbol{s} \simeq_{A\&B} \boldsymbol{t} \overset{\text{df.}}{\Leftrightarrow} (\pi_2 \circ \pi_1)^*(\boldsymbol{s}) = (\pi_2 \circ \pi_1)^*(\boldsymbol{t}) \wedge \boldsymbol{s} \upharpoonright \mathscr{W} \simeq_A \boldsymbol{t} \upharpoonright \mathscr{W} \wedge \boldsymbol{s} \upharpoonright \mathscr{E} \simeq_B \boldsymbol{t} \upharpoonright \mathscr{E}$;
- $\Delta_{A\&B}([(m, X)]_{\boldsymbol{e}}) \overset{\text{df.}}{=} \begin{cases} [(m', \mathscr{W})]_{\boldsymbol{e}} & \text{if } X = \mathscr{W}\text{, where } \Delta_A([m]_{\boldsymbol{e}}) = [m']_{\boldsymbol{e}}; \\ [(m'', \mathscr{E})]_{\boldsymbol{e}} & \text{if } X = \mathscr{E}\text{, where } \Delta_B([m]_{\boldsymbol{e}}) = [m'']_{\boldsymbol{e}}. \end{cases}$

In contrast to product, our formalization of *pairing* (Definition 3.3.19) is slightly involved. Therefore, let us first sketch how we implement 'tags' for pairing. For a pairing $\langle L, R \rangle$ of dynamic games L and R such that $\mathcal{H}^\omega(L) \trianglelefteq C \multimap A$ and $\mathcal{H}^\omega(R) \trianglelefteq C \multimap B$ for some normalized dynamic games A, B and C, we implement 'tags' for the disjoint union $M_{\langle L,R \rangle} \overset{\text{df.}}{=} M_C + (M_L \setminus M_C) + (M_R \setminus M_C)$ (Definition 3.3.19) by:

- Adding no tags for external moves of the form $[(c, \mathscr{W})]_{\boldsymbol{e}}$ of L or R, where $[c]_{\boldsymbol{e}}$ must be a move of C by the definition of tags for $\multimap$ (Definition 4.3.15);
- Changing external moves of the form $[(a, \mathscr{E})]_{\boldsymbol{f}}$ of L, where $[a]_{\boldsymbol{f}}$ must be a move of A, into $[((a, \mathscr{W}), \mathscr{E})]_{\boldsymbol{f}}$;
- Changing external moves of the form $[(b, \mathscr{E})]_{\boldsymbol{g}}$ of R, where $[b]_{\boldsymbol{g}}$ must be a move of B, into $[((b, \mathscr{E}), \mathscr{E})]_{\boldsymbol{g}}$;
- Changing internal moves $[l]_{\boldsymbol{h}}$ of L into $[(l, \mathscr{S})]_{\boldsymbol{h}}$;
- Changing internal moves $[r]_{\boldsymbol{k}}$ of R into $[(r, \mathscr{N})]_{\boldsymbol{k}}$

where note that we have distinguished the cases by pattern matching on the last digit of the inner tag and the internal/external (I/E-) parity of each move. These tags are of course not canonical at all, but they would certainly achieve the required subgame relation $\mathcal{H}^\omega(\langle L, R \rangle) \trianglelefteq C \multimap A\&B$ (see Definition 3.3.19 below).

Then, we formalize the labeling function, the enabling relation and the dummy function of $\langle L, R \rangle$ by the obvious pattern matching on inner tags; positions of $\langle L, R \rangle$ and the identification of them are formalized in the obvious manner.

However, the enabling relation of $\langle L, R \rangle$ is rather involved; thus, for convenience, we define the *peeling* $\mathit{peel}_{\langle L,R \rangle}(m) \in M_L \cup M_R$ of each move $m \in M_{\langle L,R \rangle}$ such that changing the inner tag of $\mathit{peel}_{\langle L,R \rangle}(m)$ as defined above produces m, and also the *attribute* $\mathit{att}_{\langle L,R \rangle}(m) \in \{L, R, C\}$ of m by:

$$\mathit{att}_{\langle L,R \rangle}(m) \overset{\text{df.}}{=} \begin{cases} L & \text{if } \mathit{peel}_{\langle L,R \rangle}(m) \in M_L \setminus M_C; \\ R & \text{if } \mathit{peel}_{\langle L,R \rangle}(m) \in M_R \setminus M_C; \\ C & \text{otherwise (i.e., if } \mathit{peel}_{\langle L,R \rangle}(m) \in M_C). \end{cases}$$

The enabling relation $m \vdash_{\langle L,R \rangle} n$ is then easily defined as the conjunction of:

- $att_{\langle L,R \rangle}(m) = att_{\langle L,R \rangle}(n) \vee att_{\langle L,R \rangle}(m) = C \vee att_{\langle L,R \rangle}(n) = C$;
- $peel_{\langle L,R \rangle}(m) \vdash_L peel_{\langle L,R \rangle}(n) \vee peel_{\langle L,R \rangle}(m) \vdash_R peel_{\langle L,R \rangle}(n)$.

Formally, we implement 'tags' for pairing of dynamic games as follows:

Notation. Given a dynamic game G, we write M_G^{Ext} (resp. M_G^{Int}, M_G^{Init}) for the set of all external (resp. internal, initial) moves of G.

Definition 4.3.18 (Pairing of dynamic games revisited)**.** In the rest of the chapter, the ***pairing*** $\langle L, R \rangle$ of dynamic games L and R such that $\mathcal{H}^\omega(L) \trianglelefteq C \multimap A$ and $\mathcal{H}^\omega(R) \trianglelefteq C \multimap B$ for some normalized dynamic games A, B and C is given by:

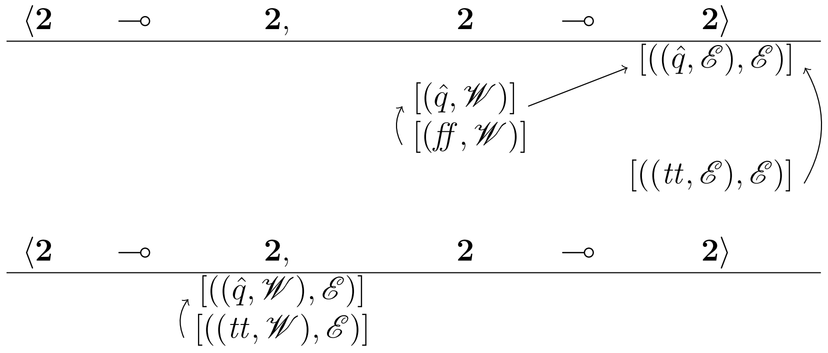
- $M_{\langle L,R \rangle} \overset{\text{df.}}{=} \{[(c, \mathscr{W})]_{\boldsymbol{e}} \mid [(c, \mathscr{W})]_{\boldsymbol{e}} \in M_L^{\mathsf{Ext}} \cup M_R^{\mathsf{Ext}}, [c]_{\boldsymbol{e}} \in M_C\}$
$\cup \{[((a, \mathscr{W}), \mathscr{E})]_{\boldsymbol{f}} \mid [(a, \mathscr{E})]_{\boldsymbol{f}} \in M_L^{\mathsf{Ext}}, [a]_{\boldsymbol{f}} \in M_A\}$
$\cup \{[((b, \mathscr{E}), \mathscr{E})]_{\boldsymbol{g}} \mid [(b, \mathscr{E})]_{\boldsymbol{g}} \in M_R^{\mathsf{Ext}}, [b]_{\boldsymbol{g}} \in M_B\}$
$\cup \{[(l, \mathscr{S})]_{\boldsymbol{h}} \mid [l]_{\boldsymbol{h}} \in M_L^{\mathsf{Int}}\} \cup \{[(r, \mathscr{N})]_{\boldsymbol{k}} \mid [r]_{\boldsymbol{k}} \in M_R^{\mathsf{Int}}\}$;
- $\lambda_{\langle L,R \rangle}([(m, X)]_{\boldsymbol{e}}) \overset{\text{df.}}{=} \begin{cases} \overline{\lambda_C}([m]_{\boldsymbol{e}}) & \text{if } X = \mathscr{W}; \\ \lambda_A([a]_{\boldsymbol{e}}) & \text{if } X = \mathscr{E} \text{ and } m \text{ is of the form } (a, \mathscr{W}); \\ \lambda_B([b]_{\boldsymbol{e}}) & \text{if } X = \mathscr{E} \text{ and } m \text{ is of the form } (b, \mathscr{E}); \\ \lambda_L([m]_{\boldsymbol{e}}) & \text{if } X = \mathscr{S}; \\ \lambda_R([m]_{\boldsymbol{e}}) & \text{if } X = \mathscr{N}; \end{cases}$
- $\star \vdash_{\langle L,R \rangle} [(m, X)]_{\boldsymbol{e}} \overset{\text{df.}}{\Leftrightarrow} X = \mathscr{E} \wedge (\exists [a]_{\boldsymbol{e}} \in M_A^{\mathsf{Init}}.\, m = (a, \mathscr{W}) \vee \exists [b]_{\boldsymbol{e}} \in M_B^{\mathsf{Init}}.\, m = (b, \mathscr{E}))$;
- $[(m, X)]_{\boldsymbol{e}} \vdash_{\langle L,R \rangle} [(n, Y)]_{\boldsymbol{f}} \overset{\text{df.}}{\Leftrightarrow} (att_{\langle L,R \rangle}([(m, X)]_{\boldsymbol{e}}) = att_{\langle L,R \rangle}([(n, Y)]_{\boldsymbol{f}}) \vee att_{\langle L,R \rangle}([(m, X)]_{\boldsymbol{e}}) = C \vee att_{\langle L,R \rangle}([(n, Y)]_{\boldsymbol{f}}) = C) \wedge (peel_{\langle L,R \rangle}([(m, X)]_{\boldsymbol{e}}) \vdash_L peel_{\langle L,R \rangle}([(n, Y)]_{\boldsymbol{f}}) \vee peel_{\langle L,R \rangle}([(m, X)]_{\boldsymbol{e}}) \vdash_R peel_{\langle L,R \rangle}([(n, Y)]_{\boldsymbol{f}}))$, where the map $att_{\langle L,R \rangle} : M_{\langle L,R \rangle} \to \{L, R, C\}$ is defined by $[(c, \mathscr{W})]_{\boldsymbol{e}} \mapsto C$, $[((a, \mathscr{W}), \mathscr{E})]_{\boldsymbol{f}} \mapsto L$, $[((b, \mathscr{E}), \mathscr{E})]_{\boldsymbol{g}} \mapsto R$, $[(l, \mathscr{S})]_{\boldsymbol{h}} \mapsto L$, $[(r, \mathscr{N})]_{\boldsymbol{k}} \mapsto R$, and the map $peel_{\langle L,R \rangle} : M_{\langle L,R \rangle} \to M_L \cup M_R$ by $[(c, \mathscr{W})]_{\boldsymbol{e}} \mapsto [(c, \mathscr{W})]_{\boldsymbol{e}}$, $[((a, \mathscr{W}), \mathscr{E})]_{\boldsymbol{f}} \mapsto [(a, \mathscr{E})]_{\boldsymbol{f}}$, $[((b, \mathscr{E}), \mathscr{E})]_{\boldsymbol{g}} \mapsto [(b, \mathscr{E})]_{\boldsymbol{g}}$, $[(l, \mathscr{S})]_{\boldsymbol{h}} \mapsto [l]_{\boldsymbol{h}}$, $[(r, \mathscr{N})]_{\boldsymbol{k}} \mapsto [r]_{\boldsymbol{k}}$;
- $P_{\langle L,R \rangle} \overset{\text{df.}}{=} \{\boldsymbol{s} \in \mathscr{L}_{\langle L,R \rangle} \mid (\boldsymbol{s} \upharpoonright L \in P_L, \boldsymbol{s} \upharpoonright B = \boldsymbol{\epsilon}) \vee (\boldsymbol{s} \upharpoonright R \in P_R, \boldsymbol{s} \upharpoonright A = \boldsymbol{\epsilon})\}$, where $\boldsymbol{s} \upharpoonright L$ (resp. $\boldsymbol{s} \upharpoonright R$) is the j-subsequence of $\boldsymbol{s}$ that consists of moves x such that $peel_{\langle L,R \rangle}(x) \in M_L$ (resp. $peel_{\langle L,R \rangle}(x) \in M_R$) changed into $peel_{\langle L,R \rangle}(x)$, and $\boldsymbol{s} \upharpoonright B$ (resp. $\boldsymbol{s} \upharpoonright A$) is the j-subsequence of $\boldsymbol{s}$ that consists of moves of the form $[((b, \mathscr{E}), \mathscr{E})]_{\boldsymbol{g}}$ with $[b]_{\boldsymbol{g}} \in M_B$ (resp. $[((a, \mathscr{W}), \mathscr{E})]_{\boldsymbol{f}}$ with $[a]_{\boldsymbol{f}} \in M_A$);

- $\boldsymbol{s} \simeq_{L\&_C R} \boldsymbol{t} \overset{\mathrm{df.}}{\Leftrightarrow} (\boldsymbol{s} \upharpoonright A = \boldsymbol{\epsilon} \Leftrightarrow \boldsymbol{t} \upharpoonright A = \boldsymbol{\epsilon}) \wedge \boldsymbol{s} \upharpoonright L \simeq_L \boldsymbol{t} \upharpoonright L \vee \boldsymbol{s} \upharpoonright R \simeq_R \boldsymbol{t} \upharpoonright R$;

- $\Delta_{\langle L,R \rangle}([(m, X)]_{\boldsymbol{e}}) \overset{\mathrm{df.}}{=} \begin{cases} [(l', \mathscr{S})]_{\boldsymbol{e}} & \text{if } X = \mathscr{S}, \text{ where } \Delta_L([l]_{\boldsymbol{e}}) = [l']_{\boldsymbol{e}}; \\ [(r', \mathscr{N})]_{\boldsymbol{e}} & \text{if } X = \mathscr{N}, \text{ where } \Delta_R([r]_{\boldsymbol{e}}) = [r']_{\boldsymbol{e}}. \end{cases}$

It is easy to see that $\mathcal{H}^\omega(\langle L, R \rangle) \trianglelefteqslant C \multimap A\&B$ if $\mathcal{H}^\omega(L) \trianglelefteqslant C \multimap A$ and $\mathcal{H}^\omega(R) \trianglelefteqslant C \multimap B$, and in particular $\langle C \multimap A, C \multimap B \rangle = C \multimap A\&B$, for any normalized dynamic games A, B and C.

Example 4.3.19. Some typical plays of the pairing $\langle \mathbf{2} \multimap \mathbf{2}, \mathbf{2} \multimap \mathbf{2} \rangle$ are described in Figure 4.9.



$\langle \mathbf{2} \multimap \mathbf{2}, \mathbf{2} \multimap \mathbf{2} \rangle$
$[((\hat{q}, \mathscr{E}), \mathscr{E})]$
$[(\hat{q}, \mathscr{W})]$
$[(ff, \mathscr{W})]$
$[((tt, \mathscr{E}), \mathscr{E})]$
$\langle \mathbf{2} \multimap \mathbf{2}, \mathbf{2} \multimap \mathbf{2} \rangle$
$[((\hat{q}, \mathscr{W}), \mathscr{E})]$
$[((tt, \mathscr{W}), \mathscr{E})]$

Figure 4.9: Typical plays of the pairing $\langle \mathbf{2} \multimap \mathbf{2}, \mathbf{2} \multimap \mathbf{2} \rangle$.

Next, we implement 'tags' for *exponential* ! (Definition 2.2.25), for which we have introduced outer tags (Definition 4.3.8):

Notation. We often abbreviate expressions $\wr . \lhd .\underline{d}.\rhd$ and $\int . \lhd .\underline{d}.\rhd$, where $d \in \mathbb{N}$, as $\wr^{\underline{d}}$ and $\int^{\underline{d}}$, respectively. Given $\boldsymbol{e} \in \mathcal{T}$, we write $\boldsymbol{e}^+$ for the expression obtained from $\boldsymbol{e}$ by replacing each occurrence of $\wr^{\underline{d}}$ (resp. $\int^{\underline{d}}$) with that of $\wr^{\underline{d+1}}$ (resp. $\int^{\underline{d+1}}$).

Definition 4.3.20 (Exponential of dynamic games revisited)**.** In the rest of the chapter, the ***exponential*** $!A$ of a dynamic game A is given by:

- $M_{!A} \overset{\mathrm{df.}}{=} \{ [m]_{\wr^{\underline{0}} \boldsymbol{f} + \int^{\underline{0}} \hbar \boldsymbol{e}} \mid [m]_{\boldsymbol{e}} \in M_A, \boldsymbol{f} \in \mathcal{T} \}$;

- $\lambda_{!A}([m]_{\wr^{\underline{0}} \boldsymbol{f} + \int^{\underline{0}} \hbar \boldsymbol{e}}) \overset{\mathrm{df.}}{=} \lambda_A([m]_{\boldsymbol{e}})$;

- $\star \vdash_{!A} [m]_{\wr^{\underline{0}} \boldsymbol{f} + \int^{\underline{0}} \hbar \boldsymbol{e}} \overset{\mathrm{df.}}{\Leftrightarrow} \star \vdash_A [m]_{\boldsymbol{e}}$;

- $[m]_{\wr^{\underline{0}} \boldsymbol{f} + \int^{\underline{0}} \hbar \boldsymbol{e}} \vdash_{!A} [m']_{\wr^{\underline{0}} \boldsymbol{f'} + \int^{\underline{0}} \hbar \boldsymbol{e'}} \overset{\mathrm{df.}}{\Leftrightarrow} \boldsymbol{f} = \boldsymbol{f'} \wedge [m]_{\boldsymbol{e}} \vdash_A [m']_{\boldsymbol{e'}}$;

- $P_{!A} \overset{\mathrm{df.}}{=} \{ \boldsymbol{s} \in \mathscr{L}_{!A} \mid \forall \boldsymbol{f} \in \mathcal{T}.\, \boldsymbol{s} \upharpoonright \boldsymbol{f} \in P_A \wedge (\boldsymbol{s} \upharpoonright \boldsymbol{f} \neq \boldsymbol{\epsilon} \Rightarrow \forall \boldsymbol{g} \in \mathcal{T}.\, \boldsymbol{s} \upharpoonright \boldsymbol{g} \neq \boldsymbol{\epsilon} \Rightarrow \mathit{ede}(\boldsymbol{f}) \neq \mathit{ede}(\boldsymbol{g})) \}$, where $\boldsymbol{s} \upharpoonright \boldsymbol{f}$ is the j-subsequence of $\boldsymbol{s}$ that consists of moves of the form $[m]_{\wr \underline{0} \boldsymbol{f} + \int \underline{0} \hbar \boldsymbol{e}}$ changed into $[m]_{\boldsymbol{e}}$;

- $\boldsymbol{s} \simeq_{!A} \boldsymbol{t} \overset{\mathrm{df.}}{\Leftrightarrow} \exists \varphi \in \mathcal{P}(\mathbb{N}).\, (\pi_1 \circ \mathit{ede} \circ \pi_2)^*(\boldsymbol{s}) = (\varphi \circ \pi_1 \circ \mathit{ede} \circ \pi_2)^*(\boldsymbol{t}) \wedge \forall i \in \mathbb{N}.\, \boldsymbol{s} \upharpoonright \varphi(i) \simeq_A \boldsymbol{t} \upharpoonright i$, where $\boldsymbol{s} \upharpoonright j$ denotes $\boldsymbol{s} \upharpoonright \boldsymbol{f}$ such that $\mathit{ede}(\boldsymbol{f}) = (j)$ for each $j \in \mathbb{N}$;

- $\Delta_{!A}([m]_{\wr \underline{0} \boldsymbol{f} + \int \underline{0} \hbar \boldsymbol{e}}) \overset{\mathrm{df.}}{=} [m']_{\wr \underline{0} \boldsymbol{f} + \int \underline{0} \hbar \boldsymbol{e}}$, where $\Delta_A([m]_{\boldsymbol{e}}) = [m']_{\boldsymbol{e}}$.

Thus, this is a slight modification of Definition 2.2.25 which generalizes moves $[m]_{\underline{i} \hbar \boldsymbol{e}}$ to $[m]_{\wr \underline{0} \boldsymbol{f} + \int \underline{0} \hbar \boldsymbol{e}}$, where $[m]_{\boldsymbol{e}} \in M_A$, $i \in \mathbb{N}$ and $\boldsymbol{f} \in \mathcal{T}$. Note that an outer tag $\boldsymbol{f}$ that represents each $i \in \mathbb{N}$, i.e., $\mathit{ede}(\boldsymbol{f}) = (i)$, is unique in each position $\boldsymbol{s} \in P_{!A}$.

Example 4.3.21. Typical plays of the exponential $!\mathbf{2}$ are depicted in Figure 4.10.

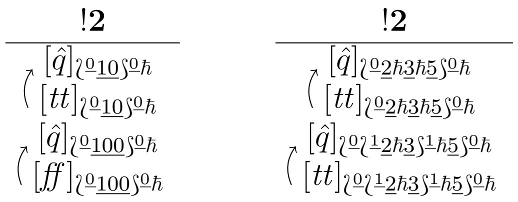

$!\mathbf{2}$
$[\hat{q}]_{\wr \underline{0} \underline{10} \int \underline{0} \hbar}$
$[tt]_{\wr \underline{0} \underline{10} \int \underline{0} \hbar}$
$[\hat{q}]_{\wr \underline{0} \underline{100} \int \underline{0} \hbar}$
$[\mathit{ff}]_{\wr \underline{0} \underline{100} \int \underline{0} \hbar}$
$!\mathbf{2}$
$[\hat{q}]_{\wr \underline{0} \underline{2} \hbar \underline{3} \hbar \underline{5} \int \underline{0} \hbar}$
$[tt]_{\wr \underline{0} \underline{2} \hbar \underline{3} \hbar \underline{5} \int \underline{0} \hbar}$
$[\hat{q}]_{\wr \underline{0} \wr \underline{1} \underline{2} \hbar \underline{3} \int \underline{1} \hbar \underline{5} \int \underline{0} \hbar}$
$[tt]_{\wr \underline{0} \wr \underline{1} \underline{2} \hbar \underline{3} \int \underline{1} \hbar \underline{5} \int \underline{0} \hbar}$

Figure 4.10: Typical plays of the exponential $!\mathbf{2}$.

Next, we formalize *promotion* $(\_)^\dagger$ on dynamic games (Definition 3.3.21). Given a dynamic game G such that $\mathcal{H}^\omega(G) \trianglelefteq\, !A \multimap B$ for some normalized dynamic games A and B, our idea is to formalize 'tags' on moves of $G^\dagger$ as follows:

- We duplicate moves of G coming from $!A$, i.e., ones of the form $[(a, \mathscr{W})]_{\wr \underline{0} \boldsymbol{f} + \int \underline{0} \hbar \boldsymbol{e}}$, as $[(a, \mathscr{W})]_{\wr \underline{0} \wr \underline{1} \boldsymbol{g} {+}{+} \int \underline{1} \hbar \wr \underline{1} \boldsymbol{f} {+}{+} \int \underline{1} \int \underline{0} \hbar \boldsymbol{e}}$ for each $\boldsymbol{g} \in \mathcal{T}$;

- We duplicate moves of G coming from B, i.e., ones of the form $[(b, \mathscr{E})]_{\boldsymbol{e}}$, as $[(b, \mathscr{E})]_{\wr \underline{0} \boldsymbol{g} + \int \underline{0} \hbar \boldsymbol{e}}$ for each $\boldsymbol{g} \in \mathcal{T}$;

- We duplicate internal moves $[m]_{\boldsymbol{e}}$ of G as $[(m, \mathscr{S})]_{\wr \underline{0} \boldsymbol{g} + \int \underline{0} \hbar \boldsymbol{e}}$ for each $\boldsymbol{g} \in \mathcal{T}$.

where note again that this way of formalizing 'tags' is far from canonical, but it achieves the required subgame relation $\mathcal{H}^\omega(G^\dagger) \trianglelefteq\, !A \multimap\, !B$ (see Definition 4.3.22).

Then, the labeling function, the enabling relation and the dummy function of $G^\dagger$ are again defined by pattern matching on inner tags in the obvious manner. Also, positions of $G^\dagger$ and the identification of them are given by a straightforward generalization of those of exponential defined in Definition 4.3.20.

Formally, employing peeling and attributes just for convenience (similarly to the case of pairing), we redefine promotion as follows:

Definition 4.3.22 (Promotion of dynamic games revisited)**.** In the rest of the chapter, given a dynamic game G such that $\mathcal{H}^{\omega}(G) \trianglelefteq \, !A \multimap B$ for some normalized dynamic games A and B, the ***promotion*** $G^{\dagger}$ of G is given by:

- $M_{G^{\dagger}} \overset{\mathrm{df.}}{=} \{[(a, \mathscr{W})]_{\wr^{\underline{0}}\wr^{\underline{1}}\boldsymbol{g}^{++}\int^{\underline{1}}\hbar\wr^{\underline{1}}\boldsymbol{f}^{++}\int^{\underline{1}}\int^{\underline{0}}\hbar\boldsymbol{e}} \mid [(a, \mathscr{W})]_{\wr^{\underline{0}}\boldsymbol{f}^{+}\int^{\underline{0}}\hbar\boldsymbol{e}} \in M_G^{\mathsf{Ext}}, \boldsymbol{g} \in \mathcal{T}\}$
  $\cup \{[(b, \mathscr{E})]_{\wr^{\underline{0}}\boldsymbol{g}^{+}\int^{\underline{0}}\hbar\boldsymbol{e}} \mid [(b, \mathscr{E})]_{\boldsymbol{e}} \in M_G^{\mathsf{Ext}}, \boldsymbol{g} \in \mathcal{T}\}$
  $\cup \{[(m, \mathscr{S})]_{\wr^{\underline{0}}\boldsymbol{g}^{+}\int^{\underline{0}}\hbar\boldsymbol{e}} \mid [m]_{\boldsymbol{e}} \in M_G^{\mathsf{Int}}, \boldsymbol{g} \in \mathcal{T}\}$;

- $\lambda_{G^{\dagger}} \overset{\mathrm{df.}}{=} \lambda_G \circ \mathit{peel}_{G^{\dagger}}$, where $\mathit{peel}_{G^{\dagger}}$ is the function $M_{G^{\dagger}} \to M_G$ that maps:

$$\begin{aligned}
[(a, \mathscr{W})]_{\wr^{\underline{0}}\wr^{\underline{1}}\boldsymbol{g}^{++}\int^{\underline{1}}\hbar\wr^{\underline{1}}\boldsymbol{f}^{++}\int^{\underline{1}}\int^{\underline{0}}\hbar\boldsymbol{e}} &\mapsto [(a, \mathscr{W})]_{\wr^{\underline{0}}\boldsymbol{f}^{+}\int^{\underline{0}}\hbar\boldsymbol{e}} \\
[(b, \mathscr{E})]_{\wr^{\underline{0}}\boldsymbol{g}^{+}\int^{\underline{0}}\hbar\boldsymbol{e}} &\mapsto [(b, \mathscr{E})]_{\boldsymbol{e}} \\
[(m, \mathscr{S})]_{\wr^{\underline{0}}\boldsymbol{g}^{+}\int^{\underline{0}}\hbar\boldsymbol{e}} &\mapsto [m]_{\boldsymbol{e}}
\end{aligned}$$

- $\star \vdash_{G^{\dagger}} [(m, X)]_{\boldsymbol{f}} \overset{\mathrm{df.}}{\Leftrightarrow} X = \mathscr{E} \wedge \exists \boldsymbol{g}, \boldsymbol{e} \in \mathcal{T}.\, \boldsymbol{f} = \wr^{\underline{0}}\boldsymbol{g}^{+}\int^{\underline{0}}\hbar\boldsymbol{e} \wedge \star \vdash_B [m]_{\boldsymbol{e}}$;

- $x \vdash_{G^{\dagger}} y \overset{\mathrm{df.}}{\Leftrightarrow} \mathit{att}_{G^{\dagger}}(x) = \mathit{att}_{G^{\dagger}}(y) \wedge \mathit{peel}_{G^{\dagger}}(x) \vdash_G \mathit{peel}_{G^{\dagger}}(y)$, where $\mathit{att}_{G^{\dagger}}$ is the function $M_{G^{\dagger}} \to \mathcal{T}$ that maps $[(a, \mathscr{W})]_{\wr^{\underline{0}}\wr^{\underline{1}}\boldsymbol{g}^{++}\int^{\underline{1}}\hbar\wr^{\underline{1}}\boldsymbol{f}^{++}\int^{\underline{1}}\int^{\underline{0}}\hbar\boldsymbol{e}} \mapsto \boldsymbol{g}$, $[(b, \mathscr{E})]_{\wr^{\underline{0}}\boldsymbol{g}^{+}\int^{\underline{0}}\hbar\boldsymbol{e}} \mapsto \boldsymbol{g}$, $[(m, \mathscr{S})]_{\wr^{\underline{0}}\boldsymbol{g}^{+}\int^{\underline{0}}\hbar\boldsymbol{e}} \mapsto \boldsymbol{g}$;

- $P_{G^{\dagger}} \overset{\mathrm{df.}}{=} \{\boldsymbol{s} \in \mathscr{L}_{G^{\dagger}} \mid \forall \boldsymbol{g} \in \mathcal{T}.\, \boldsymbol{s} \upharpoonright \boldsymbol{g} \in P_G \wedge (\boldsymbol{s} \upharpoonright \boldsymbol{g} \neq \boldsymbol{\epsilon} \Rightarrow \forall \boldsymbol{h} \in \mathcal{T}.\, \boldsymbol{s} \upharpoonright \boldsymbol{h} \neq \boldsymbol{\epsilon} \Rightarrow \mathit{ede}(\boldsymbol{g}) \neq \mathit{ede}(\boldsymbol{h}))\}$, where $\boldsymbol{s} \upharpoonright \boldsymbol{g}$ is the j-subsequence of $\boldsymbol{s}$ that consists of moves x such that $\mathit{att}_{G^{\dagger}}(x) = \boldsymbol{g}$ changed into $\mathit{peel}_{G^{\dagger}}(x)$;

- $\boldsymbol{s} \simeq_{G^{\dagger}} \boldsymbol{t} \overset{\mathrm{df.}}{\Leftrightarrow} \exists \varphi \in \mathcal{P}(\mathbb{N}).\, (\wp \circ \mathit{ede} \circ \mathit{peel}_{G^{\dagger}})^{*}(\boldsymbol{s}) = (\varphi \circ \wp \circ \mathit{ede} \circ \mathit{peel}_{G^{\dagger}})^{*}(\boldsymbol{t}) \wedge \forall i \in \mathbb{N}.\, \boldsymbol{s} \upharpoonright \varphi(i) \simeq_G \boldsymbol{t} \upharpoonright i$, where $\boldsymbol{s} \upharpoonright j$ for each $j \in \mathbb{N}$ denotes $\boldsymbol{s} \upharpoonright \boldsymbol{f}$ such that $\wp \circ \mathit{ede}(\boldsymbol{f}) = j$;

- $\Delta_{G^{\dagger}} : [(m, \mathscr{S})]_{\wr^{\underline{0}}\boldsymbol{g}^{+}\int^{\underline{0}}\hbar\boldsymbol{e}} \overset{\mathrm{df.}}{=} [(m', \mathscr{S})]_{\wr^{\underline{0}}\boldsymbol{g}^{+}\int^{\underline{0}}\hbar\boldsymbol{e}}$, where $\Delta_G([m]_{\boldsymbol{e}}) = [m']_{\boldsymbol{e}}$.

It is not hard to see that $\mathcal{H}^{\omega}(G^{\dagger}) \trianglelefteq \, !A \multimap \, !B$ if $\mathcal{H}^{\omega}(G) \trianglelefteq \, !A \multimap B$, in particular $(!A \multimap B)^{\dagger} \trianglelefteq \, !A \multimap \, !B$, for any normalized dynamic games A and B.

Next, we formalize 'tags' for *concatenation* $\ddagger$ of dynamic games (Definition 3.3.23). Again, since our formalization of concatenation is slightly more involved than other constructions, let us first sketch the idea. Given dynamic games J and K such that $\mathcal{H}^{\omega}(J) \trianglelefteq A \multimap B$ and $\mathcal{H}^{\omega}(K) \trianglelefteq B \multimap C$ for some normalized dynamic games A, B and C, we formalize 'tags' for the concatenation $J \ddagger K$ as follows:

- We do not change moves of A or C, i.e., ones of the form $[(a, \mathscr{W})]_{\boldsymbol{e}} \in M_J^{\mathsf{Ext}}$ or $[(c, \mathscr{E})]_{\boldsymbol{f}} \in M_K^{\mathsf{Ext}}$;
- We change moves of B in J, i.e., external ones of the form $[(b, \mathscr{E})]_{\boldsymbol{g}}$, into $[((b, \mathscr{E}), \mathscr{S})]_{\boldsymbol{g}}$;
- We change moves of B in K, i.e., external ones of the form $[(b, \mathscr{W})]_{\boldsymbol{g}}$, into $[((b, \mathscr{W}), \mathscr{N})]_{\boldsymbol{g}}$;
- We change internal moves $[m]_{\boldsymbol{l}}$ of J into $[(m, \mathscr{S})]_{\boldsymbol{l}}$;
- We change internal moves $[n]_{\boldsymbol{r}}$ of K into $[(n, \mathscr{N})]_{\boldsymbol{r}}$

where we distinguish the cases by the inner tag and the I/E-parity of each move. Note again that this formalization of 'tags' is not canonical at all, but it certainly achieves the required subgame relation $\mathcal{H}^\omega(J \ddagger K) \trianglelefteq A \multimap C$ (see Definition 4.3.23).

Then again, the labeling function, the enabling relation and the dummy function of $J \ddagger K$ are defined by the obvious pattern matching on inner tags, and positions of $J \ddagger K$ and the identification of them are defined as usual.

Formally, the concatenation $J \ddagger K$ of J and K is defined as follows, where it should be now clear how the *peeling* $\mathit{peel}_{J \ddagger K}$ and the *attributes* $\mathit{att}_{J \ddagger K}$ given below work:

Definition 4.3.23 (Concatenation of dynamic games revisited)**.** In the rest of the chapter, given dynamic games J and K such that $\mathcal{H}^\omega(J) \trianglelefteq A \multimap B$ and $\mathcal{H}^\omega(K) \trianglelefteq B \multimap C$ for some normalized dynamic games A, B and C, the ***concatenation*** $J \ddagger K$ of J and K is given by:

- $M_{J \ddagger K} \stackrel{\text{df.}}{=} \{[(a, \mathscr{W})]_{\boldsymbol{e}} \mid [(a, \mathscr{W})]_{\boldsymbol{e}} \in M_J^{\mathsf{Ext}}, [a]_{\boldsymbol{e}} \in M_A\}$
$\cup \{[(c, \mathscr{E})]_{\boldsymbol{f}} \mid [(c, \mathscr{E})]_{\boldsymbol{f}} \in M_K^{\mathsf{Ext}}, [c]_{\boldsymbol{f}} \in M_C\}$
$\cup \{[((b, \mathscr{E}), \mathscr{S})]_{\boldsymbol{g}} \mid [(b, \mathscr{E})]_{\boldsymbol{g}} \in M_J^{\mathsf{Ext}}, [b]_{\boldsymbol{g}} \in M_B\}$
$\cup \{[((b, \mathscr{W}), \mathscr{N})]_{\boldsymbol{g}} \mid [(b, \mathscr{W})]_{\boldsymbol{g}} \in M_K^{\mathsf{Ext}}, [b]_{\boldsymbol{g}} \in M_B\}$
$\cup \{[(m, \mathscr{S})]_{\boldsymbol{l}} \mid [m]_{\boldsymbol{l}} \in M_J^{\mathsf{Int}}\} \cup \{[(n, \mathscr{N})]_{\boldsymbol{r}} \mid [n]_{\boldsymbol{r}} \in M_K^{\mathsf{Int}}\}$;
- $$\lambda_{J \ddagger K}([(m, X)]_{\boldsymbol{e}}) \stackrel{\text{df.}}{=} \begin{cases} \lambda_J^{+\mu}([m]_{\boldsymbol{e}}) & \text{if } X = \mathscr{S} \wedge \exists [b]_{\boldsymbol{e}} \in M_B.\, [m]_{\boldsymbol{e}} = [(b, \mathscr{E})]_{\boldsymbol{e}} \in M_J^{\mathsf{Ext}}; \\ \lambda_J([m]_{\boldsymbol{e}}) & \text{if } X = \mathscr{W} \vee (X = \mathscr{S} \wedge [m]_{\boldsymbol{e}} \in M_J^{\mathsf{Int}}); \\ \lambda_K^{+\mu}([m]_{\boldsymbol{e}}) & \text{if } X = \mathscr{N} \wedge \exists [b]_{\boldsymbol{e}} \in M_B.\, [m]_{\boldsymbol{e}} = [(b, \mathscr{W})]_{\boldsymbol{e}} \in M_K^{\mathsf{Ext}}; \\ \lambda_K([m]_{\boldsymbol{e}}) & \text{if } X = \mathscr{E} \vee (X = \mathscr{N} \wedge [m]_{\boldsymbol{e}} \in M_K^{\mathsf{Int}}) \end{cases}$$
where $\mu \stackrel{\text{df.}}{=} \mathsf{Sup}(\{\mu(J), \mu(K)\}) + 1 \in \mathbb{N}$;
- $\star \vdash_{J \ddagger K} [(m, X)]_{\boldsymbol{e}} \stackrel{\text{df.}}{\Leftrightarrow} X = \mathscr{E} \wedge \star \vdash_C [m]_{\boldsymbol{e}}$;

- $[(m, X)]_{\boldsymbol{e}} \vdash_{J \ddagger K} [(n, Y)]_{\boldsymbol{f}} \overset{\text{df.}}{\Leftrightarrow} (att_{J \ddagger K}([(m, X)]_{\boldsymbol{e}}) = J = att_{J \ddagger K}([(n, Y)]_{\boldsymbol{f}}) \wedge peel_{J \ddagger K}([(m, X)]_{\boldsymbol{e}}) \vdash_J peel_{J \ddagger K}([(n, Y)]_{\boldsymbol{f}})) \vee (att_{J \ddagger K}([(m, X)]_{\boldsymbol{e}}) = K = att_{J \ddagger K}([(n, Y)]_{\boldsymbol{f}}) \wedge peel_{J \ddagger K}([(m, X)]_{\boldsymbol{e}}) \vdash_K peel_{J \ddagger K}([(n, Y)]_{\boldsymbol{f}})) \vee (X = \mathscr{N} \wedge Y = \mathscr{S} \wedge \exists [b]_{\boldsymbol{e}}, [b']_{\boldsymbol{f}} \in M_B^{\mathsf{Init}}.\, m = (b, \mathscr{W}) \wedge n = (b, \mathscr{E}))$, where the map $att_{J \ddagger K} : M_{J \ddagger K} \to \{J, K\}$ is defined by $[(a, \mathscr{W})]_{\boldsymbol{e}} \mapsto J$, $[(m, \mathscr{S})]_{\boldsymbol{l}} \mapsto J$, $[((b, \mathscr{E}), \mathscr{S})]_{\boldsymbol{g}} \mapsto J$, $[(c, \mathscr{E})]_{\boldsymbol{f}} \mapsto K$, $[(n, \mathscr{N})]_{\boldsymbol{r}} \mapsto K$, $[((b, \mathscr{W}), \mathscr{N})]_{\boldsymbol{g}} \mapsto K$, and the map $peel_{J \ddagger K} : M_{J \ddagger K} \to M_J \cup M_K$ by $[(a, \mathscr{W})]_{\boldsymbol{e}} \mapsto [(a, \mathscr{W})]_{\boldsymbol{e}}$, $[(c, \mathscr{E})]_{\boldsymbol{f}} \mapsto [(c, \mathscr{E})]_{\boldsymbol{f}}$, $[((b, \mathscr{E}), \mathscr{S})]_{\boldsymbol{g}} \mapsto [(b, \mathscr{E})]_{\boldsymbol{g}}$, $[((b, \mathscr{W}), \mathscr{N})]_{\boldsymbol{g}} \mapsto [(b, \mathscr{W})]_{\boldsymbol{g}}$, $[(m, \mathscr{S})]_{\boldsymbol{l}} \mapsto [m]_{\boldsymbol{l}}$, $[(n, \mathscr{N})]_{\boldsymbol{r}} \mapsto [n]_{\boldsymbol{r}}$;

- $P_{J \ddagger K} \overset{\text{df.}}{=} \{\boldsymbol{s} \in \mathscr{J}_{J \ddagger K} \mid \boldsymbol{s} \upharpoonright J \in P_J, \boldsymbol{s} \upharpoonright K \in P_K, \boldsymbol{s} \upharpoonright B^{[1]}, B^{[2]} \in pr_B\}$, where $\boldsymbol{s} \upharpoonright J$ (resp. $\boldsymbol{s} \upharpoonright K$) is the subsequence of $\boldsymbol{s}$ that consists of moves m such that $att_{J \ddagger K}(m) = J$ (resp. $att_{J \ddagger K}(m) = K$) but changed into $peel_{J \ddagger K}(m)$, and $\boldsymbol{s} \upharpoonright B^{[1]}, B^{[2]}$ is the j-subsequence of $\boldsymbol{s}$ that consists of moves in $B^{[1]}$ and $B^{[2]}$, i.e., moves $[((b, X), Y)]_{\boldsymbol{e}}$ such that $[b]_{\boldsymbol{e}} \in M_B \wedge ((X = \mathscr{E} \wedge Y = \mathscr{S}) \vee (X = \mathscr{W} \wedge Y = \mathscr{N}))$ but changed into $[(b, \overline{X})]_{\boldsymbol{e}}$, for which $\overline{\mathscr{E}} \overset{\text{df.}}{=} \mathscr{W}$ and $\overline{\mathscr{W}} \overset{\text{df.}}{=} \mathscr{E}$;

- $\boldsymbol{s} \simeq_{J \ddagger K} \boldsymbol{t} \overset{\text{df.}}{\Leftrightarrow} (\pi_2 \circ \pi_1)^*(\boldsymbol{s}) = (\pi_2 \circ \pi_1)^*(\boldsymbol{t}) \wedge \boldsymbol{s} \upharpoonright J \simeq_J \boldsymbol{t} \upharpoonright J \wedge \boldsymbol{s} \upharpoonright K \simeq_K \boldsymbol{t} \upharpoonright K$;

- $$\Delta_{J \ddagger K}([(m, X)]_{\boldsymbol{e}}) \overset{\text{df.}}{=} \begin{cases} [(m', \mathscr{S})]_{\boldsymbol{e}} & \text{if } X = \mathscr{S} \text{ and } \Delta_J([m]_{\boldsymbol{e}}) = [m']_{\boldsymbol{e}}; \\ [(m'', \mathscr{N})]_{\boldsymbol{e}} & \text{if } X = \mathscr{N} \text{ and } \Delta_K([m]_{\boldsymbol{e}}) = [m'']_{\boldsymbol{e}}; \\ [((b, \mathscr{W}), \mathscr{N})]_{\boldsymbol{e}} & \text{if } X = \mathscr{S},\ \Delta_J([m]_{\boldsymbol{e}}) \uparrow \text{ and } m = (b, \mathscr{E}); \\ [((b, \mathscr{E}), \mathscr{S})]_{\boldsymbol{e}} & \text{if } X = \mathscr{N},\ \Delta_K([m]_{\boldsymbol{e}}) \uparrow \text{ and } m = (b, \mathscr{W}). \end{cases}$$

It is straightforward to see that $\mathcal{H}^\omega(J \ddagger K) \trianglelefteq A \multimap C$ if $\mathcal{H}^\omega(J) \trianglelefteq A \multimap B$ and $\mathcal{H}^\omega(K) \trianglelefteq B \multimap C$, in particular $(A \multimap B) \ddagger (B \multimap C) \trianglelefteq A \multimap C$, for any normalized dynamic games A, B and C. Note also that the additional dummy functions of dynamic games (Definition 4.3.9) are motivated by the phenomenon of moves of B in the concatenation $J \ddagger K$.

Example 4.3.24. Some typical plays of the concatenation $(\mathcal{N} \multimap \mathcal{N}) \ddagger (\mathcal{N} \multimap \mathcal{N})$ are described in Figure 4.11.

Finally, we make the trivial *currying* Λ (as they just adjust inner tags [14]) precise:

Definition 4.3.25 (Currying of dynamic games revisited)**.** In the rest of the chapter, if a dynamic game G satisfies $\mathcal{H}^\omega(G) \trianglelefteq A \otimes B \multimap C$ for some normalized dynamic games A, B and C, then the ***currying*** $\Lambda(G)$ of G is given by:

- $M_{\Lambda(G)} \overset{\text{df.}}{=} \{[(a, \mathscr{W})]_{\boldsymbol{e}} \mid [((a, \mathscr{W}), \mathscr{W})]_{\boldsymbol{e}} \in M_G^{\mathsf{Ext}}, [a]_{\boldsymbol{e}} \in M_A\} \cup \{[((b, \mathscr{W}), \mathscr{E})]_{\boldsymbol{f}} \mid [((b, \mathscr{E}), \mathscr{W})]_{\boldsymbol{f}} \in M_G^{\mathsf{Ext}}, [b]_{\boldsymbol{f}} \in M_B\} \cup \{[((c, \mathscr{E}), \mathscr{E})]_{\boldsymbol{g}} \mid [(c, \mathscr{E})]_{\boldsymbol{g}} \in M_G^{\mathsf{Ext}}, [c]_{\boldsymbol{g}} \in M_C\} \cup \{[(m, \mathscr{N})]_{\boldsymbol{h}} \mid [m]_{\boldsymbol{h}} \in M_G^{\mathsf{Int}}\}$;

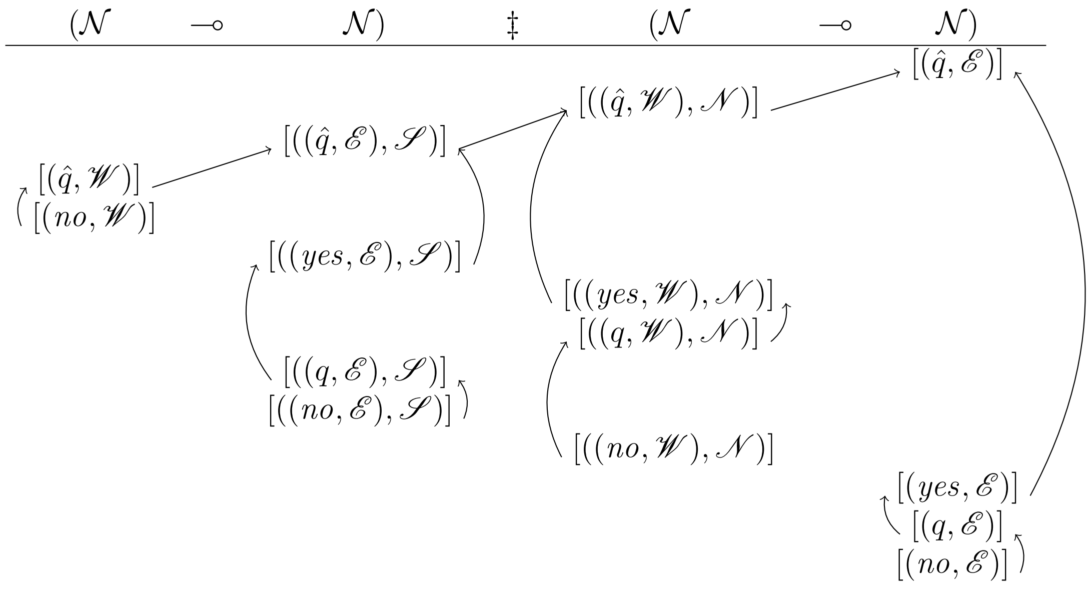

$(\mathcal{N} \multimap \mathcal{N}) \ddagger (\mathcal{N} \multimap \mathcal{N})$
$[(\hat{q}, \mathscr{E})]$
$[((\hat{q}, \mathscr{W}), \mathscr{N})]$
$[((\hat{q}, \mathscr{E}), \mathscr{S})]$
$[(\hat{q}, \mathscr{W})]$
$[(no, \mathscr{W})]$
$[((yes, \mathscr{E}), \mathscr{S})]$
$[((yes, \mathscr{W}), \mathscr{N})]$
$[((q, \mathscr{W}), \mathscr{N})]$
$[((q, \mathscr{E}), \mathscr{S})]$
$[((no, \mathscr{E}), \mathscr{S})]$
$[((no, \mathscr{W}), \mathscr{N})]$
$[(yes, \mathscr{E})]$
$[(q, \mathscr{E})]$
$[(no, \mathscr{E})]$

Figure 4.11: A typical play of the concatenation $(\mathcal{N} \multimap \mathcal{N}) \ddagger (\mathcal{N} \multimap \mathcal{N})$.

- $\lambda_{\Lambda(G)} : [(a, \mathscr{W})]_{\boldsymbol{e}} \mapsto \lambda_G([((a, \mathscr{W}), \mathscr{W})]_{\boldsymbol{e}})$, $[((b, \mathscr{W}), \mathscr{E})]_{\boldsymbol{f}} \mapsto \lambda_G([((b, \mathscr{E}), \mathscr{W})]_{\boldsymbol{f}})$, $[((c, \mathscr{E}), \mathscr{E})]_{\boldsymbol{g}} \mapsto \lambda_G([(c, \mathscr{E})]_{\boldsymbol{g}})$, $[(m, \mathscr{N})]_{\boldsymbol{h}} \mapsto \lambda_G([m]_{\boldsymbol{h}})$;
- $\star \vdash_{\Lambda(G)} [m]_{\boldsymbol{e}} \overset{\text{df.}}{\Leftrightarrow} \exists [c]_{\boldsymbol{e}} \in M_C^{\mathsf{Init}}.\, m = ((c, \mathscr{E}), \mathscr{E})$;
- $[m]_{\boldsymbol{e}} \vdash_{\Lambda(G)} [n]_{\boldsymbol{f}} \overset{\text{df.}}{\Leftrightarrow} \mathit{peel}_{\Lambda(G)}([m]_{\boldsymbol{e}}) \vdash_G \mathit{peel}_{\Lambda(G)}([n]_{\boldsymbol{f}})$, where the function $\mathit{peel}_{\Lambda(G)} : M_{\Lambda(G)} \to M_G$ maps $[(a, \mathscr{W})]_{\boldsymbol{e}} \mapsto [((a, \mathscr{W}), \mathscr{W})]_{\boldsymbol{e}}$, $[((b, \mathscr{W}), \mathscr{E})]_{\boldsymbol{f}} \mapsto [((b, \mathscr{E}), \mathscr{W})]_{\boldsymbol{f}}$, $[((c, \mathscr{E}), \mathscr{E})]_{\boldsymbol{g}} \mapsto [(c, \mathscr{E})]_{\boldsymbol{g}}$, $[(m, \mathscr{N})]_{\boldsymbol{h}} \mapsto [m]_{\boldsymbol{h}}$;
- $P_{\Lambda(G)} \overset{\text{df.}}{=} \{\boldsymbol{s} \in \mathscr{L}_{\Lambda(G)} \mid \mathit{peel}^*_{\Lambda(G)}(\boldsymbol{s}) \in P_G\}$, where the structure of justifiers in $\mathit{peel}^*_{\Lambda(G)}(\boldsymbol{s})$ is the same as $\boldsymbol{s}$;
- $\boldsymbol{s} \simeq_{\Lambda(G)} \boldsymbol{t} \overset{\text{df.}}{\Leftrightarrow} \mathit{peel}^*_{\Lambda(G)}(\boldsymbol{s}) \simeq_G \mathit{peel}^*_{\Lambda(G)}(\boldsymbol{t})$;
- $\Delta_{\Lambda(G)} : [(m, \mathscr{N})]_{\boldsymbol{e}} \mapsto [(m', \mathscr{N})]_{\boldsymbol{e}}$, where $\Delta_G : [m]_{\boldsymbol{e}} \mapsto [m']_{\boldsymbol{e}}$.

It is easy to see that $\mathcal{H}^\omega(\Lambda(G)) \trianglelefteq A \multimap (B \multimap C)$ holds if $\mathcal{H}^\omega(G) \trianglelefteq A \otimes B \multimap C$, which is a generalization of the equation $\Lambda(A \otimes B \multimap C) = A \multimap (B \multimap C)$.

Similarly, the uncurrying $\Lambda^\ominus$ of dynamic games is formalized as follows:

Definition 4.3.26 (Uncurrying of dynamic games revisited)**.** In the rest of the chapter, if a dynamic game H satisfies $\mathcal{H}^\omega(H) \trianglelefteq A \multimap (B \multimap C)$ for some normalized dynamic games A, B and C, then its ***uncurrying*** $\Lambda^\ominus(H)$ is given by:

- $M_{\Lambda^{\ominus}(H)} \stackrel{\mathrm{df.}}{=} \{[((a,\mathscr{W}),\mathscr{W})]_{\boldsymbol{e}} \mid [(a,\mathscr{W})]_{\boldsymbol{e}} \in M_H^{\mathsf{Ext}}, [a]_{\boldsymbol{e}} \in M_A\}$
  $\cup \{[((b,\mathscr{E}),\mathscr{W})]_{\boldsymbol{f}} \mid [((b,\mathscr{W}),\mathscr{E})]_{\boldsymbol{f}} \in M_H^{\mathsf{Ext}}, [b]_{\boldsymbol{f}} \in M_B\}$
  $\cup \{[(c,\mathscr{E})]_{\boldsymbol{g}} \mid [((c,\mathscr{E}),\mathscr{E})]_{\boldsymbol{g}} \in M_H^{\mathsf{Ext}}, [c]_{\boldsymbol{g}} \in M_C\} \cup \{[(m,\mathscr{S})]_{\boldsymbol{h}} \mid [m]_{\boldsymbol{h}} \in M_H^{\mathsf{Int}}\}$;

- $\lambda_{\Lambda^{\ominus}(H)} : [((a,\mathscr{W}),\mathscr{W})]_{\boldsymbol{e}} \mapsto \lambda_H([(a,\mathscr{W})]_{\boldsymbol{e}})$, $[((b,\mathscr{E}),\mathscr{W})]_{\boldsymbol{f}} \mapsto \lambda_H([((b,\mathscr{W}),\mathscr{E})]_{\boldsymbol{f}})$, $[(c,\mathscr{E})]_{\boldsymbol{g}} \mapsto \lambda_H([((c,\mathscr{E}),\mathscr{E})]_{\boldsymbol{g}}), [(m,\mathscr{S})]_{\boldsymbol{h}} \mapsto \lambda_H([m]_{\boldsymbol{h}})$;

- $\star \vdash_{\Lambda^{\ominus}(H)} [m]_{\boldsymbol{e}} \stackrel{\mathrm{df.}}{\Leftrightarrow} \exists [c]_{\boldsymbol{e}} \in M_C^{\mathsf{Init}}. \, m = (c,\mathscr{E})$;

- $[m]_{\boldsymbol{e}} \vdash_{\Lambda^{\ominus}(H)} [n]_{\boldsymbol{f}} \stackrel{\mathrm{df.}}{\Leftrightarrow} \mathit{peel}_{\Lambda^{\ominus}(H)}([m]_{\boldsymbol{e}}) \vdash_H \mathit{peel}_{\Lambda^{\ominus}(H)}([n]_{\boldsymbol{f}})$, where the function $\mathit{peel}_{\Lambda^{\ominus}(H)} : M_{\Lambda^{\ominus}(H)} \to M_H$ maps $[((a,\mathscr{W}),\mathscr{W})]_{\boldsymbol{e}} \mapsto [(a,\mathscr{W})]_{\boldsymbol{e}}$, $[((b,\mathscr{E}),\mathscr{W})]_{\boldsymbol{f}} \mapsto [((b,\mathscr{W}),\mathscr{E})]_{\boldsymbol{f}}$, $[(c,\mathscr{E})]_{\boldsymbol{g}} \mapsto [((c,\mathscr{E}),\mathscr{E})]_{\boldsymbol{g}}$, $[(m,\mathscr{S})]_{\boldsymbol{h}} \mapsto [m]_{\boldsymbol{h}}$;

- $P_{\Lambda^{\ominus}(H)} \stackrel{\mathrm{df.}}{=} \{\boldsymbol{s} \in \mathscr{L}_{\Lambda^{\ominus}(H)} \mid \mathit{peel}^*_{\Lambda^{\ominus}(H)}(\boldsymbol{s}) \in P_H\}$, where the structure of justifiers in $\mathit{peel}^*_{\Lambda^{\ominus}(H)}(\boldsymbol{s})$ is the same as $\boldsymbol{s}$;

- $\boldsymbol{s} \simeq_{\Lambda^{\ominus}(H)} \boldsymbol{t} \stackrel{\mathrm{df.}}{\Leftrightarrow} \mathit{peel}^*_{\Lambda^{\ominus}(H)}(\boldsymbol{s}) \simeq_H \mathit{peel}^*_{\Lambda^{\ominus}(H)}(\boldsymbol{t})$.

Dually to currying, $\mathcal{H}^{\omega}(\Lambda^{\ominus}(H)) \trianglelefteq A \otimes B \multimap C$ holds if $\mathcal{H}^{\omega}(H) \trianglelefteq A \multimap (B \multimap C)$, which is a generalization of the equation $\Lambda^{\ominus}(A \multimap (B \multimap C)) = A \otimes B \multimap C$. Moreover, Λ and $\Lambda^{\ominus}$ are inverses to each other for normalized dynamic games, i.e., $\Lambda \circ \Lambda^{\ominus}(A \multimap (B \multimap C)) = A \multimap (B \multimap C)$ and $\Lambda^{\ominus} \circ \Lambda(A \otimes B \multimap C) = A \otimes B \multimap C$ for any normalized dynamic games A, B and C.

4.3.2 Constructions on Dynamic Strategies Revisited

Having formalized 'tags' for constructions on dynamic games, it is automatically determined for most cases how to adjust dynamic strategies accordingly. Nevertheless, for the aim of clarity of our formalization, let us present explicitly such formalized dynamic strategies and constructions on them (in Section 3.3.5) in this section.

Notation. Henceforth, we often indicate the form of tags of moves $[m_{X_1 X_2 \dots X_k}]_{\boldsymbol{e}}$ of a dynamic game G informally by $[G_{X_1 X_2 \dots X_k}]_{\boldsymbol{e}}$, especially when the tags are complex.

Let us first see how PCF-atomic ones (Definition 4.2.9) are formalized:

Example 4.3.27. In the rest of the chapter, the zero strategy $\mathit{zero}_A : [!A_{\mathscr{W}}]_{\wr\underline{0}\boldsymbol{e}+\int\underline{0}\hbar} \multimap [\mathcal{N}_{\mathscr{E}}]$ on a normalized dynamic game A (Definition 4.2.2) is given by:

$$\mathit{zero}_A \stackrel{\mathrm{df.}}{=} \{\boldsymbol{\epsilon}, [\hat{q}_{\mathscr{E}}][\mathit{no}_{\mathscr{E}}]\}.$$

The canonical play by zero_A is as in Figure 4.12, which corresponds to Figure 4.1.

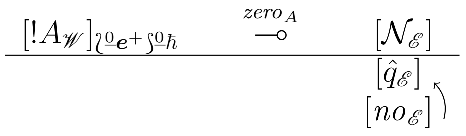

$[!A_{\mathscr{W}}]_{\wr\underline{0}\boldsymbol{e}+\int\underline{0}\hbar}$
$\overset{zero_A}{\multimap}$
$[\mathcal{N}_{\mathscr{E}}]$
$[\hat{q}_{\mathscr{E}}]$
$[no_{\mathscr{E}}]$

Figure 4.12: The computation of the zero strategy $zero_A : A \Rightarrow \mathcal{N}$ revisited.

Example 4.3.28. In the rest of the chapter, the successor strategy $succ : [!\mathcal{N}_{\mathscr{W}}]_{\wr\underline{0}\boldsymbol{e}+\int\underline{0}\hbar} \multimap [\mathcal{N}_{\mathscr{E}}]$ (Definition 4.2.3) is given by:

$$succ \overset{\text{df.}}{=} \mathsf{Pref}(\{[\hat{q}_{\mathscr{E}}][\hat{q}_{\mathscr{W}}]_{\wr\underline{0}\int\underline{0}\hbar}([yes_{\mathscr{W}}]_{\wr\underline{0}\int\underline{0}\hbar}[yes_{\mathscr{E}}][q_{\mathscr{E}}][q_{\mathscr{W}}]_{\wr\underline{0}\int\underline{0}\hbar})^i[no_{\mathscr{W}}]_{\wr\underline{0}\int\underline{0}\hbar}[yes_{\mathscr{E}}][q_{\mathscr{E}}][no_{\mathscr{E}}] \mid i \in \mathbb{N}\})^{\mathsf{Even}}.$$

The computation of *succ* is described in Figure 4.13, corresponding to Figure 4.2.

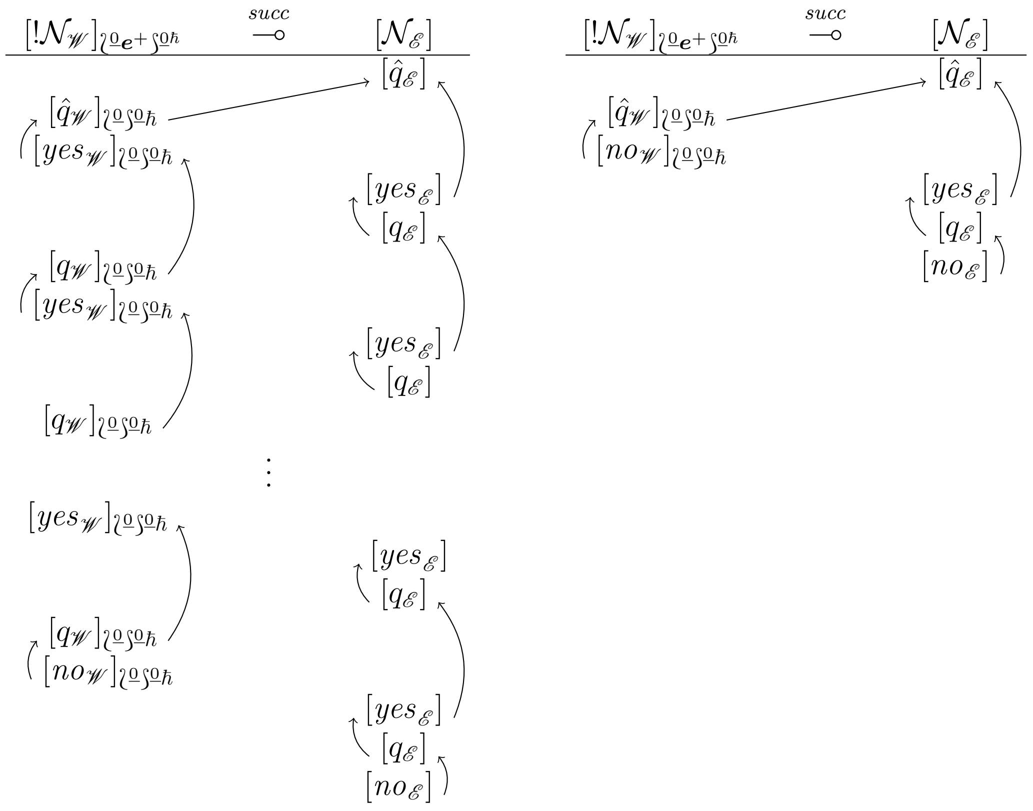

$[!\mathcal{N}_{\mathscr{W}}]_{\wr\underline{0}\boldsymbol{e}+\int\underline{0}\hbar}$
$\overset{succ}{\multimap}$
$[\mathcal{N}_{\mathscr{E}}]$
$[\hat{q}_{\mathscr{E}}]$
$[\hat{q}_{\mathscr{W}}]_{\wr\underline{0}\int\underline{0}\hbar}$
$[yes_{\mathscr{W}}]_{\wr\underline{0}\int\underline{0}\hbar}$
$[yes_{\mathscr{E}}]$
$[q_{\mathscr{E}}]$
$[q_{\mathscr{W}}]_{\wr\underline{0}\int\underline{0}\hbar}$
$[no_{\mathscr{W}}]_{\wr\underline{0}\int\underline{0}\hbar}$
$[no_{\mathscr{E}}]$

Figure 4.13: The computation of the successor strategy $succ : \mathcal{N} \Rightarrow \mathcal{N}$ revisited.

Example 4.3.29. In the rest of the chapter, the predecessor strategy $pred : [!\mathcal{N}_{\mathscr{W}}]_{\wr\underline{0}\boldsymbol{e}+\int\underline{0}\hbar} \multimap [\mathcal{N}_{\mathscr{E}}]$ (Definition 4.2.4) is given by:

$$\begin{aligned} pred \overset{\text{df.}}{=} \mathsf{Pref}(\{&[\hat{q}_{\mathscr{E}}][\hat{q}_{\mathscr{W}}]_{\wr\underline{0}\int\underline{0}\hbar}[yes_{\mathscr{W}}]_{\wr\underline{0}\int\underline{0}\hbar}[q_{\mathscr{W}}]_{\wr\underline{0}\int\underline{0}\hbar}([yes_{\mathscr{W}}]_{\wr\underline{0}\int\underline{0}\hbar}[yes_{\mathscr{E}}][q_{\mathscr{E}}][q_{\mathscr{W}}]_{\wr\underline{0}\int\underline{0}\hbar})^i[no_{\mathscr{W}}]_{\wr\underline{0}\int\underline{0}\hbar}[no_{\mathscr{E}}] \mid \\ &i \in \mathbb{N}\} \cup \{[\hat{q}_{\mathscr{E}}][\hat{q}_{\mathscr{W}}]_{\wr\underline{0}\int\underline{0}\hbar}[no_{\mathscr{W}}]_{\wr\underline{0}\int\underline{0}\hbar}[no_{\mathscr{E}}]\})^{\mathsf{Even}}. \end{aligned}$$

The computation of *pred* is described in Figure 4.14, corresponding to Figure 4.3.

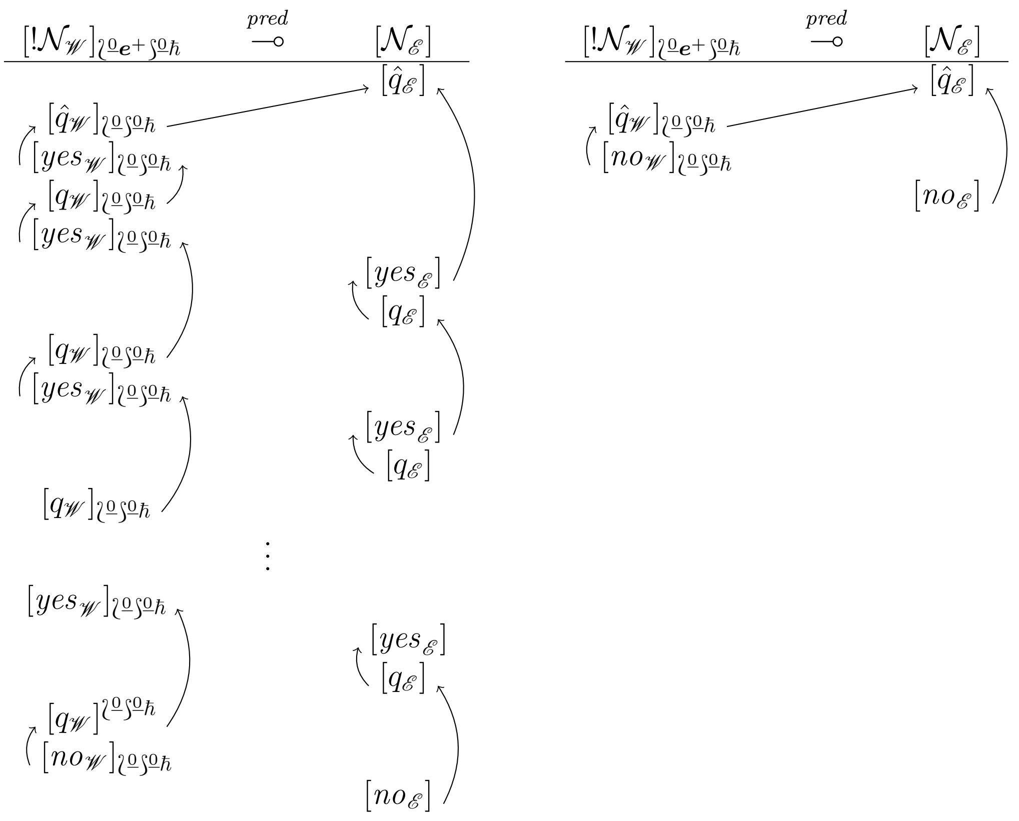


Figure 4.14: The computation of the predecessor strategy $pred : \mathcal{N} \Rightarrow \mathcal{N}$ revisited.

Example 4.3.30. In the rest of the chapter, the dereliction $der_A : [!A_{\mathscr{W}}]_{\wr\underline{o}\boldsymbol{f}+\int\underline{o}\hbar\boldsymbol{e}} \multimap [A_{\mathscr{E}}]_{\boldsymbol{e}'}$ on a normalized dynamic game A (Definition 2.3.26) is given by:

$$der_A \overset{\text{df.}}{=} \{\boldsymbol{s} \in P^{\mathsf{Even}}_{A \Rightarrow A} \mid \forall \boldsymbol{t} \preceq \boldsymbol{s}.\ \mathsf{Even}(\boldsymbol{t}) \Rightarrow \boldsymbol{t} \upharpoonright (\mathscr{W})_{\wr\underline{o}\int\underline{o}\hbar\_} = \boldsymbol{t} \upharpoonright (\mathscr{E})_{\_}\}$$

where $\boldsymbol{t} \upharpoonright (\mathscr{W})_{\wr\underline{o}\int\underline{o}\hbar\_}$ (resp. $\boldsymbol{t} \upharpoonright (\mathscr{E})_{\_}$) is the j-subsequence of $\boldsymbol{t}$ that consists of moves of the form $[(a, \mathscr{W})]_{\wr\underline{o}\int\underline{o}\hbar\boldsymbol{e}}$ (resp. $[(a', \mathscr{E})]_{\boldsymbol{e}'}$) changed into $[a]_{\boldsymbol{e}}$ (resp. $[a']_{\boldsymbol{e}'}$). The computation of der_A may be described as in Figure 4.15.

Now, it should be clear how the following two dynamic strategies are formalized, and therefore let us skip depicting diagrams of their computations:

Example 4.3.31. The case strategy $case_A : [A_{\mathscr{W}\mathscr{W}\mathscr{W}}]_{\wr\underline{o}\boldsymbol{f}+\int\underline{o}\hbar\boldsymbol{e}} \& [A_{\mathscr{E}\mathscr{W}\mathscr{W}}]_{\wr\underline{o}\boldsymbol{f}'+\int\underline{o}\hbar\boldsymbol{e}'} \& [\mathbf{2}_{\mathscr{E}\mathscr{W}}]_{\wr\underline{o}\boldsymbol{f}''+\int\underline{o}\hbar} \Rightarrow [A_{\mathscr{E}}]_{\boldsymbol{e}''}$ on a normalized dynamic game A (Definition 4.2.5) in the rest of the chapter is given by:

$$\begin{aligned} case_A \overset{\text{df.}}{=} \mathsf{Pref}(&\{[a_{\mathscr{E}}]_{\boldsymbol{e}}[\hat{q}_{\mathscr{E}\mathscr{W}}]_{\wr\underline{o}\boldsymbol{e}+\int\underline{o}\hbar}[tt_{\mathscr{E}\mathscr{W}}]_{\wr\underline{o}\boldsymbol{e}+\int\underline{o}\hbar}[a_{\mathscr{W}\mathscr{W}\mathscr{W}}]_{\wr\underline{o}\int\underline{o}\hbar\boldsymbol{e}}.\boldsymbol{s} \mid [a_{\mathscr{E}}]_{\boldsymbol{e}}[a_{\mathscr{W}\mathscr{W}\mathscr{W}}]_{\wr\underline{o}\int\underline{o}\hbar\boldsymbol{e}}.\boldsymbol{s} \in der^{\mathscr{W}}_A\} \\ &\cup \{[a_{\mathscr{E}}]_{\boldsymbol{f}}[\hat{q}_{\mathscr{E}\mathscr{W}}]_{\wr\underline{o}\boldsymbol{f}+\int\underline{o}\hbar}[f\!f_{\mathscr{E}\mathscr{W}}]_{\wr\underline{o}\boldsymbol{f}+\int\underline{o}\hbar}[a_{\mathscr{E}\mathscr{W}\mathscr{W}}]_{\wr\underline{o}\int\underline{o}\hbar\boldsymbol{f}}.\boldsymbol{t} \mid [a_{\mathscr{E}}]_{\boldsymbol{f}}[a_{\mathscr{E}\mathscr{W}\mathscr{W}}]_{\wr\underline{o}\int\underline{o}\hbar\boldsymbol{f}}.\boldsymbol{t} \in der^{\mathscr{E}}_A\})^{\mathsf{Even}} \end{aligned}$$

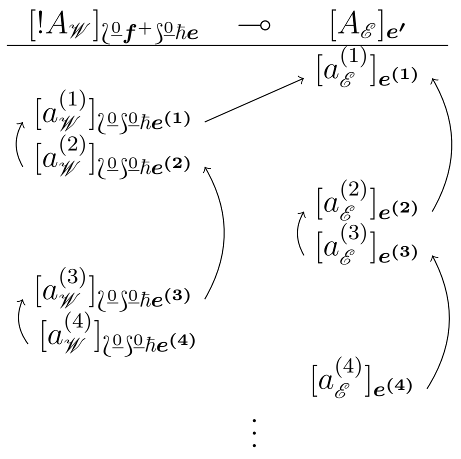

$[!A_{\mathscr{W}}]_{\ell\underline{0}\boldsymbol{f}+\int\underline{0}\hbar\boldsymbol{e}}$
$\multimap$
$[A_{\mathscr{E}}]_{\boldsymbol{e'}}$
$[a^{(1)}_{\mathscr{E}}]_{\boldsymbol{e^{(1)}}}$
$[a^{(1)}_{\mathscr{W}}]_{\ell\underline{0}\int\underline{0}\hbar\boldsymbol{e^{(1)}}}$
$[a^{(2)}_{\mathscr{W}}]_{\ell\underline{0}\int\underline{0}\hbar\boldsymbol{e^{(2)}}}$
$[a^{(2)}_{\mathscr{E}}]_{\boldsymbol{e^{(2)}}}$
$[a^{(3)}_{\mathscr{E}}]_{\boldsymbol{e^{(3)}}}$
$[a^{(3)}_{\mathscr{W}}]_{\ell\underline{0}\int\underline{0}\hbar\boldsymbol{e^{(3)}}}$
$[a^{(4)}_{\mathscr{W}}]_{\ell\underline{0}\int\underline{0}\hbar\boldsymbol{e^{(4)}}}$
$[a^{(4)}_{\mathscr{E}}]_{\boldsymbol{e^{(4)}}}$
$\vdots$

Figure 4.15: The computation of the dereliction $der_A : A \Rightarrow A$ revisited.

where $der^{\mathscr{W}}_A : [A_{\mathscr{W}\mathscr{W}\mathscr{W}}]_{\ell\underline{0}\boldsymbol{f}+\int\underline{0}\hbar\boldsymbol{e}} \Rightarrow [A_{\mathscr{E}}]_{\boldsymbol{g}}$ and $der^{\mathscr{E}}_A : [A_{\mathscr{E}\mathscr{W}\mathscr{W}}]_{\ell\underline{0}\boldsymbol{f'}+\int\underline{0}\hbar\boldsymbol{e'}} \Rightarrow [A_{\mathscr{E}}]_{\boldsymbol{g}}$ are the same as the dereliction $der_A : [A_{\mathscr{W}}]_{\ell\underline{0}\boldsymbol{f}+\int\underline{0}\hbar\boldsymbol{e}} \Rightarrow [A_{\mathscr{E}}]_{\boldsymbol{g}}$ up to inner tags.

Example 4.3.32. In the rest of the chapter, the ifzero strategy $zero? : [!\mathcal{N}_{\mathscr{W}}]_{\ell\underline{0}\boldsymbol{f}+\int\underline{0}\hbar\boldsymbol{e}} \multimap [\mathbf{2}_{\mathscr{E}}]_{\boldsymbol{e'}}$ (Definition 4.2.6) is given by:

$$zero? \overset{\mathrm{df.}}{=} \mathsf{Pref}(\{[\hat{q}_{\mathscr{E}}][\hat{q}_{\mathscr{W}}]_{\ell\underline{0}\int\underline{0}\hbar}[no_{\mathscr{W}}]_{\ell\underline{0}\int\underline{0}\hbar}[tt_{\mathscr{E}}], [\hat{q}_{\mathscr{E}}][\hat{q}_{\mathscr{W}}]_{\ell\underline{0}\int\underline{0}\hbar}[yes_{\mathscr{W}}]_{\ell\underline{0}\int\underline{0}\hbar}[f\!f_{\mathscr{E}}]\})^{\mathsf{Even}}.$$

Now, let us see how fixed-point strategies are formalized:

Example 4.3.33. The fixed-point strategy $fix_A : ([A_{\mathscr{W}\mathscr{W}}]_{\ell\underline{0}\boldsymbol{g}+\int\underline{0}\hbar\ell\underline{0}\boldsymbol{f}+\int\underline{0}\hbar\boldsymbol{e}} \Rightarrow [A_{\mathscr{E}\mathscr{W}}]_{\ell\underline{0}\boldsymbol{g'}+\int\underline{0}\hbar\boldsymbol{e'}}) \Rightarrow [A_{\mathscr{E}}]_{\boldsymbol{e''}}$ on a normalized dynamic game A (Definition 4.2.7) in the rest of the chapter plays as follows:

- After an initial occurrence $[a_{\mathscr{E}}]_{\boldsymbol{e}}$, fix_A copies it and performs the move $[a_{\mathscr{E}\mathscr{W}}]_{\ell\underline{0}\int\underline{0}\hbar\boldsymbol{e}}$ with the pointer towards the initial occurrence $[a_{\mathscr{E}}]_{\boldsymbol{e}}$;
- If Opponent initiates a new thread in the inner implication by $[a'_{\mathscr{W}\mathscr{W}}]_{\ell\underline{0}\boldsymbol{g}+\int\underline{0}\hbar\ell\underline{0}\boldsymbol{f}+\int\underline{0}\hbar\boldsymbol{e}}$, then fix_A copies it and launches a new thread in the outer implication by performing the move $[a'_{\mathscr{E}\mathscr{W}}]_{\ell\underline{0}\ell\underline{1}\boldsymbol{g}++\int\underline{1}\hbar\ell\underline{1}\boldsymbol{f}++\int\underline{1}\int\underline{0}\hbar\boldsymbol{e}}$ with the pointer towards the justifier of the second last move of the current P-view;
- If Opponent makes a move $[a''_{\mathscr{W}\mathscr{W}}]_{\ell\underline{0}\boldsymbol{g}+\int\underline{0}\hbar\ell\underline{0}\boldsymbol{f}+\int\underline{0}\hbar\boldsymbol{e}}$ (resp. $[a''_{\mathscr{E}\mathscr{W}}]_{\ell\underline{0}\int\underline{0}\hbar\boldsymbol{e}}$, $[a''_{\mathscr{E}\mathscr{W}}]_{\ell\underline{0}\ell\underline{1}\boldsymbol{g}++\int\underline{1}\hbar\ell\underline{1}\boldsymbol{f}++\int\underline{1}\int\underline{0}\hbar\boldsymbol{e}}$, $[a''_{\mathscr{E}}]_{\boldsymbol{e}}$) in an existing thread, then fix_A copies it and performs the next move $[a''_{\mathscr{E}\mathscr{W}}]_{\ell\underline{0}\ell\underline{1}\boldsymbol{g}++\int\underline{1}\hbar\ell\underline{1}\boldsymbol{f}++\int\underline{1}\int\underline{0}\hbar\boldsymbol{e}}$ (resp. $[a''_{\mathscr{E}}]_{\boldsymbol{e}}$, $[a''_{\mathscr{W}\mathscr{W}}]_{\ell\underline{0}\boldsymbol{g}+\int\underline{0}\hbar\ell\underline{0}\boldsymbol{f}+\int\underline{0}\hbar\boldsymbol{e}}$, $[a''_{\mathscr{E}\mathscr{W}}]_{\ell\underline{0}\int\underline{0}\hbar\boldsymbol{e}}$) in the dual thread with the pointer towards the third last move of the P-view.

A typical play by fix_A is depicted in Figure 4.16, which corresponds to Figure 4.6.

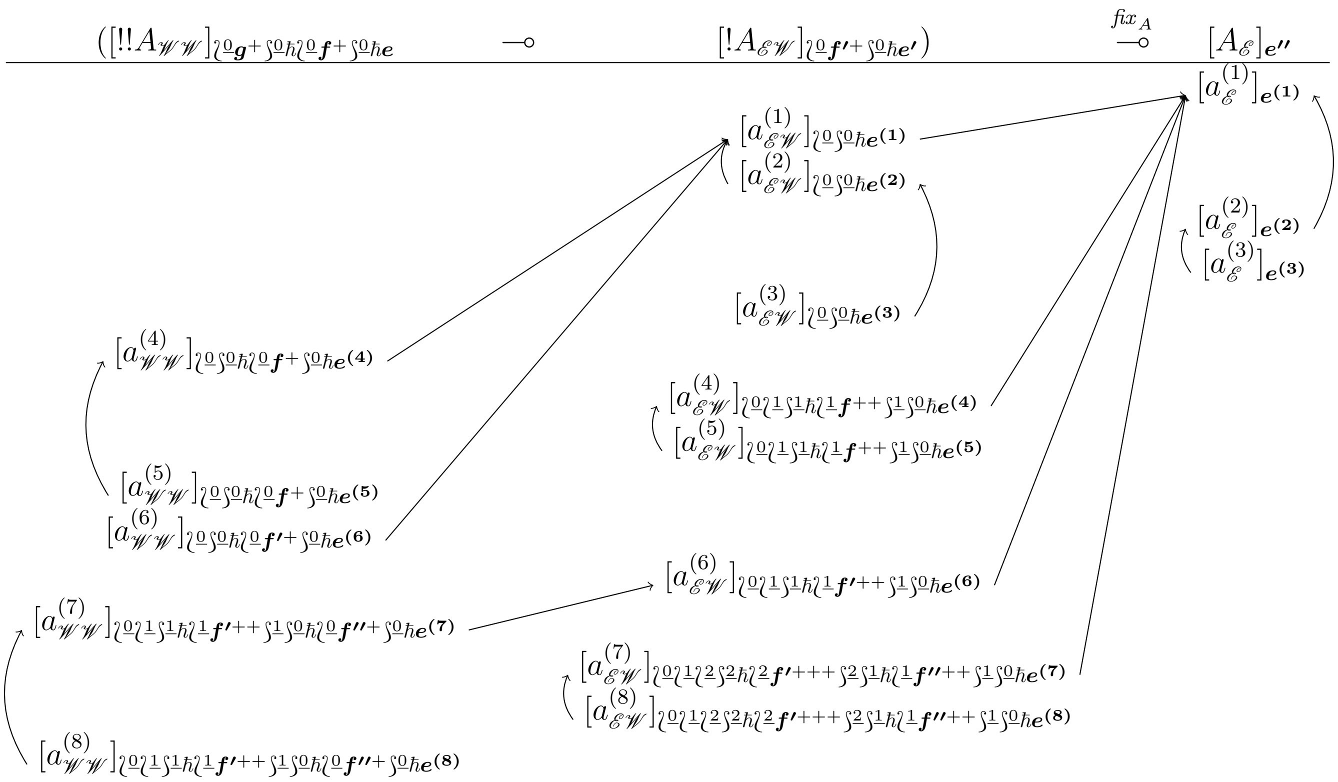


Figure 4.16: A play by $\mathit{fix}_A : (A \Rightarrow A) \Rightarrow A$ revisited.

Next, let us see by simple examples how the constructions on dynamic strategies in Section 3.3.5 are formalized.

Example 4.3.34. Consider the tensor $\mathit{succ} \otimes \mathit{pred} : [!\mathcal{N}_{\mathscr{W}\mathscr{W}}]_{\wr^{\underline{0}}\boldsymbol{e}+\int^{\underline{0}}\hbar} \otimes [!\mathcal{N}_{\mathscr{E}\mathscr{W}}]_{\wr^{\underline{0}}\boldsymbol{e'}+\int^{\underline{0}}\hbar} \multimap [\mathcal{N}_{\mathscr{W}\mathscr{E}}] \otimes [\mathcal{N}_{\mathscr{E}\mathscr{E}}]$. A typical play by $\mathit{succ} \otimes \mathit{pred}$ is depicted in Figure 4.17.

Similarly, consider the pairing $\langle \mathit{succ}, \mathit{pred} \rangle : [!\mathcal{N}_{\mathscr{W}}]_{\wr^{\underline{0}}\boldsymbol{e}+\int^{\underline{0}}\hbar} \multimap [\mathcal{N}_{\mathscr{W}\mathscr{E}}]\&[\mathcal{N}_{\mathscr{E}\mathscr{E}}]$. Some typical plays by $\langle \mathit{succ}, \mathit{pred} \rangle$ are described in Figure 4.18.

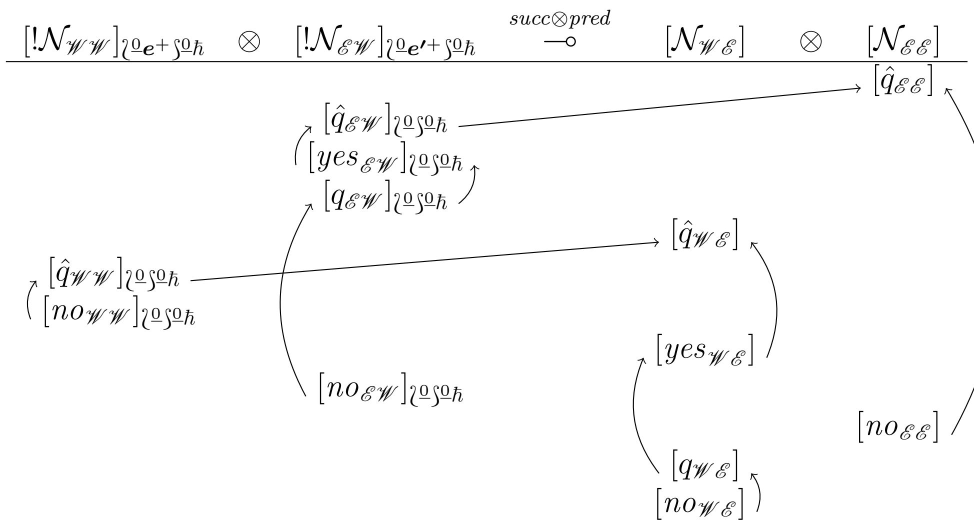


Figure 4.17: A typical play by the tensor *succ* $\otimes$ *pred* : $\mathcal{N} \otimes \mathcal{N} \multimap \mathcal{N} \otimes \mathcal{N}$ revisited.

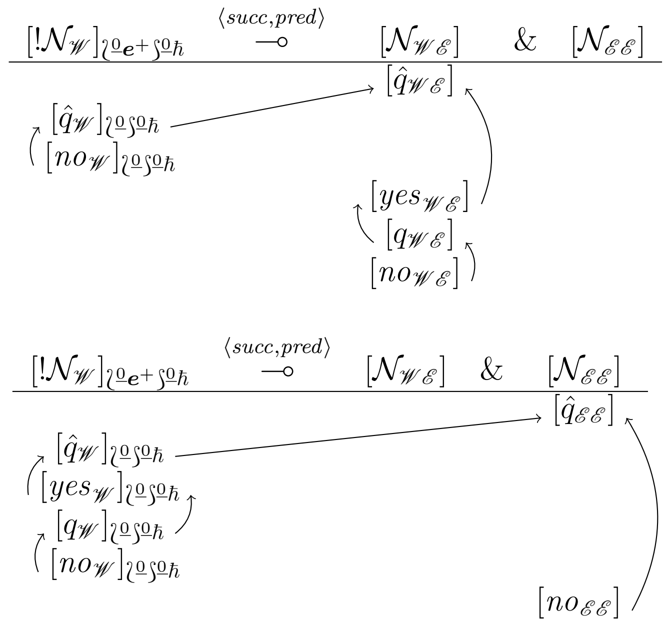


Figure 4.18: Typical plays by the pairing $\langle \mathit{succ}, \mathit{pred} \rangle : {!\mathcal{N}} \multimap \mathcal{N} \& \mathcal{N}$ revisited.

Example 4.3.35. Given a dynamic strategy $\phi : G$ such that $\mathcal{H}^{\omega}(G) \trianglelefteq {!A} \multimap {!B}$ for some normalized dynamic games A and B, the promotion $\phi^{\dagger} : G^{\dagger}$ of ϕ is given by:

$$\phi^{\dagger} \overset{\mathrm{df.}}{=} \{ \boldsymbol{s} \in \mathscr{L}_{G^{\dagger}} \mid \forall \boldsymbol{g} \in \mathcal{T}.\, \boldsymbol{s} \upharpoonright \boldsymbol{g} \in \phi \}$$

where $\boldsymbol{s} \upharpoonright \boldsymbol{g}$ is the j-subsequence of $\boldsymbol{s}$ that consists of moves of the form $[(b, \mathscr{E})]_{\wr\underline{0}\boldsymbol{g}+\int\underline{0}\hbar\boldsymbol{e}}$ with $[b]_{\boldsymbol{e}} \in M_B$, $[(a, \mathscr{W})]_{\wr\underline{0}\wr\underline{1}\boldsymbol{g}++\int\underline{1}\hbar\wr\underline{1}\boldsymbol{f}++\int\underline{1}\int\underline{0}\hbar\boldsymbol{e}}$ with $[a]_{\boldsymbol{e}} \in M_A$, or $[(m, \mathscr{S})]_{\wr\underline{0}\boldsymbol{g}+\int\underline{0}\hbar\boldsymbol{e}}$ with $[m]_{\boldsymbol{e}} \in M_G^{\mathsf{Int}}$, which are respectively changed into $[(b, \mathscr{E})]_{\boldsymbol{e}}$, $[(a, \mathscr{W})]_{\wr\underline{0}\boldsymbol{f}+\int\underline{0}\hbar\boldsymbol{e}}$ and $[m]_{\boldsymbol{e}}$.

For instance, consider the promotion $succ^{\dagger} : [!\mathcal{N}_{\mathscr{W}}]_{\wr \underline{0} \boldsymbol{e} + \int \underline{0} \hbar} \multimap [!\mathcal{N}_{\mathscr{E}}]_{\wr \underline{0} \boldsymbol{e'} + \int \underline{0} \hbar}$. A typical play by $succ^{\dagger}$ is depicted in Figure 4.19, in which there are two threads, and $succ^{\dagger}$ behaves as $succ$ in both of the threads.

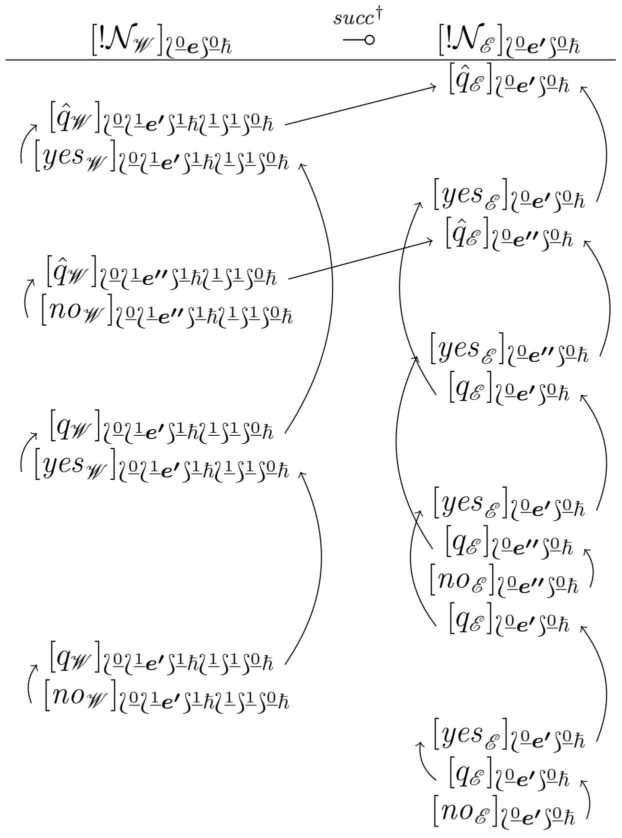


Figure 4.19: A typical play by the promotion $succ^{\dagger} : !\mathcal{N} \multimap !\mathcal{N}$ revisited.

Example 4.3.36. Consider the concatenation $succ^{\dagger} \ddagger pred : ([!\mathcal{N}_{\mathscr{W}}]_{\wr \underline{0} \boldsymbol{f} + \int \underline{0} \hbar} \multimap [!\mathcal{N}_{\mathscr{E}\mathscr{S}}]_{\wr \underline{0} \boldsymbol{e} + \int \underline{0} \hbar}) \ddagger ([!\mathcal{N}_{\mathscr{W}\mathscr{N}}]_{\wr \underline{0} \boldsymbol{e} + \int \underline{0} \hbar} \multimap [\mathcal{N}_{\mathscr{E}}])$. A typical play by $succ^{\dagger} \ddagger pred$ is described in Figure 4.20.

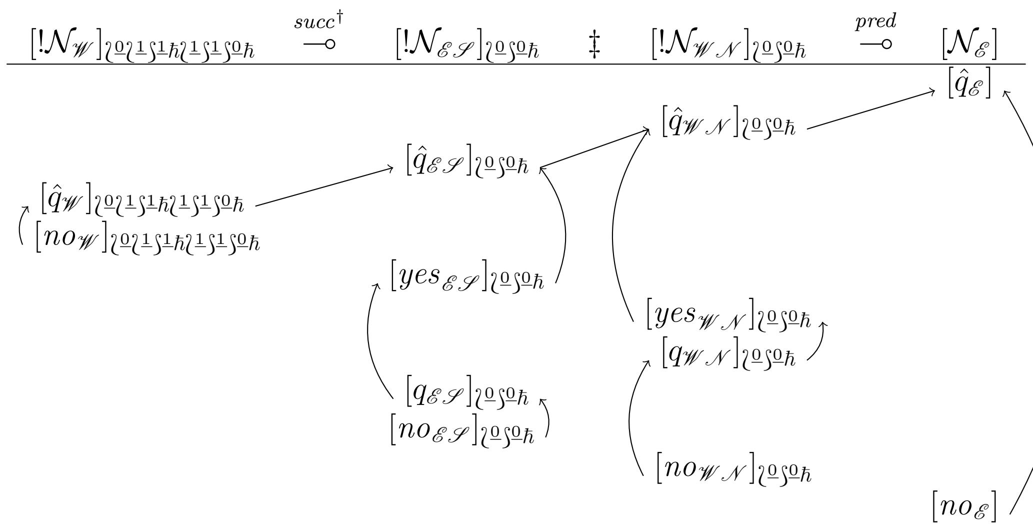

$[!\mathcal{N}_{\mathscr{W}}]_{\wr \underline{0} \wr \underline{1} \wr \underline{1} \hbar \wr \underline{1} \wr \underline{1} \wr \underline{0} \hbar}$
$succ^{\dagger}$
$[!\mathcal{N}_{\mathscr{E}\mathscr{S}}]_{\wr \underline{0} \wr \underline{0} \hbar}$
$\ddagger$
$[!\mathcal{N}_{\mathscr{W}\mathscr{N}}]_{\wr \underline{0} \wr \underline{0} \hbar}$
$pred$
$[\mathcal{N}_{\mathscr{E}}]$
$[\hat{q}_{\mathscr{E}}]$
$[\hat{q}_{\mathscr{W}\mathscr{N}}]_{\wr \underline{0} \wr \underline{0} \hbar}$
$[\hat{q}_{\mathscr{E}\mathscr{S}}]_{\wr \underline{0} \wr \underline{0} \hbar}$
$[\hat{q}_{\mathscr{W}}]_{\wr \underline{0} \wr \underline{1} \wr \underline{1} \hbar \wr \underline{1} \wr \underline{1} \wr \underline{0} \hbar}$
$[no_{\mathscr{W}}]_{\wr \underline{0} \wr \underline{1} \wr \underline{1} \hbar \wr \underline{1} \wr \underline{1} \wr \underline{0} \hbar}$
$[yes_{\mathscr{E}\mathscr{S}}]_{\wr \underline{0} \wr \underline{0} \hbar}$
$[yes_{\mathscr{W}\mathscr{N}}]_{\wr \underline{0} \wr \underline{0} \hbar}$
$[q_{\mathscr{W}\mathscr{N}}]_{\wr \underline{0} \wr \underline{0} \hbar}$
$[q_{\mathscr{E}\mathscr{S}}]_{\wr \underline{0} \wr \underline{0} \hbar}$
$[no_{\mathscr{E}\mathscr{S}}]_{\wr \underline{0} \wr \underline{0} \hbar}$
$[no_{\mathscr{W}\mathscr{N}}]_{\wr \underline{0} \wr \underline{0} \hbar}$
$[no_{\mathscr{E}}]$

Figure 4.20: A typical play by the concatenation *succ*† ‡ *pred* revisited.

Let us skip currying of dynamic strategies is formalized as it is rather trivial. Note that we have described how the inductive constructions of elements of the set $\mathcal{DPCF}$ (Definition 4.2.9) are formalized. In other words, we have implemented 'tags' for all the dynamic strategies that model PCF.

4.4 Viable Strategies

Having formalized 'tags' for dynamic games and strategies in the previous sections, we are now ready to introduce a novel notion of 'effective computability' or *viability* of strategies, and show that viable *dynamic* strategies subsume all the elements of the set $\mathcal{DPCF}$ (Definition 4.2.9), and therefore they are *PCF-complete.*

Remark. Most of the concepts introduced below make sense not only for dynamic games and strategies but also for conventional ones, and thus we often refer to the latter, rather than the former, indicating that the concepts are defined for both. Nevertheless, the main theorem (Theorem 4.4.18) holds only for the dynamic variant.

4.4.1 Viable Strategies

The idea of viable strategies is as follows. First, it seems necessary for strategies to refer only to a *bounded* number of previous moves in each odd-length position of the ambient game (to calculate the next P-move) since the set of all odd-length positions of the game can be infinite, e.g., consider the dynamic game $\mathcal{N}$ (Definition 4.3.10).[5]

[5]This is analogous to TMs which look at only one cell of an infinite tape at a time.

Here, we assume by the axiom D (Definition 4.3.9) that it is 'effective' for Player to compute internal O-moves[6], and therefore we shall not consider such computation for brevity. We then focus on *innocent* strategies (Definition 2.3.7) as a means to narrow down previous moves to be concerned with.[7] In fact, as already shown by Lemma 4.2.12, every dynamic strategy that models a term of PCF (Definition 4.2.9) only needs to read off at most the *last three moves* of each P-view and the *state* (Definition 4.2.11) of the P-view, which are 'effectively obtainable' in an informal sense. Thus, it remains to formulate how these dynamic strategies may 'effectively' compute the next P-move from the last three moves and the state.

Then, since the set $\pi_1(M_G)$ is finite for any dynamic game G (Definition 4.3.9), innocent strategies that are *finitary* in the sense that their partial function representations or *view functions* [100, 14] are finite seem sufficient. However, fixed-point strategies (Example 4.3.33) in the set $\mathcal{DPCF}$ (Definition 4.2.9) need to initiate new threads *unboundedly many times*; also, we need promotion (Example 4.3.35) for the construction $(\iota : J, \kappa : K) \mapsto \iota^\dagger \ddagger \kappa : J^\dagger \ddagger K$ of $\mathcal{DPCF}$, in which *unboundedly many outer tags occur*. Thus, finitary dynamic strategies are not strong enough.

Then, how can we define a stronger notion of 'effectivity' of the next P-move from (a finite number of) previous moves *solely in terms of games and strategies*? Our solution is to define a strategy $\sigma : G$ to be 'effective' or *viable* if it is 'describable' by a finitary strategy, called an *instruction strategy*. To state it more precisely, we define for each move $m = [m']_{\boldsymbol{e}}$ of a game G, where $\boldsymbol{e} = e_1.e_2 \dots e_k$, the strategy $\underline{m}$ on a suitable game $\mathcal{G}(M_G)$, called the *instruction game* of G, that plays as $\hat{q}.\mathsf{m}'.q.\mathsf{e}_1.q.\mathsf{e}_2 \dots q.\mathsf{e}_\mathsf{k}.q.\checkmark$, where each non-initial occurrence in the position is justified by the previous occurrence (n.b., we omit this trivial justification below). Note that the font difference between the moves is just for clarity. In this manner, the strategy $\underline{m} : \mathcal{G}(M_G)$ encodes the move m. Then, viability of strategies is defined more precisely as follows: A strategy $\sigma : G$ is defined to be *viable* iff 1. it is representable by a partial function $(\boldsymbol{s}, m_3, m_2, m_1) \mapsto m$, where m_1, m_2 and m_3 are the last move, the second last move and the third last move of the current P-view, respectively, and $\boldsymbol{s}$ is the state of the P-view; and 2. it is 'implementable' by a finitary strategy $\mathcal{A}(\sigma)^{\circledS}_{\boldsymbol{s}} : \mathcal{G}(M_G) \& \mathcal{G}(M_G) \& \mathcal{G}(M_G) \Rightarrow \mathcal{G}(M_G)$, called an *instruction strategy* for σ, in the sense that $\mathcal{A}(\sigma)^{\circledS}_{\boldsymbol{s}} \circ \langle \underline{m_3}^T, \underline{m_2}^T, \underline{m_1}^T \rangle^\dagger = \underline{m}^T : T \Rightarrow \mathcal{G}(M_G)$ for all input-output pairs

[6]Recall that we need to consider such computation as internal moves are 'invisible' to Opponent.

[7]Of course, there might be another means to 'effectively' eliminate irrelevant moves from the history of previous moves; in fact, we need more than P-views in order to model languages with *states* [14] (n.b., they are different from states of P-views), which is left as future work.

$(\boldsymbol{s}, m_3, m_2, m_1) \mapsto m$ of σ, where $\langle \underline{m_3}^T, \underline{m_2}^T, \underline{m_1}^T \rangle : T \Rightarrow \mathcal{G}(M_G) \& \mathcal{G}(M_G) \& \mathcal{G}(M_G)$ is the ternary pairing of the strategies $\underline{m_i}^T : T \Rightarrow \mathcal{G}(M_G)$ (i = 1, 2, 3).[8]

For instance, consider the dynamic strategy $\mathit{min} : \mathcal{N} \Rightarrow \mathcal{N}$ that implements the minimization $(\mathbb{N} \rightharpoonup \mathbb{N}) \rightharpoonup \mathbb{N}$ that maps a given partial function $f : \mathbb{N} \rightharpoonup \mathbb{N}$ to the least $n \in \mathbb{N}$ such that $f(n) = 0$ if it exists. For simplicity, the domain of *min* is $!\mathcal{N}$, not $\mathcal{N} \Rightarrow \mathcal{N}$; *min* informs an input computation of an input number $n \in \mathbb{N}$ by the outer tag $[\_]_{\wr \underline{0} \ell^n \int \underline{0} \hbar}$. As described in Figure 4.21 below, *min* simply investigates if the input computation gives back zero just by checking the first digit (*yes* or *no*) and adds *yes* to the output if the input computation gives back non-zero (i.e., if the first digit is *yes*). Note that *min* only needs to refer to at most the last three moves of each odd-length position (n.b., in this case, they are the last three moves of the P-view of the position as well). The point here is that the number of outer tags that *min* has to manipulate is *unbounded* (though the manipulation is very simple), and therefore the strategy is not finitary; however, it is easy to show that the strategy is viable as follows. First, its partial function representation can be given by the following *infinitary* table (n.b., n for $[\_]_{\wr \underline{0} \ell^n \int \underline{0} \hbar}$ given below ranges over *all* natural numbers):

$$(\Box, \Box, [\hat{q}_{\mathscr{E}}]) \mapsto [\hat{q}_{\mathscr{W}}]_{\wr \underline{0} \int \underline{0} \hbar} \mid ([\hat{q}_{\mathscr{E}}], [\hat{q}_{\mathscr{W}}]_{\wr \underline{0} \int \underline{0} \hbar}, [yes_{\mathscr{W}}]_{\wr \underline{0} \int \underline{0} \hbar}) \mapsto [yes_{\mathscr{E}}] \mid$$
$$([q_{\mathscr{E}}], [q_{\mathscr{W}}]_{\wr \underline{0} \ell^n \int \underline{0} \hbar}, [yes_{\mathscr{W}}]_{\wr \underline{0} \ell^n \int \underline{0} \hbar}) \mapsto [yes_{\mathscr{E}}] \mid ([yes_{\mathscr{W}}]_{\wr \underline{0} \ell^n \int \underline{0} \hbar}, [yes_{\mathscr{E}}], [q_{\mathscr{E}}]) \mapsto [q_{\mathscr{W}}]_{\wr \underline{0} \ell^{n+1} \int \underline{0} \hbar} \mid$$
$$([q_{\mathscr{E}}], [q_{\mathscr{W}}]_{\wr \underline{0} \ell^n \int \underline{0} \hbar}, [no_{\mathscr{W}}]_{\wr \underline{0} \ell^n \int \underline{0} \hbar}) \mapsto [no_{\mathscr{E}}] \mid ([\hat{q}_{\mathscr{E}}], [\hat{q}_{\mathscr{W}}]_{\wr \underline{0} \int \underline{0} \hbar}, [no_{\mathscr{W}}]_{\wr \underline{0} \int \underline{0} \hbar}) \mapsto [no_{\mathscr{E}}]$$

This high-level computational process is 'implementable' by an instruction strategy $\mathcal{A}(\mathit{min})^{\circledS} : \mathcal{G}(M_{\mathcal{N} \Rightarrow \mathcal{N}}) \& \mathcal{G}(M_{\mathcal{N} \Rightarrow \mathcal{N}}) \& \mathcal{G}(M_{\mathcal{N} \Rightarrow \mathcal{N}}) \Rightarrow \mathcal{G}(M_{\mathcal{N} \Rightarrow \mathcal{N}})$, which computes as in Figure 4.22, where the rather trivial pointers and 'tags' $(\_)^{[i]}$ for the game $\mathcal{G}(M_{\mathcal{N} \Rightarrow \mathcal{N}})^{[0]} \& \mathcal{G}(M_{\mathcal{N} \Rightarrow \mathcal{N}})^{[1]} \& \mathcal{G}(M_{\mathcal{N} \Rightarrow \mathcal{N}})^{[2]} \Rightarrow \mathcal{G}(M_{\mathcal{N} \Rightarrow \mathcal{N}})^{[3]}$ are omitted. Clearly, there is a *finite* table for $\mathcal{A}(\mathit{min})^{\circledS}$ that maps the last k-moves in the P-view of each odd-length position to the next P-move for some fixed $k \in \mathbb{N}$, proving viability of *min*. Observe in particular how the infinitary manipulation of outer tags by *min* is reduced to a finitary computation by $\mathcal{A}(\mathit{min})^{\circledS}$.

This example illustrates why we need *viable* (not only finitary) dynamic strategies for Turing completeness, where recall that minimization (in the general form) or an equivalent construction is vital to construct all partial recursive functions [46]. Also, it should be clear now why we have to employ composition of strategies *without* hiding, i.e., concatenation: An instruction strategy for the concatenation $\phi^\dagger \ddagger \phi : J^\dagger \ddagger K$ of strategies $\phi : J$ and $\psi : K$ can be obtained simply as the disjoint union of instruction

[8]Recall that given a strategy $\alpha : A$ we define another $\alpha^T : T \Rightarrow A$ that is α up to tags.

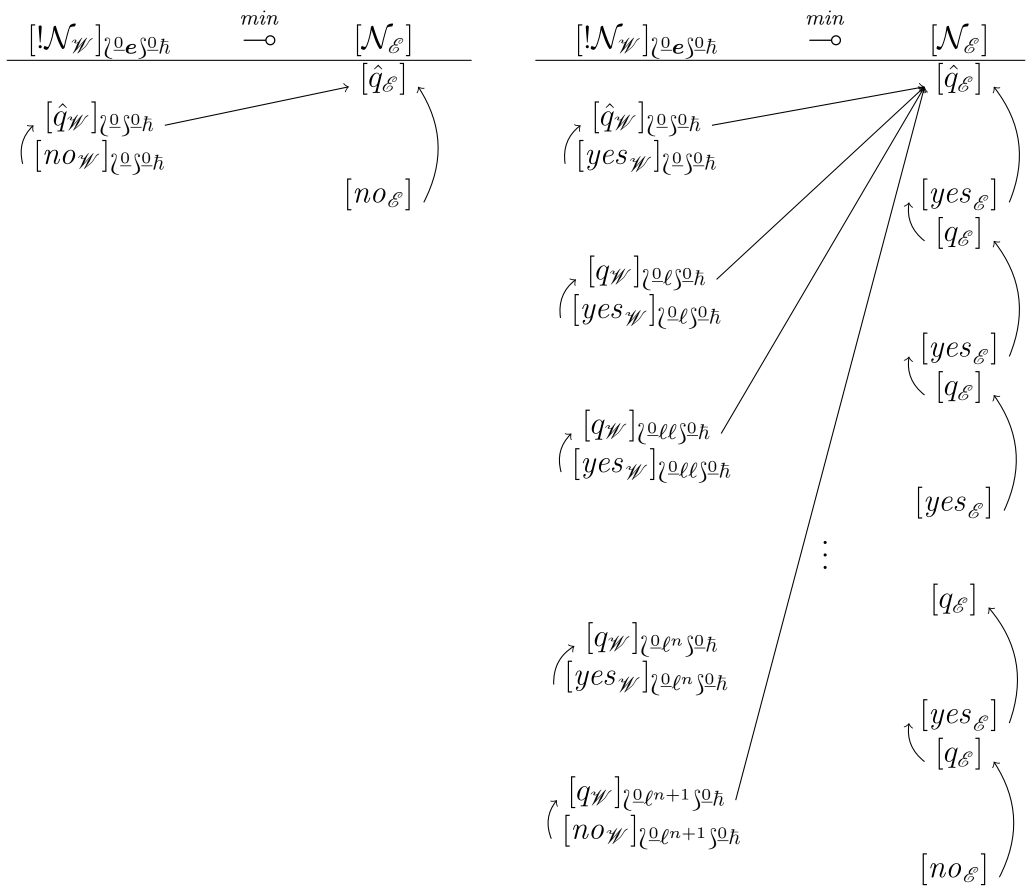


Figure 4.21: The minimization strategy $min : \mathcal{N} \Rightarrow \mathcal{N}$.

$$\begin{array}{ccccccc}
\mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}}) & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}}) & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}}) & \overset{\mathcal{A}(min)^{\circledS}}{\Rightarrow} & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}}) \\
\hline
 & & & & & & \hat{q} \\
 & & & & \hat{q} & & \\
 & & & & \hat{\mathsf{q}}_{\mathsf{E}} & & \\
 & & & & & & \hat{\mathsf{q}}_{\mathsf{W}} \\
 & & & & & & q \\
 & & & & & & \checkmark
\end{array}$$

$$\begin{array}{ccccccc}
\mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}}) & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}}) & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}}) & \overset{\mathcal{A}(min)^{\circledS}}{\Rightarrow} & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}}) \\
\hline
 & & & & & & \hat{q} \\
 & & & & \hat{q} & & \\
 & & & & \mathsf{yes}_{\mathsf{W}}(\mathsf{no}_{\mathsf{W}}) & & \\
 & & & & & & \mathsf{yes}_{\mathsf{E}}(\mathsf{no}_{\mathsf{E}}) \\
 & & & & & & q \\
 & & & & & & \checkmark
\end{array}$$

$$\begin{array}{ccccccc}
\mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}}) & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}}) & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}}) & \overset{\mathcal{A}(min)^{\circledS}}{\Rightarrow} & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}}) \\
\hline
 & & & & & & \hat{q} \\
 & & & & \hat{q} & & \\
 & & & & \mathsf{q}_{\mathsf{E}} & & \\
 & & & & & & \mathsf{q}_{W} \\
 & & & & & & q \\
\hat{q} & & & & & & \\
\mathsf{yes}_{\mathsf{W}} & & & & & & \\
q & & & & & & \\
\ell & & & & & & \\
 & & & & & & \ell \\
 & & & & & & q \\
q & & & & & & \\
\ell & & & & & & \\
 & & & & & & \ell \\
 & & & \vdots & & & \\
 & & & & & & q \\
q & & & & & & \\
\ell & & & & & & \\
 & & & & & & \ell \\
 & & & & & & q \\
q & & & & & & \\
\checkmark & & & & & & \\
 & & & & & & \ell \\
 & & & & & & q \\
 & & & & & & \checkmark
\end{array}$$

Figure 4.22: An instruction strategy $\mathcal{A}(min)^{\circledS}$ for $min : \mathcal{N} \Rightarrow \mathcal{N}$.

strategies for $\phi^\dagger$ and ψ (see the proof of Theorem 4.4.13 for the detail), but it is not possible for composition *with* hiding (in fact, there is no obvious way to construct an instruction strategy for the composition of $\phi^\dagger$ and ψ with hiding).

Having explained the idea of viability of strategies, let us proceed to make it mathematically precise. Let us first formalize *instruction games* (Definition 4.4.1):

Notation. We assign a symbol m to each element $m \in \pi_1(M_G)$ for a game G, where we assume that these symbols are *pairwise distinct* since the set $\pi_1(M_G)$ is finite, and define $\mathsf{Sym}(\pi_1(M_G)) \overset{\text{df.}}{=} \{\mathsf{m} \mid m \in \pi_1(M_G)\}$. Also, we assign symbols to elements of outer tags by the map $\mathscr{C} : \ell \mapsto \prime,\ \hbar \mapsto \sharp,\ \wr \mapsto \langle,\ \int \mapsto \rangle\ \lhd \mapsto \ll,\ \rhd \mapsto \gg$; let $\mathsf{T} \overset{\text{df.}}{=} \{\prime, \sharp, \langle, \rangle, \ll, \gg\}$. Technically, such a symbolic representation is not necessary at all; it is rather for conceptual clarity and readability.

Definition 4.4.1 (Instruction games)**.** The ***instruction game*** on a game G is the dynamic game $\mathcal{G}(M_G)$ given by:

- $M_{\mathcal{G}(M_G)} \overset{\text{df.}}{=} \{[\hat{q}], [q], [\Box], [\checkmark]\} \cup \{[\mathsf{m}] \mid \mathsf{m} \in \mathsf{Sym}(\pi_1(M_G))\} \cup \{[\mathsf{e}] \mid \mathsf{e} \in \mathsf{T}\}$, where the elements of $M_{\mathcal{G}(M_G)}$ are assumed to be pairwise distinct;
- $\lambda_{\mathcal{G}(M_G)} : [\hat{q}] \mapsto \mathsf{OQ}0,\ [q] \mapsto \mathsf{OQ}0,\ [\Box] \mapsto \mathsf{PA}0,\ [\checkmark] \mapsto \mathsf{PA}0,\ [\mathsf{m}] \mapsto \mathsf{PA}0,\ [\mathsf{e}] \mapsto \mathsf{PA}0$;
- $\vdash_{\mathcal{G}(M_G)} \overset{\text{df.}}{=} \{(\star, [\hat{q}]), ([\hat{q}], [\Box])\} \cup \{([\hat{q}], [\mathsf{m}]) \mid \mathsf{m} \in \mathsf{Sym}(\pi_1(M_G))\} \cup \{([\mathsf{m}], [q]) \mid \mathsf{m} \in \mathsf{Sym}(\pi_1(M_G))\} \cup \{([q], [\mathsf{e}]) \mid \mathsf{e} \in \mathsf{T}\} \cup \{([\mathsf{e}], [q]) \mid \mathsf{e} \in \mathsf{T}\} \cup \{([q], [\checkmark])\}$;
- $P_{\mathcal{G}(M_G)} \overset{\text{df.}}{=} \mathsf{Pref}(\{[\hat{q}][\Box]\} \cup \{[\hat{q}][\mathsf{m}][q][\mathscr{C}(e_1)][q][\mathscr{C}(e_2)] \dots [q][\mathscr{C}(e_k)][q][\checkmark] \mid \mathsf{m} \in \mathsf{Sym}(\pi_1(M_G)), e_1 e_2 \dots e_k \in \mathcal{T}\})$, where each non-initial occurrence is justified by the previous occurrence;
- $\boldsymbol{s} \simeq_{\mathcal{G}(M_G)} \boldsymbol{t} \overset{\text{df.}}{\Leftrightarrow} \boldsymbol{s} = \boldsymbol{t}$;
- $\Delta_{\mathcal{G}(M_G)} \overset{\text{df.}}{=} \emptyset$.

Convention. We loosely call the dynamic games of the form $\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G)$, where G is some game, *instruction games* as well.

The positions $[\hat{q}][\mathsf{m}][q][\mathscr{C}(e_1)][q][\mathscr{C}(e_2)] \dots [q][\mathscr{C}(e_k)][q][\checkmark]$ and $[\hat{q}].[\Box]$ of $\mathcal{G}(M_G)$ are to represent the move $[m]_{e_1 e_2 \dots e_k} \in M_G$ and 'no element', respectively. Note that pointers in these positions are trivial, and thus we usually omit them.

Notation. Let G be a game, and $[m]_{\boldsymbol{e}} \in M_G$ with $\boldsymbol{e} = e_1 e_2 \dots e_k$. We write $\underline{[m]_{\boldsymbol{e}}}$ for the strategy on $\mathcal{G}(M_G)$ given by:

$$\underline{[m]_{\boldsymbol{e}}} \overset{\text{df.}}{=} \mathsf{Pref}([\hat{q}][\mathsf{m}][q][\mathscr{C}(e_1)][q][\mathscr{C}(e_2)] \dots [q][\mathscr{C}(e_k)][q][\checkmark])^{\mathsf{Even}}$$

and similarly, we define $\underline{[\Box]} \stackrel{\mathrm{df.}}{=} \mathsf{Pref}(\{[\hat{q}][\Box]\})^{\mathsf{Even}} : \mathcal{G}(M_G)$. Given a finite sequence $\boldsymbol{s} = [m_l]_{\boldsymbol{e}^{(l)}}[m_{l-1}]_{\boldsymbol{e}^{(l-1)}} \dots [m_1]_{\boldsymbol{e}^{(1)}} \in M_G^*$ and a natural number $n \geqslant l$, we define $\underline{\boldsymbol{s}}_n \stackrel{\mathrm{df.}}{=} \langle \underbrace{\underline{[\Box]}, \dots, \underline{[\Box]}}_{n-l}, \underline{[m_l]_{\boldsymbol{e}^{(l)}}}, \underline{[m_{l-1}]_{\boldsymbol{e}^{(l-1)}}}, \dots, \underline{[m_1]_{\boldsymbol{e}^{(1)}}} \rangle : \mathcal{G}(M_G)^n \stackrel{\mathrm{df.}}{=} \underbrace{\mathcal{G}(M_G) \& \mathcal{G}(M_G) \dots \& \mathcal{G}(M_G)}_{n}$, where the n-ary pairing and product abbreviate the $(n-1)$-times iteration of the usual (binary) ones from the left. Given any strategy $\sigma : \mathcal{G}(M_G)$, we define $\mathcal{M}(\sigma)$ to be the unique move in M_G such that $\underline{\mathcal{M}(\sigma)} = \sigma$ if it exists, and undefined otherwise.

Next, let us introduce a new concept, called *m-views* (Definition 4.4.3), for the calculation of 'relevant' (and finite) part of outer tags represented by moves occurring in a position of an instruction game, which is necessary because the number of all outer tags is infinite.

Definition 4.4.2 (Depths in instruction games). Let G be a game, and $[m]_{\boldsymbol{e}} \in M_G$, where $\boldsymbol{e} = e_1 e_2 \dots e_k$. The ***depth*** of an occurrence of $\langle$ or $\rangle$ in a position $[\hat{q}][\mathsf{m}][q][\mathscr{C}(e_1)][q][\mathscr{C}(e_2)] \dots [q][\mathscr{C}(e_k)][q][\checkmark]$ of the instruction game $\mathcal{G}(M_G)$ is the depth of the corresponding occurrence of $\wr$ or $\int$ in $\boldsymbol{e}$ (Definition 4.3.5).

Definition 4.4.3 (M-views). Let G be a game, $\boldsymbol{s} \in P_{\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G)}$ and $d \in \mathbb{N}$. The ***matching view*** (***m-view***) $[\![\boldsymbol{s}]\!]_G^d$ of $\boldsymbol{s}$ up to depth d is the subsequence of $\boldsymbol{s}$ that consists of occurrences of $\langle$ or $\rangle$ of depth $\leqslant d$.

Note that it is clearly 'effective' to calculate the m-view of a given position $\boldsymbol{s}$ of an instruction game $P_{\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G)}$ with the help of the explicit representation of depths of occurrences of $\langle$ or $\rangle$ in $\boldsymbol{s}$, which is inherited from Definition 4.3.5.

Let us now formalize finitary strategies $\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G)$, called *instruction strategies* (Definition 4.4.5), that 'describe' the low-level computational process of a given strategy $\sigma : G$, whose finite tables are called *st-algorithms* (Definition 4.4.4).

Notation. Given a finite sequence $\boldsymbol{s} = x_k x_{k-1} \dots x_1$ and a natural number $l \in \mathbb{N}$, we define $\boldsymbol{s} \downharpoonleft l \stackrel{\mathrm{df.}}{=} \begin{cases} \boldsymbol{s} & \text{if } l \geqslant k; \\ x_l x_{l-1} \dots x_1 & \text{otherwise.} \end{cases}$ A function $f : \pi_1(M_G) \to \{\top, \bot\}$, where G is a game, and $\top$ and $\bot$ are arbitrarily fixed, distinct symbols, induces another function $f^\star : M_G^* \to \pi_1(M_G)^*$ defined by $f^\star([m_k]_{\boldsymbol{e}^{(k)}}[m_{k-1}]_{\boldsymbol{e}^{(k-1)}} \dots [m_1]_{\boldsymbol{e}^{(1)}}) \stackrel{\mathrm{df.}}{=} m_{i_l} m_{i_{l-1}} \dots m_{i_1}$, where $l \leqslant k$ and $m_{i_l} m_{i_{l-1}} \dots m_{i_1}$ is the subsequence of $m_k m_{k-1} \dots m_1$ that consists of m_{i_j} such that $f(m_{i_j}) = \top$ for $j = 1, 2, \dots, l$.

Definition 4.4.4 (St-algorithms). An ***st-algorithm*** $\mathcal{A}$ on a game G, written $\mathcal{A} :: G$, is a family $\mathcal{A} = (\mathcal{A}_{\boldsymbol{m}}, |\mathcal{A}_{\boldsymbol{m}}|, \|\mathcal{A}_{\boldsymbol{m}}\|)_{\boldsymbol{m} \in \mathcal{S}_{\mathcal{A}}}$ of triples of a *finite* partial function $\mathcal{A}_{\boldsymbol{m}} : \partial_{\boldsymbol{m}}(P^{\mathsf{Odd}}_{\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G) \& \boldsymbol{2}}) \rightharpoonup M_{\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G) \& \boldsymbol{2}}$ and natural numbers $|\mathcal{A}_{\boldsymbol{m}}|, \|\mathcal{A}_{\boldsymbol{m}}\| \in \mathbb{N}$, called the ***view-scope*** and the ***mate-scope*** of $\mathcal{A}_{\boldsymbol{m}}$, respectively, where:

- $\mathcal{S}_{\mathcal{A}} \subseteq \pi_1(M_G)^*$ is a *finite* set, whose elements are called ***states***;
- $\partial_{\boldsymbol{m}}(\boldsymbol{t}x) \stackrel{\mathrm{df.}}{=} (\boldsymbol{t}x \upharpoonright |\mathcal{A}_{\boldsymbol{m}}|, [\![\boldsymbol{t}x]\!]_G^{\|\mathcal{A}_{\boldsymbol{m}}\|})$ for all $\boldsymbol{t}x \in P^{\mathsf{Odd}}_{\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G) \& \boldsymbol{2}}$

equipped with the ***query (function)*** $\mathcal{Q}_{\mathcal{A}} : \pi_1(M_G) \to \{\top, \bot\}$ that satisfies:

- (Q) $[m]_e \in M_G^{\mathsf{Init}} \Rightarrow \mathcal{Q}_{\mathcal{A}}(m) = \top$.

We will see that states of st-algorithms assigned to dynamic strategies for PCF formalize states of P-views (Definition 4.2.11), and so the axiom Q makes sense.

Remark. Note that it does not make difference if each st-algorithm $\mathcal{A} :: G$ focuses on the P-views of each $\boldsymbol{t}x \in P^{\mathsf{Odd}}_{\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G) \& \boldsymbol{2}}$ for $\lceil \boldsymbol{t}x \rceil = \boldsymbol{t}x$ by the trivial pointers.

Definition 4.4.5 (Instruction strategies)**.** Given a game G, an st-algorithm $\mathcal{A} :: G$ and a state $\boldsymbol{m} \in \mathcal{S}_{\mathcal{A}}$, the ***instruction strategy*** $\mathcal{A}^{\circledS}_{\boldsymbol{m}}$ of $\mathcal{A}$ at $\boldsymbol{m}$ is the strategy on the game $\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G) \& \boldsymbol{2}$ defined by:

$$\mathcal{A}^{\circledS}_{\boldsymbol{m}} \stackrel{\mathrm{df.}}{=} \{\boldsymbol{\epsilon}\} \cup \{\boldsymbol{t}xy \in P^{\mathsf{Even}}_{\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G) \& \boldsymbol{2}} \mid \boldsymbol{t} \in \mathcal{A}^{\circledS}_{\boldsymbol{m}}, \mathcal{A}_{\boldsymbol{m}} \circ \partial_{\boldsymbol{m}}(\boldsymbol{t}x) \downarrow, y = \mathcal{A}_{\boldsymbol{m}} \circ \partial_{\boldsymbol{m}}(\boldsymbol{t}x)\}.$$

where the justifier of y in $\boldsymbol{t}xy$ is the obvious, canonical one.

Convention. Given an st-algorithm $\mathcal{A} :: G$ and a state $\boldsymbol{m} \in \mathcal{S}_{\mathcal{A}}$, the instruction strategy $\mathcal{A}^{\circledS}_{\boldsymbol{m}}$ has to specify pointers in P_G, which in the present work are always either the last or the third last occurrence, or the justifier of the second occurrence of the P-view of each odd-length position of G (as we shall see); it is why $\mathcal{A}^{\circledS}_{\boldsymbol{m}}$ is on the game $\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G) \& \boldsymbol{2}$, not the instruction game $\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G)$, so that it may specify the ternary choice on the justifiers in the component game $\boldsymbol{2}$ (by *tt*, *ff* or 'no answer'). However, since justifiers in P_G occurring in this paper are all obvious ones, we henceforth regard $\mathcal{A}_{\boldsymbol{m}}$ and $\mathcal{A}^{\circledS}_{\boldsymbol{m}}$ as $\mathcal{A}_{\boldsymbol{m}} : \partial_{\boldsymbol{m}}(P^{\mathsf{Odd}}_{\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G)}) \rightharpoonup M_{\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G)}$ and $\mathcal{A}^{\circledS}_{\boldsymbol{m}} : \mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G)$, respectively, keeping the justifiers *implicit*.

Remark. Since an st-algorithm $\mathcal{A} :: G$ refers to m-views *only occasionally*, we regard each $\mathcal{A}_{\boldsymbol{m}}$ as a partial function $\{\boldsymbol{t}x \upharpoonright |\mathcal{A}_{\boldsymbol{m}}| \mid \boldsymbol{t}x \in P^{\mathsf{Odd}}_{\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G)}\} \rightharpoonup M_{\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G)}$ in most cases. Accordingly, $\mathcal{A}^{\circledS}_{\boldsymbol{m}}$ is mostly a strategy on the game $\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G)$ whose partial function representation $\mathcal{A}_{\boldsymbol{m}}$ is finite.

Thus, an instruction strategy is a strategy on the game $\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G)$, where G is a game, that is *finitary* in the sense that it is representable by a finite partial function, and therefore it is clearly 'effective' in an informal sense. Note that the number 3 on $\mathcal{G}(M_G)^3$ comes from Lemma 4.2.12. As already mentioned, our idea is to utilize such an instruction strategy as a 'description' of a strategy on G, which may be 'effectively' read-off by Player:

Definition 4.4.6 (Realizability)**.** The strategy $\mathsf{st}(\mathcal{A}) : G$ ***realized*** by an st-algorithm $\mathcal{A} :: G$ is defined by:

$$\mathsf{st}(\mathcal{A}) \stackrel{\mathrm{df.}}{=} \{\boldsymbol{\epsilon}\} \cup \{\boldsymbol{s}ab \in P_G^{\mathsf{Even}} \mid \boldsymbol{s} \in \mathsf{st}(\mathcal{A}), \mathcal{A}^{\circledS}(\lceil \boldsymbol{s}a \rceil \upharpoonright 3) \downarrow, b = \mathcal{A}^{\circledS}(\lceil \boldsymbol{s}a \rceil \upharpoonright 3)\}$$

where $\mathcal{A}^{\circledS}(\lceil \boldsymbol{s}a \rceil \upharpoonright 3) \stackrel{\mathrm{df.}}{\simeq} \mathcal{M}(\mathcal{A}^{\circledS}_{\mathcal{Q}^\star_{\mathcal{A}}(\lceil \boldsymbol{s}a \rceil)} \circ (\underline{\lceil \boldsymbol{s}a \rceil \upharpoonright 3}_3)^\dagger)$, $\mathcal{A}^{\circledS}(\lceil \boldsymbol{s}a \rceil \upharpoonright 3) \downarrow$ presupposes $\mathcal{Q}^\star_{\mathcal{A}}(\lceil \boldsymbol{s}a \rceil) \in \mathcal{S}_{\mathcal{A}}$, and the justifier of b in $\boldsymbol{s}ab$ is the occurrence specified by $\mathcal{A}^{\circledS}_{\mathcal{Q}^\star_{\mathcal{A}}(\lceil \boldsymbol{s}a \rceil)}$ (see the convention below Definition 4.4.5).

Clearly, $\mathcal{A} :: G \Rightarrow \mathsf{st}(\mathcal{A}) : G$ holds. We are now ready to define the central notion of the present work, namely, 'effective computability' of strategies:

Definition 4.4.7 (Viability of strategies)**.** A strategy $\sigma : G$ is ***viable*** if there exists an st-algorithm $\mathcal{A} :: G$ that realizes σ, i.e., $\mathsf{st}(\mathcal{A}) = \sigma$.

That is, a strategy $\sigma : G$ is viable if there is a finitary strategy on $\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G)$ that 'describes' the computation of σ. The terms *realize* and *realizability* come from mathematical logic, in which a *realizer* refers to some computational information that 'realizes' the constructive truth of a mathematical statement [179].

Given an st-algorithm $\mathcal{A} :: G$ that realizes a strategy $\sigma : G$, P may 'effectively' execute $\mathcal{A}$ to compute σ roughly as follows:

1. Given $\boldsymbol{s}a \in P_G^{\mathsf{Odd}}$, P calculates the current state $\boldsymbol{m} \stackrel{\mathrm{df.}}{=} \mathcal{Q}^\star_{\mathcal{A}}(\lceil \boldsymbol{s}a \rceil)$ and the last (up to) three moves $\lceil \boldsymbol{s}a \rceil \upharpoonright 3$ in the P-view by the query function $\mathcal{Q}_{\mathcal{A}}$ and pointers;[9] if $\boldsymbol{m} \notin \mathcal{S}_{\mathcal{A}}$, then she stops, i.e., the next move is undefined;
2. Otherwise, she composes $(\underline{\lceil \boldsymbol{s}a \rceil \upharpoonright 3}_3)^\dagger$ with $\mathcal{A}^{\circledS}_{\boldsymbol{m}}$, calculating $\mathcal{A}^{\circledS}_{\boldsymbol{m}} \circ (\underline{\lceil \boldsymbol{s}a \rceil \upharpoonright 3}_3)^\dagger$;
3. Finally, she reads off the next move $\mathcal{M}(\mathcal{A}^{\circledS}_{\boldsymbol{m}} \circ (\underline{\lceil \boldsymbol{s}a \rceil \upharpoonright 3}_3)^\dagger)$ (and its justifier) and performes that move.

For conceptual clarity, here we assume that P may write down moves $[m]_{\boldsymbol{e}}$ in P-views as $[\mathsf{m}]_{\mathscr{C}^*(\boldsymbol{e})}$ and execute strategies on instruction games symbolically on her 'scratch pad', and also she may read off strategies $\sigma : \mathcal{G}(M_G)$ on the 'scratch pad' and reproduce them as moves $\mathcal{M}(\sigma) \in M_G$. This procedure is clearly 'effective' in an informal sense, which is our justification of the notion of viable strategies.

Note that there are two kinds of processes in viable strategies $\sigma : G$. The first one is the process of σ per se whose atomic steps are $(\boldsymbol{s}a \in P_G^{\mathsf{Odd}}) \mapsto \boldsymbol{s}a.\sigma(\lceil \boldsymbol{s}a \rceil)$,

[9]Here, we assume that pointers are represented by *directed edges* between moves occurring in positions, and Player may 'effectively' calculate P-views by tracing the edges. Note also that the set $\mathcal{S}_{\mathcal{A}}$ and the map $\mathcal{Q}_{\mathcal{A}}$ are both *finite*, and therefore Step 1 is certainly 'effective'.

and the second one is the process of its st-algorithm $\mathcal{A}$ whose atomic steps are $\boldsymbol{t}x \in P^{\mathsf{Odd}}_{\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G)} \mapsto \boldsymbol{t}x.\mathcal{A}_{\boldsymbol{m}} \circ \partial_{\boldsymbol{m}}(\boldsymbol{t}x)$, where $\boldsymbol{m}$ is the current state. The former is abstract and high-level, while the latter is symbolic and low-level. In this manner, we have achieved a mathematical formulation of high-level and low-level computational processes and 'effective computability' of the former in terms of the latter.

Henceforth, in order to establish Theorem 4.4.13 later, we shall focus on the following st-algorithms:

Definition 4.4.8 (Standard st-algorithms)**.** An st-algorithm $\mathcal{A} :: G$ is ***standard*** iff:

1. The symbol $\square$ does not occur in $\mathcal{A}_{\boldsymbol{m}}$ for any $\boldsymbol{m} \in \mathcal{S}_{\mathcal{A}}$;
2. It does not refer to any input outer tag when it computes an inner element, i.e., if $\hat{q}^{[3]}.\boldsymbol{s}.\mathsf{n}^{[3]} \in \mathcal{A}^{\circledS}_{\boldsymbol{m}} : \mathcal{G}(M_G)^{[0]} \& \mathcal{G}(M_G)^{[1]} \& \mathcal{G}(M_G)^{[2]} \Rightarrow \mathcal{G}(M_G)^{[3]}$, where $\boldsymbol{m} \in \mathcal{S}_{\mathcal{A}}$ and $n \in \pi_1(M_G)$, then $q^{[0]}$, $q^{[1]}$ and $q^{[2]}$ do not occur in $\boldsymbol{s}$;
3. If it refers to an input outer tag, then it must belongs to the last move of the current P-view of G, i.e., if q occurs as a P-move in some $\boldsymbol{s} \in \mathcal{A}^{\circledS}_{\boldsymbol{m}}$, where $\boldsymbol{m} \in \mathcal{S}_{\mathcal{A}}$, then the 'tag' on the move is $(\_)^{[2]}$.

Convention. Henceforth, ***st-algorithms*** refer to *standard* ones by default. Of course, standardness is closed under all constructions on st-algorithms (see the proof of Theorem 4.4.13), and st-algorithms given below are all standard.

Notation. We write tags for $\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G)$ *informally*: We omit outer tags and write only inner tags informally by numbers, e.g., $\mathcal{G}(M_G)^{[0]} \& \mathcal{G}(M_G)^{[1]} \& \mathcal{G}(M_G)^{[2]} \Rightarrow \mathcal{G}(M_G)^{[3]}$, $\hat{q}^{[0]}$, $\mathsf{m}^{[1]}$, $\sharp^{[2]}$, etc. For an outer tag $\boldsymbol{e} = e_1.e_2 \dots e_k \in \mathcal{T}$, we write $\boldsymbol{e}^{[i]}$ for the sequence $e_1^{[i]}.q^{[i]}.e_2.q^{[i]} \dots e_{k-1}.q^{[i]}.e_k^{[i]}$. We write $\langle @d$ and $\rangle @d$, where $d \in \mathbb{N}$, for the expressions $\langle \ll \prime^d \gg$ and $\rangle \ll \prime^d \gg$, respectively.

Example 4.4.9. The zero strategy $\mathit{zero}_A : [A_{\mathscr{W}}]_{\wr \underline{0}_{\boldsymbol{e}} + \int \underline{0}_{\hbar}} \Rightarrow [\mathcal{N}_{\mathscr{E}}]$ on any normalized dynamic game A (Example 4.3.27) is viable as we may give an st-algorithm $\mathcal{A}(\mathit{zero}_A)$ that realizes zero_A: $\mathcal{Q}_{\mathcal{A}(\mathit{zero}_A)}(m) \stackrel{\mathrm{df.}}{=} \begin{cases} \top & \text{if } m = \hat{q}_{\mathscr{E}}; \\ \bot & \text{otherwise} \end{cases}$, $\mathcal{S}_{\mathcal{A}(\mathit{zero}_A)} \stackrel{\mathrm{df.}}{=} \{\hat{q}_{\mathscr{E}}\}$, $|\mathcal{A}(\mathit{zero}_A)_{\hat{q}_{\mathscr{E}}}| \stackrel{\mathrm{df.}}{=} 3$, $\|\mathcal{A}(\mathit{zero}_A)_{\hat{q}_{\mathscr{E}}}\| \stackrel{\mathrm{df.}}{=} 0$ and $\mathcal{A}(\mathit{zero}_A)_{\hat{q}_{\mathscr{E}}} : \hat{q}^{[3]} \mapsto \mathsf{no_E}^{[3]} \mid \hat{q}^{[3]}\mathsf{no_E}^{[3]}q^{[3]} \mapsto \checkmark^{[3]}$. Then, the instruction strategy $\mathcal{A}(\mathit{zero}_A)^{\circledS}_{\hat{q}_{\mathscr{E}}}$ plays as:

$$\begin{array}{ccccccc} \mathcal{G}(M_{A \Rightarrow \mathcal{N}})^{[0]} & \& & \mathcal{G}(M_{A \Rightarrow \mathcal{N}})^{[1]} & \& & \mathcal{G}(M_{A \Rightarrow \mathcal{N}})^{[2]} & \stackrel{\mathcal{A}(\mathit{zero}_A)^{\circledS}_{\hat{q}_{\mathscr{E}}}}{\Rightarrow} & \mathcal{G}(M_{A \Rightarrow \mathcal{N}})^{[3]} \\ \hline &&&&&& \hat{q}^{[3]} \\ &&&&&& \mathsf{no_E}^{[3]} \\ &&&&&& q^{[3]} \\ &&&&&& \checkmark^{[3]} \end{array}$$

Clearly, $\mathsf{st}(\mathcal{A}(zero_A)) = zero_A$, proving viability of $zero_A$, and $\mathcal{A}(zero_A)$ is standard.

Example 4.4.10. Let us give an st-algorithm $\mathcal{A}(succ)$ that realizes the successor strategy $succ : [\mathcal{N}_{\mathscr{W}}]_{\wr\underline{0}\boldsymbol{e}+\int\underline{0}\hbar} \Rightarrow [\mathcal{N}_{\mathscr{E}}]$ (Example 4.3.28) by defining $\mathcal{Q}_{\mathcal{A}(succ)}$ similarly to $\mathcal{Q}_{\mathcal{A}(zero)}$, $\mathcal{S}_{\mathcal{A}(succ)} \overset{\mathrm{df.}}{=} \{\hat{q}_{\mathscr{E}}\}$, $|\mathcal{A}(succ)_{\hat{q}_{\mathscr{E}}}| \overset{\mathrm{df.}}{=} 17$, $\|\mathcal{A}(succ)_{\hat{q}_{\mathscr{E}}}\| \overset{\mathrm{df.}}{=} 0$, and $\mathcal{A}(succ)_{\hat{q}_{\mathscr{E}}}$ as:

$\hat{q}^{[3]} \mapsto \hat{q}^{[2]} \mid \hat{q}^{[3]}\hat{q}^{[2]}\hat{\mathsf{q}}_{\mathsf{E}}^{[2]} \mapsto \hat{\mathsf{q}}_{\mathsf{E}}^{[3]} \mid \hat{q}^{[3]}\hat{q}^{[2]}\hat{\mathsf{q}}_{\mathsf{E}}^{[2]}\hat{\mathsf{q}}_{\mathsf{W}}^{[3]}q^{[3]} \mapsto \langle^{[3]} \mid \hat{q}^{[3]}\hat{q}^{[2]}\hat{\mathsf{q}}_{\mathsf{E}}^{[2]}\hat{\mathsf{q}}_{\mathsf{W}}^{[3]}q^{[3]}\langle^{[3]}q^{[3]} \mapsto \ll^{[3]} \mid$
$\hat{q}^{[3]}\hat{q}^{[2]}\hat{\mathsf{q}}_{\mathsf{E}}^{[2]}\hat{\mathsf{q}}_{\mathsf{W}}^{[3]}q^{[3]}\langle^{[3]}q^{[3]} \ll^{[3]} q^{[3]} \mapsto \gg^{[3]} \mid \hat{q}^{[3]}\hat{q}^{[2]}\hat{\mathsf{q}}_{\mathsf{E}}^{[2]}\hat{\mathsf{q}}_{\mathsf{W}}^{[3]}q^{[3]}\langle^{[3]}q^{[3]} \ll^{[3]} q^{[3]} \gg^{[3]} q^{[3]} \mapsto \rangle^{[3]} \mid$
$\hat{q}^{[3]}\hat{q}^{[2]}\hat{\mathsf{q}}_{\mathsf{E}}^{[2]}\hat{\mathsf{q}}_{\mathsf{W}}^{[3]}q^{[3]}\langle^{[3]}q^{[3]} \ll^{[3]} q^{[3]} \gg^{[3]} q^{[3]}\rangle^{[3]}q^{[3]} \mapsto \ll^{[3]} \mid$
$\hat{q}^{[3]}\hat{q}^{[2]}\hat{\mathsf{q}}_{\mathsf{E}}^{[2]}\hat{\mathsf{q}}_{\mathsf{W}}^{[3]}q^{[3]}\langle^{[3]}q^{[3]} \ll^{[3]} q^{[3]} \gg^{[3]} q^{[3]}\rangle^{[3]}q^{[3]} \ll^{[3]} q^{[3]} \mapsto \gg^{[3]} \mid$
$\hat{q}^{[3]}\hat{q}^{[2]}\hat{\mathsf{q}}_{\mathsf{E}}^{[2]}\hat{\mathsf{q}}_{\mathsf{W}}^{[3]}q^{[3]}\langle^{[3]}q^{[3]} \ll^{[3]} q^{[3]} \gg^{[3]} q^{[3]}\rangle^{[3]}q^{[3]} \ll^{[3]} q^{[3]} \gg^{[3]} q^{[3]} \mapsto \sharp^{[3]} \mid$
$\hat{q}^{[3]}\hat{q}^{[2]}\hat{\mathsf{q}}_{\mathsf{E}}^{[2]}\hat{\mathsf{q}}_{\mathsf{W}}^{[3]}q^{[3]}\langle^{[3]}q^{[3]} \ll^{[3]} q^{[3]} \gg^{[3]} q^{[3]}\rangle^{[3]}q^{[3]} \ll^{[3]} q^{[3]} \gg^{[3]} q^{[3]}\sharp^{[3]}q^{[3]} \mapsto \checkmark^{[3]} \mid$
$\hat{q}^{[3]}\hat{q}^{[2]}\mathsf{q_E}^{[2]} \mapsto \hat{q}^{[0]} \mid \hat{q}^{[3]}\hat{q}^{[2]}\mathsf{q_E}^{[2]}\hat{q}^{[0]}\mathsf{no_W}^{[0]} \mapsto \mathsf{no_E}^{[3]} \mid \hat{q}^{[3]}\hat{q}^{[2]}\mathsf{q_E}^{[2]}\hat{q}^{[0]}\mathsf{no_W}^{[0]}\mathsf{no_E}^{[3]}q^{[3]} \mapsto \checkmark^{[3]} \mid$
$\hat{q}^{[3]}\hat{q}^{[2]}\mathsf{q_E}^{[2]}\hat{q}^{[0]}\mathsf{yes_W}^{[0]} \mapsto \mathsf{q_W}^{[3]} \mid \hat{q}^{[3]}\hat{q}^{[2]}\mathsf{q_E}^{[2]}\hat{q}^{[0]}\mathsf{yes_W}^{[0]}\mathsf{q_W}^{[3]}q^{[3]} \mapsto \langle^{[3]} \mid$
$\hat{q}^{[3]}\hat{q}^{[2]}\mathsf{q_E}^{[2]}\hat{q}^{[0]}\mathsf{yes_W}^{[0]}\mathsf{q_W}^{[3]}q^{[3]}\langle^{[3]}q^{[3]} \mapsto \ll^{[3]} \mid \hat{q}^{[3]}\hat{q}^{[2]}\mathsf{q_E}^{[2]}\hat{q}^{[0]}\mathsf{yes_W}^{[0]}\mathsf{q_W}^{[3]}q^{[3]}\langle^{[3]}q^{[3]} \ll^{[3]} q^{[3]} \mapsto \gg^{[3]} \mid$
$\hat{q}^{[3]}\hat{q}^{[2]}\mathsf{q_E}^{[2]}\hat{q}^{[0]}\mathsf{yes_W}^{[0]}\mathsf{q_W}^{[3]}q^{[3]}\langle^{[3]}q^{[3]} \ll^{[3]} q^{[3]} \gg^{[3]} q^{[3]} \mapsto \rangle^{[3]} \mid$
$\hat{q}^{[3]}\hat{q}^{[2]}\mathsf{q_E}^{[2]}\hat{q}^{[0]}\mathsf{yes_W}^{[0]}\mathsf{q_W}^{[3]}q^{[3]}\langle^{[3]}q^{[3]} \ll^{[3]} q^{[3]} \gg^{[3]} q^{[3]}\rangle^{[3]}q^{[3]} \mapsto \ll^{[3]} \mid$
$\hat{q}^{[3]}\hat{q}^{[2]}\mathsf{q_E}^{[2]}\hat{q}^{[0]}\mathsf{yes_W}^{[0]}\mathsf{q_W}^{[3]}q^{[3]}\langle^{[3]}q^{[3]} \ll^{[3]} q^{[3]} \gg^{[3]} q^{[3]}\rangle^{[3]}q^{[3]} \ll^{[3]} q^{[3]} \mapsto \gg^{[3]} \mid$
$\hat{q}^{[3]}\hat{q}^{[2]}\mathsf{q_E}^{[2]}\hat{q}^{[0]}\mathsf{yes_W}^{[0]}\mathsf{q_W}^{[3]}q^{[3]}\langle^{[3]}q^{[3]} \ll^{[3]} q^{[3]} \gg^{[3]} q^{[3]}\rangle^{[3]}q^{[3]} \ll^{[3]} q^{[3]} \gg^{[3]} q^{[3]} \mapsto \sharp^{[3]} \mid$
$\hat{q}^{[3]}\hat{q}^{[2]}\mathsf{q_E}^{[2]}\hat{q}^{[0]}\mathsf{yes_W}^{[0]}\mathsf{q_W}^{[3]}q^{[3]}\langle^{[3]}q^{[3]} \ll^{[3]} q^{[3]} \gg^{[3]} q^{[3]}\rangle^{[3]}q^{[3]} \ll^{[3]} q^{[3]} \gg^{[3]} q^{[3]}\sharp^{[3]}q^{[3]} \mapsto \checkmark^{[3]} \mid$
$\hat{q}^{[3]}\hat{q}^{[2]}\mathsf{yes_W}^{[2]} \mapsto \mathsf{yes_E}^{[3]} \mid \hat{q}^{[3]}\hat{q}^{[2]}\mathsf{yes_W}^{[2]}\mathsf{yes_E}^{[3]}q^{[3]} \mapsto \checkmark^{[3]} \mid$
$\hat{q}^{[3]}\hat{q}^{[2]}\mathsf{no_W}^{[2]} \mapsto \mathsf{yes_E}^{[3]} \mid \hat{q}^{[3]}\hat{q}^{[2]}\mathsf{no_W}^{[2]}\mathsf{yes_E}^{[3]}q^{[3]} \mapsto \checkmark^{[3]}$

Consequently, $\mathcal{A}(succ)^{\circledS}_{\hat{q}_{\mathscr{E}}}$ plays as:

$\mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[0]}$	$\&$	$\mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[1]}$	$\&$	$\mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[2]}$	$\overset{\mathcal{A}(succ)^{\circledS}_{\hat{q}_{\mathscr{E}}}}{\Rightarrow}$	$\mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[3]}$
						$\hat{q}^{[3]}$
				$\hat{q}^{[2]}$		
				$\hat{\mathsf{q}}_{\mathsf{E}}^{[2]}$		
						$\hat{\mathsf{q}}_{\mathsf{W}}^{[3]}$
						$q^{[3]}$
						$\langle @0^{[3]}$
						$q^{[3]}$
						$\rangle @0^{[3]}$
						$q^{[3]}$
						$\sharp^{[3]}$
						$q^{[3]}$
						$\checkmark^{[3]}$

$$\begin{array}{ccccccc}
\mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[0]} & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[1]} & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[2]} & \overset{\mathcal{A}(succ)^{\circledS}_{\hat{q}_{\mathscr{W}}}}{\Rightarrow} & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[3]} \\ \hline
 & & & & & & \hat{q}^{[3]} \\
 & & & & \hat{q}^{[2]} & & \\
 & & & & \mathsf{q_E}^{[2]} & & \\
\hat{q}^{[0]} & & & & & & \\
\mathsf{no_W}^{[0]} & & & & & & \\
 & & & & & & \mathsf{no_E}^{[3]} \\
 & & & & & & q^{[3]} \\
 & & & & & & \checkmark^{[3]}
\end{array}$$

$$\begin{array}{ccccccc}
\mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[0]} & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[1]} & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[2]} & \overset{\mathcal{A}(succ)^{\circledS}_{\hat{q}_{\mathscr{W}}}}{\Rightarrow} & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[3]} \\ \hline
 & & & & & & \hat{q}^{[3]} \\
 & & & & \hat{q}^{[2]} & & \\
 & & & & \mathsf{q_E}^{[2]} & & \\
\hat{q}^{[0]} & & & & & & \\
\mathsf{yes_W}^{[0]} & & & & & & \\
 & & & & & & \mathsf{q_W}^{[3]} \\
 & & & & & & q^{[3]} \\
 & & & & & & \langle @0^{[3]} \\
 & & & & & & q^{[3]} \\
 & & & & & & \rangle @0^{[3]} \\
 & & & & & & q^{[3]} \\
 & & & & & & \sharp^{[3]} \\
 & & & & & & q^{[3]} \\
 & & & & & & \checkmark^{[3]}
\end{array}$$

$$\begin{array}{ccccccc}
\mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[0]} & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[1]} & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[2]} & \overset{\mathcal{A}(succ)^{\circledS}_{\hat{q}_{\mathscr{W}}}}{\Rightarrow} & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[3]} \\ \hline
 & & & & & & \hat{q}^{[3]} \\
 & & & & \hat{q}^{[2]} & & \\
 & & & & \mathsf{x_W}^{[2]} & & \\
 & & & & & & \mathsf{yes_E}^{[3]} \\
 & & & & & & q^{[3]} \\
 & & & & & & \checkmark^{[3]}
\end{array}$$

where x is either yes or no. Clearly, $\mathsf{st}(\mathcal{A}(succ)) = succ$, and therefore we have establishes viability of $succ$. Also, it is straightforward to see that $\mathcal{A}(succ)$ is standard.

Example 4.4.11. Similarly to *zero* and *succ*, we may give an st-algorithm $\mathcal{A}(pred)$ that realizes the predecessor strategy $pred : [!\mathcal{N}_{\mathscr{W}}]_{\wr \underline{0} e + \wr \underline{0} \hbar} \multimap [\mathcal{N}_{\mathscr{E}}]$ (Example 4.3.29) as follows. We define the states, the view- and the mate-scopes, and the query function of $\mathcal{A}(pred)$ to be the same as those of $\mathcal{A}(succ)$. At this point, it should suffice to depict diagrams for $\mathcal{A}(pred)^{\circledS}_{\hat{q}_{\mathscr{E}}}$ since it is now clear that there is a finite table for the partial function $\mathcal{A}(pred)_{\hat{q}_{\mathscr{E}}}$:

$$\begin{array}{ccccccc}
\mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[0]} & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[1]} & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[2]} & \overset{\mathcal{A}(pred)^{\circledS}_{\hat{q}_{\mathscr{E}}}}{\Rightarrow} & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[3]} \\
\hline
 & & & & & & \hat{q}^{[3]} \\
 & & & & \hat{q}^{[2]} & & \\
 & & & & \hat{\mathsf{q}}_{\mathsf{E}}^{[2]}\ (\mathsf{q}_{\mathsf{E}}{}^{[2]}) & & \\
 & & & & & & \hat{\mathsf{q}}_{\mathsf{W}}^{[3]}\ (\mathsf{q}_{\mathsf{W}}{}^{[3]}) \\
 & & & & & & q^{[3]} \\
 & & & & & & \langle @0^{[3]} \\
 & & & & & & q^{[3]} \\
 & & & & & & \rangle @0^{[3]} \\
 & & & & & & q^{[3]} \\
 & & & & & & \sharp^{[3]} \\
 & & & & & & q^{[3]} \\
 & & & & & & \checkmark^{[3]}
\end{array}$$

$$\begin{array}{ccccccc}
\mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[0]} & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[1]} & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[2]} & \overset{\mathcal{A}(pred)^{\circledS}_{\hat{q}_{\mathscr{E}}}}{\Rightarrow} & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[3]} \\
\hline
 & & & & & & \hat{q}^{[3]} \\
 & & & & \hat{q}^{[2]} & & \\
 & & & & \mathsf{no}_{\mathsf{W}}{}^{[2]} & & \\
 & & & & & & \mathsf{no}_{\mathsf{E}}{}^{[3]} \\
 & & & & & & q^{[3]} \\
 & & & & & & \checkmark^{[3]}
\end{array}$$

$$\begin{array}{ccccccc}
\mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[0]} & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[1]} & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[2]} & \overset{\mathcal{A}(pred)^{\circledS}_{\hat{q}_{\mathscr{E}}}}{\Rightarrow} & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathcal{N}})^{[3]} \\
\hline
 & & & & & & \hat{q}^{[3]} \\
 & & & & \hat{q}^{[2]} & & \\
 & & & & \mathsf{yes}_{\mathsf{W}}{}^{[2]} & & \\
 & & \hat{q}^{[1]} & & & & \\
 & & \hat{\mathsf{q}}_{\mathsf{W}}^{[1]} & & & & \\
 & & & & & & \mathsf{q}_{\mathsf{W}}{}^{[3]} \\
 & & & & & & q^{[3]} \\
 & & & & & & \langle @0^{[3]} \\
 & & & & & & q^{[3]} \\
 & & & & & & \rangle @0^{[3]} \\
 & & & & & & q^{[3]} \\
 & & & & & & \sharp^{[3]} \\
 & & & & & & q^{[3]} \\
 & & & & & & \checkmark^{[3]}
\end{array}$$

$$\begin{array}{ccccccc}
\mathcal{G}(M_{\mathcal{N} \Rightarrow \mathcal{N}})^{[0]} & \& & \mathcal{G}(M_{\mathcal{N} \Rightarrow \mathcal{N}})^{[1]} & \& & \mathcal{G}(M_{\mathcal{N} \Rightarrow \mathcal{N}})^{[2]} & \overset{\mathcal{A}(\mathit{pred})^{\circledS}_{\hat{q}_{\mathscr{E}}}}{\Rightarrow} & \mathcal{G}(M_{\mathcal{N} \Rightarrow \mathcal{N}})^{[3]} \\ \hline
&&&&&& \hat{q}^{[3]} \\
&&&& \hat{q}^{[2]} && \\
&&&& \mathsf{yes_W}^{[2]} && \\
&& \hat{q}^{[1]} &&&& \\
&& \mathsf{q_W}^{[1]} &&&& \\
&&&&&& \mathsf{yes_E}^{[3]} \\
&&&&&& q^{[3]} \\
&&&&&& \checkmark^{[3]}
\end{array}$$

Clearly $\mathsf{st}(\mathcal{A}(\mathit{pred})) = \mathit{pred}$, establishing viability of *pred*. Also, it is easy to see that $\mathcal{A}(\mathit{pred})$ is standard.

Example 4.4.12. Consider the fixed-point strategy $\mathit{fix}_A : ([A_{\mathscr{W}\mathscr{W}}]_{\wr^{\underline{0}}\boldsymbol{g}+\int^{\underline{0}}\hbar\wr^{\underline{0}}\boldsymbol{f}+\int^{\underline{0}}\hbar\boldsymbol{e}} \Rightarrow [A_{\mathscr{E}\mathscr{W}}]_{\wr^{\underline{0}}\boldsymbol{g'}+\int^{\underline{0}}\hbar\boldsymbol{e'}}) \Rightarrow [A_{\mathscr{E}}]_{\boldsymbol{e''}}$ on any normalized game A (Example 4.3.33). Clearly, fix_A is not finitary for the calculation of outer tags (as already pointed out before). It is, however, viable (for any A), which is perhaps surprising to many readers. Here, let us just informally describe an st-algorithm $\mathcal{A}(\mathit{fix}_A)$ that realizes fix_A as a preparation for Section 4.4.2. Let $\mathcal{Q}_{\mathcal{A}(\mathit{fix}_A)}(m) = \top \overset{\mathrm{df.}}{\Leftrightarrow} m \in \pi_1(M^{\mathsf{Init}}_{(A \Rightarrow A) \Rightarrow A})$ and $\mathcal{S}_{\mathcal{A}(\mathit{fix}_A)} \overset{\mathrm{df.}}{=} \pi_1(M^{\mathsf{Init}}_{(A \Rightarrow A) \Rightarrow A})$. Since $\mathcal{A}(\mathit{fix}_A)_m$ does not depend on m, fix an arbitrary state $m \in \mathcal{S}_{\mathcal{A}(\mathit{fix}_A)}$. The instruction strategy $\mathcal{A}(\mathit{fix}_A)^{\circledS}_m$ computes by a case analysis on the rightmost component of input strategies on $\mathcal{G}(M_{(A \Rightarrow A) \Rightarrow A})^3$ for $\mathcal{A}(\mathit{fix}_A)^{\circledS}_m$ (which corresponds to the last occurrence of the current P-view of $(A \Rightarrow A) \Rightarrow A$):

- If the rightmost component is of the form $(\underline{[a_{\mathscr{E}}]_{\boldsymbol{f}}})^{\dagger}$, then $\mathcal{A}(\mathit{fix}_A)^{\circledS}_m$ recognizes it by the inner tag $\mathscr{E}$, and calculates the next move $[a_{\mathscr{E}\mathscr{W}}]_{\wr^{\underline{0}}\int^{\underline{0}}\hbar\boldsymbol{f}}$ once and for all for the inner element $a_{\mathscr{E}\mathscr{W}}$ and 'digit-by-digit' for the outer tag $\wr^{\underline{0}}\int^{\underline{0}}\hbar\boldsymbol{f}$ (by producing $\wr^{\underline{0}}\int^{\underline{0}}\hbar$ and then copying $\boldsymbol{f}$);

- If the rightmost component is of the form $(\underline{[a_{\mathscr{E}\mathscr{W}}]_{\wr^{\underline{0}}\int^{\underline{0}}\hbar\boldsymbol{f}}})^{\dagger}$, then the leftmost component (which corresponds to the third last occurrence of the current P-view of $(A \Rightarrow A) \Rightarrow A$) is of the form $(\underline{[a'_{\mathscr{E}}]_{\boldsymbol{f}}})^{\dagger}$, and thus $\mathcal{A}(\mathit{fix}_A)^{\circledS}_m$ recognizes this case by the inner tags $\mathscr{E}\mathscr{W}$ and $\mathscr{E}$, and calculates the next move $[a_{\mathscr{E}}]_{\boldsymbol{f}}$ once and for all for the inner element $a_{\mathscr{E}}$ and 'digit-by-digit' for the outer tag $\boldsymbol{f}$ (by ignoring $\wr^{\underline{0}}\int^{\underline{0}}\hbar$ and copying $\boldsymbol{f}$);

- If the rightmost component is of the form $(\underline{[a_{\mathscr{E}\mathscr{W}}]_{\wr^{\underline{0}}\wr^{\underline{1}}\boldsymbol{e'}++\int^{\underline{1}}\hbar\wr^{\underline{1}}\boldsymbol{e}++\int^{\underline{1}}\int^{\underline{0}}\hbar\boldsymbol{f}}})^{\dagger}$, then $\mathcal{A}(\mathit{fix}_A)^{\circledS}_m$ calculates the next move $[a_{\mathscr{W}\mathscr{W}}]_{\wr^{\underline{0}}\boldsymbol{e'}+\int^{\underline{0}}\hbar\wr^{\underline{0}}\boldsymbol{e}+\int^{\underline{0}}\hbar\boldsymbol{f}}$ similarly to the above case yet with the help of m-views for the calculation of the outer tag (see Example 4.4.17 for the detail);

- If the rightmost component is of the form $(\underline{[a_{\mathscr{W}\mathscr{W}}]_{\wr\underline{0}\boldsymbol{e'}+\int\underline{0}\hbar\wr\underline{0}\boldsymbol{e}+\int\underline{0}\hbar\boldsymbol{f}}})^{\dagger}$, then $\mathcal{A}(\mathit{fix}_A)^{\circledS}_m$ calculates the next move $[a_{\mathscr{E}\mathscr{W}}]_{\wr\underline{0}\wr\underline{1}\boldsymbol{e'}++\int\underline{1}\hbar\wr\underline{1}\boldsymbol{e}\int\underline{1}\int\underline{0}\hbar\boldsymbol{f}}$ in a similar manner to the above case again with the help of m-views (see Example 4.4.17 for the detail).

We now turn to a key theorem, which states that viability of *dynamic* strategies are preserved under constructions in Section 3.3.5:

Theorem 4.4.13 (Preservation of viability). *Viable dynamic strategies are closed under tensor $\otimes$, pairing $\langle \_, \_ \rangle$, promotion $(\_)^{\dagger}$, concatenation $\ddagger$, currying Λ and uncurrying $\Lambda^{\ominus}$ if the underlying st-algorithms are standard, where the standardness of the st-algorithms is also preserved.*

Proof. Let us first show that tensor $\otimes$ preserves viability of dynamic strategies. Let $\sigma : [A_{\mathscr{W}}]_{\boldsymbol{e}} \multimap [C_{\mathscr{E}}]_{\boldsymbol{e'}}$ and $\tau : [B_{\mathscr{W}}]_{\boldsymbol{f}} \multimap [D_{\mathscr{E}}]_{\boldsymbol{f'}}$ be viable strategies with st-algorithms $\mathcal{A}(\sigma)$ and $\mathcal{A}(\tau)$ realizing σ and τ, respectively. We have to construct an st-algorithm $\mathcal{A}(\sigma \otimes \tau)$ such that $\mathsf{st}(\mathcal{A}(\sigma \otimes \tau)) = \sigma \otimes \tau : [A_{\mathscr{W}\mathscr{W}}]_{\boldsymbol{e}} \otimes [B_{\mathscr{E}\mathscr{W}}]_{\boldsymbol{f}} \multimap [C_{\mathscr{W}\mathscr{E}}]_{\boldsymbol{e'}} \otimes [D_{\mathscr{E}\mathscr{E}}]_{\boldsymbol{f'}}$. Let us define the finite set $\mathcal{S}_{\mathcal{A}(\sigma\otimes\tau)}$ of states and the query $\mathcal{Q}_{\mathcal{A}(\sigma\otimes\tau)}$ by:

$$\begin{aligned}
\mathcal{S}_{\mathcal{A}(\sigma\otimes\tau)} &\overset{\text{df.}}{=} \{m^{(k)}_{\mathscr{W}X_k} m^{(k-1)}_{\mathscr{W}X_{k-1}} \dots m^{(1)}_{\mathscr{W}X_1} \mid m^{(k)}_{X_k} m^{(k-1)}_{X_{k-1}} \dots m^{(1)}_{X_1} \in \mathcal{S}_{\mathcal{A}(\sigma)}\} \\
&\quad \cup \{n^{(l)}_{\mathscr{E}Y_l} n^{(l-1)}_{\mathscr{E}Y_{l-1}} \dots n^{(1)}_{\mathscr{E}Y_1} \mid n^{(l)}_{Y_l} n^{(l-1)}_{Y_{l-1}} \dots n^{(1)}_{Y_1} \in \mathcal{S}_{\mathcal{A}(\tau)}\} \\
\mathcal{Q}_{\mathcal{A}(\sigma\otimes\tau)} &: a_{\mathscr{W}\mathscr{W}} \mapsto \mathcal{Q}_{\mathcal{A}(\sigma)}(a_{\mathscr{W}}), b_{\mathscr{E}\mathscr{W}} \mapsto \mathcal{Q}_{\mathcal{A}(\tau)}(b_{\mathscr{W}}), c_{\mathscr{W}\mathscr{E}} \mapsto \mathcal{Q}_{\mathcal{A}(\sigma)}(c_{\mathscr{E}}), \\
&\quad d_{\mathscr{E}\mathscr{E}} \mapsto \mathcal{Q}_{\mathcal{A}(\tau)}(d_{\mathscr{E}})
\end{aligned}$$

where $X_k, X_{k-1}, \dots, X_1, Y_l, Y_{l-1}, \dots, Y_1 \in \{\mathscr{W}, \mathscr{E}\}$. Clearly, $\mathcal{Q}_{\mathcal{A}(\sigma\otimes\tau)}$ satisfies the axiom Q (Definition 4.4.4). For each $m^{(k)}_{X_k} m^{(k-1)}_{X_{k-1}} \dots m^{(1)}_{X_1} \in \mathcal{S}_{\mathcal{A}(\sigma)}$ and $n^{(l)}_{Y_l} n^{(l-1)}_{Y_{l-1}} \dots n^{(1)}_{Y_1} \in \mathcal{S}_{\mathcal{A}(\tau)}$, we construct the finite partial functions $\mathcal{A}(\sigma\otimes\tau)_{m^{(k)}_{\mathscr{W}X_k} m^{(k-1)}_{\mathscr{W}X_{k-1}} \dots m^{(1)}_{\mathscr{W}X_1}}$ and $\mathcal{A}(\sigma\otimes\tau)_{n^{(l)}_{\mathscr{E}Y_l} n^{(l-1)}_{\mathscr{E}Y_{l-1}} \dots n^{(1)}_{\mathscr{E}Y_1}}$ from $\mathcal{A}(\sigma)_{m^{(k)}_{X_k} m^{(k-1)}_{X_{k-1}} \dots m^{(1)}_{X_1}}$ and $\mathcal{A}(\tau)_{n^{(l)}_{Y_l} n^{(l-1)}_{Y_{l-1}} \dots n^{(1)}_{Y_1}}$ simply by changing symbols $\mathsf{m_X} \in \mathsf{Sym}(\pi_1(M_{A\multimap C}))$ and $\mathsf{n_Y} \in \mathsf{Sym}(\pi_1(M_{B\multimap D}))$ into $\mathsf{m_{WX}}$ and $\mathsf{n_{EY}}$ respectively in their finite tables, where the view-scopes are defined by:

$$\begin{aligned}
|\mathcal{A}(\sigma\otimes\tau)_{m^{(k)}_{\mathscr{W}X_k} m^{(k-1)}_{\mathscr{W}X_{k-1}} \dots m^{(1)}_{\mathscr{W}X_1}}| &\overset{\text{df.}}{=} |\mathcal{A}(\sigma)_{m^{(k)}_{X_k} m^{(k-1)}_{X_{k-1}} \dots m^{(1)}_{X_1}}| \\
|\mathcal{A}(\sigma\otimes\tau)_{n^{(l)}_{\mathscr{E}l} n^{(l-1)}_{\mathscr{E}Y_{l-1}} \dots n^{(1)}_{\mathscr{E}Y_1}}| &\overset{\text{df.}}{=} |\mathcal{A}(\tau)_{n^{(l)}_{Y_l} n^{(l-1)}_{Y_{l-1}} \dots n^{(1)}_{Y_1}}|
\end{aligned}$$

and the mate-scopes are defined similarly. Then, because a P-view in $\sigma \otimes \tau$ is either a P-view in σ or τ (which is shown by induction on the length of positions of $\sigma \otimes \tau$), it is easy to see that $\mathsf{st}(\mathcal{A}(\sigma \otimes \tau)) = \sigma \otimes \tau$ holds. Also, it is clear that $\mathcal{A}(\sigma \otimes \tau)$ is standard since so are $\mathcal{A}(\sigma)$ and $\mathcal{A}(\tau)$. Intuitively, $\mathcal{A}(\sigma \otimes \tau)$ sees the new digit ($\mathscr{W}$ or

$\mathscr{E}$) of the current state $\boldsymbol{s} \in \mathcal{S}_{\mathcal{A}(\sigma \otimes \tau)}$ and decides $\mathcal{A}(\sigma)$ or $\mathcal{A}(\tau)$ to apply (n.b. since $\mathcal{Q}_{\mathcal{A}(\sigma \otimes \tau)}$ tracks every initial move by Q, a state must be non-empty).

It is then clear that pairing of dynamic strategies may be handled in a completely similar manner; currying and uncurrying are even simpler. Hence, let us skip the proof for these constructions.

Now, consider the concatenation $\iota \ddagger \kappa : J \ddagger K$ of viable dynamic strategies $\iota : J$ and $\kappa : K$ such that $\mathcal{H}^\omega(J) \trianglelefteqslant A \multimap B$ and $\mathcal{H}^\omega(K) \trianglelefteqslant B \multimap C$ for some normalized dynamic games A, B and C. Let $\mathcal{A}(\iota)$ and $\mathcal{A}(\kappa)$ be standard st-algorithms such that $\mathsf{st}(\mathcal{A}(\iota)) = \iota$ and $\mathsf{st}(\mathcal{A}(\kappa)) = \kappa$. We define the set $\mathcal{S}_{\mathcal{A}(\iota \ddagger \kappa)}$ and the map $\mathcal{Q}_{\mathcal{A}(\iota \ddagger \kappa)}$ by:

$$\begin{aligned} \mathcal{S}_{\mathcal{A}(\iota \ddagger \kappa)} &\overset{\text{df.}}{=} \{ n^{(k)}_{G(Y_k)} n^{(k-1)}_{G(Y_{k-1})} \dots n^{(1)}_{G(Y_1)} m^{(l)}_{F(X_l)} m^{(l-1)}_{F(X_{l-1})} \dots m^{(1)}_{F(X_1)} \mid \\ &\qquad m^{(l)}_{X_l} m^{(l-1)}_{X_{l-1}} \dots m^{(1)}_{X_1} \in \mathcal{S}_{\mathcal{A}(\iota)}, n^{(k)}_{Y_k} n^{(k-1)}_{Y_{k-1}} \dots n^{(1)}_{Y_1} \in \mathcal{S}_{\mathcal{A}(\kappa)} \} \\ \mathcal{Q}_{\mathcal{A}(\iota \ddagger \kappa)} &: m^{(i)}_{F(X_i)} \mapsto \mathcal{Q}_{\mathcal{A}(\iota)}(m^{(i)}_{X_i}), n^{(j)}_{G(Y_j)} \mapsto \mathcal{Q}_{\mathcal{A}(\kappa)}(n^{(j)}_{Y_j}) \end{aligned}$$

where $F(X_i) \overset{\text{df.}}{=} \begin{cases} X_i & \text{if } m^{(i)}_{X_i} \in M^{\mathsf{Ext}}_J \wedge X_i = \mathscr{W}; \\ X_i \mathscr{S} & \text{otherwise} \end{cases}$ for $i = 1, 2, \dots, l$ and $G(Y_j) \overset{\text{df.}}{=}$ $\begin{cases} Y_j & \text{if } n^{(j)}_{Y_j} \in M^{\mathsf{Ext}}_K \wedge Y_j = \mathscr{E}; \\ Y_j \mathscr{N} & \text{otherwise} \end{cases}$ for $j = 1, 2, \dots, k$. We construct the finite partial function $\mathcal{A}(\iota \ddagger \kappa)_{n^{(k)}_{G(Y_k)} n^{(k-1)}_{G(Y_{k-1})} \dots n^{(1)}_{G(Y_1)} m^{(l)}_{F(X_l)} m^{(l-1)}_{F(X_{l-1})} \dots m^{(1)}_{F(X_1)}}$ (as well as the view- and the mate-scopes) from $\mathcal{A}(\kappa)_{n^{(k)}_{Y_k} n^{(k-1)}_{Y_{k-1}} \dots n^{(1)}_{Y_1}}$ if $l = 0$, and from $\mathcal{A}(\iota)_{m^{(l)}_{X_l} m^{(l-1)}_{X_{l-1}} \dots m^{(1)}_{X_1}}$ otherwise, by modifying the symbols in the table similarly to the case of tensor (where the view- and the mate-scopes are just inherited). Again, $\mathcal{Q}_{\mathcal{A}(\iota \ddagger \kappa)}$ clearly satisfies the axiom Q. Because a P-view of $\iota \ddagger \kappa$ is a one of κ or a one of ι followed by a one of κ (n.b., it is crucial here that $\mathcal{Q}_{\mathcal{A}(\iota)}$ tracks initial moves by the axiom Q, and $\Box$ does not occue in $\mathcal{A}(\iota)$ for it is standard), we may conclude that $\mathsf{st}(\mathcal{A}(\iota \ddagger \kappa)) = \iota \ddagger \kappa$. Moreover, $\mathcal{A}(\iota \ddagger \kappa)$ is clearly standard as so are $\mathcal{A}(\iota)$ and $\mathcal{A}(\kappa)$.

Finally, assume that $\varphi^\dagger : [!A_{\mathscr{W}}]_{\wr \underline{0} \boldsymbol{e} + \int \underline{0} \hbar \boldsymbol{f}} \multimap [!B_{\mathscr{E}}]_{\wr \underline{0} \boldsymbol{e'} + \int \underline{0} \hbar \boldsymbol{f'}}$ is the promotion of a viable strategy $\varphi : [!A_{\mathscr{W}}]_{\wr \underline{0} \boldsymbol{e} + \int \underline{0} \hbar \boldsymbol{f}} \multimap [B_{\mathscr{E}}]_{\boldsymbol{f'}}$ with an st-algorithm $\mathcal{A}(\varphi)$ that realizes φ. As the more general case $\varphi : G$, where $\mathcal{H}^\omega(G) \trianglelefteqslant A \Rightarrow B$, is similar (as internal moves of $G^\dagger$ retain new digits of outer tags on moves of $!B$; see Definition 4.3.22), we focus on the case $\varphi : A \Rightarrow B$ for simplicity. We define $\mathcal{S}_{\mathcal{A}(\varphi^\dagger)} \overset{\text{df.}}{=} \mathcal{S}_{\mathcal{A}(\varphi)}$ and $\mathcal{Q}_{\mathcal{A}(\varphi^\dagger)} \overset{\text{df.}}{=} \mathcal{Q}_{\mathcal{A}(\varphi)}$. Then, roughly, the idea is that if φ makes the next P-move $[a_{\mathscr{W}}]_{\wr \underline{0} \boldsymbol{e} + \int \underline{0} \hbar \boldsymbol{f}}$ (resp. $[b_{\mathscr{E}}]_{\boldsymbol{f'}}$) at an odd-length position $\boldsymbol{t}x$ of $[!A_{\mathscr{W}}]_{\wr \underline{0} \boldsymbol{e} + \int \underline{0} \hbar \boldsymbol{f}} \multimap [B_{\mathscr{E}}]_{\boldsymbol{f'}}$, then $\varphi^\dagger$ at an odd-length position $\boldsymbol{t'}x'$ of $[!A_{\mathscr{W}}]_{\wr \underline{0} \boldsymbol{e} + \int \underline{0} \hbar \boldsymbol{f}} \multimap [!B_{\mathscr{E}}]_{\wr \underline{0} \boldsymbol{e'} + \int \underline{0} \hbar \boldsymbol{f'}}$ that begins with an initial move $[\hat{b}_{\mathscr{E}}]_{\wr \underline{0} \hat{\boldsymbol{e}} + \int \underline{0} \hbar \hat{\boldsymbol{f}}}$ and satisfies $\boldsymbol{t'}x' \upharpoonright \hat{\boldsymbol{e}} = \boldsymbol{t}x$ (see Definition 4.3.22 for the definition of $\boldsymbol{t'}x' \upharpoonright \hat{\boldsymbol{e}}$) makes the corresponding next P-move $[a_{\mathscr{W}}]_{\wr \underline{0} \wr \underline{1} \hat{\boldsymbol{e}} ++ \int \underline{1} \hbar \wr \underline{1} \boldsymbol{e} ++ \int \underline{1} \int \underline{0} \hbar \boldsymbol{f}}$ (resp. $[b_{\mathscr{E}}]_{\wr \underline{0} \hat{\boldsymbol{e}} + \int \underline{0} \hbar \boldsymbol{f'}}$). As opposed to the other constructions, however, the formal definition of the finite table of each $\mathcal{A}(\varphi^{\dagger})_{\boldsymbol{m}}$, where $\boldsymbol{m} \in \mathcal{S}_{\mathcal{A}(\varphi^{\dagger})}$, is rather involved; thus, we just informally describe how to obtain the table of $\mathcal{A}(\varphi^{\dagger})_{\boldsymbol{m}}$ from that of $\mathcal{A}(\varphi)_{\boldsymbol{m}}$, which should suffice for the reader to see how to construct the tables if he or she wishes. Fix any $\boldsymbol{s} \in \mathcal{S}_{\mathcal{A}(\varphi^{\dagger})} = \mathcal{S}_{\mathcal{A}(\varphi)}$.

1. Let $\boldsymbol{t}[m]_{\tilde{\boldsymbol{e}}}[n]_{\boldsymbol{e}} \in \varphi$ and $\boldsymbol{t'}[m]_{\tilde{\boldsymbol{e}}'}[n]_{\boldsymbol{e}'} \in \varphi^{\dagger}$ such that $\mathcal{Q}^{\star}_{\mathcal{A}(\varphi^{\dagger})}(\lceil \boldsymbol{t'}[m]_{\tilde{\boldsymbol{e}}} \rceil) = \boldsymbol{s}$ and $\boldsymbol{t'}[m]_{\tilde{\boldsymbol{e}}'}[n]_{\boldsymbol{e}'} \upharpoonright \hat{\boldsymbol{e}} = \boldsymbol{t}[m]_{\tilde{\boldsymbol{e}}}[n]_{\boldsymbol{e}}$, where the first move of the current thread of $\varphi^{\dagger}$ is of the form $[\hat{b}]_{\wr \hat{\boldsymbol{e}} \int \hbar \hat{\boldsymbol{f}}}$. Let us describe how $\mathcal{A}(\varphi^{\dagger})_{\boldsymbol{s}}$ calculates the representation $\mathsf{n}.\mathscr{C}^{*}(\boldsymbol{e'})$ of the next move $[n]_{\boldsymbol{e}'}$ by a case analysis on m and n:

   - If m and n both belong to A, which $\mathcal{A}(\varphi^{\dagger})_{\boldsymbol{s}}$ may recognize by the method described below, then $\tilde{\boldsymbol{e}}$, $\boldsymbol{e}$, $\tilde{\boldsymbol{e}}'$ and $\boldsymbol{e'}$ are respectively of the form $\wr^{\underline{0}} \tilde{\boldsymbol{g}}^{+} \int^{\underline{0}} \hbar \tilde{\boldsymbol{h}}$, $\wr^{\underline{0}} \boldsymbol{g}^{+} \int^{\underline{0}} \hbar \boldsymbol{h}$, $\wr^{\underline{0}} \wr^{\underline{1}} \hat{\boldsymbol{e}}^{++} \int^{\underline{1}} \hbar \wr^{\underline{1}} \tilde{\boldsymbol{g}}^{++} \int^{\underline{1}} \int^{\underline{0}} \hbar \tilde{\boldsymbol{h}}$ and $\wr^{\underline{0}} \wr^{\underline{1}} \hat{\boldsymbol{e}}^{++} \int^{\underline{1}} \hbar \wr^{\underline{1}} \boldsymbol{g}^{++} \int^{\underline{1}} \int^{\underline{0}} \hbar \boldsymbol{h}$.[10] Then, with the help of m-views, $\mathcal{A}(\varphi^{\dagger})_{\boldsymbol{s}}$ first calculates $\mathsf{n}.\langle @0 \langle @1 \mathscr{C}^{*}(\hat{\boldsymbol{e}}^{++}) \rangle @1 \sharp$ by simulating the computation of n by $\mathcal{A}(\varphi)_{\boldsymbol{s}}$ and referring to $\mathscr{C}^{*}(\tilde{\boldsymbol{e}}')$, and computes the remaining $\langle @1 \mathscr{C}^{*}(\boldsymbol{g}^{++}) \rangle @1 \rangle @0 \sharp \mathscr{C}^{*}(\boldsymbol{h})$ by simulating the computation of $\mathscr{C}^{*}(\boldsymbol{e})$ by $\mathcal{A}(\varphi)_{\boldsymbol{s}}$ (where adding symbols to adjust depths of $\langle$ and $\rangle$, and inserting $\rangle @0$ before $\sharp$). Clearly, $\mathcal{A}(\varphi^{\dagger})_{\boldsymbol{s}}$ is standard.
   - If m and n belong to A and B, respectively, then $\tilde{\boldsymbol{e}}$, $\boldsymbol{e}$, $\tilde{\boldsymbol{e}}'$ and $\boldsymbol{e'}$ are of the form $\wr^{\underline{0}} \tilde{\boldsymbol{g}}^{+} \int^{\underline{0}} \hbar \tilde{\boldsymbol{h}}$, $\boldsymbol{h}$, $\wr^{\underline{0}} \wr^{\underline{1}} \hat{\boldsymbol{e}}^{++} \int^{\underline{1}} \hbar \wr^{\underline{1}} \tilde{\boldsymbol{g}}^{++} \int^{\underline{1}} \int^{\underline{0}} \hbar \tilde{\boldsymbol{h}}$ and $\wr^{\underline{0}} \hat{\boldsymbol{e}} \int^{\underline{0}} \hbar \boldsymbol{h}$, respectively. Again, with the help of m-views, $\mathcal{A}(\varphi^{\dagger})_{\boldsymbol{s}}$ first calculates $\mathsf{n}.\langle @0 \mathscr{C}^{*}(\hat{\boldsymbol{e}}^{+}) \rangle @0 \sharp$ by simulating the computation of n by $\mathcal{A}(\varphi)_{\boldsymbol{s}}$ and referring to $\mathscr{C}^{*}(\tilde{\boldsymbol{e}}')$, and then computes $\mathscr{C}^{*}(\boldsymbol{h})$ by simulating the computation of $\mathscr{C}^{*}(\boldsymbol{e})$ by $\mathcal{A}(\varphi)_{\boldsymbol{s}}$. Again, $\mathcal{A}(\varphi^{\dagger})_{\boldsymbol{s}}$ is clearly standard.
   - The remaining two cases are completely analogous.

2. It remains to stipulate how $\mathcal{A}(\varphi^{\dagger})_{\boldsymbol{s}}$ distinguishes the above four cases. Assume $\hat{q}^{[3]} \boldsymbol{v} \mathsf{n}^{[3]} \in \mathcal{A}(\varphi)_{\boldsymbol{s}}$ when $\mathcal{A}(\varphi)_{\boldsymbol{s}}$ has computed the inner element n. Note that $\boldsymbol{v}$ has enough information to identify n, and moves occurring in $\boldsymbol{v}$ all have 'tags' $(\_)^{[0]}$, $(\_)^{[1]}$ or $(\_)^{[2]}$ (but not $(\_)^{[3]}$). Thus, by simulating this computation of n by $\mathcal{A}(\varphi)_{\boldsymbol{s}}$ but replacing $\mathsf{n}^{[3]}$ with $\hat{q}^{[2]}$ to learn about m, $\mathcal{A}(\varphi^{\dagger})_{\boldsymbol{s}}$ may recognize the current case out of the four described above. Specifically, if $\hat{q}^{[3]} \boldsymbol{v} \mathsf{n}^{[3]} q^{[3]} \mapsto x$ is the first step for $\mathcal{A}(\varphi)_{\boldsymbol{s}}$ to compute $\boldsymbol{e}$, then correspondingly $\mathcal{A}(\varphi^{\dagger})_{\boldsymbol{s}}$ computes as $\hat{q}^{[3]} \boldsymbol{v} \mathsf{n}^{[3]} q^{[3]} \mapsto v_1$, $\hat{q}^{[3]} \boldsymbol{v} \mathsf{n}^{[3]} q^{[3]} v_1 v_2 \mapsto v_3$, $\dots$, $\hat{q}^{[3]} \boldsymbol{v} \mathsf{n}^{[3]} q^{[3]} \boldsymbol{v} \mapsto q^{[2]}$,

[10]Note that the outer tags which $\mathcal{A}(\varphi)_{\boldsymbol{s}}$ refers to for computing $\boldsymbol{e}$ are only $\tilde{\boldsymbol{e}}$ as $\mathcal{A}(\varphi)$ is standard, and therefore it is clearly possible to adjust the computation of $\mathscr{C}^{*}(\boldsymbol{e})$ by $\mathcal{A}(\varphi)$ to the computation of the latter half $\langle @1 \mathscr{C}^{*}(\boldsymbol{g}^{++}) \rangle @1 \rangle @0 \sharp \mathscr{C}^{*}(\boldsymbol{h})$ of $\mathscr{C}^{*}(\boldsymbol{e'})$ by $\mathcal{A}(\varphi^{\dagger})$ given below.

$\hat{q}^{[3]}\boldsymbol{v}\mathsf{n}^{[3]}q^{[3]}\boldsymbol{v}q^{[2]}\mathsf{m}^{[2]} \mapsto x'$, where x' is the first step for $\mathcal{A}(\varphi^\dagger)_{\boldsymbol{s}}$ to compute $\boldsymbol{e'}$; and if $\mathcal{A}(\varphi)_{\boldsymbol{s}}$ next computes $\hat{q}^{[3]}\boldsymbol{v}\mathsf{n}^{[3]}q^{[3]}xy \mapsto z$, then correspondingly $\mathcal{A}(\varphi^\dagger)_{\boldsymbol{s}}$ computes as $\hat{q}^{[3]}\boldsymbol{v}\mathsf{n}^{[3]}q^{[3]}\boldsymbol{v}q^{[2]}\mathsf{m}^{[2]}x'y' \mapsto v_1$, $\hat{q}^{[3]}\boldsymbol{v}\mathsf{n}^{[3]}q^{[3]}\boldsymbol{v}q^{[2]}\mathsf{m}^{[2]}x'y'v_1v_2 \mapsto v_3, \dots,$ $\hat{q}^{[3]}\boldsymbol{v}\mathsf{n}^{[3]}q^{[3]}\boldsymbol{v}q^{[2]}\mathsf{m}^{[2]}x'y'\boldsymbol{v} \mapsto q^{[2]}$, $\hat{q}^{[3]}\boldsymbol{v}\mathsf{n}^{[3]}q^{[3]}\boldsymbol{v}q^{[2]}\mathsf{m}^{[2]}x'y'\boldsymbol{v}q^{[2]}\mathsf{m}^{[2]} \mapsto z'$, where y', z' are the second and the third steps for $\mathcal{A}(\varphi^\dagger)_{\boldsymbol{s}}$ to compute $\boldsymbol{e'}$, and so on. That is, $\mathcal{A}(\varphi^\dagger)_{\boldsymbol{s}}$ remembers the current case by inserting the sequence $\boldsymbol{v}q^{[2]}\mathsf{m}^{[2]}$ (of length $\leqslant 8$ without loss of generality) between each computational step. Note that $\mathcal{A}(\varphi^\dagger)$ remains to be standard.

It should be clear from the above description how to construct $\mathcal{A}(\varphi^\dagger)$ from $\mathcal{A}(\varphi)$, completing the proof. □

Remark. It is straightforward to see that states of st-algorithms which we have given for dynamic strategies modeling PCF (Definition 4.2.9) are a precise formalization of states of P-views (Definition 4.2.11) applied to the dynamic strategies.

4.4.2 Examples of Viable Dynamic Strategies

This section presents various examples of viable dynamic strategies (realized by standard st-algorithms). These dynamic strategies except fixed-point strategies fix_A are actually finitary; thus, we need viability only for promotion $(\_)^\dagger$ and fix_A, both of which are necessary for the main theorem (Theorem 4.4.18) in Section 4.4.3.

Example 4.4.14. Given a normalized dynamic game A, we define an st-algorithm $\mathcal{A}(\mathit{der}_A)$ that realizes the dereliction $\mathit{der}_A : [!A_{\mathscr{W}}]_{\wr\underline{0}\int\underline{0}\hbar\boldsymbol{e}} \multimap [A_{\mathscr{E}}]_{\boldsymbol{e'}}$ (Example 4.3.30) by $\mathcal{Q}_{\mathit{der}_A}(m) \overset{\mathrm{df.}}{=} \begin{cases} \top & \text{if } m \in \pi_1(M^{\mathsf{Init}}_{A \Rightarrow A}) \\ \bot & \text{otherwise} \end{cases}$, $\mathcal{S}_{\mathit{der}_A} \overset{\mathrm{df.}}{=} \pi_1(M^{\mathsf{Init}}_{A \Rightarrow A})$, $|\mathcal{A}(\mathit{der}_A)_m| \overset{\mathrm{df.}}{=} 3$ and $\|\mathcal{A}(\mathit{der}_A)_m\| \overset{\mathrm{df.}}{=} 0$ for all $m \in \mathcal{S}_{\mathit{der}_A}$; given $m \in \mathcal{S}_{\mathit{der}_A}$, $\mathcal{A}(\mathit{der}_A)^{\circledS}_m$ computes as in the following diagrams (n.b., we skip defining the table of $\mathcal{A}(\mathit{der}_A)_m$):

$\mathcal{G}(M_{A\Rightarrow A})^{[0]}$	&	$\mathcal{G}(M_{A\Rightarrow A})^{[1]}$	&	$\mathcal{G}(M_{A\Rightarrow A})^{[2]}$	$\overset{\mathcal{A}(der_A)^{\circledS}_m}{\Rightarrow}$	$\mathcal{G}(M_{A\Rightarrow A})^{[3]}$
						$\hat{q}^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{a_E}^{[2]}$		
						$\mathsf{a_W}^{[3]}$
						$q^{[3]}$
						$\langle @0^{[3]}$
						$q^{[3]}$
						$\rangle @0^{[3]}$
						$q^{[3]}$
						$\sharp^{[3]}$
						$q^{[3]}$
				$q^{[2]}$		
				$\mathscr{C}(e_1)^{[2]}$		
						$\mathscr{C}(e_1)^{[3]}$
						$q^{[3]}$
				$q^{[2]}$		
				$\mathscr{C}(e_2)^{[2]}$		
						$\mathscr{C}(e_2)^{[3]}$
					$\vdots$	
						$q^{[3]}$
				$q^{[2]}$		
				$\mathscr{C}(e_k)^{[2]}$		
						$\mathscr{C}(e_k)^{[3]}$
						$q^{[3]}$
				$q^{[2]}$		
				$\checkmark^{[2]}$		
						$\checkmark^{[3]}$

$\mathcal{G}(M_{A\Rightarrow A})^{[0]}$	$\&$	$\mathcal{G}(M_{A\Rightarrow A})^{[1]}$	$\&$	$\mathcal{G}(M_{A\Rightarrow A})^{[2]}$	$\overset{\mathcal{A}(der_A)^{\circledS}_m}{\Rightarrow}$	$\mathcal{G}(M_{A\Rightarrow A})^{[3]}$
						$\hat{q}^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{a_W}^{[2]}$		
						$\mathsf{a_E}^{[3]}$
						$q^{[3]}$
				$q^{[2]}$		
				$\langle @0^{[2]}$		
				$q^{[2]}$		
				$\rangle @0^{[2]}$		
				$q^{[2]}$		
				$\sharp^{[2]}$		
				$q^{[2]}$		
				$\mathscr{C}(e_1)^{[2]}$		
						$\mathscr{C}(e_1)^{[3]}$
						$q^{[3]}$
				$q^{[2]}$		
				$\mathscr{C}(e_2)^{[2]}$		
						$\mathscr{C}(e_2)^{[3]}$
					$\vdots$	
						$q^{[3]}$
				$q^{[2]}$		
				$\mathscr{C}(e_k)^{[2]}$		
						$\mathscr{C}(e_k)^{[3]}$
						$q^{[3]}$
				$q^{[2]}$		
				$\checkmark^{[2]}$		
						$\checkmark^{[3]}$

It is straightforward to see that $\mathsf{st}(\mathcal{A}(der_A)) = der_A$ holds, showing viability of der_A. Also, $\mathcal{A}(der_A)$ is clearly standard.

Example 4.4.15. Given any normalized dynamic game A, we define an st-algorithm $\mathcal{A}(case_A)$ that realizes the case strategy $case_A$ (Example 4.3.31) as follows.

Notation. Given $\mathsf{e} \in \mathsf{T}$, let us define $\mathsf{e}^+ \overset{\text{df.}}{=} \begin{cases} \ll \mathsf{'} & \text{if } \mathsf{e} = \ll; \\ \mathsf{e} & \text{otherwise.} \end{cases}$

States and the query function of $\mathcal{A}(case_A)$ are similar to those of $\mathcal{A}(der_A)$, and the instruction strategy $\mathcal{A}(case_A)^{\circledS}_m$ for each $m \in \mathcal{S}_{\mathcal{A}(case_A)}$ plays as follows (again, we skip writing down the table of $\mathcal{A}(case_A)_m$), where we write $A^{A^2\&\mathbf{2}}$ for $A\&A\&\mathbf{2} \Rightarrow A$:

$$
\begin{array}{ccccccc}
\mathcal{G}(M_{A^{A^2\&\mathbf{2}}})^{[0]} & \& & \mathcal{G}(M_{A^{A^2\&\mathbf{2}}})^{[1]} & \& & \mathcal{G}(M_{A^{A^2\&\mathbf{2}}})^{[2]} & \overset{\mathcal{A}(case_A)^{\circledS}_m}{\Rightarrow} & \mathcal{G}(M_{A^{A^2\&\mathbf{2}}})^{[3]} \\
\hline
&&&&&& \hat{q}^{[3]} \\
&&&& \hat{q}^{[2]} && \\
&&&& \hat{\mathsf{a}}_{\mathsf{E}}^{[2]} && \\
&&&&&& \hat{\mathsf{q}}_{\mathsf{EW}}^{[3]} \\
&&&&&& q^{[3]} \\
&&&&&& \langle @0^{[3]} \\
&&&&&& q^{[3]} \\
&&&& q^{[2]} && \\
&&&& \mathscr{C}(e_1)^{[2]} && \\
&&&&&& (\mathscr{C}(e_1)^{+})^{[3]} \\
&&&&&& q^{[3]} \\
&&&& \hat{q}^{[2]} && \\
&&&& \hat{\mathsf{a}}_{\mathsf{E}}^{[2]} && \\
&&&& q^{[2]} && \\
&&&& \mathscr{C}(e_2)^{[2]} && \\
&&&&&& (\mathscr{C}(e_2)^{+})^{[3]} \\
&&&&& \vdots & \\
&&&&&& q^{[3]} \\
&&&& \hat{q}^{[2]} && \\
&&&& \hat{\mathsf{a}}_{\mathsf{E}}^{[2]} && \\
&&&& q^{[2]} && \\
&&&& \mathscr{C}(e_k)^{[2]} && \\
&&&&&& (\mathscr{C}(e_k)^{+})^{[3]} \\
&&&&&& q^{[3]} \\
&&&& \hat{q}^{[2]} && \\
&&&& \hat{\mathsf{a}}_{\mathsf{E}}^{[2]} && \\
&&&& q^{[2]} && \\
&&&& \checkmark^{[2]} && \\
&&&&&& \rangle @0^{[3]} \\
&&&&&& q^{[3]} \\
&&&&&& \sharp^{[3]} \\
&&&&&& q^{[3]} \\
&&&&&& \checkmark^{[3]}
\end{array}
$$

where $\hat{a}_{\mathscr{E}} \in \pi_1(M^{\mathsf{Init}}_{A^{A^2\&\mathbf{2}}})$, which can be recognized as the set $\pi_1(M_{A^{A^2\&\mathbf{2}}})$ is finite. The iteration of $\hat{q}^{[2]}.\hat{\mathsf{a}}_{\mathsf{E}}^{[2]}$ is to distinguish the case from other cases. This remark is applied to the diagrams in the rest of the chapter.

$\mathcal{G}(M_{A^{A^2 \& \mathbf{2}}})^{[0]}$	$\&$	$\mathcal{G}(M_{A^{A^2 \& \mathbf{2}}})^{[1]}$	$\&$	$\mathcal{G}(M_{A^{A^2 \& \mathbf{2}}})^{[2]}$	$\overset{\mathcal{A}(case_A)^{\circledS}_m}{\Rightarrow}$	$\mathcal{G}(M_{A^{A^2 \& \mathbf{2}}})^{[3]}$
						$\hat{q}^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{tt_{EW}}^{[2]}$ $(\mathsf{ff_{EW}}^{[2]})$		
$\hat{q}^{[0]}$						
$\mathsf{a_E}^{[0]}$						
						$\mathsf{a_{WWW}}^{[3]}$ $(\mathsf{a_{EWW}}^{[3]})$
						$q^{[3]}$
						$\langle @0^{[3]}$
						$q^{[3]}$
						$\rangle @0^{[3]}$
						$q^{[3]}$
						$\sharp^{[3]}$
						$q^{[3]}$
				$q^{[2]}$		
				$\langle @0^{[2]}$		
				$q^{[2]}$		
				$\mathscr{C}(e_1)^{[2]}$		
						$\mathscr{C}(e_1)^{[3]}$
						$q^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{tt_{EW}}^{[2]}$ $(\mathsf{ff_{EW}}^{[2]})$		
				$q^{[2]}$		
				$\mathscr{C}(e_2)^{[2]}$		
						$\mathscr{C}(e_2)^{[3]}$
					$\vdots$	
						$q^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{tt_{EW}}^{[2]}$ $(\mathsf{ff_{EW}}^{[2]})$		
				$q^{[2]}$		
				$\mathscr{C}(e_k)^{[2]}$		
						$\mathscr{C}(e_k)^{[3]}$
						$q^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{tt_{EW}}^{[2]}$ $(\mathsf{ff_{EW}}^{[2]})$		
				$q^{[2]}$		
				$\rangle @0^{[2]}$		
				$\hat{q}^{[2]}$		
				$\mathsf{tt_{EW}}^{[2]}$ $(\mathsf{ff_{EW}}^{[2]})$		
				$q^{[2]}$		
				$\sharp^{[2]}$		
				$\hat{q}^{[2]}$		
				$\mathsf{tt_{EW}}^{[2]}$ $(\mathsf{ff_{EW}}^{[2]})$		
				$q^{[2]}$		
				$\checkmark^{[2]}$		
						$\checkmark^{[3]}$

$\mathcal{G}(M_{A^{A^2\&\mathbf{2}}})^{[0]}$	$\&$	$\mathcal{G}(M_{A^{A^2\&\mathbf{2}}})^{[1]}$	$\&$	$\mathcal{G}(M_{A^{A^2\&\mathbf{2}}})^{[2]}$	$\overset{\mathcal{A}(case_A)^{\circledS}_m}{\Rightarrow}$	$\mathcal{G}(M_{A^{A^2\&\mathbf{2}}})^{[3]}$
						$\hat{q}^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{a}_{\mathsf{WWW}}{}^{[2]}$ $(\mathsf{a}_{\mathsf{EWW}}{}^{[2]})$		
						$\mathsf{a}_{\mathsf{E}}{}^{[3]}$
						$q^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{a}_{\mathsf{WWW}}{}^{[2]}$ $(\mathsf{a}_{\mathsf{EWW}}{}^{[2]})$		
				$q^{[2]}$		
				$\langle @0^{[2]}$		
				$q^{[2]}$		
				$\rangle @0^{[2]}$		
				$q^{[2]}$		
				$\sharp^{[2]}$		
				$q^{[2]}$		
				$\mathscr{C}(e_1)^{[2]}$		
						$\mathscr{C}(e_1)^{[3]}$
						$q^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{a}_{\mathsf{WWW}}{}^{[2]}$ $(\mathsf{a}_{\mathsf{EWW}}{}^{[2]})$		
				$q^{[2]}$		
				$\mathscr{C}(e_2)^{[2]}$		
						$\mathscr{C}(e_2)^{[3]}$
					$\vdots$	
						$q^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{a}_{\mathsf{WWW}}{}^{[2]}$ $(\mathsf{a}_{\mathsf{EWW}}{}^{[2]})$		
				$q^{[2]}$		
				$\mathscr{C}(e_k)^{[2]}$		
						$\mathscr{C}(e_k)^{[3]}$
						$q^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{a}_{\mathsf{WWW}}{}^{[2]}$ $(\mathsf{a}_{\mathsf{EWW}}{}^{[2]})$		
				$q^{[2]}$		
				$\checkmark^{[2]}$		
						$\checkmark^{[3]}$

$\mathcal{G}(M_{A^{A^2\&\mathbf{2}}})^{[0]}$	$\&$	$\mathcal{G}(M_{A^{A^2\&\mathbf{2}}})^{[1]}$	$\&$	$\mathcal{G}(M_{A^{A^2\&\mathbf{2}}})^{[2]}$	$\overset{\mathcal{A}(case_A)^{\circledS}_m}{\Rightarrow}$	$\mathcal{G}(M_{A^{A^2\&\mathbf{2}}})^{[3]}$
						$\hat{q}^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{a}_{\mathsf{E}}^{[2]}$		
$\hat{q}^{[0]}$						
$\tilde{\mathsf{a}}_{\mathsf{WWW}}^{[0]}$ $(\tilde{\mathsf{a}}_{\mathsf{EWW}}^{[0]})$						
						$\mathsf{a}_{\mathsf{WWW}}{}^{[3]}$ $(\mathsf{a}_{\mathsf{EWW}}{}^{[3]})$
						$q^{[3]}$
						$\langle @0^{[3]}$
						$q^{[3]}$
						$\rangle @0^{[3]}$
						$q^{[3]}$
						$\sharp^{[3]}$
						$q^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{a}_{\mathsf{E}}^{[2]}$		
				$q^{[2]}$		
				$\mathscr{C}(e_1)^{[2]}$		
						$\mathscr{C}(e_1)^{[3]}$
						$q^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{a}_{\mathsf{E}}^{[2]}$		
				$q^{[2]}$		
				$\mathscr{C}(e_2)^{[2]}$		
						$\mathscr{C}(e_2)^{[3]}$
					$\vdots$	
						$q^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{a}_{\mathsf{E}}^{[2]}$		
				$q^{[2]}$		
				$\mathscr{C}(e_k)^{[2]}$		
						$\mathscr{C}(e_k)^{[3]}$
						$q^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{a}_{\mathsf{E}}^{[2]}$		
				$q^{[2]}$		
				$\checkmark^{[2]}$		
						$\checkmark^{[3]}$

where $a_{\mathscr{E}} \notin \pi_1(M^{\mathsf{Init}}_{A^{A^2\&\mathbf{2}}})$. Clearly, $\mathsf{st}(\mathcal{A}(case_A)) = case_A$, and therefore $case_A$ is viable. Also, it is easy to see that $\mathcal{A}(case_A)$ is standard.

Example 4.4.16. Let us next give an st-algorithm $\mathcal{A}(zero?)$ that realizes the ifzero strategy *zero*? (Example 4.3.32): Let $\mathcal{Q}_{\mathcal{A}(zero?)}(m) \overset{\mathrm{df.}}{=} \begin{cases} \top & \text{if } m = \hat{q}_{\mathscr{E}}; \\ \bot & \text{otherwise} \end{cases}$, $\mathcal{S}_{\mathcal{A}(zero?)} \overset{\mathrm{df.}}{=} \{\hat{q}_{\mathscr{E}}\}$, $|\mathcal{A}(zero?)_{\hat{q}_{\mathscr{E}}}| \overset{\mathrm{df.}}{=} 13$, $\|\mathcal{A}(zero?)_{\hat{q}_{\mathscr{E}}}\| \overset{\mathrm{df.}}{=} 0$, and $\mathcal{A}(zero?)^{\circledS}_{\hat{q}_{\mathscr{E}}}$ plays as in the following diagrams (again, we omit the table of $\mathcal{A}(zero?)_{\hat{q}_{\mathscr{E}}}$ as it should be clear at this point):

$$\begin{array}{ccccccc} \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathbf{2}})^{[0]} & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathbf{2}})^{[1]} & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathbf{2}})^{[2]} & \overset{\mathcal{A}(zero?)^{\circledS}_{\hat{q}_{\mathscr{E}}}}{\Rightarrow} & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathbf{2}})^{[3]} \\ \hline &&&&&& \hat{q}^{[3]} \\ &&&& \hat{q}^{[2]} && \\ &&&& \hat{\mathsf{q}}_{\mathsf{E}}^{[2]} && \\ &&&&&& \hat{\mathsf{q}}_{\mathsf{W}}^{[3]} \\ &&&&&& q^{[3]} \\ &&&&&& \langle @0^{[3]} \\ &&&&&& q^{[3]} \\ &&&&&& \rangle @0^{[3]} \\ &&&&&& q^{[3]} \\ &&&&&& \sharp^{[3]} \\ &&&&&& q^{[3]} \\ &&&&&& \checkmark^{[3]} \end{array}$$

$$\begin{array}{ccccccc} \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathbf{2}})^{[0]} & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathbf{2}})^{[1]} & \& & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathbf{2}})^{[2]} & \overset{\mathcal{A}(zero?)^{\circledS}_{\hat{q}_{\mathscr{E}}}}{\Rightarrow} & \mathcal{G}(M_{\mathcal{N}\Rightarrow\mathbf{2}})^{[3]} \\ \hline &&&&&& \hat{q}^{[3]} \\ &&&& \hat{q}^{[2]} && \\ &&&& \mathsf{no_W}^{[2]}\ (\mathsf{yes_W}^{[2]}) && \\ &&&&&& \mathsf{tt_E}^{[3]}\ (\mathsf{ff_E}^{[3]}) \\ &&&&&& q^{[3]} \\ &&&&&& \checkmark^{[3]} \end{array}$$

We clearly have $\mathsf{st}(\mathcal{A}(zero?)) = zero?$, and $\mathcal{A}(zero?)$ is standard.

Example 4.4.17. Finally, let us give an st-algorithm $\mathcal{A}(\mathit{fix}_A)$ that realizes the fixed-point strategy $\mathit{fix}_A : ([A_{\mathscr{W}\mathscr{W}}]_{\wr^{\underline{0}}\boldsymbol{g}+\int^{\underline{0}}\hbar\wr^{\underline{0}}\boldsymbol{f}+\int^{\underline{0}}\hbar\boldsymbol{e}} \Rightarrow [A_{\mathscr{E}\mathscr{W}}]_{\wr^{\underline{0}}\boldsymbol{g'}\int^{\underline{0}}\hbar\boldsymbol{e'}}) \Rightarrow [A_{\mathscr{E}}]_{\boldsymbol{e''}}$ on a given normalized dynamic game A. Note that we have already described fix_A informally in Example 4.3.33; here let us give a more detailed account, but again, it should suffice to just give diagrams for $\mathcal{A}(\mathit{fix}_A)^{\circledS}_m$ (where $m \in \mathcal{S}_{\mathit{fix}_A}$ is arbitrary).

Notation. Vertical double dots in the middle of $(\_)^{[i]}$ and $(\_)^{[i+1]}$ moves in the following diagram represent 'copy-cats' between $(\_)^{[i]}$ and $(\_)^{[i+1]}$ moves.

$\mathcal{G}(M_{(A\Rightarrow A)\Rightarrow A})^{[0]}$	$\&$	$\mathcal{G}(M_{(A\Rightarrow A)\Rightarrow A})^{[1]}$	$\&$	$\mathcal{G}(M_{(A\Rightarrow A)\Rightarrow A})^{[2]}$	$\overset{\mathcal{A}(\mathit{fix}_A)^{\circledS}_m}{\Rightarrow}$	$\mathcal{G}(M_{(A\Rightarrow A)\Rightarrow A})^{[3]}$
						$\hat{q}^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{a_E}^{[2]}$		
						$\mathsf{a_{EW}}^{[3]}$
						$q^{[3]}$
						$\langle @0^{[3]}$
						$q^{[3]}$
						$\rangle @0^{[3]}$
						$q^{[3]}$
						$\sharp^{[3]}$
						$q^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{a_E}^{[2]}$		
				$q^{[2]}$		
				$\mathscr{C}(e_1)^{[2]}$		
						$\mathscr{C}(e_1)^{[3]}$
						$q^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{a_E}^{[2]}$		
				$q^{[2]}$		
				$\mathscr{C}(e_2)^{[2]}$		
						$\mathscr{C}(e_2)^{[3]}$
					$\vdots$	
						$q^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{a_E}^{[2]}$		
				$q^{[2]}$		
				$\mathscr{C}(e_k)^{[2]}$		
						$\mathscr{C}(e_k)^{[3]}$
						$q^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{a_E}^{[2]}$		
				$q^{[2]}$		
				$\checkmark^{[2]}$		
						$\checkmark^{[3]}$

$\mathcal{G}(M_{(A\Rightarrow A)\Rightarrow A})^{[0]}$	&	$\mathcal{G}(M_{(A\Rightarrow A)\Rightarrow A})^{[1]}$	&	$\mathcal{G}(M_{(A\Rightarrow A)\Rightarrow A})^{[2]}$	$\overset{\mathcal{A}(\mathit{fix}_A)^{\circledS}_m}{\Rightarrow}$	$\mathcal{G}(M_{(A\Rightarrow A)\Rightarrow A})^{[3]}$
						$\hat{q}^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{a}_{\mathsf{WW}}{}^{[2]}$		
						$\mathsf{a}_{\mathsf{EW}}{}^{[3]}$
						$q^{[3]}$
						$\langle @0^{[3]}$
						$q^{[3]}$
				$q^{[2]}$		
				$\langle @0^{[2]}$		
						$\langle @1^{[3]}$
					$\vdots\ \vdots$	
						$q^{[3]}$
				$q^{[2]}$		
				$\rangle @0^{[2]}$		
						$\rangle @1^{[3]}$
						$q^{[3]}$
				$q^{[2]}$		
				$\sharp^{[2]}$		
						$\sharp^{[3]}$
						$q^{[3]}$
				$q^{[2]}$		
				$\langle @0^{[2]}$		
						$\langle @1^{[3]}$
					$\vdots\ \vdots$	
						$q^{[3]}$
				$q^{[2]}$		
				$\rangle @0^{[2]}$		
						$\rangle @1^{[3]}$
						$q^{[3]}$
						$\rangle @0^{[3]}$
						$q^{[3]}$
				$q^{[2]}$		
				$\sharp^{[2]}$		
						$\sharp^{[3]}$
					$\vdots\ \vdots$	
						$q^{[3]}$
				$q^{[2]}$		
				$\checkmark^{[2]}$		
						$\checkmark^{[3]}$

where we have omitted the iteration of $\hat{q}^{[2]}.\mathsf{a}_{\mathsf{WW}}{}^{[2]}$ for the lack of space.

$\mathcal{G}(M_{(A\Rightarrow A)\Rightarrow A})^{[0]}$ &	$\mathcal{G}(M_{(A\Rightarrow A)\Rightarrow A})^{[1]}$ &	$\mathcal{G}(M_{(A\Rightarrow A)\Rightarrow A})^{[2]}$	$\overset{\mathcal{A}(\mathit{fix}_A)^{\circledS}_m}{\Rightarrow}$	$\mathcal{G}(M_{(A\Rightarrow A)\Rightarrow A})^{[3]}$
				$\hat{q}^{[3]}$
		$\hat{q}^{[2]}$		
		$\mathsf{a_{EW}}^{[2]}$		
$\hat{q}^{[0]}$				
$\mathsf{a'_E}^{[0]}$				
				$\mathsf{a_E}^{[3]}$
				$q^{[3]}$
		$q^{[2]}$		
		$\langle @0^{[2]}$		
		$q^{[2]}$		
		$\rangle @0^{[2]}$		
		$q^{[2]}$		
		$\sharp^{[2]}$		
		$q^{[2]}$		
		$\mathscr{C}(e_1)^{[2]}$		
				$\mathscr{C}(e_1)^{[3]}$
				$q^{[3]}$
		$\hat{q}^{[2]}$		
		$\mathsf{a_{EW}}^{[2]}$		
		$q^{[2]}$		
		$\mathscr{C}(e_2)^{[2]}$		
				$\mathscr{C}(e_2)^{[3]}$
			$\vdots$	
				$q^{[3]}$
		$\hat{q}^{[2]}$		
		$\mathsf{a_{EW}}^{[2]}$		
		$q^{[2]}$		
		$\mathscr{C}(e_k)^{[2]}$		
				$\mathscr{C}(e_k)^{[3]}$
				$q^{[3]}$
		$\hat{q}^{[2]}$		
		$\mathsf{a_{EW}}^{[2]}$		
		$q^{[2]}$		
		$\checkmark^{[2]}$		
				$\checkmark^{[3]}$

where we have omitted the iteration of $\hat{q}^{[2]}.\mathsf{a_{EW}}^{[2]}.\hat{q}^{[0]}.\mathsf{a'_E}^{[0]}$ for the lack of space.

$\mathcal{G}(M_{(A\Rightarrow A)\Rightarrow A})^{[0]}$	$\&$	$\mathcal{G}(M_{(A\Rightarrow A)\Rightarrow A})^{[1]}$	$\&$	$\mathcal{G}(M_{(A\Rightarrow A)\Rightarrow A})^{[2]}$	$\overset{\mathcal{A}(\mathit{fix}_A)^{\circledS}_m}{\Rightarrow}$	$\mathcal{G}(M_{(A\Rightarrow A)\Rightarrow A})^{[3]}$
						$\hat{q}^{[3]}$
				$\hat{q}^{[2]}$		
				$\mathsf{a}_{\mathsf{EW}}{}^{[2]}$		
$\hat{q}^{[0]}$						
$\mathsf{a}'_{\mathsf{WW}}{}^{[0]}$						
						$\mathsf{a}_{\mathsf{WW}}{}^{[3]}$
						$q^{[3]}$
				$q^{[2]}$		
				$\langle @0^{[2]}$		
				$q^{[2]}$		
				$\langle @1^{[2]}$		
						$\langle @0^{[3]}$
					$\vdots\vdots$	
						$q^{[3]}$
				$q^{[2]}$		
				$\rangle @1^{[2]}$		
						$\rangle @0^{[3]}$
						$q^{[3]}$
				$q^{[2]}$		
				$\sharp^{[2]}$		
						$\sharp^{[3]}$
						$q^{[3]}$
				$q^{[2]}$		
				$\langle @1^{[2]}$		
						$\langle @0^{[3]}$
					$\vdots\vdots$	
						$q^{[3]}$
				$q^{[2]}$		
				$\rangle @1^{[2]}$		
						$\rangle @0^{[3]}$
						$q^{[3]}$
				$q^{[2]}$		
				$\rangle @0^{[2]}$		
				$q^{[2]}$		
				$\sharp^{[2]}$		
						$\sharp^{[3]}$
					$\vdots\vdots$	
						$q^{[3]}$
				$q^{[2]}$		
				$\checkmark^{[2]}$		
						$\checkmark^{[3]}$

where we have omitted the iteration of $\hat{q}^{[2]}.\mathsf{a_{EW}}^{[2]}.\hat{q}^{[0]}.\mathsf{a'_{WW}}^{[0]}$ for the lack of space.

With m-views, there is clearly a finite table $\mathcal{A}(\mathit{fix}_A)_m$ that implements $\mathcal{A}(\mathit{fix}_A)^{\circledS}_m$. It is then not hard to see that $\mathsf{st}(\mathcal{A}(\mathit{fix}_A)) = \mathit{fix}_A$ holds, showing that fix_A is viable. Also, it is easy to see that $\mathcal{A}(\mathit{fix}_A)$ is standard.

Note that we have shown that every PCF-atomic elements of the set $\mathcal{DPCF}$ (Definition 4.2.9) is realized by a standard st-algorithm.

4.4.3 PCF-Completeness of Viable Dynamic Strategies

In the last two sections, we have seen through examples that PCF-atomic dynamic strategies (Definition 4.2.9) are all viable (and realized by *standard* st-algorithms). In addition, Theorem 4.4.13 shows that constructions on dynamic strategies preserve viability. From these two facts, our main theorem immediately follows:

Theorem 4.4.18 (Main theorem I)**.** *Every static strategy $\sigma : S_\sigma$ definable by PCF has a viable dynamic strategy $\phi_\sigma : D_\sigma$ that satisfies $\sigma = \mathcal{H}^\omega(\phi_\sigma) : \mathcal{H}^\omega(D_\sigma) \trianglelefteq S_\sigma$.*

Proof. By lemma 4.2.10, examples in Sections 4.4.1-4.4.2 and Theorem 4.4.13. □

Since PCF is *Turing complete* [82, 122], this result particularly implies:

Corollary 4.4.19 (Turing completeness)**.** *Every partial recursive function $f : \mathbb{N}^k \rightharpoonup \mathbb{N}$, where $k \in \mathbb{N}$ and $\mathbb{N}^k \overset{\text{df.}}{=} \underbrace{\mathbb{N} \times \mathbb{N} \cdots \times \mathbb{N}}_{k}$, has a viable dynamic strategy $\phi_f : D_f$ that satisfies $\mathcal{H}^\omega(D_f) \trianglelefteq \mathcal{N}^k \Rightarrow \mathcal{N}$, where $\mathcal{N}^k \overset{\text{df.}}{=} \underbrace{\mathcal{N} \& \mathcal{N} \dots \& \mathcal{N}}_{k}$, and, for all $(n_1, n_2, \dots, n_k) \in \mathbb{N}^k$, $\mathcal{H}^\omega(\langle \underline{n_1}, \underline{n_2}, \dots, \underline{n_k} \rangle^\dagger \ddagger \phi_f) \simeq \underline{f(n_1, n_2, \dots, n_k)}$.*

Proof. Let $\mathsf{x_1} : \mathsf{N}, \mathsf{x_2} : \mathsf{N}, \dots, \mathsf{x_k} : \mathsf{N} \vdash \mathsf{F} : \mathsf{N}$ be a term of PCF that implements a given partial recursive function $f : \mathbb{N}^k \rightharpoonup \mathbb{N}$, i.e., $\mathsf{F}[\mathsf{n_1}/\mathsf{x_1}, \mathsf{n_2}/\mathsf{x_2}, \dots, \mathsf{n_k}/\mathsf{x_k}]$ evaluates to $\mathsf{f(n_1, n_2, \dots, n_k)}$ if $f(n_1, n_2, \dots, n_k)$ is defined and diverges otherwise, for all $n_i \in \mathbb{N}$ ($i = 1, 2, \dots, k$), where $\mathsf{n} : \mathsf{N}$ is the n^{th}-numeral, and $\mathsf{F}[\mathsf{n_1}/\mathsf{x_1}, \mathsf{n_2}/\mathsf{x_2}, \dots, \mathsf{n_k}/\mathsf{x_k}]$ is the result of substituting $\mathsf{n_i}$ for $\mathsf{x_i}$ in F for $i = 1, 2, \dots, k$ (see, e.g., [82, 122] for how to construct F from f). Then, there is a normalized dynamic strategy $\sigma_f : \mathcal{N}^k \Rightarrow \mathcal{N}$ in $\mathcal{PCF}$ that interprets F, following the static game semantics of PCF [14]. By Theorem 4.4.18, there is a viable dynamic strategy $\phi_f : D_f$ such that $\mathcal{H}^\omega(\phi_f) = \sigma_f$ and $\mathcal{H}^\omega(D_f) \trianglelefteq \mathcal{N}^k \Rightarrow \mathcal{N}$. Hence, $\mathcal{H}^\omega(\langle \underline{n_1}, \underline{n_2}, \dots, \underline{n_k} \rangle^\dagger \ddagger \phi_f) = \mathcal{H}^\omega(\langle \underline{n_1}, \underline{n_2}, \dots, \underline{n_k} \rangle^\dagger); \mathcal{H}^\omega(\phi_f) = \langle \underline{n_1}, \underline{n_2}, \dots, \underline{n_k} \rangle^\dagger; \sigma_f \simeq \underline{f(n_1, n_2, \dots, n_k)}$, where the last Kleene equality $\simeq$ is by the *adequacy* [18] of the static game semantics of PCF [100, 14]. □

Remark. Crucially, there is clearly a partial recursive function $f : \mathbb{N}^k \rightharpoonup \mathbb{N}$ such that σ_f is *not* viable (but ϕ_f is viable) by the finitary nature of tables of st-algorithms.

As our game-semantic model of computation is Turing complete, some of the well-known theorems in computability theory [46, 157] are immediately generalized (in the sense that they are not restricted to computation on natural numbers):

Corollary 4.4.20 (Generalized smn-theorem)**.** *If dynamic strategies $\sigma_i : T \Rightarrow A_i$, $i = 1, 2, \dots, n$, and $\phi : D$ such that $\mathcal{H}^\omega(D) \trianglelefteq A_1 \& A_2 \dots \& A_n \& B_1 \& B_2 \dots \& B_m \Rightarrow C$ are all realized by (standard) st-algorithms, then we may obtain a (standard) st-algorithm that realizes a viable dynamic strategy $\phi_{\sigma_1, \sigma_2, \dots, \sigma_n} : D_{A_1, A_2, \dots, A_n}$ such that:*

1. *$\mathcal{H}^\omega(\phi_{\sigma_1, \sigma_2, \dots, \sigma_n}) : \mathcal{H}^\omega(D_{A_1, A_2, \dots, A_n}) \trianglelefteq T \& B_1 \& B_2 \dots \& B_m \Rightarrow C$;*
2. *$\langle \{\boldsymbol{\epsilon}\}, \tau_1, \tau_2, \dots, \tau_m \rangle^\dagger \ddagger \phi_{\sigma_1, \sigma_2, \dots, \sigma_n} \simeq \langle \sigma_1, \sigma_2, \dots, \sigma_n, \tau_1, \tau_2, \dots, \tau_m \rangle^\dagger \ddagger \phi$ for any dynamic strategies $\tau_j : T \Rightarrow B_j$ ($j = 1, 2, \dots, m$).*

Proof. We define $\phi_{\sigma_1, \sigma_2, \dots, \sigma_n} \overset{\text{df.}}{=} \underbrace{\Lambda^\ominus(\cdots \Lambda^\ominus}_{m}(\langle \sigma_1, \sigma_2, \dots, \sigma_n \rangle^\dagger \ddagger \underbrace{\Lambda(\cdots \Lambda}_{m}(\phi) \cdots)) \cdots)$, where the proof of Theorem 4.4.13 describes how to 'effectively' obtain a standard st-algorithm that realizes $\phi_{\sigma_1, \sigma_2, \dots, \sigma_n}$ in an informal sense.[11] Note that Corollary 4.4.19 implies that it is in fact a generalization of the conventional smn-theorem [46]. □

Corollary 4.4.21 (Generalized FRT)**.** *Given a viable dynamic strategy $\varphi : D$ realized by a standard st-algorithm such that $\mathcal{H}^\omega(D) \trianglelefteq A \Rightarrow A$ for some normalized dynamic game A, there is another viable dynamic strategy $\sigma_\varphi : D_\varphi$ realized by a standard st-algorithm such that $\mathcal{H}^\omega(D_\varphi) \trianglelefteq T \Rightarrow A$ and $\mathcal{H}^\omega(\sigma_\varphi^\dagger \ddagger \varphi) = \mathcal{H}^\omega(\sigma_\varphi) : T \Rightarrow A$.*

Proof. Let $D_\varphi \overset{\text{df.}}{=} (T \Rightarrow D)^\dagger \ddagger ((A \Rightarrow A) \Rightarrow A)$ and $\sigma_\varphi \overset{\text{df.}}{=} (\varphi^T)^\dagger \ddagger \mathit{fix}_A$. Then, $\mathcal{H}^\omega(D_\varphi) = \mathcal{H}^\omega((T \Rightarrow D)^\dagger \ddagger ((A \Rightarrow A) \Rightarrow A)) \trianglelefteq \mathcal{H}^\omega((T \Rightarrow !(A \Rightarrow A)) \ddagger ((A \Rightarrow A) \Rightarrow A)) \trianglelefteq T \Rightarrow A$ by Lemmata 3.3.29 and 3.3.30, and $\mathcal{H}^\omega(\sigma_\varphi^\dagger \ddagger \varphi) = (\mathcal{H}^\omega(\varphi^T)^\dagger ; \mathit{fix}_A)^\dagger ; \mathcal{H}^\omega(\varphi) = \mathcal{H}^\omega(\varphi^T)^\dagger ; \mathit{fix}_A = \mathcal{H}^\omega((\varphi^T)^\dagger \ddagger \mathit{fix}_A) = \mathcal{H}^\omega(\sigma_\varphi)$ by Lemma 3.3.52, where [101] shows that fix_A in fact computes fixed-points. Again, Corollary 4.4.19 implies that it is in fact a generalization of the conventional first recursion theorem (FRT) [46]. □

[11] It is interesting future work to formalize this informal 'effective computability' by certain viable strategies.

4.5 Conclusion and Future Work of the Chapter

We have given a novel notion of 'effective computability' in game semantics, namely, *viability* of strategies. Due to its *intrinsic*, *non-inductive* and *non-axiomatic* natures, it gives a fundamental investigation of 'effective' computation beyond classical one, where note that viability of strategies makes sense *universally*, i.e., regardless of the underlying games (e.g., games do not have to correspond to types of PCF). Furthermore, our game-semantic model of computation formulates both high-level and low-level computational processes, and defines 'effectivity' of the former in terms of the latter, which sheds new light on the very notion of computation. For instance, dynamic strategies $\underline{n} : \mathcal{N}$ may be seen as the *definition* of natural numbers, and thus a viable dynamic strategy of the form $\phi : \mathcal{N}^k \Rightarrow \mathcal{N}$ can be regarded as an abstract, high-level computational process on natural numbers, not on their representations, and (the table of) an st-algorithm $\mathcal{A}(\phi)$ that realizes ϕ can be seen as a 'symbolic implementation' of ϕ. Moreover, the instruction strategies induced by $\mathcal{A}(\phi)$ can be seen as the corresponding low-level computational processes.

There are various directions for further work. First, we need to analyze the exact computational power of viable dynamic strategies, in comparison with other known notion of higher-order computability [122]. Also, as an application, the present framework may give an accurate measure for computational complexity [115], where note that dynamic games and strategies have already given such a measure via internal moves, but the present work may refine it further since two single steps in a dynamic game G may take different numbers of steps in the instruction game $\mathcal{G}(M_G)^3 \Rightarrow \mathcal{G}(M_G)$. Moreover, it may be possible to define computational complexity *relative* to that of oracle (or Opponent) computation, which would be an accurate measure for computational complexity of *higher-order* computation. It is also of theoretical interest to see which theorems in recursion theory can be generalized by the present framework in addition to the smn- and the first recursion theorems. Nevertheless, the most imminent future work is, by exploiting the flexibility of game semantics, to enlarge the scope of the present work (i.e., not only the language PCF) in order to establish a computational model of various logics and programming languages. We are particularly interested in how to apply our approach to *non-innocent* strategies.

Next, although the solely game-semantic approach gives a novel mathematical model of computation, one may wonder in terms of the well-established hierarchy of automata (i.e., the *Chomsky hierarchy* [33, 169]) which class of automata suffices to 'implement' the computation of viable dynamic strategies modeling PCF. Of course it is not very surprising if TMs suffice, but all the symbol manipulations executed in the low-level computational processes for viable dynamic strategies modeling PCF are actually *very* simple and also *non-erasing* (i.e., a move in a play will never be erased or replaced once performed). Hence, we are naturally led to ask:

> Are there any automata that are strictly weaker than TMs yet powerful enough to 'implement' all the dynamic strategies modeling PCF?

We leave it as future work to answer this question.

Finally, let us propose two more open questions. Since the definition of viable strategies is somewhat reflexive (as it is via strategies), we may naturally consider strategies *that can be realized by a viable strategy.* Let us define such strategies to be *2-viable.* More generally, rephrasing viability as *1-viability*, we define a strategy to be $(n+1)$*-viable* if it can be realized by an n-viable strategy for each $n \in \mathbb{N}$. Clearly, any n-viable strategy is 'effective' in an informal sense. Then, the first questions is:

> Is the class of all $(n+1)$-viable dynamic strategies strictly larger than that of all n-viable strategies for each $n \in \mathbb{N}$?

This question seems highly interesting from a theoretical perspective.[12] If the answer is positive for each $n \in \mathbb{N}$, then there would be an infinite hierarchy of generalized viable dynamic strategies. It is then natural to ask the following second question:

> Does the hierarchy, if it exists, correspond to any known hierarchy (perhaps in recursion theory or proof theory)?

We aim to answer these questions as future work as well.

[12]Note that this question would not have arisen in the first place if we had not defined 'effective computability' of strategies *solely in terms of games and strategies.*

Chapter 5

Game Semantics of MLTT

5.1 Introduction to the Chapter

In this last main chapter, we give a game semantics of *Martin-Löf type theory (MLTT)* for the motivation explained in Section 1.3.

5.1.1 Why Difficult?

Since the early 1990's, game semantics has been highly successful in giving semantics of various programming languages [9, 100, 139, 117, 94, 13, 12, 97, 6, 8]; thus, it is a surprising fact that there has been only one game semantics of MLTT established so far [10], which is not a complete solution either (see Section 5.1.3 for this point).

In contrast, a set-theoretic interpretation of MLTT is relatively straightforward. A *dependent type* may be interpreted as a family $B = (B(a))_{a \in A}$ of sets $B(a)$ indexed by elements a of another set A. Then, a *dependent function type* (or a $\mathsf{\Pi}$-*type*) from A to B is modeled by a set of functions $f : A \to \bigcup_{a \in A} B(a)$ such that $f(a) \in B(a)$ for all $a \in A$, called *dependent functions*, which is a generalization of a set of functions. Also, a *dependent pair type* (or a $\mathsf{\Sigma}$-*type*) of A and B is interpreted as a set of pairs (a, b) of elements $a \in A$ and $b \in B(a)$, called *dependent pairs*, which is a generalization a cartesian product of sets. In fact, there is a domain-theoretic model of MLTT [147] obtained by sophisticating this idea by order-theoretic structures (though it models *partial* (i.e., not every program normalizes) MLTT).

Thus, it would be helpful for understanding the content of the chapter to think of why it is difficult to give a game semantics of MLTT. A possible answer is:

It is not clear how to interpret type dependency via intensional processes.

Note that such an intensional semantics of MLTT would be meaningful as explained in Section 1.1 and therefore worth pursuing. A dependent type is not a problem; we may interpret it as a family B of games $B(\sigma)$ indexed by strategies σ on another game A. However, since a strategy $\sigma : A$ is usually 'gradually revealed' as a play proceeds, it seems impossible for Player in the game for the Π-type from A to B to determine, at the beginning of a play, Opponent's strategy $\sigma : A$ in the domain, let alone the component game $B(\sigma)$ in the codomain. Moreover, even when a play has been completed, there might be more than one strategy $\sigma : A$ that corresponds to the play. That is, the problem is in the point that we cannot determine a *fibre* $\sigma \to B(\sigma)$ to play. A similar point is applied to games for Σ-types.

5.1.2 Our Solution in a Nutshell

Our solution for the problem is to incorporate an extensional structure into games to model type dependency in MLTT that is in accordance with the intensional structure of games. More specifically, we require that the participants of a game have to 'declare' their strategies in mind before a play of the game begins. To respect the intensional nature of games mentioned above, we define the 'declarations' by the participants as 'invisible' to each other, so that the chosen strategies are still 'gradually revealed' as a play proceeds. In this way, we model type dependency by games in a similar manner to the domain-theoretic model of MLTT, keeping the intensional nature of games.

This idea leads to a generalization of games, called *predicative (p-) games.* The mathematical structure of p-games is radically different from that of conventional games, and thus it is challenging to give a CCC, let alone a categorical model of MLTT, of p-games. The main technical achievement of the chapter is to overcome it.

5.1.3 Related Work and Contribution of the Chapter

In the literature, several interpretations of MLTT that may be seen as a mathematical formalization of the meaning explanation (or the BHK-interpretation) have been proposed: *domain models* [147] and *realizability models* [156, 170, 20, 17, 39]. However, domain models interpret programs as (continuous) functions, which cannot capture *dynamics* or *intensionality* of MLTT; realizability models take, as *realizers*, e.g., (codes of) TMs or λ-terms, but TMs are *too 'low-level'* to capture type-theoretic phenomena, and λ-terms are *syntax* (not suitable for our aim; see Chapter 1).

The game semantics of dependent types [10] is certainly mathematical, syntax-independent and intensional. This work is definitely a significant achievement: It is the first game semantics of dependent type theories, and it gives a full completeness result for a certain fragment of (a variant of) MLTT. However, their mathematical structures are not very novel; their games and strategies are just the variant of [9]; consequently, their model is rather involved and not completely game-semantic in the following sense. For instance, their interpretation of Π-types is defined by a complex induction, following the recipe in [8] for the interpretation of (second-order) universal quantification. Even worse, they interpret Σ-types not by games but (finite) lists of *games with dependency* (which are certain families of games indexed by strategies), called *context games*; the name is misleading as a context game is not a game.

More specifically, they define:

- A ***game with dependency*** on a game A to be a pair of a game $\mathscr{U}(B)$ and a function $B : \mathsf{str}(A) \to \mathsf{sub}(\mathscr{U}(B))$, where $\mathsf{str}(A)$ is the set of all strategies on A, and $\mathsf{sub}(\mathscr{U}(B))$ is the set of all subgames of $\mathscr{U}(B)$;
- A ***context game*** to be a finite list $(X_1, X_2, \dots, X_k)$ such that each element X_i $(i = 1, 2, \dots, k)$ is a game with dependency on $\mathscr{U}(X_1) \& \mathscr{U}(X_2) \& \dots \& \mathscr{U}(X_{i-1})$.

In other words, they do not give a game-semantic counterpart of Σ-types but just employ the general *list* construction which does not exploit any game-semantic structure. Consequently, they model types and terms of MLTT respectively by context games and *lists* of strategies. In this sense, one may say that their model is not completely game-semantic, and to some degree they lost the conceptual naturality of game semantics. Moreover, their game semantics does not interpret universes at all.

In contrast, our game semantics of MLTT consists of a novel variant of games, viz., p-games, which naturally generalizes conventional games. Notably, they are defined *in terms of strategies*, which matches the meaning explanation that defines a formula by specifying its proofs (see Section 5.2); mathematically, it allows us to interpret Σ-types elegantly. Consequently, we may interpret types and terms of MLTT respectively by p-games and strategies on them, rather than lists of games and strategies, and give reasonably simple, natural interpretation of Π- and Σ-types as well as a cumulative hierarchy of universes. In particular, the interpretation of universes clarifies the difference between (dependent) types and terms of universes.[1]

On the other hand, our full completeness result is much weaker than theirs since it relies on an *inductive* construction of p-games and strategies though in the presence of universes a non-inductive full completeness seems much harder to establish.

[1]This distinction is not always clear, e.g., types and terms of universes are *identified* in [184].

5.1.4 Chapter Outline

This longest chapter is structured as follows. We first recall the syntax of MLTT in Section 5.2, where we also explain briefly how the meaning explanation extends and refines the BHK-interpretation of intuitionistic logic. Next, in Section 5.3, we develop the central notion of p-games, based on which, as the first highlight of the chapter, we give a *category with families (CwF)*, an abstract model of MLTT, of p-games in Section 5.4. The CwF of p-games induces an *injective* (for types built without N- and Id-types) model of the *intensional* variant of MLTT equipped with $\mathsf{1}$-, $\mathsf{0}$-, N-, $\mathsf{\Pi}$-, $\mathsf{\Sigma}$- and Id-types. It, however, cannot interpret universes completely; also, it is not surjective. We fix these problems in Section 5.5, by carving out certain elements in the CwF, giving rise to a *recursive*, *bijective* (again for types built without N- and Id-types) subCwF that interprets types mentioned above as well as the cumulative hierarchy of universes. We then investigate the degree of intensionality of the resulting game semantics in terms of several type-theoretic principles in Section 5.6: It refutes the principles of *equality reflection (EqRelf)* and *function extensionality (FunExt)* as well as *univalence axiom (UA)*, and it validates the principle of *uniqueness of identity proofs (UIP)* and Streicher's three *criteria of intensionality (CoI)*. We finally draw a conclusion and propose further work in Section 5.7.

5.2 Martin-Löf Type Theory

Let us begin with recalling our target language, (the *intensional* variant of) MLTT in the style of [92] except that universes are *cumulative*, and judgements for types contain additional information (for universes): the *ranks* of types. For injectivity of our interpretation, we adopt the *uniqueness (or η-) rules* of $\mathsf{1}$-, $\mathsf{\Pi}$- and $\mathsf{\Sigma}$-types.

Note that here we mainly focus on the syntax and just briefly explain the meaning explanation; for a comprehensive introduction to MLTT, see, e.g., [140, 126].

5.2.1 Judgements

MLTT is a formal logical system similar to *natural deduction* [65, 180] except that vertices of a derivation tree are ***judgements***, for which we usually write $\mathcal{J}$. There are the following six kinds of judgements (followed by their intended meanings):

- $\vdash \mathsf{\Gamma}\ \mathsf{ctx}$ ($\mathsf{\Gamma}$ is a ***context***);
- $\mathsf{\Gamma} \vdash \mathsf{A}\ \mathsf{type_i}$ (A is a ***type*** of ***rank*** $i \in \mathbb{N}^+$ in the context $\mathsf{\Gamma}$);

- $\Gamma \vdash \mathsf{a} : \mathsf{A}$ (a is a ***term*** (or ***program***) of type A in the context Γ);
- $\vdash \Gamma \equiv \Delta\ \mathsf{ctx}$ (Γ and Δ are ***judgmentally equal*** contexts);
- $\Gamma \vdash \mathsf{A} \equiv \mathsf{B}\ \mathsf{type_i}$ (A and B are ***judgmentally equal*** types of rank i in Γ);
- $\Gamma \vdash \mathsf{a} \equiv \mathsf{a}' : \mathsf{A}$ (a and a' are ***judgmentally equal*** terms of type A in Γ)

where we often omit the subscript i in $\mathsf{type_i}$ when the rank i is irrelevant (n.b., we may recover the usual syntax if we completely ignore the ranks). That is, MLTT consists of *axioms* $\frac{}{\mathcal{J}}$ and *(inference) rules* $\frac{\mathcal{J}_1\ \mathcal{J}_2 \dots \mathcal{J}_k}{\mathcal{J}_0}$, which are to make a *conclusion* from *hypotheses* by constructing a derivation (tree) exactly as natural deduction does.

In Sections 5.2.2–5.2.10 below, we present the axioms and rules of MLTT.

Remark. One may say that there is just a single kind of judgements $\Gamma \vdash \mathsf{A}$ in natural deduction, which are intended to mean that 'the formula A is true in the context Γ'.

Each type construction in MLTT is defined by its ***formation***, ***introduction***, ***elimination*** and ***computation*** rules. The formation rule stipulates how to form the type, and the introduction rule defines the *canonical terms* (see Section 5.2.11 below) of the type. Then, the elimination and computation rules describe respectively how to consume the canonical terms and the result of such a consumption (in the form of an equation), both of which are easily justified by the introduction rule.

As often expressed by 'MLTT *internalizes* the Curry-Howard isomorphism', its contexts, types and programs are meant to be *assumptions* (or *premises*), *formulas* and *proofs* in logic, respectively, as well. Therefore, e.g., the judgement $\Gamma \vdash \mathsf{a} : \mathsf{A}$ can be read as 'the formula A has the proof a under the assumption Γ', and so on.

5.2.2 Contexts

A ***context*** is a finite sequence $\mathsf{x}_1 : \mathsf{A}_1, \mathsf{x}_2 : \mathsf{A}_2, \dots, \mathsf{x}_n : \mathsf{A}_n$ of pairs of a variable x_i and a type A_i such that the variables $\mathsf{x}_1, \mathsf{x}_2, \dots, \mathsf{x}_n$ are pair-wise distinct. We write $\diamondsuit$ for the *empty context*, i.e., the empty sequence $\boldsymbol{\epsilon}$; we usually omit $\diamondsuit$ when it appears on the left-hand side of the symbol $\vdash$ in a judgement.

We have the following axiom and rules for contexts:

$$\frac{}{\vdash \diamondsuit\ \mathsf{ctx}}\ (\textsc{Ctx-Emp}) \qquad \frac{\Gamma \vdash \mathsf{A}\ \mathsf{type_i}}{\vdash \Gamma, \mathsf{x} : \mathsf{A}\ \mathsf{ctx}}\ (\textsc{Ctx-Ext})$$

$$\frac{\vdash \Gamma \equiv \Delta\ \mathsf{ctx} \quad \Gamma \vdash \mathsf{A} \equiv \mathsf{B}\ \mathsf{type_i}}{\vdash \Gamma, \mathsf{x} : \mathsf{A} \equiv \Delta, \mathsf{y} : \mathsf{B}\ \mathsf{ctx}}\ (\textsc{Ctx-ExtEq})$$

where x (resp. y) does not occur in Γ (resp. Δ).

The rules Ctx-Emp and Ctx-Ext determine that contexts are exactly finite lists of pairs of a variable and a type. The rule Ctx-ExtEq is a *congruence rule*, i.e., it states that judgmental equality is preserved under 'context extension'.

Convention. We will skip writing down congruence rules for the other constructions.

5.2.3 Structural Rules

Here, we collect the rules applied to all types as ***structural rules***:

$$\frac{\vdash \mathsf{x}_1 : \mathsf{A}_1, \mathsf{x}_2 : \mathsf{A}_2, \dots, \mathsf{x}_n : \mathsf{A}_n \ \mathsf{ctx}}{\mathsf{x}_1 : \mathsf{A}_1, \mathsf{x}_2 : \mathsf{A}_2, \dots, \mathsf{x}_n : \mathsf{A}_n \vdash \mathsf{x}_j : \mathsf{A}_j} \ (\textsc{Var})$$

$$\frac{\vdash \Gamma \ \mathsf{ctx}}{\vdash \Gamma \equiv \Gamma \ \mathsf{ctx}} \ (\textsc{Ctx-EqRefl}) \qquad \frac{\vdash \Gamma \equiv \Delta \ \mathsf{ctx}}{\vdash \Delta \equiv \Gamma \ \mathsf{ctx}} \ (\textsc{Ctx-EqSym})$$

$$\frac{\vdash \Gamma \equiv \Delta \ \mathsf{ctx} \quad \vdash \Delta \equiv \Theta \ \mathsf{ctx}}{\vdash \Gamma \equiv \Theta \ \mathsf{ctx}} \ (\textsc{Ctx-EqTrans})$$

$$\frac{\Gamma \vdash \mathsf{A} \ \mathsf{type_i}}{\Gamma \vdash \mathsf{A} \equiv \mathsf{A} \ \mathsf{type_i}} \ (\textsc{Ty-EqRefl}) \qquad \frac{\Gamma \vdash \mathsf{A} \equiv \mathsf{B} \ \mathsf{type_i}}{\Gamma \vdash \mathsf{B} \equiv \mathsf{A} \ \mathsf{type_i}} \ (\textsc{Ty-EqSym})$$

$$\frac{\Gamma \vdash \mathsf{A} \equiv \mathsf{B} \ \mathsf{type_i} \quad \Gamma \vdash \mathsf{B} \equiv \mathsf{C} \ \mathsf{type_i}}{\Gamma \vdash \mathsf{A} \equiv \mathsf{C} \ \mathsf{type_i}} \ (\textsc{Ty-EqTrans})$$

$$\frac{\Gamma \vdash \mathsf{a} : \mathsf{A}}{\Gamma \vdash \mathsf{a} \equiv \mathsf{a} : \mathsf{A}} \ (\textsc{Tm-EqRefl}) \qquad \frac{\Gamma \vdash \mathsf{a} \equiv \mathsf{a}' : \mathsf{A}}{\Gamma \vdash \mathsf{a}' \equiv \mathsf{a} : \mathsf{A}} \ (\textsc{Tm-EqSym})$$

$$\frac{\Gamma \vdash \mathsf{a} \equiv \mathsf{a}' : \mathsf{A} \quad \Gamma \vdash \mathsf{a}' \equiv \mathsf{a}'' : \mathsf{A}}{\Gamma \vdash \mathsf{a} \equiv \mathsf{a}'' : \mathsf{A}} \ (\textsc{Tm-EqTrans})$$

$$\frac{\vdash \Gamma \equiv \Delta \ \mathsf{ctx} \quad \Gamma \vdash \mathsf{A} \ \mathsf{type_i}}{\Delta \vdash \mathsf{A} \ \mathsf{type_i}} \ (\textsc{Ty-Conv})$$

$$\frac{\Gamma \vdash \mathsf{a} : \mathsf{A} \quad \vdash \Gamma \equiv \Delta \ \mathsf{ctx} \quad \Gamma \vdash \mathsf{A} \equiv \mathsf{B} \ \mathsf{type_i}}{\Delta \vdash \mathsf{a} : \mathsf{B}} \ (\textsc{Tm-Conv})$$

where $j \in \{1, 2, \dots, n\}$.

The rule Var states the reasonable idea that we may give an element $\mathsf{x}_j : \mathsf{A}_j$ if it occurs in the context just by 'copy-catting' it. The next nine rules stipulate that every judgmental equality is an equivalence relation. Finally, the rules Ty-Conv and

Tm-Conv formalize the natural phenomenon that judgements are preserved under the exchange of judgmentally equal contexts and/or types.

The following ***weakening*** and ***substitution*** rules are *admissible* (or *derivable*) in MLTT, but it is convenient to present them explicitly:

$$\frac{\mathsf{\Gamma, \Delta \vdash J \quad \Gamma \vdash A\ type_i}}{\mathsf{\Gamma, x : A, \Delta \vdash J}}\ (\textsc{Weak}) \qquad \frac{\mathsf{\Gamma, x : A, \Delta \vdash J \quad \Gamma \vdash a : A}}{\mathsf{\Gamma, \Delta[a/x] \vdash J[a/x]}}\ (\textsc{Subst})$$

where x does not occur in $\mathsf{\Gamma}$ or $\mathsf{\Delta}$ for Weak, and not in $\mathsf{\Gamma}$ for Subst, and $\mathsf{J[a/x]}$ (resp. $\mathsf{\Delta[a/x]}$) denotes the *capture-free substitution* [85, 92] of a for x in J[2] (resp. Δ).

5.2.4 Unit Type

We proceed to introduce specific type constructions. Let us begin with the simplest type, called the ***unit type*** (or the ***1-type***) 1, which is the type that has just one canonical term $\star$. Thus, from the logical point of view, it is the simplest true formula.

Its rules are the following:

$$\frac{\mathsf{\vdash \Gamma\ ctx}}{\mathsf{\Gamma \vdash 1\ type_1}}\ (1\text{-}\textsc{Form}) \qquad \frac{\mathsf{\vdash \Gamma\ ctx}}{\mathsf{\Gamma \vdash \star : 1}}\ (1\text{-}\textsc{Intro}) \qquad \frac{\mathsf{\Gamma \vdash t : 1}}{\mathsf{\Gamma \vdash t \equiv \star : 1}}\ (1\text{-}\textsc{Uniq})$$

$$\frac{\mathsf{\Gamma, x : 1 \vdash C\ type_i \quad \Gamma \vdash c : C[\star/x] \quad \Gamma \vdash t : 1}}{\mathsf{\Gamma \vdash R^1(C, c, t) : C[t/x]}}\ (1\text{-}\textsc{Elim})$$

$$\frac{\mathsf{\Gamma, x : 1 \vdash C\ type_i \quad \Gamma \vdash c : C[\star/x]}}{\mathsf{\Gamma \vdash R^1(C, c, \star) \equiv c : C[\star/x]}}\ (1\text{-}\textsc{Comp})$$

Note that 1-Uniq implies 1-Elim and 1-Comp if we define $\mathsf{R^1(C, c, t)} \overset{\text{df.}}{\equiv} \mathsf{c}$.

The formation rule 1-Form states that it is an *atomic* type in the sense that we may form it without assuming any other type. It has the rank 1 as other atomic types do; this point will be clear when we have introduced our game-semantic counterpart of ranks. The introduction rule 1-Intro defines that it has just one canonical term, viz., $\star$. Thus, the uniqueness rule 1-Uniq should make sense, from which the remaining rules 1-Elim and 1-Comp immediately follow.

5.2.5 Empty Type

Next, let us introduce the ***empty type*** (or the ***0-type***), which is the type that has no canonical term. Thus, it corresponds in logic to the simplest false formula.

[2]Here, J denotes the righthand side of any judgement.

Its rules are the following:

$$\frac{\vdash \Gamma \ \mathsf{ctx}}{\Gamma \vdash 0 \ \mathsf{type}_1} \ (0\text{-}\textsc{Form}) \qquad \frac{\Gamma, x : 0 \vdash C \ \mathsf{type}_i \quad \Gamma \vdash a : 0}{\Gamma \vdash R^0(C, a) : C[a/x]} \ (0\text{-}\textsc{Elim})$$

The formation rule 0-Form is the same as that of the 1-type. The elimination rule 0-Elim corresponds in logic to *ex falso*, i.e., ‘anything follows from a contradiction’. The 0-type has no introduction or computation rule since it has no canonical term.

5.2.6 Natural Number Type

We proceed to introduce an important atomic type, called the ***natural number type*** (or the **N-*type***), which is, as the name indicates, the type of natural numbers.

Its rules are the following:

$$\frac{\vdash \Gamma \ \mathsf{ctx}}{\Gamma \vdash N \ \mathsf{type}_1} \ (N\text{-}\textsc{Form}) \qquad \frac{\vdash \Gamma \ \mathsf{ctx}}{\Gamma \vdash \mathsf{zero} : N} \ (N\text{-}\textsc{IntroZ}) \qquad \frac{\Gamma \vdash n : N}{\Gamma \vdash \mathsf{succ}(n) : N} \ (N\text{-}\textsc{IntroS})$$

$$\frac{\Gamma, x : N \vdash C \ \mathsf{type}_i \quad \Gamma \vdash c_z : C[\mathsf{zero}/x] \quad \Gamma, x : N, y : C \vdash c_s : C[\mathsf{succ}(x)/x] \quad \Gamma \vdash n : N}{\Gamma \vdash R^N(C, c_z, c_s, n) : C[n/x]} \ (N\text{-}\textsc{Elim})$$

$$\frac{\Gamma, x : N \vdash C \ \mathsf{type}_i \quad \Gamma \vdash c_z : C[\mathsf{zero}/x] \quad \Gamma, x : N, y : C \vdash c_s : C[\mathsf{succ}(x)/x]}{\Gamma \vdash R^N(C, c_z, c_s, \mathsf{zero}) \equiv c_z : C[\mathsf{zero}/x]} \ (N\text{-}\textsc{CompZ})$$

$$\frac{\begin{array}{c}\Gamma, x : N \vdash C \ \mathsf{type}_i \quad \Gamma \vdash c_z : C[\mathsf{zero}/x] \\ \Gamma, x : N, y : C \vdash c_s : C[\mathsf{succ}(x)/x] \quad \Gamma \vdash n : N\end{array}}{\Gamma \vdash R^N(C, c_z, c_s, \mathsf{succ}(n)) \equiv c_s[n/x, R^N(C, c_z, c_s, n)/y] : C[\mathsf{succ}(n)/x]} \ (N\text{-}\textsc{CompS})$$

Again, the formation rule N-Form states that the N-type is an atomic type. The introduction rules N-IntroZ and N-IntroZ inductively determine the canonical terms: zero (for $0 \in \mathbb{N}$) and succ(n) if so is n (for $n \in \mathbb{N} \Rightarrow n + 1 \in \mathbb{N}$). The elimination rule N-Elim represents both *mathematical induction* and *primitive recursion*: To show a predicate C over N, it suffices to prove C(zero) and C(n) implies C(succ(n)), or equivalently under the Curry-Howard isomorphism, to define a (dependent) function f from N to C, it suffices to define its outputs f(zero) on zero and f(succ(n)) on succ(n) in terms of f(n) and n. It makes sense since canonical terms of the N-type are exactly zero and successors. Finally, the computation rules N-CompZ and N-CompS stipulate the expected behavior of proofs or computations generated by N-Elim.

Notation. Given a context $\vdash \Gamma \ \mathsf{ctx}$ and a natural number $n \in \mathbb{N}$, we define the term $\Gamma \vdash \underline{n} : N$, called the ***$n^{th}$-numeral***, by induction on n as follows:

- (Base Case) $\Gamma \vdash \underline{0} \overset{\text{df.}}{\equiv} \mathsf{zero} : \mathsf{N}$;
- (Inductive Step) $\Gamma \vdash \underline{\mathsf{n}+1} \overset{\text{df.}}{\equiv} \mathsf{succ}(\mathsf{n}) : \mathsf{N}$.

The n^{th}-numeral is intended to represent the natural number n.

5.2.7 Dependent Function Types

Now, let us introduce a central non-atomic type construction, called ***dependent function types*** (or ***Π-types***). The Π-type $\Pi_{\mathsf{x}:\mathsf{A}}\mathsf{B}(\mathsf{x})$ is intended to represent the space of *dependent functions* from A to B as described in Section 5.1.1.

The rules of Π-types are the following:

$$\frac{\Gamma \vdash \mathsf{A}\ \mathsf{type}_{\mathsf{i}} \quad \Gamma, \mathsf{x} : \mathsf{A} \vdash \mathsf{B}\ \mathsf{type}_{\mathsf{j}}}{\Gamma \vdash \Pi_{\mathsf{x}:\mathsf{A}}\mathsf{B}\ \mathsf{type}_{\mathsf{max(i,j)}}}\ (\Pi\text{-}\textsc{Form}) \qquad \frac{\Gamma, \mathsf{x} : \mathsf{A} \vdash \mathsf{b} : \mathsf{B}}{\Gamma \vdash \lambda \mathsf{x}^{\mathsf{A}}.\mathsf{b} : \Pi_{\mathsf{x}:\mathsf{A}}\mathsf{B}}\ (\Pi\text{-}\textsc{Intro})$$

$$\frac{\Gamma \vdash \mathsf{f} : \Pi_{\mathsf{x}:\mathsf{A}}\mathsf{B} \quad \Gamma \vdash \mathsf{a} : \mathsf{A}}{\Gamma \vdash \mathsf{f}(\mathsf{a}) : \mathsf{B}[\mathsf{a}/\mathsf{x}]}\ (\Pi\text{-}\textsc{Elim}) \qquad \frac{\Gamma, \mathsf{x} : \mathsf{A} \vdash \mathsf{b} : \mathsf{B} \quad \Gamma \vdash \mathsf{a} : \mathsf{A}}{\Gamma \vdash (\lambda \mathsf{x}^{\mathsf{A}}.\mathsf{b})(\mathsf{a}) \equiv \mathsf{b}[\mathsf{a}/\mathsf{x}] : \mathsf{B}[\mathsf{a}/\mathsf{x}]}\ (\Pi\text{-}\textsc{Comp})$$

$$\frac{\Gamma \vdash \mathsf{f} : \Pi_{\mathsf{x}:\mathsf{A}}\mathsf{B}}{\Gamma \vdash \lambda \mathsf{x}^{\mathsf{A}}.\mathsf{f}(\mathsf{x}) \equiv \mathsf{f} : \Pi_{\mathsf{x}:\mathsf{A}}\mathsf{B}}\ (\Pi\text{-}\textsc{Uniq})$$

where x does not occur free in f for Π-Uniq.

The formation rule Π-Form states that we may form the Π-type $\Pi_{\mathsf{x}:\mathsf{A}}\mathsf{B}$ from types A and B, where B may depend on A. The introduction rule Π-Intro defines how to construct the canonical terms of $\Pi_{\mathsf{x}:\mathsf{A}}\mathsf{B}$; it is the usual way of defining a function f from A to B, i.e., to specify its output $\mathsf{f}(\mathsf{a}) : \mathsf{B}[\mathsf{a}/\mathsf{x}]$ on each input $\mathsf{a} : \mathsf{A}$. Then, the elimination and computation rules Π-Elim and Π-Comp should make sense. Finally, the uniqueness rule Π-Uniq stipulates that terms of Π-types are only canonical ones.

5.2.8 Dependent Pair Types

Another important type construction is ***dependent sum types*** (or ***Σ-types***), which represent the spaces of *dependent pairs* again as explained in Section 5.1.1.

The rules of Σ-types are the following:

$$\frac{\Gamma \vdash \mathsf{A\ type_i} \quad \Gamma, \mathsf{x} : \mathsf{A} \vdash \mathsf{B\ type_j}}{\Gamma \vdash \Sigma_{\mathsf{x:A}}\mathsf{B\ type_{max(i,j)}}} \ (\Sigma\text{-}\textsc{Form})$$

$$\frac{\Gamma, \mathsf{x} : \mathsf{A} \vdash \mathsf{B\ type} \quad \Gamma \vdash \mathsf{a} : \mathsf{A} \quad \Gamma \vdash \mathsf{b} : \mathsf{B}[\mathsf{a}/\mathsf{x}]}{\Gamma \vdash \langle \mathsf{a}, \mathsf{b} \rangle : \Sigma_{\mathsf{x:A}}\mathsf{B}} (\Sigma\text{-}\textsc{Intro})$$

$$\frac{\Gamma, \mathsf{z} : \Sigma_{\mathsf{x:A}}\mathsf{B} \vdash \mathsf{C\ type_i} \quad \Gamma, \mathsf{x} : \mathsf{A}, \mathsf{y} : \mathsf{B} \vdash \mathsf{g} : \mathsf{C}[\langle \mathsf{x}, \mathsf{y} \rangle/\mathsf{z}] \quad \Gamma \vdash \mathsf{p} : \Sigma_{\mathsf{x:A}}\mathsf{B}}{\Gamma \vdash \mathsf{R}^{\Sigma}([\mathsf{z} : \Sigma_{\mathsf{x:A}}\mathsf{B}]\mathsf{C}, [\mathsf{x} : \mathsf{A}, \mathsf{y} : \mathsf{B}]\mathsf{g}, \mathsf{p}) : \mathsf{C}[\mathsf{p}/\mathsf{z}]} \ (\Sigma\text{-}\textsc{Elim})$$

$$\frac{\Gamma, \mathsf{z} : \Sigma_{\mathsf{x:A}}\mathsf{B} \vdash \mathsf{C\ type_i} \quad \Gamma, \mathsf{x} : \mathsf{A}, \mathsf{y} : \mathsf{B} \vdash \mathsf{g} : \mathsf{C}[\langle \mathsf{x}, \mathsf{y} \rangle/\mathsf{z}] \quad \Gamma \vdash \mathsf{a} : \mathsf{A} \quad \Gamma \vdash \mathsf{b} : \mathsf{B}[\mathsf{a}/\mathsf{x}]}{\Gamma \vdash \mathsf{R}^{\Sigma}([\mathsf{z} : \Sigma_{\mathsf{x:A}}\mathsf{B}]\mathsf{C}, [\mathsf{x} : \mathsf{A}, \mathsf{y} : \mathsf{B}]\mathsf{g}, \langle \mathsf{a}, \mathsf{b} \rangle) \equiv \mathsf{g}[\mathsf{a}/\mathsf{x}, \mathsf{b}/\mathsf{y}] : \mathsf{C}[\langle \mathsf{a}, \mathsf{b} \rangle/\mathsf{z}]} \ (\Sigma\text{-}\textsc{Comp})$$

$$\frac{\Gamma \vdash \mathsf{p} : \Sigma_{\mathsf{x:A}}\mathsf{B}}{\Gamma \vdash \langle \pi_1^{\mathsf{A,B}}(\mathsf{p}), \pi_2^{\mathsf{A,B}}(\mathsf{p}) \rangle \equiv \mathsf{p} : \Sigma_{\mathsf{x:A}}\mathsf{B}} \ (\Sigma\text{-}\textsc{Uniq})$$

where

$$\Gamma \vdash \pi_1^{\mathsf{A,B}}(\mathsf{p}) \overset{\text{df.}}{\equiv} \mathsf{R}^{\Sigma}([\mathsf{z} : \Sigma_{\mathsf{x:A}}\mathsf{B}]\mathsf{A}, [\mathsf{x} : \mathsf{A}, \mathsf{y} : \mathsf{B}]\mathsf{x}, \mathsf{p}) : \mathsf{A};$$
$$\Gamma \vdash \pi_2^{\mathsf{A,B}}(\mathsf{p}) \overset{\text{df.}}{\equiv} \mathsf{R}^{\Sigma}([\mathsf{z} : \Sigma_{\mathsf{x:A}}\mathsf{B}]\mathsf{B}[\pi_1^{\mathsf{A,B}}(\mathsf{z})/\mathsf{x}], [\mathsf{x} : \mathsf{A}, \mathsf{y} : \mathsf{B}]\mathsf{y}, \mathsf{p}]) : \mathsf{B}[\pi_1^{\mathsf{A,B}}(\mathsf{p})/\mathsf{x}]$$

are *projections* constructed by Σ-Elim.

The formation rule Σ-Form is the same as that of Π-types. The introduction rule Σ-Intro specifies that canonical terms of a Σ-type $\Sigma_{\mathsf{x:A}}\mathsf{B}$ are dependent pairs $\langle \mathsf{a}, \mathsf{b} \rangle : \Sigma_{\mathsf{x:A}}\mathsf{B}$ of terms $\mathsf{a} : \mathsf{A}$ and $\mathsf{b} : \mathsf{B}[\mathsf{a}/\mathsf{x}]$. Again, the elimination and computation rules Σ-Elim and Σ-Comp should make sense by the introduction rule. Finally, the uniqueness rule Σ-Uniq stipulates that terms of Σ-types are only canonical ones, i.e., what are obtained by the introduction rule.

5.2.9 Identity Types

Note that a judgmental equality $\Gamma \vdash \mathsf{a} \equiv \mathsf{a}' : \mathsf{A}$ is a judgement, not a formula, and thus it cannot be used in a context nor derived by an induction principle. For this point, the following ***(intensional) identity types*** (or **Id**-***types***) have been introduced[3]:

[3]We can then for instance formulate and prove *Peano axioms* [148] in the presence of Id-types.

$$\frac{\Gamma \vdash \mathsf{A}\ \mathsf{type_i} \quad \Gamma \vdash \mathsf{a} : \mathsf{A} \quad \Gamma \vdash \mathsf{a'} : \mathsf{A}}{\Gamma \vdash \mathsf{a} =_{\mathsf{A}} \mathsf{a'}\ \mathsf{type_i}}\ (=\text{-}\textsc{Form}) \qquad \frac{\Gamma \vdash \mathsf{A}\ \mathsf{type_i} \quad \Gamma \vdash \mathsf{a} : \mathsf{A}}{\Gamma \vdash \mathsf{refl_a} : \mathsf{a} =_{\mathsf{A}} \mathsf{a}}\ (=\text{-}\textsc{Intro})$$

$$\frac{\begin{array}{l}\Gamma, \mathsf{x} : \mathsf{A}, \mathsf{y} : \mathsf{A}, \mathsf{p} : \mathsf{x} =_{\mathsf{A}} \mathsf{y} \vdash \mathsf{C}\ \mathsf{type_i} \quad \Gamma, \mathsf{z} : \mathsf{A} \vdash \mathsf{c} : \mathsf{C}[\mathsf{z}/\mathsf{x}, \mathsf{z}/\mathsf{y}, \mathsf{refl_z}/\mathsf{p}] \\ \Gamma \vdash \mathsf{a} : \mathsf{A} \quad \Gamma \vdash \mathsf{a'} : \mathsf{A} \quad \Gamma \vdash \mathsf{q} : \mathsf{a} =_{\mathsf{A}} \mathsf{a'}\end{array}}{\Gamma \vdash \mathsf{R}^{=}(\mathsf{C}, \mathsf{c}, \mathsf{a}, \mathsf{a'}, \mathsf{q}) : \mathsf{C}[\mathsf{a}/\mathsf{x}, \mathsf{a'}/\mathsf{y}, \mathsf{q}/\mathsf{p}]}\ (=\text{-}\textsc{Elim})$$

$$\frac{\Gamma, \mathsf{x} : \mathsf{A}, \mathsf{y} : \mathsf{A}, \mathsf{p} : \mathsf{x} =_{\mathsf{A}} \mathsf{y} \vdash \mathsf{C}\ \mathsf{type_i} \quad \Gamma, \mathsf{z} : \mathsf{A} \vdash \mathsf{c} : \mathsf{C}[\mathsf{z}/\mathsf{x}, \mathsf{z}/\mathsf{y}, \mathsf{refl_z}/\mathsf{p}] \quad \Gamma \vdash \mathsf{a} : \mathsf{A}}{\Gamma \vdash \mathsf{R}^{=}(\mathsf{C}, \mathsf{c}, \mathsf{a}, \mathsf{a}, \mathsf{refl_a}) \equiv \mathsf{c}[\mathsf{a}/\mathsf{z}] : \mathsf{C}[\mathsf{a}/\mathsf{x}, \mathsf{a}/\mathsf{y}, \mathsf{refl_a}/\mathsf{p}]}\ (=\text{-}\textsc{Comp})$$

The formation rule =-Form states that we may form an Id-type $\mathsf{a} =_{\mathsf{A}} \mathsf{a'}$ from a type A and terms $\mathsf{a}, \mathsf{a'} : \mathsf{A}$, whose rank is the same as that of A. The introduction rule =-Intro defines that there is just one canonical term $\mathsf{refl_a}$ of an Id-type $\mathsf{a} =_{\mathsf{A}} \mathsf{a'}$ if $\mathsf{a} \equiv \mathsf{a'}$ (and there is non otherwise). Therefore, again, the elimination and computation rules =-Elim and =-Comp make sense for the introduction rule.

5.2.10 Universes

As the last type construction, we assume the existence of a ***cumulative hierarchy of universes*** $\mathsf{U}_0, \mathsf{U}_1, \mathsf{U}_2, \dots$ The initial idea by Martin-Löf was to have a 'type U of all types' to increase the proof-theoretic strength of MLTT, e.g., it allows one to obtain, by N-Elim, a family of types $\mathsf{n} : \mathsf{N} \vdash \mathsf{FS^N(n)} : \mathsf{U}$[4] such that $\mathsf{FS^N(n)}$ is the type of n-tuples of natural numbers (where note that it is impossible for $\mathsf{n} : \mathsf{N} \vdash \mathsf{FS^N(n)}\ \mathsf{type}$). In fact, an early version of MLTT has such a ***single universe*** U, but it in particular implies $\Gamma \vdash \mathsf{U} : \mathsf{U}$, leading to its inconsistency known as *Girard's paradox* [69].

For this problem, Martin-Löf later excluded the judgement $\Gamma \vdash \mathsf{U} : \mathsf{U}$ [127], and further proposed the *cumulative hierarchy of universes* [124] in the *Tarski-style* [126], so that every type $\Gamma \vdash \mathsf{A}\ \mathsf{type}$ has its 'code' $\Gamma \vdash \mathsf{c} : \mathsf{U_i}$ for some $i \in \mathbb{N}$ such that $\Gamma \vdash \mathsf{El(c)} \equiv \mathsf{A}\ \mathsf{type}$, where $\mathsf{x} : \mathsf{U_i} \vdash \mathsf{El(x)}\ \mathsf{type}$ is a 'decoding' operation equipped with the universe $\mathsf{U_i}$. Then, in particular, every universe $\Gamma \vdash \mathsf{U_i}\ \mathsf{type}$ has its 'code' $\Gamma \vdash \mathsf{u_i} : \mathsf{U_j}$ for some $j > i$ such that $\Gamma \vdash \mathsf{El(u_i)} \equiv \mathsf{U_i}\ \mathsf{type}$ without inconsistency. We adopt this cumulative hierarchy of Tarski-style universes.

[4]Originally, the judgements $\Gamma \vdash \mathsf{A}\ \mathsf{type}$ and $\Gamma \vdash \mathsf{A} : \mathsf{U}$ are rather *identified* as in [184]. This variant is called the *Russell-style universes* [146].

Their basic inference rules are the following:

$$\frac{\vdash \Gamma\ \mathsf{ctx} \quad i \in \mathbb{N}}{\Gamma \vdash \mathsf{U_i}\ \mathsf{type_{i+2}}}\ (\text{U-Form}) \qquad \frac{\Gamma \vdash \mathsf{c} : \mathsf{U_i} \quad \forall j < i.\, \Gamma \nvdash \mathsf{c} : \mathsf{U_j}}{\Gamma \vdash \mathsf{El(c)}\ \mathsf{type_{i+1}}}\ (\text{U-Elim})$$

$$\frac{\Gamma \vdash \mathsf{c} : \mathsf{U_i}}{\Gamma \vdash \mathsf{c} : \mathsf{U_{i+1}}}\ (\text{U-Cumul})$$

where U-Cumul explains why they are called *cumulative.* The increments +2 (U-Form) and +1 (U-Elim) will be clear when we define games to interpret universes.

However, universes are not yet guaranteed to have the 'code' of every type. For this point, a standard approach is to introduce *constructions on the 'codes' of types* that correspond to constructions on 'smaller types' [126, 92], e.g., the 'code' $\Gamma \vdash \underline{\mathsf{N}} : \mathsf{U_0}$ such that $\Gamma \vdash \mathsf{El}(\underline{\mathsf{N}}) \equiv \mathsf{N}\ \mathsf{type_1}$ and the construction $\Gamma \vdash \underline{\Pi}(\underline{\mathsf{A}}, \underline{\mathsf{B}}) : \mathsf{U_{max(i,j)}}$ such that $\Gamma \vdash \mathsf{El}(\underline{\Pi}(\underline{\mathsf{A}}, \underline{\mathsf{B}})) \equiv \Pi(\mathsf{A}, \mathsf{B})\ \mathsf{type_{max(i,j)+1}}$ for any $\Gamma \vdash \underline{\mathsf{A}} : \mathsf{U_i}$ and $\Gamma, \mathsf{x} : \mathsf{El}(\underline{\mathsf{A}}) \vdash \underline{\mathsf{B}} : \mathsf{U_j}$.

Consequently, the following introduction and computation rules are *admissible*:

$$\frac{\Gamma \vdash \mathsf{A}\ \mathsf{type_i}}{\Gamma \vdash \mathit{En}(\mathsf{A}) : \mathsf{U_{i-1}}}\ (\text{U-Intro}) \qquad \frac{\Gamma \vdash \mathsf{A}\ \mathsf{type_i}}{\Gamma \vdash \mathsf{El}(\mathit{En}(\mathsf{A})) \equiv \mathsf{A}\ \mathsf{type_i}}\ (\text{U-Comp})$$

where $\mathit{En}(\mathsf{A})$ denotes some term that is assigned to the type A. Note that we have introduced the ranks of types solely for U-Intro. Also, it is straightforward to see that type-checking remains decidable in the presence of this formulation of universes.

Remark. The 'decoding' operation El is part of the syntax (it is a dependent type), while the dual 'encoding' operation *En* is a *meta-notation*, i.e., $\mathit{En}(\mathsf{A})$ represents a specific term of a universe assigned to each type A, since an expression of the form 'A type' cannot be in a context. Accordingly, we do not have a congruence rule for *En*. In fact, we do not require *reflection of equality (RoE)*, i.e., $\Gamma \vdash \mathsf{A} \equiv \mathsf{B}\ \mathsf{type_i}$ does not imply $\Gamma \vdash \mathit{En}(\mathsf{A}) \equiv \mathit{En}(\mathsf{B}) : \mathsf{U_{i-1}}$ [146], which appears conceptually natural as it means that there may be more than one 'code' of a type.[5] Hence, the operation El is surjective but not injective, and thus the uniqueness rule of universes does not hold.

The difference between types $\Gamma \vdash \mathsf{A}\ \mathsf{type}$ and terms of universes $\Delta \vdash \mathsf{u} : \mathsf{U}$ is often blurred, e.g., these two notions are simply *identified* in [184]. In our formulation of universes, types and terms of universes are corresponding (not bijectively though), but they are *distinct* concepts. From the viewpoint of categorical logic [150, 41, 102], the latter should be interpreted in game semantics as strategies, but what about the former? Moreover, what are the game-semantic counterpart of the ranks of types and the '(de)coding' operations? We shall answer these questions in Section 5.5.

[5]Indeed, RoE is not always assumed; for instance, see [146].

5.2.11 Meaning Explanation

Martin-Löf elaborated the ***meaning explanation*** of MLTT in order to explain and justify the syntax [126]. Thus, it plays the role of a *standard model* (or an *intended model*) of MLTT in the sense of model theory though it is pre-mathematical.

In spite of its pre-mathematical nature, the meaning explanation gives a highly convincing, systematic explanation of the syntax. In fact, our explanation of the syntax given above is an abbreviated, informal version of the meaning explanation; see [126, 54, 140] for more details. More significantly, the meaning explanation *per se* (i.e., independently of the syntax) is a fundamental underpinning of the very notion of constructive mathematics.[6] Since we propose our game semantics of MLTT as a mathematical formalization of the meaning explanation, we briefly introduce it below.

First, the meaning explanation is an *extension* of the BHK-interpretation in the sense that the former applies the 'proofs-as-computations' translation of the latter not only to the logical part but also the non-logical part of MLTT. In other words, it clarifies not only what proofs are but also what mathematical objects are.

Also, the former is a *refinement* of the latter by introducing the distinction between *canonical* and *non-canonical* proofs (or terms) as well as *equality* between them. Recall that the BHK-interpretation regards a formula as the set of its proofs. Similarly but more elaborately, the meaning explanation argues that a formula is the set of its proofs equipped with an equality between them, which has been defined as soon as its canonical proofs and an equality between them have been specified: Any computation that evaluates to a canonical proof of a formula is a proof of the formula, and two proofs of a formula are equal if they evaluate to equal canonical proofs. Intuitively, canonical proofs may be thought of as *values* in computation or *normalized proofs* in logic. Notably, each judgement is completely (though pre-mathematically) explained in terms of proofs and equalities between them defined as such [126, 54, 140].

5.3 Predicative Games

Having introduced MLTT, this section presents a generalization of games, called *predicative (p-) games*, to model *type dependency* in MLTT. The basic idea is as follows. In a game G, every position $\boldsymbol{s} \in P_G$ belongs to some strategy $\sigma : G$ in the sense that σ may lead to $\boldsymbol{s}$[7]; thus, it makes no essential difference for Player to first 'declare' the *name* of a strategy in such a way that is 'invisible' to Opponent and then

[6]This perspective is shared with some logicians and mathematicians, e.g., see [154].

[7]For instance, take $\sigma \overset{\mathrm{df.}}{=} \mathsf{Pref}(\{\boldsymbol{s}\})^{\mathsf{Even}}$.

play by the 'declared' one. In this view, a game corresponds to a set of valid strategies with some constraint (Theorem 5.3.19); by relaxing the constraint and introducing the *ranks* of games[8] to model the cumulative hierarchy of universes, we arrive at the notion of p-games that can be seen as *families of games*, modeling type dependency.

5.3.1 Valid Strategies as Deterministic Games

We first reformulate valid strategies as a particular kind of games. This would enable us to talk about valid strategies *without underlying games* (n.b., see the last few paragraphs of Section 5.3.2 on the technical and conceptual reasons why we need it).

We first need the following:

Definition 5.3.1 (Identification of sets of positions)**.** The ***identification of sets of positions*** of a game G is the relation $\simeq_G$ on subsets of P_G given by:

$$\begin{aligned} S \simeq_G T \overset{\mathrm{df.}}{\Leftrightarrow} & \forall \boldsymbol{s}m \in S, \boldsymbol{t} \in T.\, \boldsymbol{s} \simeq_G \boldsymbol{t}.\, \exists \boldsymbol{t}l \in T.\, \boldsymbol{s}m \simeq_G \boldsymbol{t}l \\ & \wedge \forall \boldsymbol{t}l \in T, \boldsymbol{s} \in S.\, \boldsymbol{t} \simeq_G \boldsymbol{s}.\, \exists \boldsymbol{s}m \in S.\, \boldsymbol{t}l \simeq_G \boldsymbol{s}m \end{aligned}$$

for all $S, T \subseteq P_G$.

In a similar manner to the case of identification of strategies (Corollary 2.3.5), we may show that the identification $\simeq_G$ of sets $S \subseteq P_G$ of positions that satisfy

$$(\textsc{Tree})\ S \neq \emptyset \wedge \forall \boldsymbol{s}m \in S.\, \boldsymbol{s} \in S$$

is a PER for any game G:

Lemma 5.3.2 (Second PER lemma)**.** *Let G be a game, and $S \simeq_G T \subseteq P_G$; assume that S and T both satisfy tree. Then, $\forall \boldsymbol{s} \in S.\, \exists \boldsymbol{t} \in T.\, \boldsymbol{s} \simeq_G \boldsymbol{t} \wedge \forall \boldsymbol{t} \in T.\, \exists \boldsymbol{s} \in S.\, \boldsymbol{t} \simeq_G \boldsymbol{s}$.*

Proof. By a straightforward induction on the length of positions. □

Corollary 5.3.3 (PER on sets of positions)**.** *Given a game G, the identification $\simeq_G$ of sets of positions, when restricted to subsets $S \subseteq P_G$ that satisfy tree, is a PER.*

Proof. The symmetry is obvious. For the transitivity, let $S, T, U \subseteq P_G$ be any subsets that satisfy tree, $S \simeq_G T$ and $T \simeq_G U$; assume $\boldsymbol{s}m \in S$, $\boldsymbol{u} \in U$ and $\boldsymbol{s} \simeq_G \boldsymbol{u}$; it suffices to find some $\boldsymbol{u}p \in U$ such that $\boldsymbol{s}m \simeq_G \boldsymbol{u}p$ since the other direction is symmetric. By Lemma 5.3.2, there is some $\boldsymbol{t}l \in T$ such that $\boldsymbol{s}m \simeq_G \boldsymbol{t}l$. Finally, for $\boldsymbol{u} \simeq_G \boldsymbol{t}$ and $U \simeq_G T$, there is some $\boldsymbol{u}p \in U$ such that $\boldsymbol{t}l \simeq_G \boldsymbol{u}p$, whence $\boldsymbol{s}m \simeq_G \boldsymbol{u}p$. □

[8]As expected, we shall see that they correspond to the ranks of types introduced in Section 5.2.

Therefore, as in the case of strategies (Definition 2.3.6), it makes sense to define:

Definition 5.3.4 (Validity of sets of positions)**.** Given a game G, a subset $S \subseteq P_G$ is ***valid*** if it satisfies the axiom tree and $S \simeq_G S$.

For instance, the set P_G of all positions of G is valid for any game G. Also, validity is clearly preserved under union but not intersection of sets.

We are now ready to characterize strategies as certain sets of positions (Lemma 5.3.6).

Definition 5.3.5 (Strategies as trees)**.** Given a strategy $\sigma : G$, the ***tree-form*** of σ with respect to G is the subset $(\overline{\sigma})_G \subseteq P_G$ given by:

$$(\overline{\sigma})_G \stackrel{\text{df.}}{=} \sigma \cup \{ \boldsymbol{s}m \in P_G \mid \boldsymbol{s} \in \sigma \}.$$

Notation. We often omit the subscript G when it is obvious.

Clearly, we may recover σ from $\overline{\sigma}$ by removing odd-length positions. Thus, σ and $\overline{\sigma}$ are essentially the same (in the context of G), just in different forms. Moreover, we may restrict the codomain of the map $\sigma \mapsto \overline{\sigma}$ so that it becomes a bijection:

Lemma 5.3.6 (Strategies in second-form)**.** *Given a game G, there is a bijection f between strategies $\sigma : G$ and subsets $S \subseteq P_G$ that satisfy:*

- (Tree) *Non-empty and prefix-closed: $S \neq \emptyset \wedge \forall \boldsymbol{s}m \in S.\, \boldsymbol{s} \in S$;*
- (Edet) *Deterministic: $\forall \boldsymbol{s}mn, \boldsymbol{s}mn' \in S^{\mathsf{Even}}.\, \boldsymbol{s}mn = \boldsymbol{s}mn'$;*
- (Oinc) *Inclusive on odd-length positions: $\forall \boldsymbol{s}m \in P_G^{\mathsf{Odd}}.\, \boldsymbol{s} \in S \Rightarrow \boldsymbol{s}m \in S$.*

Moreover, $\sigma \simeq_G \tau$ (Definition 2.3.3) iff $f(\sigma) \simeq_G f(\tau)$ for any strategies $\sigma, \tau : G$.

Proof. Let us define $f(\sigma) \stackrel{\text{df.}}{=} \overline{\sigma}$ for all $\sigma : G$. First, it is straightforward to see that for each strategy $\sigma : G$ the subset $\overline{\sigma} \subseteq P_G$ satisfies the three conditions of the lemma, e.g., $\overline{\sigma}$ is non-empty because $\boldsymbol{\epsilon} \in \overline{\sigma}$, and it is prefix-closed: For any $\boldsymbol{s}m \in \overline{\sigma}$, if $\boldsymbol{s} \in \sigma$, then $\boldsymbol{s} \in \overline{\sigma}$; otherwise, i.e., $\boldsymbol{s}m \in \sigma$, we may write $\boldsymbol{s} = \boldsymbol{t}n \in P_G$ with $\boldsymbol{t} \in \sigma$, whence $\boldsymbol{s} \in \overline{\sigma}$. For the converse, assume that a subset $S \subseteq P_G$ satisfies the three conditions; we then clearly have $S^{\mathsf{Even}} : G$.

Next, we show $(\_)^{\mathsf{Even}} = f^{-1}$. Clearly $(\overline{\sigma})^{\mathsf{Even}} = \sigma$ for all $\sigma : G$. It remains to establish $\overline{S^{\mathsf{Even}}} = S$ for all $S \subseteq P_G$ satisfying the three conditions. Let $S \subseteq P_G$ be such a subset. It is immediate that $\boldsymbol{s} \in \overline{S^{\mathsf{Even}}}$ iff $\boldsymbol{s} \in S$ for any $\boldsymbol{s} \in P_G^{\mathsf{Even}}$. If $\boldsymbol{t}m \in \overline{S^{\mathsf{Even}}}$ is of odd-length, then $\boldsymbol{t} \in S^{\mathsf{Even}}$ and $\boldsymbol{t}m \in P_G$, and thus $\boldsymbol{t}m \in S$ as S satisfies oinc. Conversely, if $\boldsymbol{u}n \in S^{\mathsf{Odd}}$, then $\boldsymbol{u} \in S^{\mathsf{Even}}$ by tree and $\boldsymbol{u}n \in P_G$, whence $\boldsymbol{u}n \in \overline{S^{\mathsf{Even}}}$.

Finally, for the logical equivalence of the two kinds of validity, the 'if' direction is immediate from their definitions, and the 'only if' direction is by I3 on G. □

Lemma 5.3.7 (Valid strategies as subgames)**.** *Given a valid strategy $\sigma : G$, we have:*

$$(\hat{\sigma})_G \overset{\mathrm{df.}}{=} (M_{(\hat{\sigma})_G}, \lambda_G \upharpoonright M_{(\hat{\sigma})_G}, \vdash_{(\hat{\sigma})_G}, (\overline{\sigma})_G, \simeq_G \cap ((\overline{\sigma})_G \times (\overline{\sigma})_G)) \trianglelefteq G$$

for which we often omit the subscript G, where $M_{\hat{\sigma}} \subseteq M_G$ is the set of moves of G that occur in a position of $\overline{\sigma}$, and $\vdash_{\hat{\sigma}} \subseteq \vdash_G \cap ((\{\star\} \cup M_{\hat{\sigma}}) \times M_{\hat{\sigma}})$ contains pairs that occur in a position of $\overline{\sigma}$ as $\star$ and an initial occurrence or as an occurrence m and a non-initial occurrence justified by m.

Proof. By Lemma 5.3.6, $P_{\hat{\sigma}} = \overline{\sigma}$ satisfies tree, and $\simeq_{\hat{\sigma}}$ satisfies I3 for σ is required to be valid. The remaining conditions for $\hat{\sigma} \trianglelefteq G$ are easy to verify. □

Let us call a game whose positions satisfy the axiom edet a ***deterministic game***. We are now ready to establish the main theorem of the present section:

Theorem 5.3.8 (Valid strategies as deterministic games)**.** *Given a game G, valid strategies $\sigma : G$ and subgames $H \trianglelefteq G$ such that P_H is valid and satisfies edet and oinc (with respect to P_G) are in one-to-one correspondence. In particular, valid strategies (on some games) and deterministic games are in one-to-one correspondence.*

Proof. The second statement is a corollary of the first one which follows from Lemmata 5.3.6 and 5.3.7, where the assumption that games are economical is crucial. □

Remark. This one-to-one correspondence between valid strategies and deterministic games implies in some sense that validity of strategies is a reasonable condition to impose by default, not only for a fully complete interpretation of syntax.

If we *identify* valid strategies with deterministic games, then we may talk about them *without underlying games.* To justify this view further, let us show that the bijection $(\sigma : G) \overset{\sim}{\mapsto} (\hat{\sigma})_G \trianglelefteq G$, where σ is valid, *commutes* with constructions on games and strategies (Theorem 5.3.14). For this, we first need:

Definition 5.3.9 (Composition of games)**.** Given games A, B and C and subgames $J \trianglelefteq A \multimap B$ and $K \trianglelefteq B \multimap C$, all of which we regard as normalized dynamic games (Definition 3.3.11), the ***composition*** $J;K$ of J and K is defined by $J;K \overset{\mathrm{df.}}{=} \mathcal{H}(J \ddagger K)$.

Notation. We also write $K \circ J$ for $J;K$.

Clearly, by Theorem 3.3.25, games are closed under composition. Also, note that $J;K \trianglelefteq A \multimap C$ if $J \trianglelefteq A \multimap B$ and $K \trianglelefteq B \multimap C$ by Lemma 3.3.30.

Lemma 5.3.10 (Characterization of composition of games)**.** *The composition $J;K$ of games $J \trianglelefteq A \multimap B$ and $K \trianglelefteq B \multimap C$ can be equivalently defined by:*

- $M_{J;K} \overset{\text{df.}}{=} (M_J \cap M_A) + (M_K \cap M_C)$;
- $\lambda_{J;K} \overset{\text{df.}}{=} [\lambda_J \upharpoonright M_A, \lambda_K \upharpoonright M_C]$;
- $\star \vdash_{J;K} m \overset{\text{df.}}{\Leftrightarrow} \star \vdash_K m$;
- $m \vdash_{J;K} n \ (m \neq \star) \overset{\text{df.}}{\Leftrightarrow} m \vdash_J n \vee m \vdash_K n \vee \exists b \in M_B.\, m \vdash_K b \wedge b \vdash_J n$;
- $P_{J;K} \overset{\text{df.}}{=} \{\boldsymbol{s} \upharpoonright A, C \mid \boldsymbol{s} \in \mathscr{J}_{((A \multimap B_{[1]}) \multimap B_{[2]}) \multimap C}, \boldsymbol{s} \upharpoonright A, B_{[1]} \in P_J, \boldsymbol{s} \upharpoonright B_{[2]}, C \in P_K,$ $\boldsymbol{s} \upharpoonright B_{[1]}, B_{[2]} \in pr_B\}$;
- $\boldsymbol{s} \simeq_{J;K} \boldsymbol{t} \overset{\text{df.}}{\Leftrightarrow} \boldsymbol{s} \upharpoonright A \simeq_A \boldsymbol{t} \upharpoonright A \wedge \boldsymbol{s} \upharpoonright C \simeq_C \boldsymbol{t} \upharpoonright C \wedge \forall i \in \mathbb{N}.\, \boldsymbol{s}(i) \in M_A \Leftrightarrow \boldsymbol{t}(i) \in M_A$.[9]

Proof. Clear from the definition. $\square$

We also need the following three technical lemmata:

Lemma 5.3.11 (O-view lemma)**.** *Let A, B and C be games and $J \trianglelefteq A \multimap B$ and $K \trianglelefteq B \multimap C$ subgames, all of which we regard as normalized dynamic games; assume that $\boldsymbol{s} \in P_{J \ddagger K}^{\mathsf{Even}}$ does not end with a move of B. Then, $\lfloor \boldsymbol{s} \upharpoonright A, C \rfloor_{A \multimap C} \preceq \lfloor \boldsymbol{s} \rfloor_{J \ddagger K} \upharpoonright A, C$.*

Proof. By induction on the length of $\boldsymbol{s}$. The base case $\boldsymbol{s} = \boldsymbol{\epsilon}$ is trivial; assume $\boldsymbol{s} = \boldsymbol{t}mn$. Let us focus on the case $n \in M_A$ since the other case $n \in M_C$ is simpler.

First, assume that $\boldsymbol{t}$ is of the form $\boldsymbol{u}l\boldsymbol{v}$, where l justifies n in $\boldsymbol{s}$; the other case (i.e., when m justifies n) is handled below. Note that $l \in M_A + M_B$. If $l \in M_A$, then:

$$\begin{aligned}
\lfloor \boldsymbol{s} \upharpoonright A, C \rfloor_{A \multimap C} &= \lfloor \boldsymbol{u}l\boldsymbol{v}mn \upharpoonright A, C \rfloor_{A \multimap C} \\
&= \lfloor \boldsymbol{u} \upharpoonright A, C \rfloor_{A \multimap C}.ln \\
&\preceq (\lfloor \boldsymbol{u} \rfloor_{J \ddagger K} \upharpoonright A, C).ln \text{ (by the induction hypothesis)} \\
&= \lfloor \boldsymbol{u} \rfloor_{J \ddagger K}.ln \upharpoonright A, C \\
&= \lfloor \boldsymbol{u}l\boldsymbol{v}mn \rfloor_{J \ddagger K} \upharpoonright A, C \\
&= \lfloor \boldsymbol{s} \rfloor_{J \ddagger K} \upharpoonright A, C.
\end{aligned}$$

On the other hand, if $l \in M_B$, then $n \in M_A^{\mathsf{Init}}$ and $l \in M_B^{\mathsf{Init}}$; thus, we may write $\boldsymbol{s} = \boldsymbol{w_1}c\boldsymbol{w_2}ll\boldsymbol{v}mn$, where $c \in M_C^{\mathsf{Init}}$ and it justifies the left occurrence of l, which in

[9]Strictly speaking, we need to take the subsets of $M_{J;K}$ and $\vdash_{J;K}$ in such a way that makes $J;K$ *economical* (Definition 2.2.15) in the obvious manner.

turn justifies the right occurrence of l. Hence, we may conclude that:

$$\begin{aligned}
\lfloor \boldsymbol{s} \upharpoonright A, C \rfloor_{A \multimap C} &= \lfloor \boldsymbol{w_1} c \boldsymbol{w_2} l l \boldsymbol{v} m n \upharpoonright A, C \rfloor_{A \multimap C} \\
&= \lfloor \boldsymbol{w_1} \upharpoonright A, C \rfloor_{A \multimap C} . cn \\
&\preceq (\lfloor \boldsymbol{w_1} \rfloor_{A \multimap C} \upharpoonright A, C) . cn \text{ (by the induction hypothesis)} \\
&= \lfloor \boldsymbol{w_1} \rfloor_{A \multimap C} . clln \upharpoonright A, C \\
&= \lfloor \boldsymbol{w_1} c \boldsymbol{w_2} l \rfloor_{J \ddagger K} . ln \upharpoonright A, C \\
&= \lfloor \boldsymbol{w_1} c \boldsymbol{w_2} l l \boldsymbol{v} m n \rfloor_{J \ddagger K} \upharpoonright A, C \\
&= \lfloor \boldsymbol{s} \rfloor_{J \ddagger K} \upharpoonright A, C.
\end{aligned}$$

Finally, consider the case where m justifies n in $\boldsymbol{s}$. It is just reduced to the induction hypothesis if $m \in M_A$; and it is handled in the same manner as the second case above if $m \in M_B^{\mathsf{Init}}$ (in this case, we may write $\boldsymbol{s} = \boldsymbol{u} c \boldsymbol{v} m m n$, where $c \in M_C$ and it justifies the left occurrence of m), completing the proof. □

Lemma 5.3.12 (Covering lemma 1)**.** *Let A, B, C and D be games, and $\phi : A \multimap B$ and $\psi : C \multimap D$ strategies; assume $\boldsymbol{s} \in \phi \otimes \psi$ and $\boldsymbol{s}m \in P_{A \otimes C \multimap B \otimes D}$. Then, $\boldsymbol{s}m \upharpoonright A, C \in \mathscr{L}_{A \otimes C} \wedge \boldsymbol{s}m \upharpoonright B, D \in \mathscr{L}_{B \otimes D} \Leftrightarrow \boldsymbol{s}m \upharpoonright A, B \in \mathscr{L}_{A \multimap B} \wedge \boldsymbol{s}m \upharpoonright C, D \in \mathscr{L}_{C \multimap D}$.*

Proof. We proceed by the case analysis on m. Let us just focus on the case $m \in M_A$ since the other three cases are similar or simpler. In this case, we have to show that $(\boldsymbol{s} \upharpoonright A, C).m \in \mathscr{L}_{A \otimes C} \Leftrightarrow (\boldsymbol{s} \upharpoonright A, B).m \in \mathscr{L}_{A \multimap B}$. Clearly, the two alternation conditions are logically equivalent.

Now, note that $\mathcal{J}_{\boldsymbol{s}m}(m) \in M_A$ since m is an O-move. Thus, the two justification conditions are logically equivalent. Finally, since only Player may switch between the domain and codomain of a linear implication, it is not hard to see that the two visibility conditions are also logically equivalent, completing the proof. □

Lemma 5.3.13 (Covering lemma 2)**.** *Let A, B and C be games, and assume that $\boldsymbol{s}m$ is an odd-length j-sequence of the arena $((A \multimap B_{[1]}) \multimap B_{[2]}) \multimap C$ such that $m \in M_{A \multimap C}$, $\boldsymbol{s} \upharpoonright A, B_{[1]} \in \mathscr{L}_{A \multimap B_{[1]}}$, $\boldsymbol{s} \upharpoonright B_{[2]}, C \in \mathscr{L}_{B_{[2]} \multimap C}$ and $\boldsymbol{s} \upharpoonright B_{[1]}, B_{[2]} \in pr_B$. Then, $\boldsymbol{s}m \upharpoonright A, C \in \mathscr{L}_{A \multimap C} \Leftrightarrow \boldsymbol{s}m \upharpoonright A, B_{[1]} \in \mathscr{L}_{A \multimap B_{[1]}} \wedge \boldsymbol{s}m \upharpoonright B_{[2]}, C \in \mathscr{L}_{B_{[2]} \multimap C}$.*

Proof. The implication $\Leftarrow$ has been well-established in the literature; see, e.g., [129] for the detail. Let us show the other implication $\Rightarrow$; assume that $\boldsymbol{s}m \upharpoonright A, C \in \mathscr{L}_{A \multimap C}$. First, it is clear that $\boldsymbol{s}m \upharpoonright A, B_{[1]}$ (resp. $\boldsymbol{s}m \upharpoonright B_{[2]}, C$) is a j-sequence of the arena $A \multimap B_{[1]}$ (resp. $B_{[2]} \multimap C$) that satisfies alternation by Table 3.2. It remains to establish visibility. Let us focus on the case $m \in M_A$ as the other case $m \in M_C$ is similar. Again by Table 3.2, we may write $\boldsymbol{s}m = \boldsymbol{t}a_1m$ with $a_1 \in M_A$. Note that the O-view $\lfloor \boldsymbol{t}a_1 \rfloor$ does not have moves of $B_{[1]}$, $B_{[2]}$ or C after the justifier $l \in M_A$ of m in $\boldsymbol{s}$ occurs because otherwise it cannot contain l, contradicting the visibility of $\boldsymbol{s}m \upharpoonright A, C$ for $\lfloor \boldsymbol{t}a_1 \upharpoonright A, C \rfloor \preceq \lfloor \boldsymbol{t}a_1 \rfloor \upharpoonright A, C$ (by Lemmata 5.3.10 and 5.3.11). Thus, $\lfloor \boldsymbol{t}a_1 \rfloor$ is of the form $\boldsymbol{u}la_{2k}a_{2k-1} \dots a_4a_3a_2a_1$, where $a_{2i} \in M_A$ justifies $a_{2i-1} \in M_A$ for $i = 1, 2, \dots, k$. Therefore, the O-view $\lfloor \boldsymbol{s} \upharpoonright A, B_{[1]} \rfloor$ is of the form $\boldsymbol{v}la_{2k}a_{2k-1} \dots a_4a_3a_2a_1$, and so it in particular contains $l = \mathcal{J}_{\boldsymbol{s}}(m)$. Hence, $\boldsymbol{s}m \upharpoonright A, B_{[1]}$ satisfies visibility (and $\boldsymbol{s}m \upharpoonright B_{[2]}, C$ trivially satisfies it). □

We can now establish the desired commutativity:

Theorem 5.3.14 (Interactions of constructions on games and strategies)**.** *Given games A, B, C and D, and valid strategies $\phi : A \multimap B$, $\psi : C \multimap D$, $\varphi : !A \multimap B$ and $\theta : A \multimap C$, we have the following four equations:*

1. $\hat{\phi} \otimes \hat{\psi} = \widehat{\phi \otimes \psi} \trianglelefteq A \otimes C \multimap B \otimes D$;
2. $\hat{\varphi}^\dagger = \widehat{\varphi^\dagger} \trianglelefteq\, !A \multimap\, !B$;
3. $\langle \hat{\phi}, \hat{\theta} \rangle = \widehat{\langle \phi, \theta \rangle} \trianglelefteq A \multimap B \& C$;
4. $\hat{\theta}; \hat{\psi} = \widehat{\theta; \psi} \trianglelefteq A \multimap D$.

Proof. Since the subgame relations are obvious by Lemmata 2.3.16, 2.3.19, 2.3.22, 2.3.25 and 5.3.7, and Theorem 2.2.29, it suffices to show the equations between the corresponding sets of positions.

Let us begin with the equation 1. For brevity, we define $\overline{\phi} \otimes \overline{\psi} \overset{\mathrm{df.}}{=} P_{\hat{\phi} \otimes \hat{\psi}}$. It is easy to see that $(\overline{\phi} \otimes \overline{\psi})^{\mathsf{Even}} = \phi \otimes \psi = (\overline{\phi \otimes \psi})^{\mathsf{Even}}$ holds. To show $(\overline{\phi} \otimes \overline{\psi})^{\mathsf{Odd}} = (\overline{\phi \otimes \psi})^{\mathsf{Odd}}$, it suffices to observe the following chain of logical equivalences:

$$
\begin{aligned}
&\boldsymbol{s}m \in (\overline{\phi \otimes \psi})^{\mathsf{Odd}} \\
\Leftrightarrow\ &\boldsymbol{s} \in \phi \otimes \psi \wedge \boldsymbol{s}m \in P_{A \otimes C \multimap B \otimes D} \\
\Leftrightarrow\ &\boldsymbol{s}m \in \mathscr{L} \wedge \boldsymbol{s} \upharpoonright A, B \in \phi \wedge \boldsymbol{s} \upharpoonright C, D \in \psi \wedge \boldsymbol{s}m \upharpoonright A, C \in P_{A \otimes C} \wedge \boldsymbol{s}m \upharpoonright B, D \in P_{B \otimes D} \\
\Leftrightarrow\ &\boldsymbol{s}m \in \mathscr{L} \wedge \boldsymbol{s} \upharpoonright A, B \in \phi \wedge \boldsymbol{s} \upharpoonright C, D \in \psi \wedge \boldsymbol{s}m \upharpoonright A, B \in P_{A \multimap B} \wedge \boldsymbol{s}m \upharpoonright C, D \in P_{C \multimap D} \\
&\text{(by Lemma 5.3.12)} \\
\Leftrightarrow\ &\boldsymbol{s}m \in \mathscr{L} \wedge \boldsymbol{s} \upharpoonright A, B \in \phi \wedge \boldsymbol{s} \upharpoonright C, D \in \psi \wedge ((\boldsymbol{s} \upharpoonright A, B).m \in P_{A \multimap B} \vee (\boldsymbol{s} \upharpoonright C, D).m \in P_{C \multimap D}) \\
\Leftrightarrow\ &\boldsymbol{s}m \in \mathscr{L} \wedge ((\boldsymbol{s} \upharpoonright A, B).m \in \overline{\phi} \wedge \boldsymbol{s} \upharpoonright C, D \in \overline{\psi}) \vee (\boldsymbol{s} \upharpoonright A, B \in \overline{\phi} \wedge (\boldsymbol{s} \upharpoonright C, D).m \in \overline{\psi}) \\
\Leftrightarrow\ &\boldsymbol{s}m \in \mathscr{L} \wedge \boldsymbol{s}m \upharpoonright A, B \in \overline{\phi} \wedge \boldsymbol{s}m \upharpoonright C, D \in \overline{\psi} \\
\Leftrightarrow\ &\boldsymbol{s}m \in (\overline{\phi} \otimes \overline{\psi})^{\mathsf{Odd}}
\end{aligned}
$$

where $\mathscr{L} \stackrel{\mathrm{df.}}{=} \mathscr{L}_{A \otimes C \multimap B \otimes D}$. The equations 2 and 3 are even simpler to prove.

It remains to establish the equation 4. Let us define $\overline{\theta};\overline{\psi} \stackrel{\mathrm{df.}}{=} P_{\hat{\theta};\hat{\psi}}$. Again, the equation $(\overline{\theta};\overline{\psi})^{\mathsf{Even}} = \theta;\psi = (\overline{\theta;\psi})^{\mathsf{Even}}$ is straightforward to show; for $(\overline{\theta};\overline{\psi})^{\mathsf{Odd}} = (\overline{\theta;\psi})^{\mathsf{Odd}}$, observe the following chain of logical equivalences:

$$
\begin{aligned}
& \boldsymbol{sm} \in (\overline{\theta};\overline{\psi})^{\mathsf{Odd}} \\
\Leftrightarrow\ & \exists \boldsymbol{tm} \in \mathscr{J}.\, \boldsymbol{tm} \upharpoonright A, D = \boldsymbol{sm} \wedge \boldsymbol{tm} \upharpoonright A, C_{[1]} \in \overline{\theta} \wedge \boldsymbol{tm} \upharpoonright C_{[2]}, D \in \overline{\psi} \\
& \quad \wedge \boldsymbol{tm} \upharpoonright C_{[1]}, C_{[2]} \in pr_C \\
\Leftrightarrow\ & \exists \boldsymbol{tm} \in \mathscr{J}.\, \boldsymbol{tm} \upharpoonright A, D = \boldsymbol{sm} \wedge \boldsymbol{t} \upharpoonright A, C_{[1]} \in \theta \wedge \boldsymbol{t} \upharpoonright C_{[2]}, D \in \psi \wedge \boldsymbol{tm} \upharpoonright C_{[1]}, C_{[2]} \in pr_C \\
& \quad \wedge ((\boldsymbol{t} \upharpoonright A, C_{[1]}).m \in P_{A \multimap C_{[1]}} \vee (\boldsymbol{t} \upharpoonright C_{[2]}, D).m \in P_{C_{[2]} \multimap D}) \\
\Leftrightarrow\ & \exists \boldsymbol{tm} \in \mathscr{J}.\, \boldsymbol{tm} \upharpoonright A, D = \boldsymbol{sm} \wedge \boldsymbol{t} \in \theta \| \psi \wedge \boldsymbol{tm} \upharpoonright C_{[1]}, C_{[2]} \in pr_C \\
& \quad \wedge \boldsymbol{tm} \upharpoonright A, C_{[1]} \in P_{A \multimap C_{[1]}} \wedge \boldsymbol{tm} \upharpoonright C_{[2]}, D \in P_{C_{[2]} \multimap D} \\
\Leftrightarrow\ & \exists \boldsymbol{tm} \in \mathscr{J}.\, \boldsymbol{tm} \upharpoonright A, D = \boldsymbol{sm} \wedge \boldsymbol{t} \in \theta \| \psi \wedge \boldsymbol{tm} \upharpoonright C_{[1]}, C_{[2]} \in pr_C \\
& \quad \wedge \boldsymbol{tm} \upharpoonright A, C_{[1]} \in \mathscr{L}_{A \multimap C_{[1]}} \wedge \boldsymbol{tm} \upharpoonright C_{[2]}, D \in \mathscr{L}_{C_{[2]} \multimap D} \wedge \boldsymbol{tm} \upharpoonright A \in P_A \wedge \boldsymbol{tm} \upharpoonright D \in P_D \\
\Leftrightarrow\ & \exists \boldsymbol{tm} \in \mathscr{J}.\, \boldsymbol{tm} \upharpoonright A, D = \boldsymbol{sm} \wedge \boldsymbol{t} \in \theta \| \psi \wedge \boldsymbol{tm} \upharpoonright C_{[1]}, C_{[2]} \in pr_C \\
& \quad \wedge \boldsymbol{tm} \upharpoonright A, D \in \mathscr{L}_{A \multimap D} \wedge \boldsymbol{tm} \upharpoonright A \in P_A \wedge \boldsymbol{tm} \upharpoonright D \in P_D \text{ (by Lemma 5.3.13)} \\
\Leftrightarrow\ & \exists \boldsymbol{tm} \in \mathscr{J}.\, \boldsymbol{t} \in \theta \| \psi \wedge \boldsymbol{sm} = \boldsymbol{tm} \upharpoonright A, D \wedge \boldsymbol{tm} \upharpoonright A, D \in P_{A \multimap D} \\
\Leftrightarrow\ & \boldsymbol{s} \in \theta;\psi \wedge \boldsymbol{sm} \in P_{A \multimap D} \\
& \text{($\Leftarrow$ holds for m, $\mathcal{J}_{\boldsymbol{sm}}(m)$ and the last move of $\boldsymbol{s}$ (if exists) all belong to A or B)} \\
\Leftrightarrow\ & \boldsymbol{sm} \in (\overline{\theta;\psi})^{\mathsf{Odd}}
\end{aligned}
$$

by Lemma 5.3.10, where $\mathscr{J} \stackrel{\mathrm{df.}}{=} \mathscr{J}_{((A \multimap C_{[1]}) \multimap C_{[2]}) \multimap D}$, completing the proof. □

To summarize the present section, Theorem 5.3.8 establishes the fact that valid strategies on a game G correspond to subgames $H \trianglelefteq G$ such that P_H is valid and satisfies the axioms edet and oinc (with respect to P_G). Moreover, as shown in Theorem 5.3.14, constructions on games and valid strategies may be identified.

These results suggest that we may reformulate valid strategies as follows:

Definition 5.3.15 (V-strategies)**.** A ***v-strategy*** is a deterministic game. The ***composition*** $\circ$, ***tensor*** $\otimes$, ***pairing*** $\langle \_, \_ \rangle$ and ***promotion*** $(\_)^\dagger$ on v-strategies are the corresponding ones on games, respectively. The ***copy-cat*** (resp. ***dereliction***) on a game A is the v-strategy $\widehat{cp}_A$ (resp. $\widehat{der}_A$), but we use the notation cp_A (resp. der_A) for it. Given a v-strategy σ and a game G, we say that σ is ***on*** G and write $\sigma : G$ if $\sigma \trianglelefteq G$ and P_σ satisfies the axiom oinc (with respect to P_G), and σ is ***innocent*** (resp. ***well-bracketed***, ***total***, ***noetherian***, ***winning***) if so is the set P_σ^{Even}.

Remark. Given a v-strategy σ, a game G that satisfies $\sigma : G$ may not be unique, e.g., $(\widehat{\{\boldsymbol{\epsilon}, q.0\}})_N$ is a v-strategy on any of the following games:

$$\begin{tikzcd}
q \arrow[d] \\
0
\end{tikzcd}
\qquad
\begin{tikzcd}
& q \arrow[dl] \arrow[d] \arrow[dr] & \\
0 & 1 & 2
\end{tikzcd}
\qquad
\begin{tikzcd}
& & q \arrow[dll] \arrow[dl] \arrow[d] \arrow[dr] & \cdots & \\
0 & 1 & 2 & 3 & \cdots
\end{tikzcd}
\qquad
\begin{tikzcd}
& q \arrow[dl] \arrow[dr] & \\
0 & & 1 \arrow[d] \\
& & q
\end{tikzcd}$$

Notation. Given $\sigma : G$, $\boldsymbol{s} \in P_\sigma^{\mathsf{Even}}$ and $\boldsymbol{s}m \in P_G$, we write $\sigma(\boldsymbol{s}m)\downarrow$ if there is a move $n \in M_G$ such that $\boldsymbol{s}mn \in P_\sigma$ (also write $\sigma(\boldsymbol{s}m) = n$) and $\sigma(\boldsymbol{s}m)\uparrow$ otherwise.

At the end of the present section, let us show:

Lemma 5.3.16 (V-lemma)**.** *If $\sigma : G$ is a strategy with no pair $\boldsymbol{s}mn, \boldsymbol{t}lr \in \sigma$ such that $\boldsymbol{s}m \simeq_G \boldsymbol{t}l$ and $\boldsymbol{s}mn \not\simeq_G \boldsymbol{t}lr$, then there is a v-strategy $\nu : G$ such that $\sigma \subseteq P_\nu^{\mathsf{Even}} : G$.*

Proof. By Theorem 5.3.8, it suffices to show the existence of a valid strategy $\nu : G$ such that $\sigma \subseteq \nu$. Let us define a sequence $(\nu_i)_{i \in \mathbb{N}}$ of strategies $\nu_i : G$ as follows. First, let $\nu_0 \stackrel{\mathrm{df.}}{=} \sigma$. Next, we obtain ν_{i+1} from ν_i by adding any unique choice of $\boldsymbol{t}lr \in P_G$ if $\boldsymbol{s}mn, \boldsymbol{t} \in \nu_i$, $|\boldsymbol{s}| = 2i$, $\boldsymbol{s}m \simeq_G \boldsymbol{t}l$ and any $\boldsymbol{t}lr' \in P_G$ such that $\boldsymbol{s}mn \simeq_G \boldsymbol{t}lr'$ is not yet in ν_i. Note that such $\boldsymbol{t}lr$ must exist by the axiom I3 on G. We then define $\nu \stackrel{\mathrm{df.}}{=} \bigcup_{i \in \mathbb{N}} \nu_i$, which is clearly a valid strategy on G such that $\sigma \subseteq \nu$. $\square$

5.3.2 Games via V-Strategies

This section introduces, based on the last section, a characterization of games *as sets of v-strategies* with some constraint. From this fact, by relaxing the constraint (and equipping *ranks*), we shall arrive at a more general notion of *predicative games*.

Then, what constraint should we impose? Well, for instance, a set of v-strategies on the same game must be *consistent* in the sense that they share the same labeling, enabling, odd-length positions and identification of positions. Thus, we define:

Definition 5.3.17 (Consistency)**.** A set $\mathcal{S}$ of v-strategies is ***consistent*** if, for all $\sigma, \tau \in \mathcal{S}$, it satisfies:

1. $\lambda_\sigma(m) = \lambda_\tau(m)$ for all $m \in M_\sigma \cap M_\tau$;

2. $\star \vdash_\sigma m \Leftrightarrow \star \vdash_\tau m$ and $m \vdash_\sigma n \Leftrightarrow m \vdash_\tau n$ for all $m, n \in M_\sigma \cap M_\tau$;

3. $\boldsymbol{s}m \in P_\sigma \Leftrightarrow \boldsymbol{s}m \in P_\tau$ for all $\boldsymbol{s} \in P_\sigma^{\mathsf{Even}} \cap P_\tau^{\mathsf{Even}}$ and $\boldsymbol{s}m \in P_\sigma \cup P_\tau$;

4. $\boldsymbol{s} \simeq_\sigma \boldsymbol{t} \Leftrightarrow \boldsymbol{s} \simeq_\tau \boldsymbol{t}$ for all $\boldsymbol{s}, \boldsymbol{t} \in P_\sigma \cap P_\tau$.

Two v-strategies σ and τ are ***consistent***, written $\sigma \asymp \tau$, if so is the set $\{\sigma, \tau\}$.

Note that the first four components of a given game G may be recovered from the consistent set $\mathsf{vs}(G)$ of all v-strategies on G because $M_G = \bigcup_{\sigma \in \mathsf{vs}(G)} M_\sigma$, $\lambda_G = \bigcup_{\sigma \in \mathsf{vs}(G)} \lambda_\sigma$, $\vdash_G = \bigcup_{\sigma \in \mathsf{vs}(G)} \vdash \sigma$ and $P_G = \bigcup_{\sigma \in \mathsf{vs}(G)} P_\sigma$ (see the proof of Theorem 5.3.19 below). However, it is not the case for the identification $\simeq_G$ of positions; for instance, consider the game $N \Rightarrow N$: $q.(q,0) \simeq_{N \Rightarrow N} q.(q,1)$, but $q.(q,0) \not\simeq_\phi q.(q,1)$ for all v-strategies $\phi : N \Rightarrow N$. This suggests that we should keep $\simeq_G$ in addition to $\mathsf{vs}(G)$.

Generally, given a pair $(\mathcal{S}, \simeq_\mathcal{S})$, called a ***consistent pair***, of a consistent set $\mathcal{S}$ of v-strategies and an equivalence relation $\simeq_\mathcal{S} \supseteq \bigcup_{\sigma \in \mathcal{S}} \simeq_\sigma$ on $\bigcup_{\sigma \in \mathcal{S}} P_\sigma$ that satisfies the axioms I1, I2 and I3 as well as the following ***consistency of validity***

$$(\text{CoV})\ \forall (\boldsymbol{s}, \boldsymbol{t}) \in \simeq_\mathcal{S} \setminus \textstyle\bigcup_{\sigma \in \mathcal{S}} \simeq_\sigma . \forall \sigma \in \mathcal{S}. \, \boldsymbol{s} \notin P_\sigma \vee \boldsymbol{t} \notin P_\sigma$$

called an ***identification*** on $\mathcal{S}$, we may construct the ***union game*** $\bigcup(\mathcal{S}, \simeq_\mathcal{S})$ of the consistent pair $(\mathcal{S}, \simeq_\mathcal{S})$ by:

$$\textstyle\bigcup(\mathcal{S}, \simeq_\mathcal{S}) \stackrel{\text{df.}}{=} (\bigcup_{\sigma \in \mathcal{S}} M_\sigma, \bigcup_{\sigma \in \mathcal{S}} \lambda_\sigma, \bigcup_{\sigma \in \mathcal{S}} \vdash_\sigma, \bigcup_{\sigma \in \mathcal{S}} P_\sigma, \simeq_\mathcal{S})$$

where the first condition of the consistency of $\mathcal{S}$ guarantees that $\bigcup(\mathcal{S}, \simeq_\mathcal{S})$ is a well-defined game, and the other three and the axiom CoV on $\simeq_\mathcal{S}$ preserve the structure of each $\sigma \in \mathcal{S}$ in $\bigcup(\mathcal{S}, \simeq_\mathcal{S})$. In other words, the consistency of $\mathcal{S}$ is equivalent to the existence of a common underlying game for all v-strategies in $\mathcal{S}$, and CoV on $\simeq_\mathcal{S}$ ensures that the restriction of $\simeq_\mathcal{S}$ to positions of each $\sigma \in \mathcal{S}$ coincides with $\simeq_\sigma$.

However, some v-strategies on $\bigcup(\mathcal{S}, \simeq_\mathcal{S})$ may not exist in $\mathcal{S}$. For example, consider a consistent set $\mathcal{S} = \{\sigma, \tau\}$ of v-strategies σ and τ such that $P_\sigma = \mathsf{Pref}(\{ac, bc\})$, $P_\tau = \mathsf{Pref}(\{ad, bd\})$, where a, b, c and d are pairwise distinct moves, and both have only the trivial identification of positions (the other components are obvious), and the trivial identification $=$ on $\mathcal{S}$. Clearly, a v-strategy $\phi : \bigcup(\mathcal{S}, \simeq_\mathcal{S})$ defined by $P_\phi \stackrel{\text{df.}}{=} \mathsf{Pref}(\{ac, bd\})$ (again, the other components are obvious), does not exist in $\mathcal{S}$. Even a simpler v-strategy $\psi : \bigcup(\mathcal{S}, \simeq_\mathcal{S})$ defined by $P_\psi \stackrel{\text{df.}}{=} \{\boldsymbol{\epsilon}, a, b\}$ does not exist in $\mathcal{S}$.

For this point, in view of Lemma 5.3.6, we define:

Definition 5.3.18 (Completeness). A consistent pair $(\mathcal{S}, \simeq_\mathcal{S})$ is ***complete*** if any valid (with respect to $\simeq_\mathcal{S}$) subset $\mathcal{A} \subseteq \bigcup_{\sigma \in \mathcal{S}} P_\sigma$ such that

- (Tree) $\mathcal{A} \neq \emptyset \wedge \forall \boldsymbol{s}m \in \mathcal{A}.\, \boldsymbol{s} \in \mathcal{A}$;
- (Edet) $\forall \boldsymbol{s}mn, \boldsymbol{s}mn' \in \mathcal{A}^{\mathsf{Even}}.\, \boldsymbol{s}mn = \boldsymbol{s}mn'$;
- (Oinc) $\forall \boldsymbol{s}m \in P_{\bigcup(\mathcal{S}, \simeq_{\mathcal{S}})} = \bigcup_{\sigma \in \mathcal{S}} P_\sigma.\, \boldsymbol{s} \in \mathcal{A}^{\mathsf{Even}} \Rightarrow \boldsymbol{s}m \in \mathcal{A}$

satisfies $(\widehat{\mathcal{A}^{\mathsf{Even}}})_{\bigcup(\mathcal{S}, \simeq_{\mathcal{S}})} \in \mathcal{S}$.

Intuitively, the completeness of a consistent pair $(\mathcal{S}, \simeq_{\mathcal{S}})$ means the closure of $\mathcal{S}$ under 'patchwork' $\mathcal{A} \subseteq \bigcup_{\sigma \in \mathcal{S}} P_\sigma$ of v-strategies. We may easily see that in the above example the pair $(\mathcal{S}, =)$ is not complete; there are total nine v-strategies on $\sigma \cup \tau$, and so we need to add ϕ, ψ and the remaining five to $\mathcal{S}$ to make it complete.

Now, we have arrived at the desired characterization:

Theorem 5.3.19 (Games as collections of v-strategies)**.** *There exists a one-to-one correspondence between games and complete pairs:*

1. *Given a game G, the pair $(\mathsf{vs}(G), \simeq_G)$ of the set $\mathsf{vs}(G) \stackrel{\text{df.}}{=} \{\sigma \mid \sigma : G\}$ of all v-strategies on G and the identification $\simeq_G$ of positions of G is complete, and $G = \bigcup(\mathsf{vs}(G), \simeq_G)$;*
2. *Given a complete pair $(\mathcal{S}, \simeq_{\mathcal{S}})$, v-strategies on the union game $\bigcup(\mathcal{S}, \simeq_{\mathcal{S}})$ are precisely elements of $\mathcal{S}$.*

Proof. For the clause 1, let G be a game. First, the pair $(\mathsf{vs}(G), \simeq_G)$ is clearly complete. For the equation $G = \bigcup(\mathsf{vs}(G), \simeq_G)$, it suffices to show $P_G = \bigcup_{\sigma \in \mathsf{vs}(G)} P_\sigma$ since then the other components clearly coincide (note that G is economical). The inclusion $\bigcup_{\sigma \in \mathsf{vs}(G)} P_\sigma \subseteq P_G$ is immediate. For the other inclusion, let $\boldsymbol{s} \in P_G$; by Lemmata 5.3.6 and 5.3.16, it suffices to show $\boldsymbol{s} \in \tau$ for some strategy $\tau : G$ satisfying the assumption of Lemma 5.3.16, but then we may just take $\tau \stackrel{\text{df.}}{=} \mathsf{Pref}(\{\boldsymbol{s}\})^{\mathsf{Even}}$.

Next, for the clause 2, let $(\mathcal{S}, \simeq_{\mathcal{S}})$ be a complete pair. To show $\mathsf{vs}(\bigcup(\mathcal{S}, \simeq_{\mathcal{S}})) \subseteq \mathcal{S}$, let ϕ be a v-strategy on $\bigcup(\mathcal{S}, \simeq_{\mathcal{S}})$; we have to show $\phi \in \mathcal{S}$. For each $i \in \mathbb{N}$, we define the subset $\mathcal{S}(\phi, i) \subseteq \mathcal{S}$ by $\tau \in \mathcal{S}(\phi, i) \stackrel{\text{df.}}{\Leftrightarrow} \{\boldsymbol{s} \in P_\tau \mid |\boldsymbol{s}| \leqslant 2i\} = \{\boldsymbol{t} \in P_\phi \mid |\boldsymbol{t}| \leqslant 2i\}$ for all $\tau \in \mathcal{S}$. Then, it suffices to show $\mathcal{S}(\phi, i) \neq \emptyset$ for all $i \in \mathbb{N}$. The base case ($i = 0$) is trivial because the completeness of $\mathcal{S}$ in particular implies $\mathcal{S} \neq \emptyset$. Consider the inductive step where we have to show $\mathcal{S}(\phi, i+1) \neq \emptyset$. Let us take an arbitrary $\tau \in \mathcal{S}(\phi, i)$ by the induction hypothesis. For $\phi : \bigcup(\mathcal{S}, \simeq_{\mathcal{S}})$, there is some $\phi_{\boldsymbol{s}m} \in \mathcal{S}$ that satisfies $\phi_{\boldsymbol{s}m}(\boldsymbol{s}m) \simeq \phi(\boldsymbol{s}m)$ for each $\boldsymbol{s}m \in P_\phi$ with $|\boldsymbol{s}m| = 2i + 1$.[10] We take

[10]If $\phi(\boldsymbol{s}m)\uparrow$ but there seems no $\tau \in \mathcal{S}$ such that $\tau(\boldsymbol{s}m)\uparrow$, then pick any $\psi \in \mathcal{S}$ with $\boldsymbol{s}m \in P_\psi$ and eliminate proper suffixes of $\boldsymbol{s}m$ from P_ψ to form $\phi_{\boldsymbol{s}m}$, which actually lies in $\mathcal{S}$ by the completeness.

$\mathcal{A} \stackrel{\mathrm{df.}}{=} \tau \cup \{ \boldsymbol{sm}.\phi_{\boldsymbol{sm}}(\boldsymbol{sm}) \mid \boldsymbol{sm} \in P_\phi, |\boldsymbol{sm}| = 2i+1, \phi(\boldsymbol{sm}) \downarrow \} \cup \{ \boldsymbol{sm}.\phi_{\boldsymbol{sm}}(\boldsymbol{sm}).n \in P^{\mathsf{Odd}}_{\bigcup(\mathcal{S}, \simeq_{\mathcal{S}})} \mid \boldsymbol{sm} \in P_\phi, |\boldsymbol{sm}| = 2i+1, \phi(\boldsymbol{sm}) \downarrow \} \subseteq P_{\bigcup(\mathcal{S}, \simeq_{\mathcal{S}})}$; clearly, $\mathcal{A} \subseteq \bigcup_{\sigma \in \mathcal{S}} P_\sigma$, and it is valid and satisfies the three conditions. Thus, by the completeness of the pair $(\mathcal{S}, \simeq_{\mathcal{S}})$, we may conclude that $\widehat{\mathcal{A}^{\mathsf{Even}}} \in \mathcal{S}$, which implies $\mathcal{S}(\sigma, i+1) \neq \emptyset$. Finally, the opposite inclusion $\mathcal{S} \subseteq \mathsf{vs}(\bigcup(\mathcal{S}, \simeq_{\mathcal{S}}))$ clearly holds. $\square$

Theorem 5.3.19 particularly implies that any game is of the form $\bigcup(\mathcal{S}, \simeq_{\mathcal{S}})$, where $(\mathcal{S}, \simeq_{\mathcal{S}})$ is a complete pair, and v-strategies on $\bigcup(\mathcal{S}, \simeq_{\mathcal{S}})$ are precisely elements of $\mathcal{S}$.

Now, given a complete pair $(\mathcal{S}, \simeq_{\mathcal{S}})$, observe that there is no essential difference between the union game $\bigcup(\mathcal{S}, \simeq_{\mathcal{S}})$ and the ***sum game***[11] $\sum(\mathcal{S}, \simeq_{\mathcal{S}})$ defined by:

- $M_{\sum \mathcal{S}} \stackrel{\mathrm{df.}}{=} \{q_{\mathcal{S}}\} \cup \|\mathcal{S}\| \cup \{(m)_{\|\sigma\|} \stackrel{\mathrm{df.}}{=} (m, \|\sigma\|) \mid \sigma \in \mathcal{S}, m \in M_\sigma\}$, where $\|\sigma\|$ is the *name* of each $\sigma \in \mathcal{S}$, i.e., $\|\_\|$ is any injection (e.g., the simplest choice is to define $\|\_\|$ to be the identity function on $\mathcal{S}$), and $q_{\mathcal{S}}$ is any element such that $q_{\mathcal{S}} \notin \|\mathcal{S}\| \cup \{(m)_{\|\sigma\|} \mid \sigma \in \mathcal{S}, m \in M_\sigma\}$;

- $\lambda_{\sum \mathcal{S}} : q_{\mathcal{S}} \mapsto \mathsf{OQ}, (\|\sigma\| \in \|\mathcal{S}\|) \mapsto \mathsf{PA}, (m)_{\|\sigma\|} \mapsto \lambda_\sigma(m)$;

- $\vdash_{\sum \mathcal{S}} \stackrel{\mathrm{df.}}{=} \{(\star, q_{\mathcal{S}})\} \cup \{(q_{\mathcal{S}}, \|\sigma\|) \mid \sigma \in \mathcal{S}\} \cup \{(\|\sigma\|, (m)_{\|\sigma\|}) \mid \sigma \in \mathcal{S}, \star \vdash_\sigma m\}$
$\cup \{((m)_{\|\sigma\|}, (n)_{\|\sigma\|}) \mid \sigma \in \mathcal{S}, m \vdash_\sigma n\}$;

- $P_{\sum \mathcal{S}} \stackrel{\mathrm{df.}}{=} \{\boldsymbol{\epsilon}, q_{\mathcal{S}}\} \cup \{q_{\mathcal{S}}.\|\sigma\|.(\boldsymbol{s})_{\|\sigma\|} \mid \sigma \in \mathcal{S}, \boldsymbol{s} \in P_\sigma\}$, where $q_{\mathcal{S}}$ justifies $\|\sigma\|$, $\|\sigma\|$ justifies initial moves occurring in $\boldsymbol{s}$, and $\boldsymbol{s} = m_1 m_2 \dots m_k$ implies $(\boldsymbol{s})_{\|\sigma\|} \stackrel{\mathrm{df.}}{=} (m_1)_{\|\sigma\|}.(m_2)_{\|\sigma\|} \dots (m_k)_{\|\sigma\|}$;

- $\simeq_{\sum \mathcal{S}} \stackrel{\mathrm{df.}}{=} \{(\boldsymbol{\epsilon}, \boldsymbol{\epsilon}), (q_{\mathcal{S}}, q_{\mathcal{S}})\}$
$\cup \{(q_{\mathcal{S}}.\|\sigma\|.(\boldsymbol{s})_{\|\sigma\|}, q_{\mathcal{S}}.\|\tau\|.(\boldsymbol{t})_{\|\tau\|}) \mid \sigma, \tau \in \mathcal{S}, \boldsymbol{s} \in P_\sigma, \boldsymbol{t} \in P_\tau, \boldsymbol{s} \simeq_{\mathcal{S}} \boldsymbol{t}\}$.

It is easy to see that $\sum(\mathcal{S}, \simeq_{\mathcal{S}})$ is a well-defined game. A position $q_{\mathcal{S}}.\|\sigma\|.(\boldsymbol{s})_{\|\sigma\|}$ of $\sum(\mathcal{S}, \simeq_{\mathcal{S}})$ is essentially a position $\boldsymbol{s}$ of $\bigcup(\mathcal{S}, \simeq_{\mathcal{S}})$ prefixed with the two moves $q_{\mathcal{S}}.\|\sigma\|$ and equipped with the 'tag' $(\_)_{\|\sigma\|}$ on subsequent moves, where σ is any (not unique) $\sigma \in \mathcal{S}$ such that $\boldsymbol{s} \in P_\sigma$ (such σ must exist as shown in the proof of Theorem 5.3.19); the difference between $\sum(\mathcal{S}, \simeq_{\mathcal{S}})$ and $\bigcup(\mathcal{S}, \simeq_{\mathcal{S}})$ is whether to specify such $\sigma \in \mathcal{S}$.

Intuitively, a play of the sum game $\sum(\mathcal{S}, \simeq_{\mathcal{S}})$ proceeds as follows. For conceptual clarity, let us introduce ***Judge*** of the game. Judge first asks Player about her v-strategy in mind by the question $q_{\mathcal{S}}$, and Player answers it by the name $\|\sigma\|$ of a v-strategy $\sigma \in \mathcal{S}$; then an actual play between Opponent and Player begins as in $\bigcup(\mathcal{S}, \simeq_{\mathcal{S}})$ except that Player must follow the declared v-strategy σ.

[11]It is similar to the *weak sum* $\oplus$ of games (Definition 2.2.27), but this point is not relevant here.

To be fair, the declared v-strategy should be 'invisible' to Opponent, and he also has to declare to Judge an *anti-strategy*, i.e., a set of odd-length positions that is non-empty, odd-prefix-closed and deterministic on odd-length positions, at the beginning of a play, which is 'invisible' to Player, and play by following it. Clearly, we may achieve 'invisibility' of v-strategies to Opponent by imposing that anti-strategies cannot depend on 'tags', i.e., any anti-strategy τ on $\sum(\mathcal{S}, \simeq_{\mathcal{S}})$ must satisfy:

$$q_{\mathcal{S}}.\|\phi\|.(m_1)_{\|\phi\|}.(m_2)_{\|\phi\|} \dots (m_{2k+1})_{\|\phi\|} \in \tau \Leftrightarrow q_{\mathcal{S}}.\|\psi\|.(m_1)_{\|\psi\|}.(m_2)_{\|\psi\|} \dots (m_{2k+1})_{\|\psi\|} \in \tau$$

for any $\phi, \psi \in \mathcal{S}$ and $m_1 m_2 \dots m_{2k} \in P_\phi \cap P_\psi$. Note that we have introduced the notion of judge for a conceptual understanding of this mathematical formulation.

Nevertheless, since the 'spirit' of game semantics is not to restrict Opponent's computational power at all, we choose *not* to incorporate anti-strategies, let alone Opponent's 'declaration' of them or their 'invisibility' condition to Player, into games.

To sum up, we may reformulate any game in the form of $\sum(\mathcal{S}, \simeq_{\mathcal{S}})$, where $\mathcal{S}$ is a consistent set of v-strategies, and $\simeq_{\mathcal{S}}$ is an identification on $\mathcal{S}$ such that the pair $(\mathcal{S}, \simeq_{\mathcal{S}})$ is complete. But what is the point of this reformulation? Well, first, since $\mathcal{S}$ becomes explicit in $\sum(\mathcal{S}, \simeq_{\mathcal{S}})$, i.e., a part of the structure, by defining v-strategies on $\sum(\mathcal{S}, \simeq_{\mathcal{S}})$ to be elements of $\mathcal{S}$ and dropping the completeness of $(\mathcal{S}, \simeq_{\mathcal{S}})$, we may regard $\sum(\mathcal{S}, \simeq_{\mathcal{S}})$ as a generalization of games; some v-strategy on $\sum(\mathcal{S}, \simeq_{\mathcal{S}})$ in the sense of Definition 5.3.15 may not correspond to any v-strategy on $\sum(\mathcal{S}, \simeq_{\mathcal{S}})$ in the sense just defined. Also, $\simeq_{\mathcal{S}}$ induces an equivalence relation on elements of $\mathcal{S}$ in the obvious manner; thus, a game $\sum(\mathcal{S}, \simeq_{\mathcal{S}})$ is defined in terms of its v-strategies and an equality between them, matching the idea of the meaning explanation which *defines* a formula in terms of a set of its proofs and an equality between them.

Remark. Although games are more primitive than strategies in conventional game semantics, Definition 5.3.15 and Theorem 5.3.19 enable us to reverse the order.

Moreover, since we take a disjoint union of sets of moves for $\sum(\mathcal{S}, \simeq_{\mathcal{S}})$, it is trivially a well-defined game even if we drop the consistency of $\mathcal{S}$; then $\sum(\mathcal{S}, \simeq_{\mathcal{S}})$ can be thought of as a *family of games* as its v-strategies may have different underlying games, where Player has an additional opportunity to 'declare' a v-strategy which simultaneously specifies an underlying game to play. It is to model *type dependency.*

This is the idea behind *predicative games*: A predicative game is a game of the form $\sum(\mathcal{S}, \simeq_{\mathcal{S}})$, where $\mathcal{S}$ is a (not necessarily consistent) set of v-strategies, $\simeq_{\mathcal{S}}$ satisfies the axiom CoV, and v-strategies on $\sum(\mathcal{S}, \simeq_{\mathcal{S}})$ are elements of $\mathcal{S}$.

Remark. If we had defined v-strategies on $\sum(\mathcal{S}, \simeq_{\mathcal{S}})$ in the sense of Definition 5.3.15, then we would not be able to enforce Player's 'declaration' of v-strategies.

However, this naive idea brings a *Russell's-like paradox* as follows. Let $\underline{G}$ be a v-strategy given by $P_{\underline{G}} \overset{\text{df.}}{=} \mathsf{Pref}(\{q.\|G\|\})$ with only the trivial identification of positions for each game G. Then, we may form a class $\mathcal{P}$ of v-strategies by:

$$\mathcal{P} \overset{\text{df.}}{=} \{\underline{G} \mid G \text{ is a game}, (\|G\|)_{\|\underline{G}\|} \notin M_G\}.$$

Now, observe that the game $\mathscr{P} \overset{\text{df.}}{=} \sum(\mathcal{P}, \{(\boldsymbol{s}, \boldsymbol{s}) \mid \underline{G} \in \mathcal{P}, \boldsymbol{s} \in P_{\underline{G}}\})$ gives rise to a paradox: If $(\|\mathscr{P}\|)_{\|\underline{\mathscr{P}}\|} \in M_{\mathscr{P}}$, then $(\|\mathscr{P}\|)_{\|\underline{\mathscr{P}}\|} \notin M_{\mathscr{P}}$, and vice versa. Our solution for this problem is the *ranks* of games, which we introduce in the next section.

Remark. One may simply have recourse to axiomatic set theory [142, 56] to circumvent the paradox because the set $M_{\mathscr{P}}$ is a proper class. However, we shall employ the ranks of games as they are the game-semantic counterpart of the ranks of types (Section 5.2), and moreover the both notions of ranks avoid the paradox in the same fashion.

At the end of the present section, let us answer the following question: One may wonder if it would be much simpler to require the declaration of strategies directly on games. However, it is not the case (and thus the present section is not meaningless). To see this point closely, let us define such a game $\mathscr{D}(G)$ for each game G by:

- $M_{\mathscr{D}(G)} \overset{\text{df.}}{=} \{q_G\} \cup \{\|\sigma\| \mid \sigma : G\} \cup M_G$;
- $\lambda_{\mathscr{D}(G)} : q_G \mapsto \mathsf{OQ}, \|\sigma\| \mapsto \mathsf{PA}, (m \in m_G) \mapsto \lambda_G(m)$;
- $\vdash_{\mathscr{D}(G)} \overset{\text{df.}}{=} \{(\star, q_G)\} \cup \{(q_G, \|\sigma\|) \mid \sigma : G\} \cup \{(\|\sigma\|, m) \mid \star \vdash_G m\}$
  $\cup \{(m, n) \mid m \neq \star, m \vdash_G n\}$;
- $P_{\mathscr{D}(G)} \overset{\text{df.}}{=} \mathsf{Pref}(\{q_G.\|\sigma\|.\boldsymbol{s} \mid \sigma : G, \boldsymbol{s} \in \sigma\})$, where q_G justifies $\|\sigma\|$ and $\|\sigma\|$ justifies initial moves occurring in $\boldsymbol{s}$;
- $\simeq_{\mathscr{D}(G)} \overset{\text{df.}}{=} \{(\boldsymbol{\epsilon}, \boldsymbol{\epsilon}), (q_G, q_G)\} \cup \{(q_G.\|\sigma\|.\boldsymbol{s}, q_G.\|\tau\|.\boldsymbol{t} \mid \boldsymbol{s} \simeq_G \boldsymbol{t}\}$.

Moreover, we may define the notion of *consistent* sets $\mathcal{S}$ of games and take the union games $\bigcup \mathcal{S}$ in the obvious manner.

Then, we may interpret a $\mathsf{\Pi}$-type $\mathsf{\Pi}(\mathsf{A}, \mathsf{B})$ by $\mathscr{D}(\Pi(A, B))$, where $\Pi(A, B)$ is the subgame of $A \Rightarrow \bigcup_{\sigma : A} B(\sigma)$ such that $\forall \boldsymbol{s} \in P_{\Pi(A,B)}, \sigma : G. \boldsymbol{s} \upharpoonright A \in \sigma \Rightarrow \boldsymbol{s} \upharpoonright B \in P_{B(\sigma)}$, where A and $(B(\sigma))_{\sigma:A}$ are interpretations of the type A and the dependent type B on A, respectively, such that $(B(\sigma))_{\sigma:A}$ is consistent. Actually, the construction $\mathscr{D}$ is not necessary at all to model $\mathsf{\Pi}$-types; the game $\Pi(A, B)$ may model $\mathsf{\Pi}(\mathsf{A}, \mathsf{B})$.

Thus, $\mathscr{D}$ is virtually to model $\mathsf{\Sigma}$-types $\mathsf{\Sigma}(\mathsf{A}, \mathsf{B})$; however, it does *not* work as unlike $\Pi(A, B)$ we cannot carve out the required subgame of $A \& \bigcup_{\sigma:A} B(\sigma)$ in terms of plays. This is why we had to reformulate games in terms of strategies in this section.

5.3.3 Predicative Games

As we have just observed, a naive formulation of sum games leads to a paradox. To circumvent this problem, we introduce the *ranks of moves*, which also induce the *ranks of p-games*. We begin with defining the ranks of moves:

Definition 5.3.20 (Ranked moves)**.** A move of a game is ***ranked*** if it is a pair (m, r) of some object m and a natural number $r \in \mathbb{N}$, which is usually written $[m]_r$. A ranked move $[m]_r$ is more specifically called an ***r^{th}-rank move***, and r is said to be the ***rank*** of the move. In particular, a 0th-rank move is called a ***mere move***.

Notation. We often write m for a ranked move $[m]_r$ when the rank r is not important.

Our intention is as follows. A mere move is just a move of a game in the usual sense, and an $(r+1)^{\text{st}}$-rank move is the *name* (Definition 5.3.22) of another game such that its moves are all ranked, and the supremum of the ranks of the moves is r:

Definition 5.3.21 (Ranked games)**.** A ***ranked game*** is a game G such that its moves are all ranked, and $\simeq_G$ respects the ranks of moves (i.e., $\boldsymbol{s} \simeq_G \boldsymbol{t}$ implies that $\boldsymbol{s}(i)$ and $\boldsymbol{t}(i)$ have the same rank for any $i \in \mathbb{N}$). The ***rank*** $\mathcal{R}(G)$ of G is given by:

$$\mathcal{R}(G) \overset{\text{df.}}{=} \begin{cases} 1 & \text{if } M_G = \emptyset \\ \mathsf{Sup}(\{ r \mid [m]_r \in M_G \}) + 1 & \text{otherwise.} \end{cases}$$

More specifically, G is called an ***$\mathcal{R}(G)^{th}$-rank game***.

Remark. One may wonder if the rank of a ranked game can be transfinite; however, as we shall see, the rank of a *predicative game* is always a natural number.

Definition 5.3.22 (Names of ranked games)**.** The ***name*** of a ranked game G, written $\|G\|$, is the pair $[G]_{\mathcal{R}(G)}$ of G (as a set) itself and its rank $\mathcal{R}(G)$.

Of course, the name of a ranked game can be a move of a ranked game; however, that name cannot be a move of the game itself because of its rank, which prevents the paradox described above (we shall show it formally as Proposition 5.3.34 below).

Notation. Given a v-strategy σ and a sequence $[\boldsymbol{s}]_{\boldsymbol{r}} = [m_1]_{r_1}[m_2]_{r_2} \dots [m_k]_{r_k}$ of ranked moves, let $[\boldsymbol{s}]_{\boldsymbol{r}}^{\|\sigma\|} \overset{\text{df.}}{=} [m_1]_{r_1}^{\|\sigma\|}.[m_2]_{r_2}^{\|\sigma\|} \dots [m_k]_{r_k}^{\|\sigma\|} \overset{\text{df.}}{=} [(m_1)_{\|\sigma\|}]_{r_1}.[(m_2)_{\|\sigma\|}]_{r_2} \dots [(m_k)_{\|\sigma\|}]_{r_k}$.

We are now ready to define the following central notion of the chapter:

Definition 5.3.23 (P-games)**.** For each integer $r \geqslant 1$, an ***r-predicative (r-p-) game*** is a quintuple $G = (M_G, \lambda_G, \vdash_G, P_G, \simeq_G)$ equipped with a set $\mathcal{VS}(G)$ of ranked v-strategies σ such that $M_\sigma \subseteq (\mathscr{B} \times \{0\}) \cup \{ \|H\| \mid H \text{ is an } l\text{-predicative game}, l < r \}$, where $\mathscr{B}$ is an arbitrarily fixed set containing q, each $n \in \mathbb{N}$, tt, ff and $\checkmark$, that satisfies:

- $\mathcal{R}(G) \overset{\mathrm{df.}}{=} \mathsf{Sup}(\{\mathcal{R}(\sigma) \mid \sigma \in \mathcal{VS}(G)\}) = r$, where $\mathcal{R}(G)$ is called the ***rank*** of G;
- $M_G = \sum_{\sigma \in \mathcal{VS}(G)} M_\sigma \overset{\mathrm{df.}}{=} \{[m]_l^{\|\sigma\|} \mid \sigma \in \mathcal{VS}(G), [m]_l \in M_\sigma\}$;
- $\lambda_G : [m]_l^{\|\sigma\|} \mapsto \lambda_\sigma([m]_l)$;
- $\vdash_G = \{(\star, [m]_l^{\|\sigma\|}) \mid \sigma \in \mathcal{VS}(G), \star \vdash_\sigma [m]_l\}$
  $\cup \{([m]_l^{\|\sigma\|}, [n]_k^{\|\sigma\|}) \mid \sigma \in \mathcal{VS}(G), [m]_l \vdash_\sigma [n]_k\}$;
- $P_G = \{\boldsymbol{\epsilon}, q_G\} \cup \{q_G.\|\sigma\|.[\boldsymbol{s}]_{\boldsymbol{l}}^{\|\sigma\|} \mid \sigma \in \mathcal{VS}(G), [\boldsymbol{s}]_{\boldsymbol{l}} \in P_\sigma\}$, where $q_G \overset{\mathrm{df.}}{=} [0]_0$;
- $\boldsymbol{\epsilon} \simeq_G \boldsymbol{\epsilon} \wedge q_G \simeq_G q_G \wedge \forall \sigma, \tau \in \mathcal{VS}(G).\, (q_G.\|\sigma\| \simeq_G q_G.\|\tau\| \Leftrightarrow \sigma \asymp \tau)$
  $\wedge \forall \sigma \in \mathcal{VS}(G).\, (q_G.\|\sigma\|.[\boldsymbol{s}]_{\boldsymbol{l}}^{\|\sigma\|} \simeq_G q_G.\|\sigma\|.[\boldsymbol{t}]_{\boldsymbol{r}}^{\|\sigma\|} \Leftrightarrow [\boldsymbol{s}]_{\boldsymbol{l}} \simeq_\sigma [\boldsymbol{t}]_{\boldsymbol{r}})$.

A ***predicative (p-) game*** is an r-predicative game for some $r \geqslant 1$. A ***v-strategy on a predicative game*** G is any element in $\mathcal{VS}(G)$, and $\sigma : G$ denotes $\sigma \in \mathcal{VS}(G)$.

Remark. Since moves of a 1-p-game are elements of the fixed set $\mathscr{B}$ (with rank 0), the class of moves of a p-game will never be a proper class.

Convention. Henceforth, every mere move is assumed to be an element of $\mathscr{B} \times \{0\}$.

Notation. We write $\mathcal{PG}_r$ (resp. $\mathcal{PG}_{\leqslant r}$) for the set of all r-p-games (resp. i-p-games with $1 \leqslant i \leqslant r$). Similar notation $\mathcal{VS}_l$ (resp. $\mathcal{VS}_{\leqslant l}$) applies to the set of all l^{th}-rank (resp. i^{th}-rank with $1 \leqslant i \leqslant l$) v-strategies.

Thus, p-games G are ranked games of the form $\sum(\mathcal{VS}(G), \simeq_G)$ inductively defined along with their ranks except that elements q_G and $\|\sigma\|$ are not counted as moves. Thus, strictly speaking, a p-game G is not a game in the sense of Definition 2.2.10 for elements q_G and $\|\sigma\|$ ($\sigma \in \mathcal{VS}(G)$) are not moves, they do not have labels, and $\|\sigma\|$ occurs in a position without a justifier. Nevertheless, we may easily fix this problem by adding q_G and $\|\sigma\|$ into M_G, defining $\lambda_G : q_G \mapsto \mathsf{OQ}, \|\sigma\| \mapsto \mathsf{PA}$, and so on; however, because the *initial protocol* $q_G.\|\sigma\|$ at the beginning of each position is conceptually made between Judge and Player, which is 'invisible' to Opponent, we have defined G as above so that the initial protocol does not appear in an O-view. Also, it prevents the name $\|\sigma\|$ of each $\sigma \in \mathcal{VS}(G)$ from affecting the rank $\mathcal{R}(G)$. Except these points, a p-game is a particular type of a ranked game.

Intuitively, a play of a p-game G proceeds as follows. At the beginning, Player has an opportunity to 'declare' a v-strategy $\sigma : G$ to Judge via an initial protocol $q_G.\|\sigma\|$, and then a play between Opponent and Player follows, where Player is forced to play by the 'declared' σ. The point is that $\sigma : G$ may range over v-strategies on different games in the conventional sense (Definition 5.3.15), and so Player may choose an underlying game when she selects σ. Thus, a p-game G is a *family of games*, where each component game corresponds to a maximal consistent subset of $\mathcal{VS}(G)$, which is to interpret *type dependency* in MLTT; see Example 5.3.27 below.

Remark. We may define an order $\leqslant$ between v-strategies on each p-game G by:

$$\sigma \leqslant \tau \overset{\mathrm{df.}}{\Leftrightarrow} \sigma \asymp \tau \wedge \sigma \trianglelefteq \tau$$

for all $\sigma, \tau : G$. Moreover, if each maximal consistent subset of $\mathcal{VS}(G)$ is complete, then G forms an *algebraic cpo* [67, 11, 173] just as games do [129]. It is then easy to show that this constraint is preserved under the constructions in Definition 5.3.40 and moreover satisfied by every p-game that models a type of MLTT; thus, we may reasonably adopt it as a part of the definition of p-games. Also, it is straightforward to see that each v-strategy on a *linear implication* (Definition 5.3.40) between such cpo-enriched p-games can be seen as a *continuous function* [67, 11, 173] with respect to the order $\leqslant$. However, we do not need such domain-theoretic structures in the rest of the chapter; thus, for simplicity, we have not imposed the condition on p-games.

As a generalization of identifications of valid strategies on games, let us define:

Definition 5.3.24 (Identification of v-strategies on p-games)**.** Given a p-game G, the equivalence relation $\simeq_G$ on v-strategies on G, called the ***identification*** of v-strategies on G, is given by $\sigma \simeq_G \tau \overset{\mathrm{df.}}{\Leftrightarrow} \mathsf{Pref}(\{q_G.\|\sigma\|.\boldsymbol{s} \mid \boldsymbol{s} \in P_\sigma\}) \simeq_G \mathsf{Pref}(\{q_G.\|\tau\|.\boldsymbol{t} \mid \boldsymbol{t} \in P_\tau\})$.

I.e., given v-strategies σ and τ on a p-game G, $\sigma \simeq_G \tau$ exactly when they are consistent and identified by $\simeq_G$ on 'actual' positions in the sense of Definition 5.3.1.

Example 5.3.25 (Example of p-games)**.** A maximal position of the 1-p-game N_1 corresponding to N is of the form $q_{N_1}.\|\underline{n}_0\|.[q]_0^{\|\underline{n}_0\|}.[n]_0^{\|\underline{n}_0\|}$, where $\underline{n}_0$ is obtained from $\underline{n}$ by changing each move m to the mere move $[m]_0$. For readability, we often abbreviate the position as $q_{N_1}.\|\underline{n}\|.q.n$, which corresponds to the position $q.n$ of N. Henceforth, we usually abbreviate N_1 as N.

Now, recall the ***terminal game*** $T \overset{\mathrm{df.}}{=} (\emptyset, \emptyset, \emptyset, \{\boldsymbol{\epsilon}\}, \{(\boldsymbol{\epsilon}, \boldsymbol{\epsilon})\})$ in Example 2.2.11 and the ***empty game*** $\mathbf{0} \overset{\mathrm{df.}}{=} \mathit{flat}(\emptyset)$ in Example 2.2.12. Again, abusing notation, we usually write T and $\mathbf{0}$ for the corresponding 1-p-games T_1 and $\mathbf{0}_1$, respectively. Also, we sometimes write $\mathbf{1}$ for T, and $\top$ and $\bot$ for the unique v-strategies $\top : T$ and $\bot : \mathbf{0}$, respectively, when we regard $\mathbf{1}$ and $\mathbf{0}$ as the simplest true and false formulas.

Let us call the p-games N, $\mathbf{1}$ (or T) and $\mathbf{0}$ the ***natural number p-game***, the ***unit p-game*** and the ***empty p-game***, respectively. Typical plays of these p-games may be depicted as in the following diagrams:

N	$\mathbf{1}$	$\mathbf{0}$
q_N	$q_{\mathbf{1}}$	$q_{\mathbf{0}}$
$\lVert\underline{n}\rVert$	$\lVert\top\rVert$	$\lVert\bot\rVert$
q		q
n		

Notice that a p-game G is completely determined by specifying a set $\mathcal{VS}(G)$ of all v-strategies on G and an identification $\simeq_G$ of positions of G. For instance, we have just defined N, $\mathbf{1}$ and $\mathbf{0}$ respectively by $\mathcal{VS}(N) \overset{\mathrm{df.}}{=} \{\bot\} \cup \{\underline{n} \mid n \in \mathbb{N}\}$, $\mathcal{VS}(\mathbf{1}) \overset{\mathrm{df.}}{=} \{\top\}$ and $\mathcal{VS}(\mathbf{0}) \overset{\mathrm{df.}}{=} \{\bot\}$, where $\simeq_N$, $\simeq_{\mathbf{1}}$ and $\simeq_{\mathbf{0}}$ are the trivial ones.

More in general, just like the sum games $\sum(\mathcal{S}, \simeq_{\mathcal{S}})$ in the last section, we may define a p-game $\oint(\mathcal{S}, \simeq_{\mathcal{S}})$ for any given pair $(\mathcal{S}, \simeq_{\mathcal{S}})$ of a set $\mathcal{S}$ of v-strategies such that there is an upper bound of the ranks of elements of $\mathcal{S}$ and an equivalence relation $\simeq_{\mathcal{S}}$ on the induced positions satisfying the axioms I1, I2 and I3 as well as the axiom

$$\text{(PI)}\ \boldsymbol{\epsilon} \simeq_{\mathcal{S}} \boldsymbol{\epsilon} \wedge q_{\oint(\mathcal{S},\simeq_{\mathcal{S}})} \simeq_{\mathcal{S}} q_{\oint(\mathcal{S},\simeq_{\mathcal{S}})} \wedge \forall \sigma, \tau \in \mathcal{S}.\, (q_{\oint(\mathcal{S},\simeq_{\mathcal{S}})}.\lVert\sigma\rVert \simeq_{\mathcal{S}} q_{\oint(\mathcal{S},\simeq_{\mathcal{S}})}.\lVert\tau\rVert \Leftrightarrow \sigma \asymp \tau)$$
$$\wedge\ \forall \sigma \in \mathcal{S}.\, (q_{\oint(\mathcal{S},\simeq_{\mathcal{S}})}.\lVert\sigma\rVert.[\boldsymbol{s}]_{\boldsymbol{l}}^{\lVert\sigma\rVert} \simeq_{\mathcal{S}} q_{\oint(\mathcal{S},\simeq_{\mathcal{S}})}.\lVert\sigma\rVert.[\boldsymbol{t}]_{\boldsymbol{r}}^{\lVert\sigma\rVert} \Leftrightarrow [\boldsymbol{s}]_{\boldsymbol{l}} \simeq_{\sigma} [\boldsymbol{t}]_{\boldsymbol{r}})$$

such that $\mathcal{VS}(\oint(\mathcal{S}, \simeq_{\mathcal{S}})) = \mathcal{S}$ and $\simeq_{\oint(\mathcal{S},\simeq_{\mathcal{S}})} = \simeq_{\mathcal{S}}$. Let us record this observation:

Definition 5.3.26 (Predicative union)**.** Given an integer $k \geqslant 1$ and a set $\mathcal{S} \subseteq \mathcal{VS}_{\leqslant k}$, the quadruple $\oint\mathcal{S} = (M_{\oint\mathcal{S}}, \lambda_{\oint\mathcal{S}}, \vdash_{\oint\mathcal{S}}, P_{\oint\mathcal{S}})$ is defined by:

- $M_{\oint\mathcal{S}} \overset{\mathrm{df.}}{=} \sum_{\sigma \in \mathcal{S}} M_\sigma = \{[m]_l^{\lVert\sigma\rVert} \mid \sigma \in \mathcal{VS}(G), [m]_l \in M_\sigma\}$;
- $\lambda_{\oint\mathcal{S}} : [m]_r^{\lVert\sigma\rVert} \mapsto \lambda_\sigma([m]_r)$;
- $\vdash_{\oint\mathcal{S}} \overset{\mathrm{df.}}{=} \{(\star, [m]_r^{\lVert\sigma\rVert}) \mid \sigma \in \mathcal{S}, \star \vdash_\sigma [m]_r\} \cup \{([m]_r^{\lVert\sigma\rVert}, [n]_l^{\lVert\sigma\rVert}) \mid \sigma \in \mathcal{S}, [m]_r \vdash_\sigma [n]_l\}$;
- $P_{\oint\mathcal{S}} \overset{\mathrm{df.}}{=} \{\boldsymbol{\epsilon}, q_{\oint\mathcal{S}}\} \cup \{q_{\oint\mathcal{S}}.\lVert\sigma\rVert.[\boldsymbol{s}]_{\boldsymbol{r}}^{\lVert\sigma\rVert} \mid \sigma \in \mathcal{S}.\, [\boldsymbol{s}]_{\boldsymbol{r}} \in P_\sigma\}$, where $q_{\oint\mathcal{S}} \overset{\mathrm{df.}}{=} [0]_0$.

A pair $(\mathcal{S}, \simeq_{\mathcal{S}})$ is a ***predicative pair*** if $\simeq_{\mathcal{S}}$ is an equivalence relation on $P_{\oint\mathcal{S}}$ that satisfies the axioms I1, I2, I3 and PI. The ***predicative union*** $\oint(\mathcal{S}, \simeq_{\mathcal{S}})$ on $(\mathcal{S}, \simeq_{\mathcal{S}})$ is the p-game defined by $\oint(\mathcal{S}, \simeq_{\mathcal{S}}) \overset{\mathrm{df.}}{=} (\oint\mathcal{S}).(\simeq_{\mathcal{S}}) = (M_{\oint\mathcal{S}}, \lambda_{\oint\mathcal{S}}, \vdash_{\oint\mathcal{S}}, P_{\oint\mathcal{S}}, \simeq_{\mathcal{S}})$.

Clearly, every p-game G is a predicative union: $G = \oint(\mathcal{VS}(G), \simeq_G)$.

Notation. Given $k \geqslant 1$ and $\mathcal{S} \subseteq \mathcal{VS}_{\leqslant k}$, abusing notation, let us define

$$\oint\mathcal{S} \overset{\mathrm{df.}}{=} \oint(\mathcal{S}, \{(\boldsymbol{\epsilon}, \boldsymbol{\epsilon}), (q_{\oint\mathcal{S}}, q_{\oint\mathcal{S}})\} \cup \{(q_{\oint\mathcal{S}}.\lVert\sigma\rVert.[\boldsymbol{s}]_{\boldsymbol{l}}^{\lVert\sigma\rVert}, q_{\oint\mathcal{S}}.\lVert\sigma\rVert.[\boldsymbol{t}]_{\boldsymbol{r}}^{\lVert\sigma\rVert}) \mid \sigma \in \mathcal{S}, [\boldsymbol{s}]_{\boldsymbol{l}} \simeq_\sigma [\boldsymbol{t}]_{\boldsymbol{r}}\}).$$

Example 5.3.27 (An example of predicative union)**.** Consider the predicative union $\oint\{\underline{100}, \top\}$, whose typical positions are as depicted in the following tables:

$$\begin{tikzcd}[row sep=tiny]
\underline{\oint\{\underline{100}, \top\}} & \underline{\oint\{\underline{100}, \top\}} \\
q_{\oint\{\underline{100}, \top\}} & q_{\oint\{\underline{100}, \top\}} \\
\|\underline{100}\| & \|\top\| \\
q & \\
100 \arrow[u, bend right] &
\end{tikzcd}$$

This p-game is not very meaningful, but it illustrates the point that a p-game can be seen as a family of games.

Also, given a game A, we define the 1-p-game A_1 by $A_1 \overset{\text{df.}}{=} \oint(\{\alpha_0 \mid \alpha : A\}, \simeq_{A_1})$, where α_0 is obtained from α by changing each move m to the mere move $[m]_0$, and $\simeq_{A_1} \overset{\text{df.}}{=} \{(\boldsymbol{\epsilon}, \boldsymbol{\epsilon}), (q_{A_1}, q_{A_1})\} \cup \{(q_{A_1}.\|\alpha_0\|.[\boldsymbol{s}]_{\mathbf{0}}^{\|\alpha_0\|}, q_{A_1}.\|\alpha'_0\|.[\boldsymbol{t}]_{\mathbf{0}}^{\|\alpha'_0\|}) \mid \alpha, \alpha' : A, \boldsymbol{s} \in \overline{\alpha}, \boldsymbol{t} \in \overline{\alpha'}, \boldsymbol{s} \simeq_A \boldsymbol{t}\}$. This construction generalizes the examples given in Example 5.3.25. Again, we usually abbreviate the p-game A_1, a v-strategy $\alpha_0 : A_1$ and a position $[\boldsymbol{s}]_{\mathbf{0}}^{\|\alpha_0\|} \in P_{A_1}$ as A, $\alpha : A$ and $\boldsymbol{s} \in P_A$, respectively.

Let us define another convenient construction:

Definition 5.3.28 (Parallel union)**.** Given an integer $k \geqslant 1$ and a set $\mathcal{S} \subseteq \mathcal{PG}_{\leqslant k}$, the ***parallel union*** $\int\mathcal{S}$ is the p-game given by:

- $M_{\int\mathcal{S}} \overset{\text{df.}}{=} \bigcup_{G \in \mathcal{S}} M_G = \{[m]_r^{\|\sigma\|} \mid \exists G \in \mathcal{S}.\, [m]_r^{\|\sigma\|} \in M_G\}$;
- $\lambda_{\int\mathcal{S}} : [m]_r^{\|\sigma\|} \mapsto \lambda_\sigma([m]_r)$;
- $\vdash_{\int\mathcal{S}} \overset{\text{df.}}{=} \{(\star, [m]_r^{\|\sigma\|}) \mid \exists G \in \mathcal{S}.\, \star \vdash_G [m]_r^{\|\sigma\|}\}$
  $\cup \{([m]_r^{\|\sigma\|}, [n]_l^{\|\sigma\|}) \mid \exists G \in \mathcal{S}.\, [m]_r^{\|\sigma\|} \vdash_G [n]_l^{\|\sigma\|}\}$;
- $P_{\int\mathcal{S}} \overset{\text{df.}}{=} \{\boldsymbol{\epsilon}, q_{\int\mathcal{S}}\} \cup \{q_{\int\mathcal{S}}\|\sigma\|[\boldsymbol{s}]_{\boldsymbol{r}}^{\|\sigma\|} \mid \exists G \in \mathcal{S}.\, q_G\|\sigma\|[\boldsymbol{s}]_{\boldsymbol{r}}^{\|\sigma\|} \in P_G\}$, where $q_{\int\mathcal{S}} \overset{\text{df.}}{=} [0]_0$;
- $\simeq_{\int\mathcal{S}} \overset{\text{df.}}{=} \{(\boldsymbol{\epsilon}, \boldsymbol{\epsilon}), (q_{\int\mathcal{S}}, q_{\int\mathcal{S}})\}$
  $\cup \{(q_{\int\mathcal{S}}\|\sigma\|[\boldsymbol{s}]_{\boldsymbol{r}}^{\|\sigma\|}, q_{\int\mathcal{S}}\|\tau\|[\boldsymbol{t}]_{\boldsymbol{l}}^{\|\tau\|}) \mid \exists G \in \mathcal{S}.\, q_G\|\sigma\|[\boldsymbol{s}]_{\boldsymbol{r}}^{\|\sigma\|} \simeq_G q_G\|\tau\|[\boldsymbol{t}]_{\boldsymbol{l}}^{\|\tau\|}\}$.

That is, the construction $\int$ forms a p-game from a set of p-games by 'unifying the first moves q_G ($G \in \mathcal{S}$)'. Clearly, each parallel union is a well-defined p-game, where note that we take union for identification of positions (n.b., intersection would not work), and a v-strategy $\sigma : G$ for some $G \in \mathcal{S}$ is again a v-strategy on $\int\mathcal{S}$.

Remark. We take union, not disjoint union, of moves for parallel union since otherwise Player would be able to see component p-games of the interpretation of Π- and Σ-types by 'tags' for the disjoint union, (partially) violating the 'invisibility' of anti-strategies. Also, such 'tags' would prohibit the *uniformity* condition in Definition 5.3.35 from functioning properly on the interpretation of Π-types.

Example 5.3.29 (An example of parallel union)**.** Consider the parallel union $\int\{N, \mathbf{1}\}$, whose typical positions are as depicted in the following tables:

$$\begin{array}{c} \int\{N,\mathbf{1}\} \\ \hline q_{\int\{N,\mathbf{1}\}} \\ \|\underline{100}\| \\ q \\ 100 \end{array} \qquad \begin{array}{c} \int\{N,\mathbf{1}\} \\ \hline q_{\int\{N,\mathbf{1}\}} \\ \|\top\| \end{array}$$

where for brevity we omit 'tags' on moves occurring after initial protocols.

Next, let us adapt the subgame relation to p-games. In view of Theorem 5.3.19, the subgame relation on games (Definition 2.2.13) and the subset relation on their sets of v-strategies are logically equivalent. Hence, it is natural to define:

Definition 5.3.30 (P-subgames)**.** A ***predicative subgame (p-subgame)*** of a p-game G is a p-game H that satisfies $\mathcal{VS}(H) \subseteq \mathcal{VS}(G)$ and $\simeq_H \, = \, \simeq_G \cap \, (P_H \times P_H)$.

Notation. We write $H \trianglelefteqslant G$ to mean that H is a p-subgame of a p-game G.

Given p-games A and B, we clearly have $A = B \Leftrightarrow A \trianglelefteqslant B \wedge B \trianglelefteqslant A$; thus, the relation $\trianglelefteqslant$ forms a *partial order* on p-games.

Example 5.3.31 (Examples of p-subgames)**.** In contrast to conventional games in Chapter 2, among which the terminal game T is the least one, the terminal p-game T_1 is *not* the least p-game, e.g., $T_1 \not\trianglelefteqslant \mathbf{0}_1$, $T_1 \not\trianglelefteqslant \mathbf{1}_1$ and $T_1 \not\trianglelefteqslant N_1$. Instead, the least p-game is the ***initial p-game*** I defined by $I \overset{\text{df.}}{=} \oint(\emptyset, \simeq_I)$, where $\simeq_I \overset{\text{df.}}{=} \{(\boldsymbol{\epsilon}, \boldsymbol{\epsilon}), (q_I, q_I)\}$.

On the other hand, the subgame relation on games has been certainly generalized to p-games for we have $A \trianglelefteqslant B \Leftrightarrow A_1 \trianglelefteqslant B_1$ for any games A and B.

We now define a certain kind of p-games to interpret universes of MLTT, which should be called *universe games*. As we are interested in MLTT with a *hierarchy* of universes, we shall construct the corresponding hierarchy of universe games.

Definition 5.3.32 (Universe games)**.** For each $k \in \mathbb{N}$, the ***k^{th}-universe game*** is the p-game $\mathcal{U}_k \overset{\text{df.}}{=} \oint\{\underline{G} \mid G \in \mathcal{PG}_{\leqslant k+1}\}$, where $\underline{G} \overset{\text{df.}}{=} \mathit{flat}(\{\|G\|\})_1$. A ***universe game*** is the k^{th}-universe game for some $k \in \mathbb{N}$, and it is often abbreviated as $\mathcal{U}$.

Notation. Given a v-strategy $\mu : T \Rightarrow \mathcal{U}$, we write $\mathit{El}(\mu)$ for the unique p-game such that $\underline{\mathit{El}(\mu)}^T = \mu$ if it exists (otherwise it is undefined). Since the identification $\simeq_{\mathcal{U}}$ of positions is just the equality $=$, we may define the operation El on the equivalence classes $[\mu]$ of v-strategies $\mu : T \Rightarrow \mathcal{U}$ as well in the obvious manner.

Proposition 5.3.33 (Predicativity of universe games). *For each $k \in \mathbb{N}$, the k^{th}-universe game $\mathcal{U}_k$ is a $(k+2)$-p-game.*

Proof. Observe that $T \in \mathcal{PG}_1$ and $\forall k \in \mathbb{N}. \mathcal{U}_k \in \mathcal{PG}_{k+2}$ by induction on k. □

As a consequence, we have $\underline{\mathcal{U}_i} : \mathcal{U}_j$ for all $i, j \in \mathbb{N}$ with $i < j$, which conceptually induces a *hierarchy of universe games*: $\mathcal{U}_0 : \mathcal{U}_1 : \mathcal{U}_2 \dots$ On the other hand, we clearly have $\underline{\mathcal{U}_k} \notin \mathcal{VS}(\mathcal{U}_k)$ for all $k \in \mathbb{N}$ because ranks of p-games prohibit a Russell's-like paradox as promised previously:

Proposition 5.3.34 (Paradox-free). *The name of a p-game is not a move of the game itself, i.e., if G is a p-game, then $\|G\| \notin M_G$.*

Proof. Let G be an r-predicative game. By the definition, each move $[m]_l \in M_G$ satisfies $l < r$. Thus, since the name $\|G\|$ has rank r, it cannot be in M_G. □

By the definition, $G \in \mathcal{PG}_{\leqslant k+1} \Leftrightarrow \underline{G} : \mathcal{U}_k$ for all $k \in \mathbb{N}$. Thus, as intended, the kth-universe game $\mathcal{U}_k$ is the 'universe' of all i-p-games with $1 \leqslant i \leqslant k+1$. Intuitively, a play of a universe game $\mathcal{U}$ starts with Opponent's question q, meaning 'What is your game?', and Player answers it by the name of a p-game such as $\|G\|$, meaning 'It is the game G!' (here we omit the initial protocol for brevity).

Remark. Now, interpreting ranks of types by ranks of p-games, the increments $+2$ (U-Form), $+1$ (U-Elim) in the syntax (see Section 5.2) should make sense.

5.3.4 The CCC of Logical Predicative Games

This section generalizes the existing constructions on games (in Chapter 2) so that they preserve *predicativity* of games, based on which we shall define the CCC $\mathcal{LPG}$ of *logical* p-games and *winning* v-strategies, generalizing the CCC $\mathcal{LMG}$ in Chapter 2.

To obtain a cartesian closed structure of p-games, however, there is a technical challenge in linear implication $\multimap$ (Definition 2.2.21). Note that a $\mathsf{\Pi}$-type $\mathsf{\Pi}_{\mathsf{a}:\mathsf{A}}\mathsf{B}(\mathsf{a})$ is a generalization of the function type $\mathsf{A} \Rightarrow \mathsf{B}$. Thus, an interpretation of $\mathsf{\Pi}_{\mathsf{a}:\mathsf{A}}\mathsf{B}(\mathsf{a})$ must be a generalization of implication $A \Rightarrow B$, where $B(\sigma)$ may vary, depending on a v-strategy $\sigma : A$ which Opponent chooses to play. Naively, it seems that we may interpret it by the p-subgame of $A \Rightarrow \int\{B(\sigma) \mid \sigma : A\}$ whose v-strategies ϕ satisfy $\phi \circ \sigma^\dagger : B(\sigma)$ for all $\sigma : A$. Now, the initial protocol becomes $q_B.q_A.\|\sigma\|.\|\phi \circ \sigma^\dagger\|$, and then a play of the p-subgame $\sigma \Rightarrow \phi \circ \sigma^\dagger \trianglelefteq \sigma \Rightarrow B(\sigma)$ follows.

This nicely captures the phenomenon of $\mathsf{\Pi}$-types, but imposes another challenge: The play described above no longer follows the initial protocol because the second move q_A is not the name of a v-strategy to follow. Even if we somehow enforce the protocol, then we would lose the initial two questions and two answers to determine the component p-game $\sigma \Rightarrow B(\sigma)$.

Our solution for this problem is the following:

Definition 5.3.35 (PoPLIs)**.** Given p-games A and B, a ***product of pointwise linear implications (PoPLIs)*** from A to B is a v-strategy of the form $\phi = \&_{\sigma:A}\phi_\sigma$, where $(\phi_\sigma)_{\sigma:A}$ is a family of v-strategies $\phi_\sigma : \sigma \multimap \pi_\phi(\sigma)$ and $\pi_\phi \in \mathcal{VS}(B)^{\mathcal{VS}(A)}$, that satisfies the following ***uniformity*** axiom:

(Uni) $\forall \sigma_1, \sigma_2 : A, \boldsymbol{s}m \in P^{\mathsf{Odd}}_{\phi_{\sigma_1}} \cap P^{\mathsf{Odd}}_{\phi_{\sigma_2}}, \boldsymbol{s}mn \in P^{\mathsf{Odd}}_{\phi_{\sigma_1}} \cup P^{\mathsf{Odd}}_{\phi_{\sigma_2}}. \boldsymbol{s}mn \in P_{\phi_{\sigma_1}} \Leftrightarrow \boldsymbol{s}mn \in P_{\phi_{\sigma_2}}$

where $\&_{\sigma:A}\phi_\sigma$ is defined by:

- $M_{\&_{\sigma:A}\phi_\sigma} \overset{\mathrm{df.}}{=} \{[m]_r^{\|\sigma\|} \mid \sigma : A, [m]_r \in M_{\phi_\sigma}\}$;
- $\lambda_{\&_{\sigma:A}\phi_\sigma} : [m]_r^{\|\sigma\|} \mapsto \lambda_{\phi_\sigma}([m]_r)$;
- $\vdash_{\&_{\sigma:A}\phi_\sigma} \overset{\mathrm{df.}}{=} \{(\star, [m]_r^{\|\sigma\|}) \mid \sigma : A, \star \vdash_{\phi_\sigma} [m]_r\}$ $\cup \{([m]_r^{\|\sigma\|}, [n]_l^{\|\sigma\|}) \mid \sigma : A, [m]_r \vdash_{\phi_\sigma} [n]_l\}$;
- $P_{\&_{\sigma:A}\phi_\sigma} \overset{\mathrm{df.}}{=} \bigcup_{\sigma:A}\{[\boldsymbol{s}]_{\boldsymbol{r}}^{\|\sigma\|} \mid [\boldsymbol{s}]_{\boldsymbol{r}} \in P_{\phi_\sigma}\}$, where $[\boldsymbol{s}]_{\boldsymbol{r}}^{\|\sigma\|} \overset{\mathrm{df.}}{=} [m_1]_{r_1}^{\|\sigma\|}[m_2]_{r_2}^{\|\sigma\|} \dots [m_k]_{r_k}^{\|\sigma\|}$ if $[\boldsymbol{s}]_{\boldsymbol{r}} = [m_1]_{r_1}[m_2]_{r_2} \dots [m_k]_{r_k}$;
- $\simeq_{\&_{\sigma:A}\phi_\sigma} \overset{\mathrm{df.}}{=} \{([\boldsymbol{s}]_{\boldsymbol{r}}^{\|\sigma\|}, [\boldsymbol{t}]_{\boldsymbol{l}}^{\|\sigma\|}) \mid \sigma : A, [\boldsymbol{s}]_{\boldsymbol{r}} \simeq_{\phi_\sigma} [\boldsymbol{t}]_{\boldsymbol{l}}\}$.

Notation. The set of all PoPLIs from A to B is written $\mathcal{LI}(A, B)$.

Definition 5.3.36 (Composition on PoPLIs)**.** Given p-games A, B and C, and PoPLIs $\phi \in \mathcal{LI}(A, B)$ and $\psi \in \mathcal{LI}(B, C)$, the ***composition*** $\psi \circ \phi$ (also written $\phi; \psi$) of ϕ and ψ is defined by:

$$\psi \circ \phi \overset{\mathrm{df.}}{=} \&_{\sigma:A}\psi_{\pi_\phi(\sigma)} \circ \phi_\sigma.$$

Lemma 5.3.37 (Well-defined composition on PoPLIs)**.** *For any p-games A, B and C, if $\phi \in \mathcal{LI}(A, B)$ and $\psi \in \mathcal{LI}(B, C)$, then $\psi \circ \phi \in \mathcal{LI}(A, C)$.*

Proof. Immediate from Lemma 2.3.16, where uniformity is clearly preserved. □

Clearly, PoPLIs $\phi \in \mathcal{LI}(A, B)$ are well-defined v-strategies. They are intended to be v-strategies on the linear implications $A \multimap B$ defined in Definition 5.3.40 below. The basic idea is as follows. When Opponent performs the first move in $A \multimap B$, he is enforced to determine a v-strategy σ on A due to 'tags' for the linear implication, which together with Player's 'declared' strategy $\phi : A \multimap B$ in turn selects her v-strategy $\pi_\phi(\sigma)$ on B. Note that ϕ has to be uniform because she should not be able to see Opponent's choice $\sigma : A$. In fact, by uniformity, PoPLIs are a natural generalization of strategies on linear implications (Definition 2.2.21) as we shall see.

Next, we generalize exponential ! of games (Definition 2.2.25). The point is that a strategy on an exponential $!A$ is not necessarily the exponential of a strategy on A, unlike tensor $\otimes$ or product &, which prohibits us from defining $!A$ in terms of strategies on A. We have to overcome this point since p-games are defined in terms of its v-strategies. However, it is not a difficult problem; it suffices to consider:

Definition 5.3.38 (C-tensor)**.** Given a countably infinite family $\sigma = (\sigma_n)_{n \in \mathbb{N}}$ of strategies σ_n on a game A, their ***countable (c-) tensor*** $\otimes\sigma = \otimes_{n\in\mathbb{N}}\sigma_n$ is given by:

- $M_{\otimes\sigma} \overset{\text{df.}}{=} \{(a, n) \mid n \in \mathbb{N}, a \in M_{\sigma_n}\}$;
- $\lambda_{\otimes\sigma} : (a, n) \mapsto \lambda_{\sigma_n}(a)$;
- $\vdash_{\otimes\sigma} \overset{\text{df.}}{=} \{(\star, (a, n)) \mid n \in \mathbb{N}, \star \vdash_{\sigma_n} a\} \cup \{((a, n), (a', n)) \mid n \in \mathbb{N}, a \vdash_{\sigma_n} a'\}$;
- $P_{\otimes\sigma} \overset{\text{df.}}{=} \{\boldsymbol{s} \in \mathscr{L}_{\otimes\sigma} \mid \forall n \in \mathbb{N}.\, \boldsymbol{s} \upharpoonright n \in P_{\sigma_n}\}$;
- $\boldsymbol{s} \simeq_{\otimes\sigma} \boldsymbol{t} \overset{\text{df.}}{\Leftrightarrow} \exists \varphi \in \mathcal{P}(\mathbb{N}).\, \pi_2^*(\boldsymbol{s}) = (\varphi \circ \pi_2)^*(\boldsymbol{t}) \wedge \forall n \in \mathbb{N}.\, \boldsymbol{s} \upharpoonright \varphi(n) \simeq_A \boldsymbol{t} \upharpoonright n$.

Lemma 5.3.39 (Well-defined c-tensor)**.** *For any countably infinite family $(\sigma_n)_{n\in\mathbb{N}}$ of v-strategies on a game A, the c-tensor $\otimes_{n\in\mathbb{N}}\sigma_n$ is a v-strategy on the exponential $!A$. Conversely, every v-strategy on $!A$ is a c-tensor of v-strategies on A.*

Proof. Straightforward. □

Remark. 'Tags' $(\_, n)$, where $n \in \mathbb{N}$, are deleted when we take a view of a position of a c-tensor similarly to exponential ! of games.

We are now ready to define constructions on p-games:

Definition 5.3.40 (Constructions on p-games)**.** Given a family $(G_i)_{i\in I}$ of p-games, let $G_1 \multimap G_2 \overset{\text{df.}}{=} \oint(\mathcal{LI}(G_1, G_2), \simeq_{G_1 \multimap G_2})$, $!G_1 \overset{\text{df.}}{=} \oint(\{\otimes_{n\in\mathbb{N}}\sigma_n \mid \forall n \in \mathbb{N}.\, \sigma_n : G_1\}, \simeq_{!G_1})$ and $\clubsuit_{i\in I}G_i \overset{\text{df.}}{=} \oint(\{\clubsuit_{i\in I}\sigma_i \mid \forall i \in I.\, \sigma_i : G_i\}, \simeq_{\clubsuit_{i\in I}G_i})$ if $\clubsuit_{i\in I}$ is $\otimes$ or &, where:

- $q_{G_1 \multimap G_2}\|\phi\|[\boldsymbol{s}]_{\boldsymbol{r}}^{\|\sigma\|} \simeq_{G_1 \multimap G_2} q_{G_1 \multimap G_2}\|\psi\|[\boldsymbol{t}]_{\boldsymbol{l}}^{\|\tau\|} \overset{\text{df.}}{\Leftrightarrow} \phi \asymp \psi \wedge q_{G_1}\|\sigma\|([\boldsymbol{s}]_{\boldsymbol{r}}^{\|\sigma\|} \upharpoonright G_1) \simeq_{G_1} q_{G_1}\|\tau\|([\boldsymbol{t}]_{\boldsymbol{l}}^{\|\tau\|} \upharpoonright G_1) \wedge q_{G_2}\|\pi_\phi(\sigma)\|([\boldsymbol{s}]_{\boldsymbol{r}}^{\|\sigma\|} \upharpoonright G_2) \simeq_{G_2} q_{G_2}\|\pi_\psi(\tau)\|([\boldsymbol{t}]_{\boldsymbol{l}}^{\|\tau\|} \upharpoonright G_2)$ $\wedge \forall j \in \mathbb{N}.\, (([\boldsymbol{s}(j)]_{\boldsymbol{r}(j)} \in M_{G_1} \Leftrightarrow [\boldsymbol{t}(j)]_{\boldsymbol{l}(j)} \in M_{G_1}) \wedge (\mathcal{J}([\boldsymbol{s}(j)]_{\boldsymbol{r}(j)}) = [\boldsymbol{s}(k)]_{\boldsymbol{r}(k)} \Leftrightarrow \mathcal{J}([\boldsymbol{t}(j)]_{\boldsymbol{l}(j)}) = [\boldsymbol{t}(k)]_{\boldsymbol{l}(k)}))$;

- $q_{!G_1} \| \otimes_{n\in\mathbb{N}} \sigma_n \| [\boldsymbol{s}]_{\boldsymbol{r}}^{\|\otimes_{n\in\mathbb{N}}\sigma_n\|} \simeq_{!G_1} q_{!G_1} \| \otimes_{n\in\mathbb{N}} \tau_n \| [\boldsymbol{t}]_{\boldsymbol{l}}^{\|\otimes_{n\in\mathbb{N}}\tau_n\|} \overset{\text{df.}}{\Leftrightarrow} \exists \varphi \in \mathcal{P}(\mathbb{N}).\, \pi_2^*(\boldsymbol{s}) = (\varphi \circ \pi_2)^*(\boldsymbol{t}) \wedge \forall n \in \mathbb{N}.\, q_{G_1} \|\sigma_{\varphi(n)}\| ([\boldsymbol{s}]_{\boldsymbol{r}} \upharpoonright \varphi(n))_{\|\sigma_{\varphi(n)}\|} \simeq_{G_1} q_{G_1} \|\tau_n\| ([\boldsymbol{t}]_{\boldsymbol{l}} \upharpoonright n)_{\|\tau_n\|}$;
- $q_{\clubsuit_{i\in I} G_i} \| \clubsuit_{i\in I} \sigma_i \| [\boldsymbol{s}]_{\boldsymbol{r}}^{\|\clubsuit_{i\in I}\sigma_i\|} \simeq_{\clubsuit_{i\in I} G_i} q_{\clubsuit_{i\in I} G_i} \| \clubsuit_{i\in I} \tau_i \| [\boldsymbol{t}]_{\boldsymbol{l}}^{\|\clubsuit_{i\in I}\tau_i\|} \overset{\text{df.}}{\Leftrightarrow} (\forall j \in \mathbb{N}.\, [\boldsymbol{s}(j)]_{\boldsymbol{r}(j)} \in M_{G_1} \Leftrightarrow [\boldsymbol{t}(j)]_{\boldsymbol{l}(j)} \in M_{G_1}) \wedge \forall i \in I.\, q_{G_i} \|\sigma_i\| ([\boldsymbol{s}]_{\boldsymbol{r}} \upharpoonright G_i)_{\|\sigma_i\|} \simeq_{G_i} q_{G_i} \|\tau_i\| ([\boldsymbol{t}]_{\boldsymbol{l}} \upharpoonright G_i)_{\|\tau_i\|}$ if $\clubsuit_{i\in I}$ is tensor $\otimes$ or product $\&$.

Notation. Henceforth, we write $\clubsuit_{i\in I}$ for $\multimap$ and ! as well.

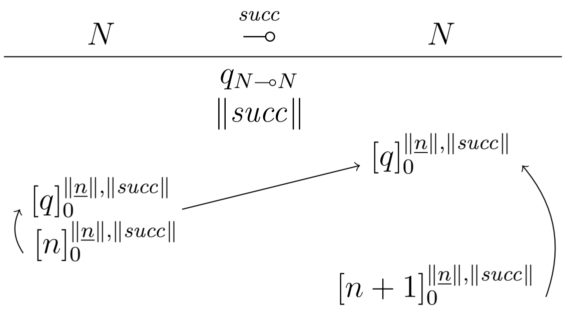
Theorem 5.3.41 (Well-defined constructions on p-games). *P-games are closed under all the constructions defined in Definition 5.3.40.*

Proof. It is not hard to see that the constructions on identifications of positions preserve the axioms I1, I2, I3 and PI. Then, the theorem follows from the well-definedness of the constructions on v-strategies, and Lemmata 5.3.37 and 5.3.39. □

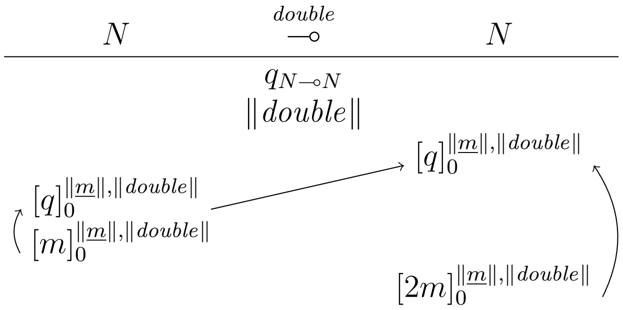
Corollary 5.3.42 (Well-defined constructions on v-strategies). *Given p-games A, B, C and D, and v-strategies $\phi : A \multimap B$, $\psi : C \multimap D$, $\varphi : !A \multimap B$, $\vartheta : A \multimap C$, the tensor $\phi \otimes \psi : A \otimes C \multimap B \otimes D$, the pairing $\langle \phi, \vartheta \rangle : A \multimap B\&C$, the promotion $\varphi^\dagger : !A \multimap !B$ and the composition $\psi \circ \vartheta : A \multimap D$ are all well-defined v-strategies.*

Proof. Immediate from Theorem 5.3.41. □

Example 5.3.43 (Examples of constructions on p-games). Consider the v-strategies $succ, double : N \multimap N$:

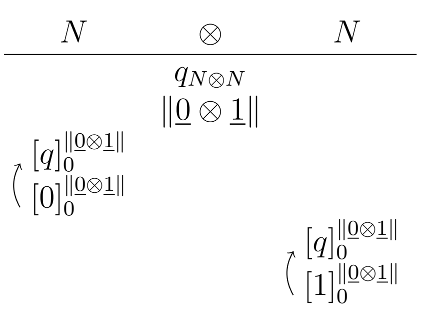


$N \overset{succ}{\multimap} N$
$q_{N \multimap N}$
$\|succ\|$
$[q]_0^{\|\underline{n}\|, \|succ\|}$
$[q]_0^{\|\underline{n}\|, \|succ\|}$
$[n]_0^{\|\underline{n}\|, \|succ\|}$
$[n+1]_0^{\|\underline{n}\|, \|succ\|}$

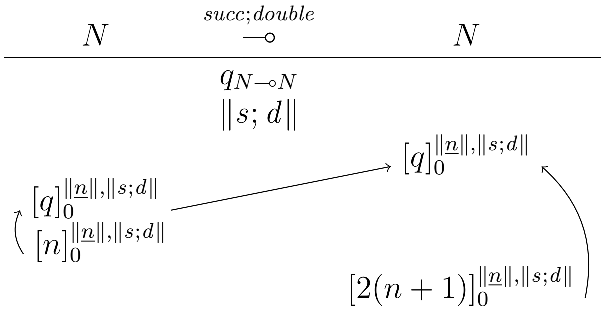


$N \overset{double}{\multimap} N$
$q_{N \multimap N}$
$\|double\|$
$[q]_0^{\|\underline{m}\|, \|double\|}$
$[q]_0^{\|\underline{m}\|, \|double\|}$
$[m]_0^{\|\underline{m}\|, \|double\|}$
$[2m]_0^{\|\underline{m}\|, \|double\|}$

The tensor $\underline{0} \otimes \underline{1} : N \otimes N$ and the composition $succ; double : N \multimap N$ play as follows:



$N \otimes N$
$q_{N \otimes N}$
$\|\underline{0} \otimes \underline{1}\|$
$[q]_0^{\|\underline{0}\otimes\underline{1}\|}$
$[0]_0^{\|\underline{0}\otimes\underline{1}\|}$
$[q]_0^{\|\underline{0}\otimes\underline{1}\|}$
$[1]_0^{\|\underline{0}\otimes\underline{1}\|}$



$N \overset{succ;double}{\multimap} N$
$q_{N \multimap N}$
$\|s; d\|$
$[q]_0^{\|\underline{n}\|, \|s;d\|}$
$[q]_0^{\|\underline{n}\|, \|s;d\|}$
$[n]_0^{\|\underline{n}\|, \|s;d\|}$
$[2(n+1)]_0^{\|\underline{n}\|, \|s;d\|}$

where $s; d$ is an abbreviation for *succ*; *double*. Note that conceptually Player (resp. Opponent) can see only the 'tags' $\|\underline{0} \otimes \underline{1}\|$ and $\|s; d\|$ (resp. $\|\underline{n}\|$ and $\|\underline{m}\|$).

Importantly, for p-games G_i corresponding to games, i.e., the pairs $(\mathcal{VS}(G_i), \simeq_{G_i})$ are all complete (Theorem 5.3.19), the constructions in Definition 5.3.40 coincide with the constructions on games in Chapter 2, where the resulting pairs $(\mathcal{VS}(\clubsuit_{i \in I} G_i), \simeq_{\mathcal{VS}(\clubsuit_{i \in I} G_i)}$) are again complete. For instance, consider the linear implication $A \multimap B$ between games A and B. Given a v-strategy $\phi : A \multimap B$, we may define a *v-substrategy*, which is just the subgame $\phi_\sigma \trianglelefteq \phi$ in the sense of Definition 2.2.13 for each $\sigma : A$ by $P_{\phi_\sigma} \overset{\mathrm{df.}}{=} \{ \boldsymbol{s} \in P_\phi \mid \boldsymbol{s} \upharpoonright A \in P_\sigma \}$; then the product $\&_{\sigma : A} (\phi_\sigma)_0$, where $(\phi_\sigma)_0$ is the first-rank v-strategy obtained from ϕ_σ by replacing moves m with $[m]_0$, clearly forms a v-strategy on the p-game $A_1 \multimap B_1$, which essentially coincides with ϕ itself. Conversely, any v-strategy $\psi : A_1 \multimap B_1$ may be seen as a v-strategy on the game $A \multimap B$, more specifically as $\bigcup (\{ (\psi_\sigma)_0^{-1} \mid \sigma : A_1 \}, \bigcup_{\sigma : A_1} (\simeq_{\psi_\sigma})_0^{-1}) : A \multimap B$ (recall *union games* in Section 5.3.2), which is essentially the same as ψ, thanks to the uniformity of ψ. Moreover, these constructions are mutually inverses again by the uniformity of ϕ. Therefore, in view of Theorem 5.3.19, we may conclude that $A \multimap B$ and $A_1 \multimap B_1$ are essentially the same. It is even simpler to see the correspondence for other constructions.

Finally, let us generalize the most basic strategies:

Definition 5.3.44 (Copy-cats on p-games)**.** The ***copy-cat (v-strategy)*** on a p-game G is the v-strategy $cp_G = \&_{\sigma : G} cp_\sigma : G \multimap G$.

Definition 5.3.45 (Derelictions on p-games)**.** The ***dereliction*** on a p-game G is the v-strategy $der_G = \&_{\otimes \tau : !G} der_{\tau_0} : !G \multimap G$, where $\otimes \tau = \otimes_{n \in \mathbb{N}} \tau_n$.

Diagrammatically, the dereliction der_A on any p-game A plays as follows:

$$\begin{tikzcd}
!A & \overset{der_A}{\multimap} & A \\
 & q_{!A \multimap A} & \\
 & \|der_A\| & \\
 & & {[a^{(1)}]^{\|\otimes\tau\|, \|der_A\|}_{r^{(1)}}} \\
{[(a^{(1)}, 0)]^{\|\otimes\tau\|, \|der_A\|}_{r^{(1)}}} \arrow[urr] & & \\
{[(a^{(2)}, 0)]^{\|\otimes\tau\|, \|der_A\|}_{r^{(2)}}} \arrow[u, bend left] & & \\
 & & {[a^{(2)}]^{\|\otimes\tau\|, \|der_A\|}_{r^{(2)}}} \arrow[uuu, bend right] \\
 & & {[a^{(3)}]^{\|\otimes\tau\|, \|der_A\|}_{r^{(3)}}} \arrow[u, bend left] \\
{[(a^{(3)}, 0)]^{\|\otimes\tau\|, \|der_A\|}_{r^{(3)}}} \arrow[uuu, bend right] & & \\
 & \vdots &
\end{tikzcd}$$

where $\otimes\tau : !A$ and $[a^{(1)}]_{r^{(1)}}[a^{(2)}]_{r^{(2)}}[a^{(3)}]_{r^{(3)}} \cdots \in P_{\tau_0}$. Note that Player (resp. Opponent) can see the 'tag' $\|der_A\|$ (resp. $\|\otimes\tau\|$) but not $\|\otimes\tau\|$ (resp. $\|der_A\|$).

We are now ready to define:

Definition 5.3.46 (The category $\mathcal{LPG}$)**.** The category $\mathcal{LPG}$ of logical p-games and winning v-strategies is defined as follows:

- Objects are well-founded p-games G such that $\mathcal{VS}(G) \neq \emptyset$, which we call ***logical p-games (lp-games)***;
- Morphisms $A \to B$ are the equivalence classes $[\phi]$ of winning v-strategies (***wv-strategies***) $\phi : A \Rightarrow B$ with respect to the equivalence relation $\simeq_{A \Rightarrow B}$;
- The composition $[\psi] \bullet [\phi] : A \to C$ of morphisms $[\phi] : A \to B$ and $[\psi] : B \to C$ is defined by $[\psi] \bullet [\phi] \stackrel{\text{df.}}{=} [\psi \bullet \phi] \stackrel{\text{df.}}{=} [\psi \circ \phi^\dagger]$;
- The identity id_A on each object A is $[der_A] : A \to A$.

Theorem 5.3.47 (Well-defined $\mathcal{LPG}$)**.** *$\mathcal{LPG}$ forms a well-defined category.*

Proof. The composition is well-defined by Lemma 2.3.16 and Corollary 5.3.42, and the identities are clearly well-defined by Lemma 2.3.27. Finally, the associativity and the unit law follow from those of $\mathcal{LMG}$ (see, e.g., [129] for the proof). □

Remark. Importantly, Opponent's anti-strategies do not have to be innocent, well-bracketed, total or noetherian at all (we have not formulated these notions precisely, but it should be clear what they are). In particular, strategies $\otimes\sigma$ for any morphism $[\phi] = [\&_{\otimes\sigma:!A}\phi_\sigma] : A \to B$ in $\mathcal{LPG}$ range over *any* v-strategies on $!A$.

Thus, the category $\mathcal{LPG}$ is a natural generalization of the category $\mathcal{LMG}$ of well-founded games and winning strategies for IPC in Chapter 2. Note that there is at least one (not necessarily winning) v-strategy on each game, corresponding to the non-emptiness of objects of $\mathcal{LPG}$, but see Section 5.3.5. As in the case of $\mathcal{LMG}$, we may think of objects and morphisms of $\mathcal{LPG}$ as *formulas* and *proofs*, respectively.

Convention. We often call objects and morphisms in $\mathcal{LPG}$ ***formulas*** and ***proofs***, respectively. Let $\mathcal{LPG}(A) \stackrel{\text{df.}}{=} \mathcal{LPG}(T, A)$ for all $A \in \mathcal{LPG}$. A formula $G \in \mathcal{LPG}$ is ***true*** if $\exists[\phi] \in \mathcal{LPG}(G)$, where $[\phi]$ is called a ***proof*** of G, and ***false*** otherwise.

Mathematically, the category $\mathcal{LPG}$ is well-behaved. Its *linear version* (in the sense of $\mathcal{LLMG}$ for $\mathcal{LMG}$ in Chapter 2) gives rise to an NSC (Definition 2.4.2), and thus $\mathcal{LPG}$ induces a CCC (by Theorem 2.4.3), in a completely similar manner to the case of $\mathcal{LLMG}$ and $\mathcal{LMG}$. In fact, the structure of $\mathcal{LPG}$ may be obtained via this route.

Nevertheless, since this result is not directly relevant to the rest of the thesis, we just state it here without giving a proof:

Theorem 5.3.48 (The NSC $\mathcal{LLPG}$)**.** *There is the NSC $\mathcal{LLPG}$ of logical p-games and winning v-strategies with the comonad* $!$ *such that* $\mathcal{LLPG}_! = \mathcal{LPG}$.

Corollary 5.3.49 (The CCC $\mathcal{LPG}$)**.** *The category $\mathcal{LPG}$ is cartesian closed.*

Proof. By Theorems 5.3.48 and 2.4.3. □

5.3.5 Coproducts of Predicative Games

At this point, one may have recognized that the mathematical structure of p-games is somewhat similar to that of the *Fam*-construction (which takes a model $\mathcal{C}$ of *call-by-name* computation and generates a model $\mathit{Fam}(\mathcal{C})$ of *call-by-value* computation) applied to the CCC $\mathcal{CMG}$ of conventional (*call-by-name*) games by Abramsky et al. [13], which is *bicartesian closed*. Therefore, it is reasonable to expect that p-games also have coproducts. The present section briefly addresses this point though it is not strictly necessary for the rest of the chapter. In order to discuss the relation between coproducts and fixed-points [98], we define the CCC $\mathcal{CPG}$ of *computational* p-games and v-strategies obtained from $\mathcal{LPG}$ by relaxing the well-foundedness condition on objects and the winning condition on morphisms.

Remark. There are, however, notable differences between p-games and call-by-value games of [13]. Firstly, morphisms in $\mathcal{LPG}$ and $\mathcal{CPG}$ are not necessarily *strict* unlike those in the BCC $\mathit{Fam}(\mathcal{CMG})$; thus, $\mathcal{LPG}$ and $\mathcal{CPG}$ both model *call-by-name* computation, which is appropriate for embodying logic since it allows proofs that completely ignore assumptions. Secondly, $\mathit{Fam}(\mathcal{CMG})$ has *families* of games and *families* of strategies as objects and morphisms, respectively. Although a family $\{A_i \mid i \in I\}$ of games and a family $\{\phi_i : !A_i \multimap \sum_{j \in J} !B_j \mid i \in I\} : \{A_i \mid i \in I\} \to \sum\{B_j \mid j \in J\}$ of strategies may be regarded as a single (*pointed*) game $\sum\{A_i \mid i \in I\} = \sum_{i \in I} A_i$, where $\sum : \mathit{Fam}(\mathcal{CMG}) \to \mathit{Fam}(\mathcal{CMG})$ is a *strong monad* [136] on $\mathit{Fam}(\mathcal{CMG})$, and a single (*strict*) strategy $\phi : \sum_{i \in I} !A_i \multimap \sum_{j \in J} !B_j$, respectively (see [13] for the detail), the linear implication $\sum_{i \in I} !A_i \multimap \sum_{j \in J} !B_j$ no longer has the canonical form $\sum(\_)$. In this sense, one of the novelties of p-games lies in the point that the structure of *families of games* and *families of strategies* is incorporated into game-semantic concepts (or formulated solely in terms of games and strategies).

5.3.5.1 Initial Objects

Let us begin with *(strong) initial objects.* It is immediate from the definition of linear implication $\multimap$ between p-games (Definition 5.3.40) that the *initial p-game* I (Example 5.3.31) would have been an initial object of $\mathcal{LPG}$ if we allow that an object of $\mathcal{LPG}$ has no v-strategy at all. However, we have excluded I from $\mathcal{LMG}$ in order to keep the *intensional* or *play-relevant* nature of p-games:

- Recall that a position $q_G.\|\sigma\|.(\boldsymbol{s})_{\|\sigma\|}$ of a p-game G is divided into the *initial protocol* $q_G.\|\sigma\|$ and the *actual position* $(\boldsymbol{s})_{\|\sigma\|}$, where we regard only the latter as the essential part since then the chosen strategy $\sigma : G$ is 'gradually revealed' as a play proceeds (which conforms to conventional games);

- Let us call this nature that only actual positions matter in logical p-games *play-relevance*;

- However, I breaks *play-relevance* of logical p-games, e.g., T and I both have the same (and trivial) actual position $\boldsymbol{\epsilon}$, but it is a win for Player in T but a defeat in I due to their difference in the initial protocols;

Alternatively, we may regard initial protocols also as an essential part of plays so that T and I are reasonably distinguished. In this case, the notion of p-games departs from conventional games (in Chapter 2) since Player reveals her strategy in mind to Opponent *in one go.* From this standpoint, we define the CCCs $\mathcal{ILPG}$ and $\mathcal{ICPG}$ obtained from $\mathcal{LPG}$ and $\mathcal{CPG}$, respectively, by simply adding I.

Although plays of p-games in $\mathcal{ILPG}$ and $\mathcal{ICPG}$ appear extensional, these CCCs are actually highly *intensional*, e.g., they are *not* well-pointed, standing in sharp contrast to extensional categories such as the category *Sets* of sets and functions. On the other hand, they are closer to *Sets* than $\mathcal{LPG}$ or $\mathcal{CPG}$ since it has the initial object $I \in \mathcal{WPG}$ similarly to the empty set $\emptyset \in Set$, and they do not have fixed-points, e.g., there is no fixed-point of $id_I : I \to I$ in $\mathcal{ILPG}$ or even in $\mathcal{ICPG}$ (which saves them from becoming *trivial*, i.e., objects are all isomorphic to T, by the argument in [98]).

Nevertheless, from the view of *proofs-as-programs* [172], the *weak* initial object or the *empty p-game* $\mathbf{0}$ (Example 5.3.25) is much more preferable as falsity than I; for instance, the negation $\neg A \overset{\text{df.}}{=} A \Rightarrow \mathbf{0}$ enables us to systematically relate classical and intuitionistic reasonings as we have seen for $\mathcal{LMG}$ in Section 2.4.7, but the negation $A \Rightarrow I$ obviously cannot. Moreover, in the presence of I, $\mathbf{0}$ cannot model falsity very well; see the proof of Lemma 5.4.32.

To summarize, we have adopted the CCC $\mathcal{LPG}$, excluding the initial object I, as the underlying category for modeling MLTT in order to conform to the *intensional* nature of conventional games and model negation in the *play-relevant* manner.

5.3.5.2 Binary Coproducts

Next, let us consider *(strong) binary coproducts.* We focus on the CCCs $\mathcal{CPG}$ and $\mathcal{ICPG}$ in this section since the case for $\mathcal{LPG}$ and $\mathcal{ILPG}$ would be completely similar.

It is easy to see that neither $\mathcal{CPG}$ nor $\mathcal{ICPG}$ has binary coproducts (they only have *weak* ones), but if we remove the *uniformity* condition on PoPLIs (Definition 5.3.35), then they both have strong ones. Let us define the *revealed* CCCs $\mathcal{RCPG}$ and $\mathcal{RICPG}$ obtained from $\mathcal{CPG}$ and $\mathcal{ICPG}$, respectively, by relaxing uniformity on morphisms.

Theorem 5.3.50 (Binary coproducts in $\mathcal{RCPG}$ and $\mathcal{RICPG}$)**.** *The CCCs $\mathcal{RCPG}$ and $\mathcal{RICPG}$ both have binary coproducts.*

Proof. We focus on $\mathcal{RCPG}$ since the case for $\mathcal{RICPG}$ is just the same. The coproduct $A \oplus B$ of $A, B \in \mathcal{RCPG}$ is given by $A \oplus B \stackrel{\text{df.}}{=} \oint(\mathcal{VS}(A) + \mathcal{VS}(B), \simeq_A + \simeq_B)$, for which the injections and copairings are the obvious ones. Then, it is straightforward to see that this structure satisfies the axioms of binary coproducts (in particular, initial protocols prohibit the problem described in Section 2.4.5 from occurring). $\square$

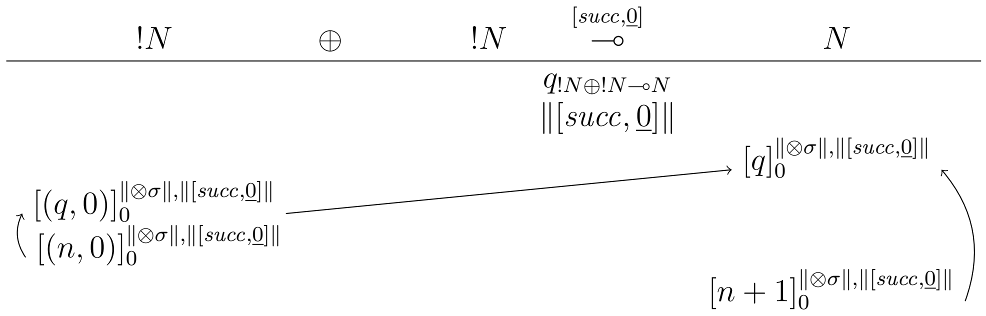
Example 5.3.51 (An example of coproduct of p-games)**.** Consider the coproduct $N \oplus \mathbf{1}$, whose typical positions are as depicted in the following tables:

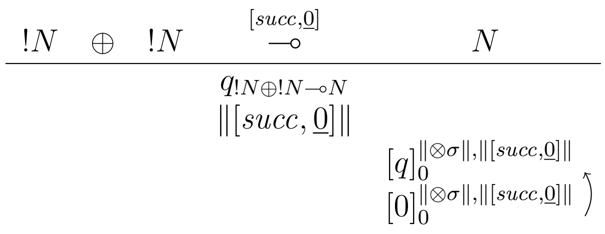
$$\begin{array}{c} N \oplus \mathbf{1} \\ \hline q_{N\oplus\mathbf{1}} \\ \|\underline{100}\| \\ q \\ 100 \end{array} \qquad \begin{array}{c} N \oplus \mathbf{1} \\ \hline q_{N\oplus\mathbf{1}} \\ \|\top\| \\ \\ \\ \end{array}$$

where for brevity we omit 'tags' for the disjoin union of the sets of moves.

The CCCs $\mathcal{RCPG}$ and $\mathcal{RICPG}$ are again *not* well-pointed, embodying *intensional* categories of computation. However, they significantly differ from the categories of conventional games for Player is allowed to see Opponent's choice of an anti-strategy:

Example 5.3.52 (An example of copairing on p-games)**.** Consider the copairing $[succ, \underline{0}] : !N \oplus !N \multimap N$:

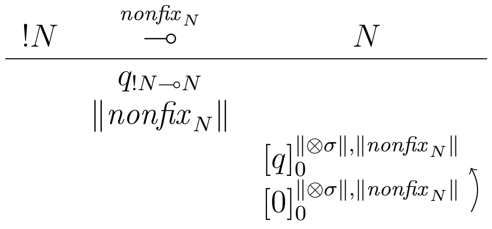




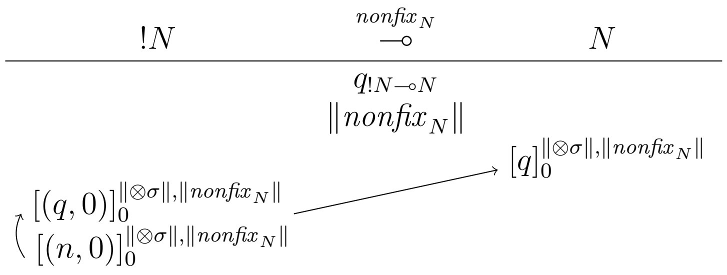
This phenomenon is unusual from the viewpoint of conventional game semantics since Opponent's choice of an anti-strategy may affect the behavior of a v-strategy.

One may wonder if $\mathcal{RCPG}$ has fixed-points (clearly, $\mathcal{RICPG}$ does not). The following example demonstrates that the answer is *no*:

Example 5.3.53 (An example of a v-strategy with no fixed-points)**.** We have the v-strategy $\mathit{nonfix}_N : !N \multimap N$ in $\mathcal{RCPG}$ that behaves as follows:



if $\sigma_0 = \bot : N$, and



otherwise (i.e., $\sigma_0 = \underline{n} : N$ for some $n \in \mathbb{N}$). Therefore, given a v-strategy $\phi : T \Rightarrow N$, we have $\mathit{nonfix}_N \circ \phi^\dagger = \begin{cases} \underline{0} & \text{if } \phi = \bot \\ \bot & \text{otherwise} \end{cases}$ so that $\mathit{nonfix}_N \bullet \phi \neq \phi$.

On the other hand, it is clear that the uniformity condition brings fixed-points into $\mathcal{RCPG}$ just similarly to the case of conventional categories of games and strategies such as $\mathcal{CMG}$. This point may be seen as a game-semantic counterpart of the general categorical phenomenon that binary coproducts and fixed-points are incompatible in a CCC unless it is trivial [98]: Conventional CCCs of games and strategies such as $\mathcal{CMG}$ have fixed-points but not binary coproducts for the (implicit) presence of the uniformity condition; they obtain binary coproducts but lose fixed-points if we relax uniformity. The generalization of games to p-games makes this point evident.

Corollary 5.3.54 (The BCCs $\mathcal{RICPG}$ and $\mathcal{RILPG}$). *The CCCs $\mathcal{RICPG}$ and $\mathcal{RILPG}$ both have (strong) coproducts.*

In this manner, we have established a *truly game-semantic coproducts* in the sense explained in Section 2.4.5. However, as already mentioned, these BCCs break:

- The idea of initial protocols in p-games (for Player cannot complete it in I);
- The interpretation of falsity by the empty p-game $\mathbf{0}$;
- The invisibility of Opponent's choice of an anti-strategy to Player.

For this reason, we take the CCCs $\mathcal{CPG}$ and $\mathcal{LPG}$ (in particular the latter) as central in this thesis though it is possible to extend them to BCCs $\mathcal{RICPG}$ and $\mathcal{RILPG}$.

Remark. It is of course possible to obtain BCCs from $\mathcal{ICPG}$ and $\mathcal{ILPG}$ by restricting objects to *pointed* ones and morphisms to *strict* ones, respectively, just similarly to the BCC of call-by-value games in [13]. Recall that a game is ***pointed*** if it is well-opened and has at most one initial move, and a strategy between pointed games is ***strict*** if it responds to the initial move in the codomain by the initial move of the domain whenever it is possible. Notably, the resulting BCCs may interpret fixed-points [136, 13]. However, to embody logic, we would like to have *non-strict* morphisms since they correspond to proofs that completely ignore assumptions; also, fixed-points would make every formula provable. Thus, we shall not consider these BCCs in the rest of the chapter either.

In the next section, which is the highlight of the present chapter, we give a game semantics of MLTT based on the category $\mathcal{LPG}$, implying that the generalization of $\mathcal{LMG}$ to $\mathcal{LPG}$ corresponds to the route from propositional logic to predicate logic.

5.4 Game Semantics of MLTT

We are now ready to present a game semantics of MLTT. Our approach is based on an abstract and algebraic model of MLTT, called *categories with families (CwFs)* [53, 92] because it is in general easier to show that a mathematical structure is an instance of an abstract model than to directly establish that it is a model of MLTT, and CwFs are closer to the syntax than other abstract or categorical semantics, so that we may directly see semantic counterparts of syntactic phenomena.

More specifically, the present section gives a CwF based on the category $\mathcal{LPG}$ (Definition 5.3.46) and equips it with *semantic type formers* [92] such as $\mathsf{1}$-, $\mathsf{0}$-, N-, Π-, Σ- and Id-types. We shall present our game-semantic universes in Section 5.5.

5.4.1 Dependent Logical Predicative Games

In MLTT, a *dependent type* over a type A is a judgement of the form $\mathsf{\Gamma, x : A, \Delta \vdash B\ type}$ [126, 125, 127]. As we have seen in Section 5.2, the syntactic construction (in fact a dependent type) El is a surjective but not injective map from terms $\mathsf{\Gamma, x : A, \Delta \vdash c : U}$ of universes to dependent types $\mathsf{\Gamma, x : A, \Delta \vdash El(c)\ type}$ with the right inverse En.

In this manner, dependent types and terms of universes are closely related. Then, how should we interpret them respectively? For simplicity, assume $\mathsf{\Gamma}$ and $\mathsf{\Delta}$ are both $\diamondsuit$. Intuitively, the former denotes merely a *family of types* $(\mathsf{B(x)})_{\mathsf{x:A}}$, while the latter represents the corresponding *computation*: $(\mathsf{x : A}) \mapsto \mathit{En}(\mathsf{B(x)})$. The point is that dependent types in MLTT are not any families of types but the ones inductively generated by rules of MLTT whose computational meanings are clear, which is why El is surjective (otherwise there would be dependent types with no their 'codes').

Now, note that a dependent type $\mathsf{x : A \vdash B\ type}$ is a *predicate* on the formula A in logic. Thus, it should be interpreted in a category $\mathcal{C}$ of formulas and proofs as a family of objects indexed by morphisms in $\mathcal{C}(T, A)$, where $T \in \mathcal{C}$ is a terminal object. In the case of $\mathcal{LPG}$, it is a family of p-games indexed by elements of $\mathcal{LPG}(A) = \mathcal{LPG}(T, A)$:

Definition 5.4.1 (Dlp-games)**.** A ***dependent logical predicative (dlp-) game*** over an lp-game $A \in \mathcal{LPG}$ is a family $B = \{B([\sigma]) \in \mathcal{LPG} \,|\, [\sigma] \in \mathcal{LPG}(A)\}$ of lp-games indexed by elements of $\mathcal{LPG}(A)$ with $\mathcal{R}(B) \overset{\text{df.}}{=} \mathsf{Sup}(\{\mathcal{R}(B([\sigma])) \,|\, [\sigma] \in \mathcal{LPG}(A)\}) \in \mathbb{N}$. It is ***constant*** if $B([\sigma]) = B([\sigma'])$ for all $[\sigma], [\sigma'] \in \mathcal{LPG}(A)$.

Notation. If B is constant, then we write $B = \{B_0\}_A$ or just $\{B_0\}$, where $B_0 = B([\sigma])$ for all $[\sigma] \in \mathcal{LPG}(A)$. We write $\mathcal{DLPG}(A)$ for the set of all dlp-games over an lp-game A. We often write $B[\sigma]$ for $B([\sigma])$. We define $\int B \overset{\text{df.}}{=} \int \{B[\sigma] \,|\, [\sigma] \in \mathcal{LPG}(A)\} \in \mathcal{LPG}$.

Example 5.4.2. Let us generalize the operation El: Given a wv-strategy $\phi : A \Rightarrow \mathcal{U}$, we define the dlp-game $El([\phi]) \in \mathcal{DLPG}(A)$ by:

$$El([\phi]) \stackrel{\mathrm{df.}}{=} \{ El([\phi \bullet \sigma]) \mid [\sigma] \in \mathcal{LPG}(A) \}.$$

It is a generalization as $El([\mu]) = \{ El([\mu \bullet \_]) \mid [\_] \in \mathcal{LPG}(T) \} = \{El([\mu])\}$ for any $\mu : T \Rightarrow \mathcal{U}$ in $\mathcal{LPG}$.

Example 5.4.3. Let $\mathcal{L}(N)$ be the dlp-game over N such that $\mathcal{L}(N)[\underline{k}]$ with $k \geqslant 1$ is the lp-game whose maximal positions are of the form $q_1 n_1 q_2 n_2 \dots q_k n_k$, where $n_1, n_2, \dots, n_k \in \mathbb{N}$, representing the k-tuple $(n_1, n_2, \dots, n_k) \in \mathbb{N}^k$, and $\mathcal{L}(N)[\underline{0}]$ is the lp-game whose only the position is $\boldsymbol{\epsilon}$, representing the empty tuple $\boldsymbol{\epsilon} \in \mathbb{N}^0$.

Example 5.4.4. The dlp-game $\mathcal{ENDO}$ over $\mathcal{U}$ is given by $\mathcal{ENDO}([\mu]) \stackrel{\mathrm{df.}}{=} El([\mu]) \Rightarrow El([\mu])$ for all $\mu : T \Rightarrow \mathcal{U}$ in $\mathcal{LPG}$. Similarly, we have $\underline{\otimes}, \underline{\multimap}, \underline{\&} \in \mathcal{DLPG}(\mathcal{U} \& \mathcal{U})$ and $\underline{!} \in \mathcal{DLPG}(\mathcal{U})$ that correspond to $\otimes$, $\multimap$, $\&$ and $!$, respectively.

5.4.2 Dependent Function Spaces

We now present our game-semantic interpretation of $\mathsf{\Pi}$-*types*. The construction here is, however, preliminary; the full interpretation of $\mathsf{\Pi}$-types is given in Section 5.4.6.1.

Definition 5.4.5 ($\widehat{\Pi}$-spaces)**.** The ***dependent function ($\widehat{\mathbf{\Pi}}$-) space*** $\widehat{\Pi}(A, B)$ of a dlp-game $B \in \mathcal{DLPG}(A)$ over an lp-game $A \in \mathcal{LPG}$ is the lp-subgame of $A \Rightarrow \int B$ whose v-strategies ϕ satisfy $\forall [\sigma] \in \mathcal{LPG}(A). [\phi \bullet \sigma] \in \mathcal{LPG}(B[\sigma])$.

The idea is best described by a set-theoretic analogy: $\widehat{\Pi}(A, B)$ represents the set of all (set-theoretic) functions $f : A \to \bigcup_{x \in A} B(x)$, where $B = (B(x))_{x \in A}$ is a family of sets indexed by elements of the set A, that satisfies $f(a) \in B(a)$ for all $a \in A$. Thus, when B is constant, $\widehat{\Pi}(A, \{B_0\})$ coincides with the implication $A \Rightarrow B_0$.

However, we will have to handle the case where A is a dlp-game; so $\widehat{\Pi}$ given above is not general enough (note that an lp-game A may be identified with the singleton dlp-game $\{A\}$). In terms of the syntax of MLTT, we can interpret the rule

$$(\Pi\text{-}\textsc{Form})\ \mathsf{\Gamma}, \mathsf{x : A} \vdash \mathsf{B(x)\ type} \Rightarrow \mathsf{\Gamma} \vdash \mathsf{\Pi_{x:A} B(x)\ type}$$

only when $\mathsf{\Gamma} = \diamondsuit$ at the moment. This is why we use the symbol $\widehat{\Pi}$ here; we shall define a more general construction Π of dependent function space shortly.

Example 5.4.6. On the $\widehat{\Pi}$-space $\widehat{\Pi}(N, \mathcal{L}(N))$, we define a wv-strategy $\phi : \widehat{\Pi}(N, \mathcal{L}(N))$ by $P_{\phi_\sigma} \stackrel{\mathrm{df.}}{=} \mathsf{Pref}(\{q_1 q k n_1 q_2 n_2 \dots q_k n_k\})$ if $\sigma_0 = \underline{k}$ with $k \geqslant 1$ and $P_{\phi_\sigma} \stackrel{\mathrm{df.}}{=} \{\boldsymbol{\epsilon}\}$ if $\sigma_0 = \underline{0}$, which represents the dependent function $(k \in \mathbb{N}) \mapsto (n_1, n_2, \dots, n_k) \in \mathbb{N}^k$.

5.4.3 Dependent Pair Spaces

In a similar manner to $\widehat{\Pi}$-space, we define an interpretation of Σ-*types*. Again, the construction here is preliminary; see Section 5.4.6.2 for the full interpretation.

Definition 5.4.7 ($\widehat{\Sigma}$-spaces)**.** The ***dependent pair ($\widehat{\Sigma}$-) space*** $\widehat{\Sigma}(A, B)$ of a dlp-game $B \in \mathcal{DLPG}(A)$ over an lp-game $A \in \mathcal{LPG}$ is the lp-subgame of the product $A \& (\int B)$ whose v-strategies $\langle \sigma, \tau \rangle$ satisfy $[\sigma^T] \in \mathcal{LPG}(A) \Rightarrow [\tau^T] \in \mathcal{LPG}(B[\sigma])$.

In terms of the set-theoretic analogy, $\widehat{\Sigma}(A, B)$ represents the set of all pairs (a, b), where $a \in A$, $b \in B(a)$. Again, this construction $\widehat{\Sigma}$ is not general enough; we shall define a more general one Σ shortly. Note that when B is a constant dlp-game $\{B_0\}$, the $\widehat{\Sigma}$-space $\widehat{\Sigma}(A, \{B_0\})$ coincides with the product $A \& B_0$.

Example 5.4.8. The $\widehat{\Sigma}$-space $\widehat{\Sigma}(N, \mathcal{L}(N))$ represents the 'space' of *dependent pairs* $(k, (n_1, n_2, \ldots, n_k))$, where $k, n_1, n_2, \ldots, n_k \in \mathbb{N}$.

5.4.4 Identity Spaces

We proceed to define lp-games that interpret Id-*types*:

Definition 5.4.9 ($\widehat{\mathsf{Id}}$- spaces)**.** Given $G \in \mathcal{LPG}$, the ***identity ($\widehat{\mathsf{Id}}$-) space*** $\widehat{\mathsf{Id}}_G([\sigma], [\tau])$ between $[\sigma], [\tau] \in \mathcal{LPG}(G)$ is defined by:

$$\widehat{\mathsf{Id}}_G([\sigma], [\tau]) \stackrel{\text{df.}}{=} \begin{cases} \mathbf{1} & \text{if } [\sigma] = [\tau]; \\ \mathbf{0} & \text{otherwise.} \end{cases}$$

Therefore, $\widehat{\mathsf{Id}}_G([\sigma], [\tau])$ is true iff $[\sigma] = [\tau]$. Again, the construction $\widehat{\mathsf{Id}}$ is not general enough; we shall define its generalization Id in Section 5.4.6.3.

Remark. It is certainly possible to define a more *intensional* version of $\widehat{\mathsf{Id}}$-spaces that compare the behavior of two given proofs of the same formula. However, such an intensional formulation cannot fully model Id-types as we shall see in Section 5.4.6.3.

5.4.5 Game-Semantic Category with Families

We now define the CwF $\mathcal{LPG}$ of lp-games and wv-strategies. Let us first recall the general definition of CwFs, where our presentation is based on [92]:

Definition 5.4.10 (CwFs [53, 92])**.** A ***category with families (CwF)*** is a tuple $\mathcal{C} = (\mathcal{C}, Ty, Tm, \_\{\_\}, T, \_.\_, p, v, \langle \_, \_ \rangle_\_)$, where:

- $\mathcal{C}$ is a category;

- Ty assigns, to each object $\Gamma \in \mathcal{C}$, a set $Ty(\Gamma)$, called the set of all ***types*** in the ***context*** Γ;
- Tm assigns, to each pair of an object $\Gamma \in \mathcal{C}$ and a type $A \in Ty(\Gamma)$, a set $Tm(\Gamma, A)$, called the set of all ***terms*** of type A in the context Γ;
- For each morphism $f : \Delta \to \Gamma$ in $\mathcal{C}$, $\_\{\_\}$ induces a function $\_\{f\} : Ty(\Gamma) \to Ty(\Delta)$, called the ***substitution on types***, and a family $(\_\{f\}_A : Tm(\Gamma, A) \to Tm(\Delta, A\{f\}))_{A \in Ty(\Gamma)}$ of functions, called the ***substitutions on terms***;
- $T \in \mathcal{C}$ is a terminal object;
- $\_.\_$ assigns, to each pair of a context $\Gamma \in \mathcal{C}$ and a type $A \in Ty(\Gamma)$, a context $\Gamma.A \in \mathcal{C}$, called the ***comprehension*** of A;
- p associates each pair of a context $\Gamma \in \mathcal{C}$ and a type $A \in Ty(\Gamma)$ with a morphism $p(A) : \Gamma.A \to \Gamma$ in $\mathcal{C}$, called the ***first projection*** associated to A;
- v associates each pair of a context $\Gamma \in \mathcal{C}$ and a type $A \in Ty(\Gamma)$ with a term $v_A \in Tm(\Gamma.A, A\{p(A)\})$ called the ***second projection*** associated to A;
- $\langle \_, \_ \rangle_\_$ assigns, to each triple of a morphism $f : \Delta \to \Gamma$ in $\mathcal{C}$, a type $A \in Ty(\Gamma)$ and a term $g \in Tm(\Delta, A\{f\})$, a morphism $\langle f, g \rangle_A : \Delta \to \Gamma.A$ in $\mathcal{C}$, called the ***extension*** of f by g

that satisfies, for all $\Gamma, \Delta, \Theta \in \mathcal{C}$, $A \in Ty(\Gamma)$, $f : \Delta \to \Gamma$, $e : \Theta \to \Delta$, $h \in Tm(\Gamma, A)$ and $g \in Tm(\Delta, A\{f\})$, the following equations:

- (Ty-Id) $A\{id_\Gamma\} = A$;
- (Ty-Comp) $A\{f \circ e\} = A\{f\}\{e\}$;
- (Tm-Id) $h\{id_\Gamma\}_A = h$;
- (Tm-Comp) $h\{f \circ e\}_A = h\{f\}_A\{e\}_{A\{f\}}$;
- (Cons-L) $p(A) \circ \langle f, g \rangle_A = f$;
- (Cons-R) $v_A\{\langle f, g \rangle_A\} = g$;
- (Cons-Nat) $\langle f, g \rangle_A \circ e = \langle f \circ e, g\{e\}_{A\{f\}} \rangle_A$;
- (Cons-Id) $\langle p(A), v_A \rangle_A = id_{\Gamma.A}$.

It is straightforward to consider a substructure relation between CwFs:

Definition 5.4.11 (SubCwFs)**.** A CwF $\mathcal{C}' = (\mathcal{C}', \mathit{Ty}', \mathit{Tm}', \_\{\_\}', T', \_.'\_, p', v', \langle\_,\_\rangle'_{\_})$ is a ***subCwF*** of a CwF $\mathcal{C} = (\mathcal{C}, \mathit{Ty}, \mathit{Tm}, \_\{\_\}, T, \_.\_, p, v, \langle\_,\_\rangle_{\_})$ if:

- $\mathcal{C}'$ is a subcategory of $\mathcal{C}$;
- $\mathit{Ty}(\Gamma') \subseteq \mathit{Ty}(\Gamma')$ for each $\Gamma' \in \mathcal{C}'$;
- $\mathit{Tm}(\Gamma', A') \subseteq \mathit{Tm}(\Gamma', A')$ for each $\Gamma' \in \mathcal{C}'$ and $A' \in \mathit{Ty}(\Gamma')$;
- $A'\{f'\}' = A'\{f'\}$ and $a'\{f'\}' = a'\{f'\}$ for all $f' : \Delta' \to \Gamma'$ in $\mathcal{C}'$, $A' \in \mathit{Ty}'(\Gamma')$ and $a' \in \mathit{Tm}'(\Gamma', A')$;
- $T' = T$ and $\Gamma'.'A' = \Gamma'.A'$ for all $\Gamma' \in \mathcal{C}'$ and $A' \in \mathit{Ty}'(\Gamma')$;
- $p'(A') = p(A')$ and $v'_{A'} = v_{A'}$ for all $\Gamma' \in \mathcal{C}'$ and $A' \in \mathit{Ty}(\Gamma')$;
- $\langle f', g' \rangle_{A'} = \langle f', g' \rangle'_{A'}$ for all $f' : \Delta' \to \Gamma'$ in $\mathcal{C}'$, $A' \in \mathit{Ty}(\Gamma')$ and $g' \in \mathit{Tm}'(\Gamma', A')$.

Next, let us recall the interpretation of MLTT in a CwF. Roughly, judgements of MLTT as presented in Section 5.2 are interpreted in a CwF $\mathcal{C}$ as follows:

$$\vdash \mathsf{\Gamma\ ctx} \mapsto [\![\mathsf{\Gamma}]\!] \in \mathcal{C}; \tag{5.1}$$
$$\mathsf{\Gamma} \vdash \mathsf{A\ type_i} \mapsto [\![\mathsf{A}]\!] \in \mathit{Ty}([\![\mathsf{\Gamma}]\!]) \text{ such that } \mathcal{R}([\![\mathsf{A}]\!]) = i; \tag{5.2}$$
$$\mathsf{\Gamma} \vdash \mathsf{a : A} \mapsto [\![\mathsf{a}]\!] \in \mathit{Tm}([\![\mathsf{\Gamma}]\!], [\![\mathsf{A}]\!]); \tag{5.3}$$
$$\vdash \mathsf{\Gamma} \equiv \mathsf{\Delta\ ctx} \Rightarrow [\![\mathsf{\Gamma}]\!] = [\![\mathsf{\Delta}]\!] \in \mathcal{C}; \tag{5.4}$$
$$\mathsf{\Gamma} \vdash \mathsf{A} \equiv \mathsf{B\ type_i} \Rightarrow [\![\mathsf{A}]\!] = [\![\mathsf{B}]\!] \in \mathit{Ty}([\![\mathsf{\Gamma}]\!]); \tag{5.5}$$
$$\mathsf{\Gamma} \vdash \mathsf{a} \equiv \mathsf{a' : A} \Rightarrow [\![\mathsf{a}]\!] = [\![\mathsf{a'}]\!] \in \mathit{Tm}([\![\mathsf{\Gamma}]\!], [\![\mathsf{A}]\!]). \tag{5.6}$$

where $[\![\_]\!]$ denotes the interpretation. The last three equations are the *soundness* of $[\![\_]\!]$; see [92] for the proof. As we shall see, the additional equality of the rank of a type and the rank of its interpretation is easily established by induction on $\mathsf{\Gamma} \vdash \mathsf{A\ type_i}$.

Remark. An interpretation is applied to judgements, and thus $[\![\mathsf{\Gamma}]\!]$, $[\![\mathsf{A}]\!]$ and $[\![\mathsf{a}]\!]$ should be strictly speaking written respectively as $[\![\vdash \mathsf{\Gamma\ ctx}]\!]$, $[\![\mathsf{\Gamma} \vdash \mathsf{A\ type}]\!]$ and $[\![\mathsf{\Gamma} \vdash \mathsf{a : A}]\!]$. For brevity, however, we shall often adopt the shorter notation if it is not confusing.

Note that for a deduction of a judgement in MLTT is not unique in the presence of the rules Ty-Con and Tm-Con, a priori we cannot define an interpretation by induction on deductions. For this point, a standard approach is to define an interpretation $[\![\_]\!]$ on *pre-syntax* which is *partial*, and show that it is well-defined on every valid syntax (i.e., judgement) and preserves judgmental equality as the corresponding semantic equality [92]. By this soundness result, a posteriori we may describe the interpretation $[\![\_]\!]$ of the syntax by induction on derivation of judgements:

Definition 5.4.12 (Interpretation of MLTT in CwFs [92])**.** The interpretation $[\![\_]\!]$ of MLTT in a CwF $\mathcal{C} = (\mathcal{C}, Ty, Tm, \_\{\_\}, T, \_.\_, p, v, \langle\_,\_\rangle\_)$ is defined as follows:

- (Ct-Emp) $[\![\vdash \diamondsuit \ \mathsf{ctx}]\!] \overset{\text{df.}}{=} T$;
- (Ct-Ext) $[\![\vdash \Gamma, \mathsf{x} : \mathsf{A} \ \mathsf{ctx}]\!] \overset{\text{df.}}{=} [\![\vdash \Gamma \ \mathsf{ctx}]\!].[\![\Gamma \vdash \mathsf{A} \ \mathsf{type}]\!]$;
- (Var) $[\![\Gamma, \mathsf{x} : \mathsf{A} \vdash \mathsf{x} : \mathsf{A}]\!] \overset{\text{df.}}{=} v_{[\![\mathsf{A}]\!]}$;
  $[\![\Gamma, \mathsf{x} : \mathsf{A}, \Delta, \mathsf{y} : \mathsf{B} \vdash \mathsf{x} : \mathsf{A}]\!] \overset{\text{df.}}{=} [\![\Gamma, \mathsf{x} : \mathsf{A}, \Delta \vdash \mathsf{x} : \mathsf{A}]\!]\{p([\![\Gamma, \mathsf{x} : \mathsf{A}, \Delta \vdash \mathsf{B} \ \mathsf{type}]\!])\}_{[\![\mathsf{A}]\!]}$;
- (Ty-Con) $[\![\Delta \vdash \mathsf{A} \ \mathsf{type}]\!] \overset{\text{df.}}{=} [\![\Gamma \vdash \mathsf{A} \ \mathsf{type}]\!]$;
- (Tm-Con) $[\![\Delta \vdash \mathsf{a} : \mathsf{B}]\!] \overset{\text{df.}}{=} [\![\Gamma \vdash \mathsf{a} : \mathsf{A}]\!]$

where the hypotheses of the rules are as presented in Section 5.2.

We leave the interpretation of $\mathsf{1}$-, $\mathsf{0}$-, N-, $\mathsf{\Pi}$-, $\mathsf{\Sigma}$- and Id-types as well as universes by *semantic type formers* [92] to the next section.

We now define our game-semantic CwF:

Definition 5.4.13 (The CwF $\mathcal{LPG}$)**.** The CwF $\mathcal{LPG}$ of lp-games and wv-strategies is the tuple $\mathcal{LPG} = (\mathcal{LPG}, Ty, Tm, \_\{\_\}, I, \_.\_, p, v, \langle\_,\_\rangle\_)$, where:

- The underlying category $\mathcal{LPG}$ has been defined in Definition 5.3.46;
- For each $\Gamma \in \mathcal{LPG}$, $Ty(\Gamma) \overset{\text{df.}}{=} \mathcal{DLPG}(\Gamma)$;
- For each pair of $\Gamma \in \mathcal{LPG}$ and $A \in \mathcal{DLPG}(\Gamma)$, $Tm(\Gamma, A)$ is the set of all equivalence classes $[\phi]$ of wv-strategies $\phi : \widehat{\Pi}(\Gamma, A)$;
- For each morphism $[\phi] : \Delta \to \Gamma$ in $\mathcal{LPG}$, the function $\_\{[\phi]\} : Ty(\Gamma) \to Ty(\Delta)$ is defined by $A\{[\phi]\} \overset{\text{df.}}{=} \{A[\phi \bullet \delta] \mid [\delta] \in \mathcal{LPG}(\Delta)\}$ for all $A \in Ty(\Gamma)$, and the functions $\_\{[\phi]\}_A : Tm(\Gamma, A) \to Tm(\Delta, A\{[\phi]\})$ are defined by $[\varphi]\{[\phi]\}_A \overset{\text{df.}}{=} [\varphi] \bullet [\phi] = [\varphi \bullet \phi]$ for all $A \in Ty(\Gamma)$ and $[\varphi] \in Tm(\Gamma, A)$;
- T is the terminal p-game defined in Example 2.2.11;
- Given $\Gamma \in \mathcal{LPG}$ and $A \in \mathcal{DLPG}(\Gamma)$, $\Gamma.A \overset{\text{df.}}{=} \widehat{\Sigma}(\Gamma, A)$;
- $p(A) \overset{\text{df.}}{=} [\mathit{fst}_{\widehat{\Sigma}(\Gamma,A)}] : \widehat{\Sigma}(\Gamma, A) \to \Gamma$, where $\mathit{fst}_{\widehat{\Sigma}(\Gamma,A)}$ is der_Γ up to 'tags';

- $v_A \overset{\text{df.}}{=} [snd_{\widehat{\Sigma}(\Gamma,A)}] : \widehat{\Pi}(\widehat{\Sigma}(\Gamma,A), A\{p(A)\})$, where $snd_{\widehat{\Sigma}(\Gamma,A)}$ is $[der_{\int A}]$ up to 'tags';
- Given $\Gamma \in \mathcal{LPG}$, $A \in \mathcal{DLPG}(\Gamma)$, $[\phi] \in \mathcal{LPG}(\Delta,\Gamma)$ and $[\tau] \in \mathit{Tm}(\Delta, A\{[\phi]\})$, $\langle[\phi],[\tau]\rangle_A \overset{\text{df.}}{=} [\langle\phi,\tau\rangle] : \Delta \to \widehat{\Sigma}(\Gamma,A)$.

Notation. We often omit subscripts A on $\_\{[\phi]\}_A$ and $\langle\_,\_\rangle_A$. We often write *fst* and *snd* respectively for $fst_{\widehat{\Sigma}(\Gamma,A)}$ and $snd_{\widehat{\Sigma}(\Gamma,A)}$ if the underlying $\widehat{\Sigma}(\Gamma,A)$ is obvious. Our choice of the notation for the projections, $p(A)$ and v_A, or $fst_{\widehat{\Sigma}(\Gamma,A)}$ and $snd_{\widehat{\Sigma}(\Gamma,A)}$, depends on the context; we just employ what is easier to read.

Theorem 5.4.14 (Well-defined $\mathcal{LPG}$)**.** *The tuple $\mathcal{LPG}$ forms a well-defined CwF.*

Proof. It is obvious that each component is well-defined except substitutions on terms and extensions. Therefore, we consider just these two. Let $\Gamma \in \mathcal{LPG}$, $A \in \mathcal{DLPG}(\Gamma)$, $[\phi] \in \mathcal{LPG}(\Delta,\Gamma)$, $[\varphi] \in \mathit{Tm}(\Gamma,A)$ and $[\tau] \in \mathit{Tm}(\Delta, A\{[\phi]\})$ in $\mathcal{LPG}$.

For the substitution $[\varphi]\{[\phi]\} = [\varphi \bullet \phi]$, note first that it does not depend on the choice of the representatives φ and ϕ, and thus it is well-defined. Moreover, $\varphi \bullet \phi$ is a wv-strategy on $\widehat{\Pi}(\Delta, A\{[\phi]\})$ because if $[\delta] \in \mathcal{LPG}(\Delta)$, then $[\varphi \bullet \phi] \bullet [\delta] = [(\varphi \bullet \phi) \bullet \delta] = [\varphi \bullet (\phi \bullet \delta)] \in \mathcal{LPG}(A[\phi \bullet \delta]) = \mathcal{LPG}(A\{[\phi]\}([\delta]))$. Thus, $\_\{[\phi]\}$ is a well-defined function from $\mathit{Tm}(\Gamma,A)$ to $\mathit{Tm}(\Delta, A\{[\phi]\})$.

For the extension $\langle[\phi],[\tau]\rangle = [\langle\phi,\tau\rangle] : \Delta \to \Gamma \& (\int A)$, let $[\delta] \in \mathcal{LPG}(\Delta)$. Again, $\langle[\phi],[\tau]\rangle$ and $\langle[\phi],[\tau]\rangle \bullet [\delta] = [\langle\phi,\tau\rangle] \bullet [\delta] = [\langle\phi,\tau\rangle \bullet \delta]$ does not depend on the choice of the representatives ϕ, τ and δ. We have to show $[\langle\phi,\tau\rangle \bullet \delta] \in \mathcal{LPG}(\widehat{\Sigma}(\Gamma,A))$, but it clearly holds as $\langle\phi,\tau\rangle \bullet \delta = \langle\phi \bullet \delta, \tau \bullet \delta\rangle$ and $[\tau \bullet \delta] = [\tau]\{[\delta]\} \in \mathcal{LPG}(A\{[\phi]\}([\delta])) = \mathcal{LPG}(A[\phi \bullet \delta])$.

Finally, we verify the required equations:

- (Ty-Id) Given $\Gamma \in \mathcal{LPG}$ and $A \in \mathcal{DLPG}(\Gamma)$,

$$A\{id_\Gamma\} = \{ A[der_\Gamma \bullet \delta] \mid [\delta] \in \mathcal{LPG}(\Delta) \} = \{ A[\delta] \mid [\delta] \in \mathcal{LPG}(\Delta) \} = A;$$

- (Ty-Comp) Given $\Delta, \Theta \in \mathcal{LPG}$ and $[\psi] : \Theta \to \Delta$, $[\phi] : \Delta \to \Gamma$ in $\mathcal{LPG}$,

$$\begin{aligned} A\{[\phi] \bullet [\psi]\} &= A\{[\phi \bullet \psi]\} \\ &= \{ A[(\phi \bullet \psi) \bullet \theta] \mid [\theta] \in \mathcal{LPG}(\Theta) \} \\ &= \{ A[\phi \bullet (\psi \bullet \theta)] \mid [\theta] \in \mathcal{LPG}(\Theta) \} \\ &= \{ A\{[\phi]\}([\psi \bullet \theta]) \mid [\theta] \in \mathcal{LPG}(\Theta) \} \\ &= \{ A\{[\phi]\}\{[\psi]\}([\theta]) \mid [\theta] \in \mathcal{LPG}(\Theta) \} \\ &= A\{[\phi]\}\{[\psi]\}; \end{aligned}$$

- (Tm-Id) Given a wv-strategy $\varphi : \widehat{\Pi}(\Gamma, A)$,

$$[\varphi]\{id_\Gamma\} = [\varphi] \bullet [der_\Gamma] = [\varphi \bullet der_\Gamma] = [\varphi];$$

- (Tm-Comp) Under the same assumption,

$$\begin{aligned}[\varphi]\{[\phi] \bullet [\psi]\} &= [\varphi] \bullet [\phi \bullet \psi] \\ &= [\varphi \bullet (\phi \bullet \psi)] \\ &= [(\varphi \bullet \phi) \bullet \psi] \\ &= [\varphi \bullet \phi] \bullet [\psi] \\ &= [\varphi]\{[\phi]\} \bullet [\psi] \\ &= [\varphi]\{[\phi]\}\{[\psi]\};\end{aligned}$$

- (Cons-L) Given a wv-strategy $\tau : \widehat{\Pi}(\Delta, A\{\phi\})$,

$$p(A) \bullet \langle [\phi], [\tau] \rangle = [\mathit{fst}] \bullet [\langle \phi, \tau \rangle] = [\mathit{fst} \bullet \langle \phi, \tau \rangle] = [\phi];$$

- (Cons-R) $v_A\{\langle [\phi], [\tau] \rangle\} = [\mathit{snd}] \bullet [\langle \phi, \tau \rangle] = [\mathit{snd} \bullet \langle \phi, \tau \rangle] = [\tau];$
- (Cons-Nat) $\langle [\phi], [\tau] \rangle \bullet [\psi] = [\langle \phi, \tau \rangle] \bullet [\psi] = [\langle \phi, \tau \rangle \bullet \psi] = [\langle \phi \bullet \psi, \tau \bullet \psi \rangle] = \langle [\phi \bullet \psi], [\tau \bullet \psi] \rangle = \langle [\phi] \bullet [\psi], [\tau]\{[\psi]\} \rangle;$
- (Cons-Id) $\langle p(A), v_A \rangle = \langle [\mathit{fst}], [\mathit{snd}] \rangle = [\langle \mathit{fst}, \mathit{snd} \rangle] = [der_{\widehat{\Sigma}(\Gamma, A)}] = id_{\Gamma.A}$

which completes the proof. □

5.4.6 Game-Semantic Type Formers

Note that a CwF handles only the 'core' of MLTT: It interprets just the syntax common to all types. Thus, for a full interpretation of MLTT, we need to equip the CwF $\mathcal{LPG}$ with additional structures to interpret 1-, 0-, N-, Π-, Σ- and Id-types. This is the aim of the present section; we consider each type in order.

5.4.6.1 Game-Semantic Dependent Function Types

We begin with Π-types. Let us first recall the general interpretation of Π-types:

Definition 5.4.15 (CwFs with Π-types [92]). A CwF $\mathcal{C}$ ***supports* Π*-types*** if:

- (Π-Form) For any $\Gamma \in \mathcal{C}$, $A \in Ty(\Gamma)$ and $B \in Ty(\Gamma.A)$, there is a type

$$\Pi(A, B) \in Ty(\Gamma);$$

- (Π-Intro) If $b \in Tm(\Gamma.A, B)$, then there is a term

$$\lambda_{A,B}(b) \in Tm(\Gamma, \Pi(A, B));$$

- (Π-Elim) If $k \in Tm(\Gamma, \Pi(A, B))$, $g \in Tm(\Gamma, A)$, then there is a term

$$App_{A,B}(k, g) \in Tm(\Gamma, B\{\overline{g}\})$$

  where $\overline{g} \stackrel{\text{df.}}{=} \langle id_\Gamma, g\rangle_A : \Gamma \to \Gamma.A$;

- (Π-Comp) $App_{A,B}(\lambda_{A,B}(b), g) = b\{\overline{g}\}$;

- (Π-Subst) Given $\Delta \in \mathcal{C}$ and $f : \Delta \to \Gamma$ in $\mathcal{C}$,

$$\Pi(A, B)\{f\} = \Pi(A\{f\}, B\{f^+\})$$

  where $f^+ \stackrel{\text{df.}}{=} \langle f \circ p(A\{f\}), v_{A\{f\}}\rangle_A : \Delta.A\{f\} \to \Gamma.A$;

- (λ-Subst) $\lambda_{A,B}(b)\{f\} = \lambda_{A\{f\},B\{f^+\}}(b\{f^+\}) \in Tm(\Delta, \Pi(A\{f\}, B\{f^+\}))$ for all $b \in Tm(\Gamma.A, B)$;

- (App-Subst) $App_{A,B}(k, g)\{f\} = App_{A\{f\},B\{f^+\}}(k\{f\}, g\{f\}) \in Tm(\Delta, B\{\overline{g}\}\{f\})$, where note that $k\{f\} \in Tm(\Delta, \Pi(A\{f\}, B\{f^+\}))$, $g\{f\} \in Tm(\Delta, A\{f\})$ and $f^+ \circ \overline{g\{f\}} = \langle f \circ p(A\{f\}), v_{A\{f\}}\rangle_A \circ \langle id_\Delta, g\{f\}\rangle_{A\{f\}} = \langle f, g\{f\}\rangle_A = \langle id_\Gamma, g\rangle_A \circ f = \overline{g} \circ f$.

Furthermore, $\mathcal{C}$ ***supports* $\boldsymbol{\Pi}$*-types in the strict sense*** if it also satisfies:

- (λ-Uniq) Given $w : \widehat{\Pi}(\Gamma, \Pi(A, B))$,

$$\lambda_{A,B}(App_{A\{p(A)\},B\{p(A)^+\}}(w\{p(A)\}, v_A)) = w.$$

Definition 5.4.16 (Interpretation of Π-types)**.** The interpretation $[\![\_]\!]$ of Π-types in a CwF $\mathcal{C}$ that supports Π-types is given by:

- (Π-Form) $[\![\mathsf{\Gamma \vdash \Pi_{x:A}B\ type}]\!] \stackrel{\text{df.}}{=} \Pi([\![\mathsf{\Gamma \vdash A\ type}]\!], [\![\mathsf{\Gamma, x : A \vdash B\ type}]\!])$;

- (Π-Intro) $[\![\mathsf{\Gamma \vdash \lambda x.\, b : \Pi_{x:A}B}]\!] \stackrel{\text{df.}}{=} \lambda_{[\![\mathsf{A}]\!],[\![\mathsf{B}]\!]}([\![\mathsf{\Gamma, x : A \vdash b : B}]\!])$;

- (Π-Elim) $[\![\mathsf{\Gamma \vdash f(a) : B[a/x]}]\!] \stackrel{\text{df.}}{=} App_{[\![\mathsf{A}]\!],[\![\mathsf{B}]\!]}([\![\mathsf{\Gamma \vdash f : \Pi_{x:A}B}]\!], [\![\mathsf{\Gamma \vdash a : A}]\!])$.

where the hypotheses of the rules are as presented in Section 5.2.

Again, the soundness of the interpretation holds for this interpretation of Π-types, where the uniqueness rule Π-Uniq is also interpreted if the CwF $\mathcal{C}$ supports Π-types *in the strict sense*; see [92] for the detail. This point holds for the other semantic type formers given below, and thus we henceforth skip pointing it out.

We now propose our game-semantic Π-types:

Lemma 5.4.17 (Π-types in $\mathcal{LPG}$)**.** *The CwF $\mathcal{LPG}$ supports Π-types in the strict sense.*

Proof. Let $\Gamma \in \mathcal{LPG}$, $A \in \mathcal{DLPG}(\Gamma)$ and $B \in \mathcal{DLPG}(\widehat{\Sigma}(\Gamma, A))$, and assume that $\varphi : \widehat{\Pi}(\widehat{\Sigma}(\Gamma, A), B)$ is a wv-strategy.

- (Π-Form) As stated before, we need to generalize the construction of $\widehat{\Pi}$-spaces as A is a dlp-game: Let $\Pi(A, B) \stackrel{\text{df.}}{=} \{\widehat{\Pi}(A[\gamma], B_{[\gamma]}) \,|\, [\gamma] \in \mathcal{LPG}(\Gamma)\} \in \mathcal{DLPG}(\Gamma)$, where $B_{[\gamma]} \stackrel{\text{df.}}{=} \{B[\langle \gamma, \sigma \rangle] \,|\, [\sigma] \in \mathcal{LPG}(A[\gamma])\} \in \mathcal{DLPG}(A[\gamma])$. Note that if $\Gamma = T$, then $\Pi(A, B) = \{\widehat{\Pi}(A[\_], B_{[\_]})\}$; therefore, Π is a generalization of $\widehat{\Pi}$, and thus we call $\Pi(A, B)$ the ***dependent function (Π-) space*** of B over A.

- (Π-Intro) Thanks to the correspondence $\widehat{\Pi}(\widehat{\Sigma}(\Gamma, A), B) \cong \widehat{\Pi}(\Gamma, \Pi(A, B))$ up to 'tags', we may obtain $\lambda_{A,B}(\beta) : \widehat{\Pi}(\Gamma, \Pi(A, B))$ from any $\beta : \widehat{\Pi}(\widehat{\Sigma}(\Gamma, A), B)$ by 'adjusting the tags'. Clearly, this operation $\lambda_{A,B}$ preserves the identification of positions. Thus, given $[\beta] \in Tm(\widehat{\Sigma}(\Gamma, A), B)$, we define $\lambda_{A,B}([\beta]) \stackrel{\text{df.}}{=} [\lambda_{A,B}(\beta)] \in Tm(\Gamma, \Pi(A, B))$. Note that $\lambda_{A,B}$ has the obvious inverse $\lambda^{-1}_{A,B}$. We often omit the subscripts A and B on $\lambda_{A,B}$ and $\lambda^{-1}_{A,B}$.

- (Π-Elim) Given $\kappa : \widehat{\Pi}(\Gamma, \Pi(A, B))$ and $\alpha : \widehat{\Pi}(\Gamma, A)$ in $\mathcal{LPG}$, we define:

$$App_{A,B}(\kappa, \alpha) \stackrel{\text{df.}}{=} \lambda^{-1}_{A,B}(\kappa) \bullet \overline{\alpha}$$

  where $\overline{\alpha} = \langle der_\Gamma, \alpha \rangle : \Gamma \Rightarrow \widehat{\Sigma}(\Gamma, A)$. As in the proof of Theorem 5.4.14, we have $\lambda^{-1}_{A,B}(\kappa) \bullet [\overline{\alpha}] : \widehat{\Pi}(\Gamma, B\{[\overline{\alpha}]\})$. Then, we define:

$$App_{A,B}([\kappa], [\alpha]) \stackrel{\text{df.}}{=} [App_{A,B}(\kappa, \alpha)] = [\lambda^{-1}_{A,B}(\kappa) \bullet \overline{\alpha}] \in Tm(\Gamma, B\{[\overline{\alpha}]\}).$$

  It is easy to see that $App_{A,B}([\kappa], [\alpha])$ does not depend on the choice of the representatives κ and α. We often omit the subscripts A, B on $App_{A,B}$.

- (Π-Comp) By a simple calculation, we obtain:

$$\begin{aligned} App_{A,B}(\lambda_{A,B}([\beta]), [\alpha]) &= [\lambda^{-1}_{A,B}(\lambda_{A,B}([\beta]))] \bullet [\overline{\alpha}] \\ &= [\beta] \bullet [\overline{\alpha}] \\ &= [\beta]\{[\overline{\alpha}]\}. \end{aligned}$$

- (Π-Subst) Given $\Delta \in \mathcal{LPG}$ and $[\phi] : \Delta \to \Gamma$ in $\mathcal{LPG}$, we have:

$$\begin{aligned}\Pi(A,B)\{[\phi]\} &= \{\widehat{\Pi}(A[\gamma], B_{[\gamma]}) \mid [\gamma] \in \mathcal{LPG}(\Gamma)\}\{[\phi]\} \\ &= \{\widehat{\Pi}(A[\phi \bullet \delta], B_{[\phi \bullet \delta]}) \mid [\delta] \in \mathcal{LPG}(\Delta)\} \\ &= \{\widehat{\Pi}(A\{[\phi]\}([\delta]), B\{[\phi^+]\}_{[\delta]}) \mid [\delta] \in \mathcal{LPG}(\Delta)\} \\ &= \Pi(A\{[\phi]\}, B\{[\phi]^+\})\end{aligned}$$

where $[\phi]^+ \overset{\text{df.}}{=} \langle [\phi] \bullet p(A\{[\phi]\}), v_{A\{[\phi]\}} \rangle : \widehat{\Sigma}(\Delta, A\{[\phi]\}) \to \widehat{\Sigma}(\Gamma, A)$ and $B\{[\phi]^+\} \in \mathcal{DLPG}(\widehat{\Sigma}(\Delta, A\{[\phi]\}))$. The third equation holds because we have:

$$\begin{aligned}B\{[\phi]^+\}_{[\delta]}([\psi]) &= B\{[\phi]^+\}\langle [\delta], [\psi] \rangle \\ &= B(\langle [\phi] \bullet p(A\{[\phi]\}), v_{A\{[\phi]\}} \rangle \bullet [\langle \delta, \psi \rangle]) \\ &= B(\langle [\phi] \bullet [\mathit{fst}], [\mathit{snd}] \rangle \bullet \langle [\delta], [\psi] \rangle) \\ &= B(\langle [\phi \bullet \mathit{fst}], [\mathit{snd}] \rangle \bullet [\langle \delta, \psi \rangle]) \\ &= B([\langle \phi \bullet \mathit{fst}, \mathit{snd} \rangle] \bullet [\langle \delta, \psi \rangle]) \\ &= B[\langle \phi \bullet \mathit{fst}, \mathit{snd} \rangle \bullet \langle \delta, \psi \rangle] \\ &= B[\langle \phi \bullet \mathit{fst} \bullet \langle \delta, \psi \rangle, \mathit{snd} \bullet \langle \delta, \psi \rangle \rangle] \\ &= B[\langle \phi \bullet \delta, \psi \rangle] \\ &= B\langle [\phi \bullet \delta], [\psi] \rangle \\ &= B\langle [\phi] \bullet [\delta], [\psi] \rangle \\ &= B_{[\phi] \bullet [\delta]}([\psi])\end{aligned}$$

for all $[\psi] \in \mathcal{LPG}(A[\phi \bullet \gamma])$.

- (λ-Subst) Given $[\beta] \in \mathit{Tm}(\widehat{\Sigma}(\Gamma, A), B)$,

$$\begin{aligned}\lambda_{A,B}([\beta])\{[\phi]\} &= [\lambda_{A,B}(\beta)] \bullet [\phi] \\ &= [\lambda_{A,B}(\beta) \bullet \phi] \\ &= [\lambda_{A\{[\phi]\}, B\{[\phi]^+\}}(\beta \bullet \langle \phi \bullet \mathit{fst}_{\widehat{\Sigma}(\Delta, A\{[\phi]\})}, \mathit{snd}_{\widehat{\Sigma}(\Delta, A\{[\phi]\})} \rangle)] \\ &= \lambda_{A\{[\phi]\}, B\{[\phi]^+\}}([\beta] \bullet \langle [\phi] \bullet [\mathit{fst}_{\widehat{\Sigma}(\Delta, A\{[\phi]\})}], [\mathit{snd}_{\widehat{\Sigma}(\Delta, A\{[\phi]\})}] \rangle) \\ &= \lambda_{A\{[\phi]\}, B\{[\phi]^+\}}([\beta] \bullet \langle [\phi] \bullet p(A\{[\phi]\}), v_{A\{[\phi]\}} \rangle) \\ &= \lambda_{A\{[\phi]\}, B\{[\phi]^+\}}(\beta\{[\phi]^+\}).\end{aligned}$$

- (App-Subst) Moreover, it is easy to see that:

$$\begin{aligned}
App_{A,B}([\kappa],[\alpha])\{[\phi]\} &= ([\lambda^{-1}_{A,B}(\kappa)] \bullet [\langle der_\Gamma, \alpha\rangle]) \bullet [\phi] \\
&= [(\lambda^{-1}_{A,B}(\kappa) \bullet \langle der_\Gamma, \alpha\rangle) \bullet \phi] \\
&= [\lambda^{-1}_{A,B}(\kappa) \bullet (\langle der_\Gamma, \alpha\rangle \bullet \phi)] \\
&= [\lambda^{-1}_{A,B}(\kappa) \bullet \langle \phi, \alpha \bullet \phi\rangle] \\
&= [\lambda^{-1}_{A\{[\phi]\},B\{[\phi]^+\}}(\kappa \bullet \phi) \bullet \langle der_\Delta, \alpha \bullet \phi\rangle] \\
&= [\lambda^{-1}_{A\{[\phi]\},B\{[\phi]^+\}}(\kappa \bullet \phi) \bullet \overline{(\alpha \bullet \phi)}] \\
&= App_{A\{[\phi]\},B\{[\phi]^+\}}([\kappa \bullet \phi],[\alpha \bullet \phi]) \\
&= App_{A\{[\phi]\},B\{[\phi]^+\}}([\kappa]\{[\phi]\},[\alpha]\{[\phi]\}).
\end{aligned}$$

where $\overline{\alpha \bullet \phi} \overset{\text{df.}}{=} \langle der_\Delta, \alpha \bullet \phi\rangle : \Delta \Rightarrow \widehat{\Sigma}(\Delta, A\{[\phi]\})$.

- (λ-Uniq) Finally, if $[\omega] \in Tm(\Gamma, \Pi(A,B))$, then we have:

$$\begin{aligned}
&\lambda_{A,B}(App_{A\{p(A)\},B\{p(A)^+\}}([\omega]\{p(A)\}, v_A)) \\
&= \lambda_{A,B}(\lambda^{-1}_{A\{p(A)\},B\{p(A)^+\}}([\omega]\{[fst_{\widehat{\Sigma}(\Gamma,A)}]\}) \bullet \langle [der_{\widehat{\Sigma}(\Gamma,A)}], [snd_{\widehat{\Sigma}(\Gamma,A)}]\rangle) \\
&= \lambda_{A,B}([\lambda^{-1}_{A\{p(A)\},B\{p(A)^+\}}(\omega \bullet fst_{\widehat{\Sigma}(\Gamma,A)})] \bullet [\langle der_{\widehat{\Sigma}(\Gamma,A)}, snd_{\widehat{\Sigma}(\Gamma,A)}\rangle]) \\
&= \lambda_{A,B}([\lambda^{-1}_{A\{p(A)\},B\{p(A)^+\}}(\omega \bullet fst_{\widehat{\Sigma}(\Gamma,A)}) \bullet \langle der_{\widehat{\Sigma}(\Gamma,A)}, snd_{\widehat{\Sigma}(\Gamma,A)}\rangle]) \\
&= [\lambda_{A,B}(\lambda^{-1}_{A\{p(A)\},B\{p(A)^+\}}(\omega \bullet fst_{\widehat{\Sigma}(\Gamma,A)}) \bullet \langle der_{\widehat{\Sigma}(\Gamma,A)}, snd_{\widehat{\Sigma}(\Gamma,A)}\rangle))] \\
&= [\lambda_{A,B}(\lambda^{-1}_{A,B}(\omega))] \\
&= [\omega]
\end{aligned}$$

completing the proof. □

5.4.6.2 Game-Semantic Dependent Pair Types

Next, we consider **Σ**-types. Again, we begin with the general definition:

Definition 5.4.18 (CwFs with Σ-types [92]). A CwF $\mathcal{C}$ ***supports Σ-types*** if:

- (Σ-Form) Given $\Gamma \in \mathcal{C}$, $A \in Ty(\Gamma)$ and $B \in Ty(\Gamma.A)$, there is a type

$$\Sigma(A,B) \in Ty(\Gamma);$$

- (Σ-Intro) There is a morphism in $\mathcal{C}$

$$Pair_{A,B} : \Gamma.A.B \to \Gamma.\Sigma(A,B);$$

- (Σ-Elim) Given $P \in \mathit{Ty}(\Gamma.\Sigma(A,B))$ and $p \in \mathit{Tm}(\Gamma.A.B, P\{\mathit{Pair}_{A,B}\})$, there is a term
$$\mathcal{R}^{\Sigma}_{A,B,P}(p) \in \mathit{Tm}(\Gamma.\Sigma(A,B), P);$$

- (Σ-Comp) $\mathcal{R}^{\Sigma}_{A,B,P}(p)\{\mathit{Pair}_{A,B}\} = p$;

- (Σ-Subst) Given $\Delta \in \mathcal{C}$ and $f : \Delta \to \Gamma$ in $\mathcal{C}$,
$$\Sigma(A,B)\{f\} = \Sigma(A\{f\}, B\{f^+\})$$
where $f^+ \stackrel{\text{df.}}{=} \langle f \circ p(A\{f\}), v_{A\{f\}} \rangle_A : \Delta.A\{f\} \to \Gamma.A$;

- (Pair-Subst) $p(\Sigma(A,B)) \circ \mathit{Pair}_{A,B} = p(A) \circ p(B)$ and $f^* \circ \mathit{Pair}_{A\{f\},B\{f^+\}} = \mathit{Pair}_{A,B} \circ f^{++}$, where $f^* \stackrel{\text{df.}}{=} \langle f \circ p(\Sigma(A,B)\{f\}), v_{\Sigma(A,B)\{f\}} \rangle_{\Sigma(A,B)} : \Delta.\Sigma(A,B)\{f\} \to \Gamma.\Sigma(A,B)$ and $f^{++} \stackrel{\text{df.}}{=} \langle f^+ \circ p(B\{f^+\}), v_{B\{f^+\}} \rangle_B : \Delta.A\{f\}.B\{f^+\} \to \Gamma.A.B$;

- ($\mathcal{R}^{\Sigma}$-Subst) $\mathcal{R}^{\Sigma}_{A,B,P}(p)\{f^*\} = \mathcal{R}^{\Sigma}_{A\{f\},B\{f^+\},P\{f^*\}}(p\{f^{++}\})$.

Moreover, $\mathcal{C}$ ***supports* $\boldsymbol{\Sigma}$*-types in the strict sense*** if it additionally satisfies:

- ($\mathcal{R}^{\Sigma}$-Uniq) If any $p \in \mathit{Tm}(\Gamma.A.B, P\{\mathit{Pair}_{A,B}\})$ and $q \in \mathit{Tm}(\Gamma.\Sigma(A,B), P)$ satisfy the equation $q\{\mathit{Pair}_{A,B}\} = p$, then $q = \mathcal{R}^{\Sigma}_{A,B,P}(p)$.

Definition 5.4.19 (Interpretation of Σ-types)**.** The interpretation $[\![\_]\!]$ of Σ-types in a CwF $\mathcal{C}$ that supports Σ-types is given by:

- (Σ-Form) $[\![\mathsf{\Gamma \vdash \Sigma_{x:A}B\ type}]\!] \stackrel{\text{df.}}{=} \Sigma([\![\mathsf{\Gamma \vdash A\ type}]\!], [\![\mathsf{\Gamma, x : A \vdash B\ type}]\!])$

- (Σ-Intro) $[\![\mathsf{\Gamma \vdash (a,b) : \Sigma_{x:A}B}]\!] \stackrel{\text{df.}}{=} \mathit{Pair}_{[\![\mathsf{A}]\!],[\![\mathsf{B}]\!]} \circ \langle \overline{[\![\mathsf{\Gamma \vdash a : A}]\!]}, [\![\mathsf{\Gamma \vdash b : B[a/x]}]\!] \rangle_{[\![\mathsf{B}]\!]}$

- (Σ-Elim) $[\![\mathsf{\Gamma \vdash R^{\Sigma}(C, g, p) : C[p/z]}]\!] \stackrel{\text{df.}}{=} \mathcal{R}^{\Sigma}_{[\![\mathsf{A}]\!],[\![\mathsf{B}]\!],[\![\mathsf{C}]\!]}([\![\mathsf{\Gamma, x : A, y : B \vdash g : C[(x,y)/z]}]\!]) \circ \overline{[\![\mathsf{\Gamma \vdash p : \Sigma_{x:A}B}]\!]}$

where the hypotheses of the rules are as presented in Section 5.2, $\overline{[\![\mathsf{\Gamma \vdash a : A}]\!]} \stackrel{\text{df.}}{=} \langle \mathit{id}_{[\![\mathsf{\Gamma}]\!]}, [\![\mathsf{a}]\!] \rangle : [\![\mathsf{\Gamma}]\!] \to [\![\mathsf{\Gamma}]\!].[\![\mathsf{A}]\!]$ and $\overline{[\![\mathsf{\Gamma \vdash p : \Sigma_{x:A}B}]\!]} \stackrel{\text{df.}}{=} \langle \mathit{id}_{[\![\mathsf{\Gamma}]\!]}, [\![\mathsf{p}]\!] \rangle : [\![\mathsf{\Gamma}]\!] \to [\![\mathsf{\Gamma}]\!].[\![\mathsf{\Sigma_{x:A}B}]\!]$.

Now, we describe our game-semantic interpretation of Σ-types:

Lemma 5.4.20 (Σ-types in $\mathcal{LPG}$)**.** *The CwF $\mathcal{LPG}$ supports Σ-types in the strict sense.*

Proof. Let $\Gamma \in \mathcal{LPG}$, $A \in \mathcal{DLPG}(\Gamma)$ and $B \in \mathcal{DLPG}(\widehat{\Sigma}(\Gamma, A))$.

- (Σ-Form) Similarly to $\widehat{\Pi}$-spaces, we generalize $\widehat{\Sigma}$-spaces to Σ-spaces by $\Sigma(A, B) \stackrel{\mathrm{df.}}{=} \{ \widehat{\Sigma}(A[\gamma], B_{[\gamma]}) \mid [\gamma] \in \mathcal{LPG}(\Gamma) \} \in \mathcal{DLPG}(\Gamma)$. Again, Σ is a generalization of $\widehat{\Sigma}$ for $\Sigma(A, B) = \{ \widehat{\Sigma}(A[\_], B_{[\_]}) \}$ if $\Gamma = T$, and we call $\Sigma(A, B)$ the ***dependent pair (Σ-) space*** of B over A.

- (Σ-Intro) By the obvious correspondence $\widehat{\Sigma}(\widehat{\Sigma}(\Gamma, A), B) \cong \widehat{\Sigma}(\Gamma, \Sigma(A, B))$ up to 'tags', we define the morphism $\mathit{Pair}_{A,B} : \widehat{\Sigma}(\widehat{\Sigma}(\Gamma, A), B) \to \widehat{\Sigma}(\Gamma, \Sigma(A, B))$ to be the equivalence class of the obvious dereliction up to 'tags', i.e., $\mathit{Pair}_{A,B} \stackrel{\mathrm{df.}}{=} [\langle \mathit{fst}_{\widehat{\Sigma}(\Gamma,A)} \bullet \mathit{fst}_{\widehat{\Sigma}(\widehat{\Sigma}(\Gamma,A),B)}, \langle \mathit{snd}_{\widehat{\Sigma}(\Gamma,A)} \bullet \mathit{fst}_{\widehat{\Sigma}(\widehat{\Sigma}(\Gamma,A),B)}, \mathit{snd}_{\widehat{\Sigma}(\widehat{\Sigma}(\Gamma,A),B)} \rangle \rangle]$. Note that $\mathit{Pair}^{-1}_{A,B} = [\langle \langle \mathit{fst}_{\widehat{\Sigma}(\Gamma,\Sigma(A,B))}, \mathit{fst}_{\int \Sigma(A,B)} \bullet \mathit{snd}_{\widehat{\Sigma}(\Gamma,\Sigma(A,B))} \rangle, \mathit{snd}_{\int \Sigma(A,B)} \bullet \mathit{snd}_{\widehat{\Sigma}(\Gamma,\Sigma(A,B))} \rangle]$.

- (Σ-Elim) Given $P \in \mathcal{DLPG}(\widehat{\Sigma}(\Gamma, \Sigma(A, B)))$ and $[\psi] \in \mathit{Tm}(\widehat{\Sigma}(\widehat{\Sigma}(\Gamma, A), B), P\{\mathit{Pair}_{A,B}\})$ in $\mathcal{LPG}$, we may construct, by the correspondence described above, the term $\mathcal{R}^{\Sigma}_{A,B,P}([\psi]) \in \mathit{Tm}(\widehat{\Sigma}(\Gamma, \Sigma(A, B)), P)$ by $\mathcal{R}^{\Sigma}_{A,B,P}([\psi]) \stackrel{\mathrm{df.}}{=} [\psi] \bullet \mathit{Pair}^{-1}_{A,B}$.

- (Σ-Comp) $\mathcal{R}^{\Sigma}_{A,B,P}([\psi])\{\mathit{Pair}_{A,B}\} = \mathcal{R}^{\Sigma}_{A,B,P}([\psi]) \bullet \mathit{Pair}_{A,B} = ([\psi] \bullet \mathit{Pair}^{-1}_{A,B}) \bullet \mathit{Pair}_{A,B} = [\psi] \bullet (\mathit{Pair}^{-1}_{A,B} \bullet \mathit{Pair}_{A,B}) = [\psi]$.

- (Σ-Subst) Given $\Delta \in \mathcal{LPG}$ and $[\phi] : \Delta \to \Gamma$ in $\mathcal{LPG}$, by the same reasoning as the case of Π-space, $\Sigma(A, B)\{[\phi]\} = \Sigma(A\{[\phi]\}, B\{[\phi]^+\})$.

- (Pair-Subst) Under the same assumption, we clearly have:

$$\begin{aligned} p(\Sigma(A, B)) \bullet \mathit{Pair}_{A,B} &= [\mathit{fst} \bullet \langle \mathit{fst} \bullet \mathit{fst}, \langle \mathit{snd} \bullet \mathit{fst}, \mathit{snd} \rangle \rangle] \\ &= [\mathit{fst} \bullet \mathit{fst}] \\ &= [\mathit{fst}] \bullet [\mathit{fst}] \\ &= p(A) \bullet p(B) \end{aligned}$$

and we also have:

$$
\begin{aligned}
&[\phi]^* \bullet \mathit{Pair}_{A\{[\phi]\},B\{[\phi]^+\}} \\
&= \langle [\phi] \bullet p(\Sigma(A,B)\{[\phi]\}), v_{\Sigma(A,B)\{[\phi]\}} \rangle \bullet \mathit{Pair}_{A\{[\phi]\},B\{[\phi]^+\}} \\
&= \langle [\phi] \bullet p(\Sigma(A\{[\phi]\}, B\{[\phi]^+\})) \bullet \mathit{Pair}_{A\{[\phi]\},B\{[\phi]^+\}}, v_{\Sigma(A,B)\{[\phi]\}} \bullet \mathit{Pair}_{A\{[\phi]\},B\{[\phi]^+\}} \rangle \\
&= \langle [\phi] \bullet p(A\{[\phi]\}) \bullet p(B\{[\phi]^+\}), v_{\Sigma(A\{[\phi]\},B\{[\phi]^+\})} \bullet \mathit{Pair}_{A\{[\phi]\},B\{[\phi]^+\}} \rangle \\
&= \langle [\phi] \bullet [\mathit{fst}] \bullet [\mathit{fst}], [\mathit{snd}] \bullet \langle [\mathit{fst}] \bullet [\mathit{fst}], \langle [\mathit{snd}] \bullet [\mathit{fst}], [\mathit{snd}] \rangle \rangle \rangle \\
&= [\langle \phi \bullet \mathit{fst} \bullet \mathit{fst}, \mathit{snd} \bullet \langle \mathit{fst} \bullet \mathit{fst}, \langle \mathit{snd} \bullet \mathit{fst}, \mathit{snd} \rangle \rangle \rangle] \\
&= [\langle \phi \bullet \mathit{fst} \bullet \mathit{fst}, \langle \mathit{snd} \bullet \mathit{fst}, \mathit{snd} \rangle \rangle] \\
&= [\langle \mathit{fst} \bullet \mathit{fst}, \langle \mathit{snd} \bullet \mathit{fst}, \mathit{snd} \rangle \rangle \bullet \langle \langle \phi \bullet \mathit{fst} \bullet \mathit{fst}, \mathit{snd} \bullet \mathit{fst} \rangle, \mathit{snd} \rangle] \\
&= [\langle \mathit{fst} \bullet \mathit{fst}, \langle \mathit{snd} \bullet \mathit{fst}, \mathit{snd} \rangle \rangle \bullet \langle \langle \phi \bullet \mathit{fst}, \mathit{snd} \rangle \bullet \mathit{fst}, \mathit{snd} \rangle] \\
&= \langle [\mathit{fst}] \bullet [\mathit{fst}], \langle [\mathit{snd}] \bullet [\mathit{fst}], [\mathit{snd}] \rangle \rangle \bullet \langle \langle [\phi] \bullet [\mathit{fst}], [\mathit{snd}] \rangle \bullet [\mathit{fst}], [\mathit{snd}] \rangle \\
&= \mathit{Pair}_{A,B} \bullet \langle \langle [\phi] \bullet p(A\{[\phi]\}), v_{A\{[\phi]\}} \rangle \bullet p(B\{[\phi]^+\}), v_{B\{[\phi]^+\}} \rangle \\
&= \mathit{Pair}_{A,B} \bullet \langle [\phi]^+ \bullet p(B\{[\phi]^+\}), v_{B\{[\phi]^+\}} \rangle \\
&= \mathit{Pair}_{A,B} \bullet [\phi]^{++}
\end{aligned}
$$

where $[\phi]^* \overset{\mathrm{df.}}{=} \langle [\phi] \bullet p(\Sigma(A,B)\{[\phi]\}), v_{\Sigma(A,B)\{[\phi]\}} \rangle : \widehat{\Sigma}(\Delta, \widehat{\Sigma}(A,B)\{[\phi]\}) \to \widehat{\Sigma}(\Gamma, \widehat{\Sigma}(A,B))$ and $[\phi]^{++} \overset{\mathrm{df.}}{=} \langle [\phi]^+ \bullet p(B\{[\phi]^+\}), v_{B\{[\phi]^+\}} \rangle : \widehat{\Sigma}(\widehat{\Sigma}(\Delta, A\{[\phi]\}), B\{[\phi]^+\}) \to \widehat{\Sigma}(\widehat{\Sigma}(\Gamma, A), B)$.

- ($\mathcal{R}^\Sigma$-Subst) Clearly, we have:

$$
\begin{aligned}
&\mathcal{R}^\Sigma_{A,B,P}([\psi])\{[\phi]^*\} \\
&= [\psi] \bullet \mathit{Pair}^{-1}_{A,B} \bullet \langle [\phi] \bullet p(\Sigma(A,B)\{[\phi]\}, v_{\Sigma(A,B)\{[\phi]\}} \rangle \\
&= [\psi] \bullet [\langle \langle \mathit{fst}, \mathit{fst} \bullet \mathit{snd} \rangle, \mathit{snd} \bullet \mathit{snd} \rangle] \bullet \langle [\phi] \bullet [\mathit{fst}], [\mathit{snd}] \rangle \\
&= [\psi \bullet \langle \langle \mathit{fst}, \mathit{fst} \bullet \mathit{snd} \rangle, \mathit{snd} \bullet \mathit{snd} \rangle \bullet \langle \phi \bullet \mathit{fst}, \mathit{snd} \rangle] \\
&= [\psi \bullet \langle \langle \phi \bullet \mathit{fst}, \mathit{fst} \bullet \mathit{snd} \rangle, \mathit{snd} \bullet \mathit{snd} \rangle] \\
&= [\psi \bullet \langle \langle \phi \bullet \mathit{fst}, \mathit{snd} \rangle \bullet \mathit{fst}, \mathit{snd} \rangle \bullet \langle \langle \mathit{fst}, \mathit{fst} \bullet \mathit{snd} \rangle, \mathit{snd} \bullet \mathit{snd} \rangle] \\
&= [\psi] \bullet \langle \langle [\phi] \bullet [\mathit{fst}], [\mathit{snd}] \rangle \bullet [\mathit{fst}], [\mathit{snd}] \rangle \bullet \langle \langle [\mathit{fst}], [\mathit{fst}] \bullet [\mathit{snd}] \rangle, [\mathit{snd}] \bullet [\mathit{snd}] \rangle \\
&= [\psi] \bullet \langle [\phi]^+ \bullet [\mathit{fst}], [\mathit{snd}] \rangle \bullet \langle \langle [\mathit{fst}], [\mathit{fst}] \bullet [\mathit{snd}] \rangle, [\mathit{snd}] \bullet [\mathit{snd}] \rangle \\
&= [\psi] \bullet \langle [\phi]^+ \bullet p(B\{[\phi]^+\}), v_{B\{[\phi]^+\}} \rangle \bullet \mathit{Pair}^{-1}_{A\{[\phi]\},B\{[\phi]^+\}} \\
&= \mathcal{R}^\Sigma_{A\{[\phi]\},B\{[\phi]^+\},P\{[\phi]^*\}}([\psi] \bullet \langle [\phi]^+ \bullet p(B\{[\phi]^+\}), v_{B\{[\phi]^+\}} \rangle) \\
&= \mathcal{R}^\Sigma_{A\{[\phi]\},B\{[\phi]^+\},P\{[\phi]^*\}}(\psi\{[\phi]^{++}\}).
\end{aligned}
$$

- ($\mathcal{R}^\Sigma$-Uniq) If $[\varphi] \in \mathit{Tm}(\widehat{\Sigma}(\Gamma, \Sigma(A,B)), P)$ satisfies $[\varphi]\{\mathit{Pair}_{A,B}\} = [\psi]$, then $[\varphi] = [\psi] \bullet \mathit{Pair}^{-1}_{A,B} = \mathcal{R}^\Sigma_{A,B,P}([\psi])$

completing the proof. $\square$

5.4.6.3 Game-Semantic Identity Types

Next, we consider Id-types. Again, we first review the general interpretation:

Definition 5.4.21 (CwFs with Id-types [92]). A CwF $\mathcal{C}$ ***supports*** Id***-types*** if:

- (Id-Form) Given $\Gamma \in \mathcal{C}$ and $A \in \mathit{Ty}(\Gamma)$, there is a type

$$\mathsf{Id}_A \in \mathit{Ty}(\Gamma.A.A^+)$$

  where $A^+ \stackrel{\mathrm{df.}}{=} A\{p(A)\} \in \mathit{Ty}(\Gamma.A)$;

- (Id-Intro) There is a morphism in $\mathcal{C}$

$$\mathit{Refl}_A : \Gamma.A \to \Gamma.A.A^+.\mathsf{Id}_A;$$

- (Id-Elim) Given $B \in \mathit{Ty}(\Gamma.A.A^+.\mathsf{Id}_A)$ and $b \in \mathit{Tm}(\Gamma.A, B\{\mathit{Refl}_A\})$, there is a term

$$\mathcal{R}^{\mathsf{Id}}_{A,B}(b) \in \mathit{Tm}(\Gamma.A.A^+.\mathsf{Id}_A, B);$$

- (Id-Comp) $\mathcal{R}^{\mathsf{Id}}_{A,B}(b)\{\mathit{Refl}_A\} = b$;

- (Id-Subst) Given $\Delta \in \mathcal{C}$ and $f : \Delta \to \Gamma$ in $\mathcal{C}$,

$$\mathsf{Id}_A\{f^{++}\} = \mathsf{Id}_{A\{f\}} \in \mathit{Ty}(\Delta.A\{f\}.A\{f\}^+)$$

  where $A\{f\}^+ \stackrel{\mathrm{df.}}{=} A\{f\}\{p(A\{f\})\} \in \mathit{Ty}(\Delta.A\{f\})$, $f^+ \stackrel{\mathrm{df.}}{=} \langle f \circ p(A\{f\}), v_{A\{f\}}\rangle_A : \Delta.A\{f\} \to \Gamma.A$, and $f^{++} \stackrel{\mathrm{df.}}{=} \langle f^+ \circ p(A^+\{f^+\}), v_{A^+\{f^+\}}\rangle_{A^+} : \Delta.A\{f\}.A^+\{f^+\} \to \Gamma.A.A^+$;

- (Refl-Subst) $\mathit{Refl}_A \circ f^+ = f^{+++} \circ \mathit{Refl}_{A\{f\}} : \Delta.A\{f\} \to \Gamma.A.A^+.\mathsf{Id}_A$, where $f^{+++} \stackrel{\mathrm{df.}}{=} \langle f^{++} \circ p(\mathsf{Id}_A\{f^{++}\}), v_{\mathsf{Id}_A\{f^{++}\}}\rangle_{\mathsf{Id}_A} : \Delta.A\{f\}.A^+\{f^+\}.\mathsf{Id}_{A\{f\}} \to \Gamma.A.A^+.\mathsf{Id}_A$;

- ($\mathcal{R}^{\mathsf{Id}}$-Subst) $\mathcal{R}^{\mathsf{Id}}_{A,B}(b)\{f^{+++}\} = \mathcal{R}^{\mathsf{Id}}_{A\{f\},B\{f^{+++}\}}(b\{f^+\})$.

Definition 5.4.22 (Interpretation of Id-types). The interpretation $[\![\_]\!]$ of $=$-types in a CwF $\mathcal{C}$ that supports Id-types is given by:

- ($=$-Form) $[\![\Gamma \vdash \mathsf{a} =_{\mathsf{A}} \mathsf{a}' \ \mathsf{type}]\!] \stackrel{\mathrm{df.}}{=} \mathsf{Id}_{[\![\mathsf{A}]\!]}\{\langle \overline{[\![\Gamma \vdash \mathsf{a} : \mathsf{A}]\!]}, [\![\Gamma \vdash \mathsf{a}' : \mathsf{A}]\!]\rangle_{[\![\mathsf{A}]\!]}\}$;

- ($=$-Intro) $[\![\Gamma \vdash \mathsf{refl}_{\mathsf{A}} : \mathsf{a} =_{\mathsf{A}} \mathsf{a}]\!] \stackrel{\mathrm{df.}}{=} v_{\mathsf{Id}_{[\![\mathsf{A}]\!]}}\{\mathit{Refl}_{[\![\mathsf{A}]\!]} \circ \overline{[\![\Gamma \vdash \mathsf{a} : \mathsf{A}]\!]}\}$;

- (=-Elim) $[\![\mathsf{\Gamma \vdash R^{=}(C, c, a, a', q) : C[a/x, a'/y, q/p]}]\!] \overset{\mathrm{df.}}{=} \mathcal{R}^{\mathsf{Id}}_{[\![\mathsf{A,C}]\!]}([\![\mathsf{c}]\!])\{\langle\langle\overline{[\![\mathsf{a}]\!]}, [\![\mathsf{a'}]\!]\rangle_{[\![\mathsf{A}]\!]}, [\![\mathsf{q}]\!]\rangle_{[\![\mathsf{a =_A a'}]\!]}\}$

where the hypotheses of the rules are as presented in Section 5.2.

We then equip the CwF $\mathcal{LPG}$ with our game-semantic Id-types:

Lemma 5.4.23 (Id-types in $\mathcal{LPG}$)**.** *The CwF $\mathcal{LPG}$ supports Id-types.*

Proof. Let $\Gamma \in \mathcal{LPG}$ and $A \in \mathcal{DLPG}(\Gamma)$.

- (Id-Form) We define the dlp-game $\mathsf{Id}_A \in \mathcal{DLPG}(\widehat{\Sigma}(\widehat{\Sigma}(\Gamma, A), A^+))$ by:

$$\mathsf{Id}_A \overset{\mathrm{df.}}{=} \{\widehat{\mathsf{Id}}_{A[\gamma \bullet \_]}([\sigma], [\sigma']) \mid \langle\langle[\gamma], [\sigma]\rangle, [\sigma']\rangle \in \mathcal{LPG}(\widehat{\Sigma}(\widehat{\Sigma}(\Gamma, A), A^+))\}.$$

  Let us also call Id_A the ***identity space*** on A.

- (Id-Intro) We define $\mathit{Refl}_A : \widehat{\Sigma}(\Gamma, A_1) \to \widehat{\Sigma}(\widehat{\Sigma}(\widehat{\Sigma}(\Gamma, A_2), A_3^+), \mathsf{Id}_A)$ to be the equivalence class of the wv-strategy that plays as the dereliction between $\widehat{\Sigma}(\Gamma, A_1)$ and $\widehat{\Sigma}(\Gamma, A_2)$, between $\int A_1$ and $\int A_3^+$, or as the unique wv-strategy on $\widehat{\Sigma}(\Gamma, A_1) \to \mathbf{1}$, where the subscripts 1, 2 and 3 are to distinguish the different copies of A. Clearly, there is the inverse $\mathit{Refl}_A^{-1} : \widehat{\Sigma}(\widehat{\Sigma}(\widehat{\Sigma}(\Gamma, A_2), A_3^+), \mathsf{Id}_A) \to \widehat{\Sigma}(\Gamma, A_1)$ which is the equivalence class of the dereliction between $\widehat{\Sigma}(\Gamma, A_2)$ and $\widehat{\Sigma}(\Gamma, A_1)$.

  Remark. Note that if we had allowed a non-canonical proof of Id_A, then Refl_A would have only the left inverse, being unable to interpret the rule Id-Elim below, which is why we have defined $\widehat{\mathsf{Id}}$-spaces as in Definition 5.4.9.

- (Id-Elim) Given $B \in \mathcal{DLPG}(\widehat{\Sigma}(\widehat{\Sigma}(\widehat{\Sigma}(\Gamma, A_2), A_3^+), \mathsf{Id}_A))$ and $[\beta] \in \mathit{Tm}(\widehat{\Sigma}(\Gamma, A), B\{\mathit{Refl}_A\})$ in $\mathcal{LPG}$, we define:

$$\mathcal{R}^{\mathsf{Id}}_{A,B}([\beta]) \overset{\mathrm{df.}}{=} [\beta] \bullet \mathit{Refl}_A^{-1} \in \mathit{Tm}(\widehat{\Sigma}(\widehat{\Sigma}(\widehat{\Sigma}(\Gamma, A_1), A_2^+), \mathsf{Id}_A), B).$$

- (Id-Comp) We then clearly have:

$$\mathcal{R}^{\mathsf{Id}}_{A,B}([\beta])\{\mathit{Refl}_A\} = \mathcal{R}^{\mathsf{Id}}_{A,B}([\beta]) \bullet \mathit{Refl}_A = [\beta] \bullet \mathit{Refl}_A^{-1} \bullet \mathit{Refl}_A = [\beta].$$

- (Id-Subst) Given $\Delta \in \mathcal{LPG}$ and $[\phi] : \Delta \to \Gamma$ in $\mathcal{LPG}$,

$$\begin{aligned}
&\mathsf{Id}_A\{[\phi]^{++}\} \\
&= \{\widehat{\mathsf{Id}}_{A[\gamma\bullet\_]}([\sigma],[\sigma']) \mid \langle\langle[\gamma],[\sigma]\rangle,[\sigma']\rangle \in \mathcal{LPG}(\widehat{\Sigma}(\widehat{\Sigma}(\Gamma,A),A^+))\,\}\{[\phi]^{++}\} \\
&= \{\mathsf{Id}_A(\langle\langle[\gamma],[\sigma]\rangle,[\sigma']\rangle) \mid \langle\langle[\gamma],[\sigma]\rangle,[\sigma']\rangle \in \mathcal{LPG}(\widehat{\Sigma}(\widehat{\Sigma}(\Gamma,A),A^+)\}\{[\phi]^{++}\} \\
&= \{\mathsf{Id}_A([\phi]^{++}\bullet\langle\langle[\delta],[\alpha]\rangle,[\alpha']\rangle) \mid \langle\langle\delta,\alpha\rangle,\alpha'\rangle \in \mathcal{LPG}(\widehat{\Sigma}(\widehat{\Sigma}(\Delta,A\{[\phi]\}),A\{[\phi]\}^+))\} \\
&= \{\mathsf{Id}_A(\langle[\phi\bullet\delta],[\alpha]\rangle,[\alpha']\rangle) \mid \langle\langle[\delta],[\alpha]\rangle,[\alpha']\rangle \in \mathcal{LPG}(\widehat{\Sigma}(\widehat{\Sigma}(\Delta,A\{[\phi]\}),A\{[\phi]\}^+))\} \\
&= \{\widehat{\mathsf{Id}}_{A[(\phi\bullet\delta)\bullet\_]}([\alpha],[\alpha']) \mid \langle\langle[\delta],[\alpha]\rangle,[\alpha']\rangle \in \mathcal{LPG}(\widehat{\Sigma}(\widehat{\Sigma}(\Delta,A\{[\phi]\}),A\{[\phi]\}^+))\} \\
&= \{\widehat{\mathsf{Id}}_{A\{[\phi]\}[\delta\bullet\_]}([\alpha],[\alpha']) \mid \langle\langle[\delta],[\alpha]\rangle,[\alpha']\rangle \in \mathcal{LPG}(\widehat{\Sigma}(\widehat{\Sigma}(\Delta,A\{[\phi]\}),A\{[\phi]\}^+))\} \\
&= \mathsf{Id}_{A\{[\phi]\}}
\end{aligned}$$

where $[\phi]^+ \stackrel{\mathrm{df.}}{=} \langle[\phi]\bullet p(A\{[\phi]\}), v_{A\{[\phi]\}}\rangle : \widehat{\Sigma}(\Delta,A\{[\phi]\}) \to \widehat{\Sigma}(\Gamma,A)$ and $[\phi]^{++} \stackrel{\mathrm{df.}}{=} \langle[\phi]^+\bullet p(A^+\{[\phi]^+\}), v_{A^+\{[\phi]^+\}}\rangle : \widehat{\Sigma}(\widehat{\Sigma}(\Delta,A\{[\phi]\}),A^+\{[\phi]^+\}) \to \widehat{\Sigma}(\widehat{\Sigma}(\Gamma,A),A^+)$. Note that for the forth equation we have:

$$\begin{aligned}
[\phi]^{++}\bullet\langle\langle[\delta],[\alpha]\rangle,[\alpha']\rangle &= \langle[\phi]^+\bullet p(A^+\{[\phi]^+\}), v_{A^+\{[\phi]^+\}}\rangle\bullet\langle\langle[\delta],[\alpha]\rangle,[\alpha']\rangle \\
&= \langle[\phi]^+\bullet p(A^+\{\phi^+\})\bullet\langle\langle[\delta],[\alpha]\rangle,[\alpha']\rangle, v_{A^+\{[\phi]^+\}}\bullet\langle\langle[\delta],[\alpha]\rangle,[\alpha']\rangle\rangle \\
&= \langle\langle[\phi\bullet\delta],[\alpha]\rangle,[\alpha']\rangle.
\end{aligned}$$

- (Refl-Subst) Also, the following equation holds:

$$\begin{aligned}
\mathit{Refl}_A\bullet[\phi]^+ &= \mathit{Refl}_A\bullet\langle[\phi]\bullet p(A\{[\phi]\}), v_{A\{[\phi]\}}\rangle \\
&= \langle\langle\langle[\phi]\bullet p(A\{[\phi]\}), v_{A\{[\phi]\}}\rangle\bullet p(A^+\{[\phi]^+\})\bullet p(\mathsf{Id}_A\{[\phi]^{++}\}), \\
&\quad v_{A^+\{[\phi]^+\}}\bullet p(\mathsf{Id}_A\{[\phi]^{++}\})\rangle, v_{\mathsf{Id}_A\{[\phi]^{++}\}}\rangle\bullet\mathit{Refl}_{A\{[\phi]\}} \\
&= \langle\langle[\phi]^+\bullet p(A^+\{[\phi]^+\}), v_{A^+\{[\phi]^+\}}\rangle\bullet p(\mathsf{Id}_A\{[\phi]^{++}\}), v_{\mathsf{Id}_A\{[\phi]^{++}\}}\rangle\bullet\mathit{Refl}_{A\{[\phi]\}} \\
&= \langle[\phi]^{++}\bullet p(\mathsf{Id}_A\{[\phi]^{++}\}), v_{\mathsf{Id}_A\{[\phi]^{++}\}}\rangle\bullet\mathit{Refl}_{A\{[\phi]\}} \\
&= [\phi]^{+++}\bullet\mathit{Refl}_{A\{[\phi]\}}
\end{aligned}$$

where $[\phi]^{+++} \stackrel{\mathrm{df.}}{=} \langle[\phi]^{++}\bullet p(\mathsf{Id}_A\{[\phi]^{++}\}), v_{\mathsf{Id}_A\{[\phi]^{++}\}}\rangle$.

- ($\mathcal{R}^{\mathsf{Id}}$-Subst) Finally, we have:

$$\begin{aligned}
\mathcal{R}^{\mathsf{Id}}_{A,B}([\beta])\{[\phi]^{+++}\} &= ([\beta]\bullet\mathit{Refl}_A^{-1})\bullet[\phi]^{+++} \\
&= [\beta]\bullet(\mathit{Refl}_A^{-1}\bullet[\phi]^{+++}) \\
&= [\beta]\bullet([\phi]^+\bullet\mathit{Refl}^{-1}_{A\{[\phi]\}}) \text{ (by Refl-Subst)} \\
&= ([\beta]\bullet[\phi]^+)\bullet\mathit{Refl}^{-1}_{A\{[\phi]\}} \\
&= \mathcal{R}^{\mathsf{Id}}_{A\{[\phi]\},B\{[\phi]^{+++}\}}([\beta]\bullet[\phi]^+) \\
&= \mathcal{R}^{\mathsf{Id}}_{A\{[\phi]\},B\{[\phi]^{+++}\}}([\beta]\{[\phi]^+\})
\end{aligned}$$

which completes the proof. $\square$

5.4.6.4 Game-Semantic Natural Number Type

We proceed to give our game-semantic natural number type. Again, we first present the abstract interpretation:

Definition 5.4.24 (CwFs with the N-type [92][12]). A CwF $\mathcal{C}$ ***supports the N-type*** or ***natural numbers*** if:

- (N-Form) Given $\Gamma \in \mathcal{C}$, there is a type

  $$N^\Gamma \in \mathit{Ty}(\Gamma)$$

  called the ***natural number type*** (in Γ), which we often abbreviate as N;

- (N-Intro) There are a term and a morphism in $\mathcal{C}$

  $$\underline{0}_\Gamma \in \mathit{Tm}(\Gamma, N)$$
  $$\mathit{succ}_\Gamma : \Gamma.N \to \Gamma.N$$

  that satisfy

  $$\underline{0}_\Gamma\{f\} = \underline{0}_\Delta \in \mathit{Tm}(\Delta, N)$$
  $$p(N) \circ \mathit{succ}_\Gamma = p(N) : \Gamma.N \to \Gamma$$
  $$\mathit{succ}_\Gamma \circ \langle g, v_N \rangle_N = \langle g, v_N\{\mathit{succ}_\Delta\}\rangle_N : \Delta.N \to \Gamma.N$$

  for any morphisms $f : \Delta \to \Gamma$ and $g : \Delta.N \to \Gamma$ in $\mathcal{C}$;

  Notation. With $\mathit{zero}_\Gamma \overset{\text{df.}}{=} \langle \mathit{id}_\Gamma, \underline{0}_\Gamma \rangle_N : \Gamma \to \Gamma.N$, we clearly have $\mathit{zero}_\Gamma \circ f = \langle f, \underline{0}_\Delta \rangle_N = \langle f, v_N\{\mathit{zero}_\Delta\}\rangle_N : \Delta \to \Gamma.N$. We often omit the subscript Γ on $\underline{0}_\Gamma$, zero_Γ and succ_Γ. We define for each $n \in \mathbb{N}$ the term $\underline{n}_\Gamma \in \mathit{Tm}(\Gamma, N)$ by:

  - $\underline{0}_\Gamma$ is already given;
  - $\underline{n+1}_\Gamma \overset{\text{df.}}{=} v_N\{\mathit{succ}_\Gamma \circ \langle \mathit{id}_\Gamma, \underline{n}_\Gamma \rangle\}$.

- (N-Elim) Given any triple of $P \in \mathit{Ty}(\Gamma.N)$, $c_z \in \mathit{Tm}(\Gamma, P\{\mathit{zero}\})$ and $c_s \in \mathit{Tm}(\Gamma.N.P, P\{\mathit{succ} \circ p(P)\})$, there is a term

  $$\mathcal{R}^N_P(c_z, c_s) \in \mathit{Tm}(\Gamma.N, P);$$

[12]In the book, the definition is actually left to the reader; accordingly, this definition is the author's solution, which may be shown to be *sound* in the same manner as in the case of Π-, Σ- and Id-types introduced above [92]. This applies for the remaining semantic type formers below as well.

- (N-Comp) We have the following equations:

$$\mathcal{R}^N_P(c_z, c_s)\{zero\} = c_z \in Tm(\Gamma, P\{zero\});$$
$$\mathcal{R}^N_P(c_z, c_s)\{succ\} = c_s\{\langle id_{\Gamma.N}, \mathcal{R}^N_P(c_z, c_s)\rangle_P\} \in Tm(\Gamma.N, P\{succ\});$$

- (N-Subst) $N^\Gamma\{f\} = N^\Delta \in Ty(\Delta)$;

- ($\mathcal{R}^N$-Subst) $\mathcal{R}^N_P(c_z, c_s)\{f^+\} = \mathcal{R}^N_{P\{f^+\}}(c_z\{f\}, c_s\{f^{++}\}) \in Tm(\Delta.N, P\{f^+\})$, where $f^+ \stackrel{\text{df.}}{=} \langle f \circ p(N), v_N\rangle_N : \Delta.N \to \Gamma.N$ and $f^{++} \stackrel{\text{df.}}{=} \langle f^+ \circ p(P\{f^+\}), v_{P\{f^+\}}\rangle_P : \Delta.N.P\{f^+\} \to \Gamma.N.P$.

Definition 5.4.25 (Interpretation of N-types)**.** The interpretation $[\![\_]\!]$ of N-types in a CwF $\mathcal{C}$ that supports the N-type is given by:

- (N-Form) $[\![\Gamma \vdash \mathsf{N\ type}]\!] \stackrel{\text{df.}}{=} N^{[\![\Gamma]\!]}$;

- (N-IntroZ) $[\![\Gamma \vdash \mathsf{zero} : \mathsf{N}]\!] \stackrel{\text{df.}}{=} \underline{0}_{[\![\Gamma]\!]}$;

- (N-IntroS) $[\![\Gamma \vdash \mathsf{succ(n)} : \mathsf{N}]\!] \stackrel{\text{df.}}{=} v_N\{succ_{[\![\Gamma]\!]} \circ \langle id_{[\![\Gamma]\!]}, [\![\Gamma \vdash \mathsf{n} : \mathsf{N}]\!]\rangle\}$;

- (N-Elim) $[\![\Gamma \vdash \mathsf{R^N(C, c_z, c_s, n)} : \mathsf{C[n/x]}]\!] \stackrel{\text{df.}}{=} \mathcal{R}^N_{[\![\mathsf{C}]\!]}([\![\mathsf{c_z}]\!], [\![\mathsf{c_s}]\!])\{\langle id_{[\![\Gamma]\!]}, [\![\mathsf{n}]\!]\rangle_N\}$

where the hypotheses of the rules are as presented in Section 5.2.

It is easy to see by mathematical induction that we have $[\![\Gamma \vdash \underline{\mathsf{n}} : \mathsf{N}]\!] = \underline{n}_{[\![\Gamma]\!]}$ for any context $\vdash \Gamma\ \mathsf{ctx}$ and natural number $n \in \mathbb{N}$.

We now propose our game-semantic natural number type:

Lemma 5.4.26 (Natural numbers in $\mathcal{LPG}$)**.** *The CwF $\mathcal{LPG}$ supports the N-type.*

Proof. Let $\Gamma, \Delta \in \mathcal{LPG}$ and $[\phi] : \Delta \to \Gamma$ in $\mathcal{LPG}$.

- (N-Form) Given $\Gamma \in \mathcal{LPG}$, we define N^Γ to be the constant dlp-game $\{N\}_\Gamma$.

- (N-Intro) We define $[\underline{0}_\Gamma] \in Tm(\Gamma, \{N\}_\Gamma)$ to be the morphism $[z_\Gamma]$ in $\mathcal{LPG}$, where $z_\Gamma = \&_{\otimes\gamma:!\Gamma} z_{\otimes\gamma}$ and $P_{z_{\otimes\gamma}} \stackrel{\text{df.}}{=} \mathsf{Pref}(\{q.0\})$. Similarly, we define $[succ_\Gamma] : \widehat{\Sigma}(\Gamma, \{N_{[1]}\}_\Gamma) \to \widehat{\Sigma}(\Gamma, \{N_{[2]}\}_\Gamma)$ to be the morphism $\langle p(\{N\}_\Gamma), [s_\Gamma]\rangle$ in $\mathcal{LPG}$, where $s_\Gamma \stackrel{\text{df.}}{=} \&_{(\otimes\gamma)\otimes(\otimes\sigma):!\Gamma\otimes!N_{[1]}} s_{(\otimes\gamma)\otimes(\otimes\sigma)} : \widehat{\Sigma}(\Gamma, \{N_{[1]}\}_\Gamma) \Rightarrow N_{[2]}$, $P_{s_{(\otimes\gamma)\otimes(\otimes\sigma)}} \stackrel{\text{df.}}{=} \mathsf{Pref}(\{q_{[2]}.q_{[1]}.n_{[1]}.(n+1)_{[2]} \mid n \in \mathbb{N}\})$ if $\sigma_0 = \underline{n}$ and $P_{s_{(\otimes\gamma)\otimes(\otimes\sigma)}} \stackrel{\text{df.}}{=} \mathsf{Pref}(\{q_{[2]}.q_{[1]}\})$ otherwise (i.e., $\sigma_0 = \bot$). Moreover, since $[\underline{0}_\Gamma] \bullet [\phi] = [\underline{0}_\Delta]$ and $[s_\Gamma] \bullet \langle [\psi], v_{\{N\}_\Delta}\rangle = [s_\Delta] = v_{\{N\}_\Delta}\{[succ_\Delta]\}$, where $[\psi] : \widehat{\Sigma}(\Delta, \{N\}_\Delta) \to \Gamma$ is any morphism in $\mathcal{LPG}$, the required equations clearly hold.

Remark. We write $[\underline{0}]$, $[\mathit{zero}_\Gamma]$ and $[\mathit{succ}_\Gamma]$, instead of $\underline{0}$, zero_Γ and succ_Γ, for consistency of the notation.

- (N-Elim) We apply the 'indirect' interpretation of fixed-point combinators as in [9, 14] for the fixed-point v-strategies in Chapter 4 are not noetherian, and thus they cannot be employed here. Given $P \in \mathcal{DLPG}(\widehat{\Sigma}(\Gamma, \{N\}))$, $[c_z] \in \mathit{Tm}(\Gamma, P\{[\mathit{zero}]\})$ and $[c_s] \in \mathit{Tm}(\widehat{\Sigma}(\widehat{\Sigma}(\Gamma, \{N\}), P), P\{[\mathit{succ}] \circ p(P)\})$ in $\mathcal{LPG}$, there are two terms

$$\widetilde{[c_z]} \in \mathit{Tm}(\widehat{\Sigma}(\widehat{\Pi}(\widehat{\Sigma}(\Gamma, \{N\}), P), \{\widehat{\Sigma}(\Gamma, \{N\})\}), P\{[\mathit{zero}] \bullet [\mathit{fst}] \bullet [\mathit{snd}]\});$$
$$\widetilde{[c_s]} \in \mathit{Tm}(\widehat{\Sigma}(\widehat{\Pi}(\widehat{\Sigma}(\Gamma, \{N\}), P), \{\widehat{\Sigma}(\Gamma, \{N\})\}), P\{[\mathit{succ}] \bullet [\mathit{pred}] \bullet [\mathit{snd}]\})$$

defined by:

$$\widetilde{[c_z]} : \widehat{\Pi}(\widehat{\Sigma}(\Gamma, \{N\}), P) \& \widehat{\Sigma}(\Gamma, \{N\}) \overset{[\mathit{snd}]}{\to} \widehat{\Sigma}(\Gamma, \{N\}) \overset{[\mathit{fst}]}{\to} \Gamma \overset{[c_z]}{\to} \textstyle\int P\{[\mathit{zero}]\};$$
$$\widetilde{[c_s]} : \widehat{\Pi}(\widehat{\Sigma}(\Gamma, \{N\}), P) \& \widehat{\Sigma}(\Gamma, \{N\}) \overset{\langle [\mathit{pred}] \bullet [\mathit{snd}], [\mathit{ev}]\{\langle [\mathit{fst}], [\mathit{pred}] \bullet [\mathit{snd}] \rangle\} \rangle}{\to} \widehat{\Sigma}(\Gamma, \{N\}) \& \textstyle\int P$$
$$\overset{[c_s]}{\to} \textstyle\int P\{[\mathit{succ}] \circ p(P)\}$$

where the term $[\mathit{ev}] \in \mathit{Tm}(\widehat{\Sigma}(\widehat{\Pi}(\widehat{\Sigma}(\Gamma, \{N\}), P), \{\widehat{\Sigma}(\Gamma, \{N\})\}), P\{[\mathit{snd}]\})$ (or written $[\mathit{ev}_P]$) is the *evaluation* over P [5] given by $[\mathit{ev}] \overset{\text{df.}}{=} \lambda^{-1}([\mathit{der}_{\widehat{\Pi}(\widehat{\Sigma}(\Gamma, N), P)}])$ (compare it with the evaluation in Definition **??**), and the morphism $[\mathit{pred}] : \widehat{\Sigma}(\Gamma, \{N\}) \to \widehat{\Sigma}(\Gamma, \{N\})$ is the *predecessor* defined in a similar manner to $[\mathit{succ}]$ such that $[\mathit{pred}] \bullet [\mathit{succ}] = [\mathit{der}_{\widehat{\Sigma}(\Gamma, \{N\})}]$ and $[\mathit{pred}] \bullet [\mathit{zero}] = [\mathit{zero}]$.

  Also, writing

$$P_z \overset{\text{df.}}{=} P\{[\mathit{zero}] \bullet p(N)\}) \in \mathit{Ty}(\widehat{\Sigma}(\Gamma, \{N\}));$$
$$P_s \overset{\text{df.}}{=} P\{[\mathit{succ}] \bullet [\mathit{pred}] \bullet p(P_z)\} \in \mathit{Ty}(\widehat{\Sigma}(\widehat{\Sigma}(\Gamma, \{N\}), P_z))$$

  we have the term

$$[\mathit{cond}] \in \mathit{Tm}(\widehat{\Sigma}(\widehat{\Sigma}(\widehat{\Sigma}(\Gamma, N), P_z), P_s), P\{p(P_z) \bullet p(P_s)\})$$

  (or written $[\mathit{cond}_\Gamma]$) that is the standard interpretation of *conditionals* in PCF [9, 100, 14, 129]: *cond* first asks an input natural number in the component N of the domain, and plays as the dereliction between P_z and $P\{p(P_z) \bullet p(P_s)\}$ if the answer is 0, and as the dereliction between P_s and $P\{p(P_z) \bullet p(P_s)\}$ otherwise.
  We then define $\mathcal{F}_P^N([c_z], [c_s]) : \widehat{\Pi}(\widehat{\Sigma}(\Gamma, \{N\}), P) \to \widehat{\Pi}(\widehat{\Sigma}(\Gamma, \{N\}), P)$ by:

$$\mathcal{F}_P^N([c_z], [c_s]) \overset{\text{df.}}{=} \lambda_{\{\widehat{\Sigma}(\Gamma, \{N\})\}, \{P\{[\mathit{snd}]\}\}}([\mathit{cond}]\{\langle \langle [\mathit{snd}], \widetilde{[c_z]} \rangle, \widetilde{[c_s]} \rangle\}).$$

Finally, we define the term $\mathcal{R}^N_P([c_z], [c_s]) \in Tm(\widehat{\Sigma}(\Gamma, \{N\}), P)$ to be the least upper bound of the following chain of terms $(\mathcal{R}^N_P([c_z], [c_s])_n \in Tm(\widehat{\Sigma}(\Gamma, \{N\}), P))_{n \in \mathbb{N}}$:

$$\mathcal{R}^N_P([c_z], [c_s])_0 \stackrel{\text{df.}}{=} [\_] \text{ up to 'tags'};$$
$$\mathcal{R}^N_P([c_z], [c_s])_{n+1} \stackrel{\text{df.}}{=} \mathcal{F}^N_P([c_z], [c_s]) \bullet \mathcal{R}^N_P([c_z], [c_s])_n.$$

- (N-Comp) By the definition, we clearly have:

  $\mathcal{R}^N_P([c_z], [c_s])\{[zero]\} = [c_z] \in Tm(\Gamma, P\{[zero]\})$;
  $\mathcal{R}^N_P([c_z], [c_s])\{[succ]\} = [c_s]\{\langle [der_{\widehat{\Sigma}(\Gamma,\{N\})}], \mathcal{R}^N_P([c_z], [c_s])\rangle\} \in Tm(\widehat{\Sigma}(\Gamma, \{N\}), P\{[succ]\})$.

- (N-Subst) By the definition, it is clear that $\{N\}_\Gamma\{[\phi]\} = \{N\}_\Delta$.

- ($\mathcal{R}^N$-Subst) Finally, by induction on $n \in \mathbb{N}$, we clearly have:

$$\begin{aligned}\mathcal{R}^N_P([c_z], [c_s])_n\{[\phi]^+\} &= \mathcal{R}^N_P([c_z], [c_s])_n \bullet \langle [\phi] \bullet p(\{N\}_\Delta), v_{\{N\}_\Delta}\rangle \\ &= \mathcal{R}^N_{P\{[\phi]^+\}}([c_z]\{[\phi]\}, [c_s]\{[\phi]^{++}\})_n\end{aligned}$$

  for all $n \in \mathbb{N}$, where $[\phi]^+ \stackrel{\text{df.}}{=} \langle [\phi] \bullet p(\{N\}_\Delta), v_{\{N\}_\Delta}\rangle : \widehat{\Sigma}(\Delta, \{N\}_\Delta) \to \widehat{\Sigma}(\Gamma, \{N\}_\Gamma)$ and $[\phi]^{++} \stackrel{\text{df.}}{=} \langle [\phi]^+ \bullet p(P\{[\phi]^+\}), v_{P\{[\phi]^+\}}\rangle : \widehat{\Sigma}(\widehat{\Sigma}(\Delta, \{N\}_\Delta), P\{[\phi]^+\}) \to \widehat{\Sigma}(\widehat{\Sigma}(\Gamma, \{N\}_\Gamma), P)$. Therefore, we may conclude that

$$\mathcal{R}^N_P([c_z], [c_s])\{[\phi]^+\} = \mathcal{R}^N_{P\{[\phi]^+\}}([c_z]\{[\phi]\}, [c_s]\{[\phi]^{++}\})$$

which completes the proof. □

5.4.6.5 Game-Semantic Unit Type

We further proceed to consider the *unit type* though it is rather simple. First, the categorical interpretation is as follows:

Definition 5.4.27 (CwFs with unit type [92]). A CwF $\mathcal{C}$ ***supports unit type*** if:

- (Unit-Form) Given $\Gamma \in \mathcal{C}$, there is a type

$$\mathbf{1}^\Gamma \in Ty(\Gamma)$$

  called the ***unity type*** (in Γ);

- (Unit-Intro) Given $\Gamma \in \mathcal{C}$, there is a term

$$\top_\Gamma \in Tm(\Gamma, \mathbf{1}^\Gamma);$$

- (Unit-Elim) Given $\Gamma \in \mathcal{C}$, $A \in Ty(\Gamma.\mathbf{1}^{\Gamma})$, $a \in Tm(\Gamma, A\{\overline{\top_{\Gamma}}\})$ and $t \in Tm(\Gamma, \mathbf{1}^{\Gamma})$, there is a term
$$\mathcal{R}^{\mathbf{1}}_{A}(a, t) \in Tm(\Gamma, A\{\bar{t}\})$$
where $\overline{\top} \overset{\text{df.}}{=} \langle id_{\Gamma}, \top_{\Gamma}\rangle_{\mathbf{1}^{\Gamma}} : \Gamma \to \Gamma.\mathbf{1}^{\Gamma}$ and $\bar{t} \overset{\text{df.}}{=} \langle id_{\Gamma}, t\rangle_{\mathbf{1}^{\Gamma}} : \Gamma \to \Gamma.\mathbf{1}^{\Gamma}$;
- (Unit-Comp) Under the same assumption, we have:
$$\mathcal{R}^{\mathbf{1}}_{A}(a, \top_{\Gamma}) = a;$$
- (Unit-Subst) Given a morphism $f : \Delta \to \Gamma$ in $\mathcal{C}$, we have:
$$\mathbf{1}^{\Gamma}\{f\} = \mathbf{1}^{\Delta} \in Ty(\Delta);$$
- ($\top$-Subst) Finally, we have:
$$\top_{\Gamma}\{f\} = \top_{\Delta} \in Tm(\Delta, \mathbf{1}^{\Delta}).$$

Moreover, $\mathcal{C}$ ***supports unit type in the strict sense*** if it additionally satisfies:

- ($\top$-Uniq) $t = \top_{\Gamma}$ for all $t \in Tm(\Gamma, \mathbf{1}^{\Gamma})$.[13]

Definition 5.4.28 (Interpretation of unit type)**.** The interpretation $[\![\_]\!]$ of $\mathsf{1}$-type in a CwF $\mathcal{C}$ that supports unit type is given by:

- ($\mathsf{1}$-Form) $[\![\mathsf{\Gamma \vdash 1\ type}]\!] \overset{\text{df.}}{=} \mathbf{1}^{[\![\mathsf{\Gamma}]\!]}$;
- ($\mathsf{1}$-Intro) $[\![\mathsf{\Gamma \vdash \star : 1}]\!] \overset{\text{df.}}{=} \top_{[\![\mathsf{\Gamma}]\!]}$;
- ($\mathsf{1}$-Elim) $[\![\mathsf{\Gamma \vdash R^{1}(C, c, t) : C[t/x]}]\!] \overset{\text{df.}}{=} \mathcal{R}^{\mathbf{1}}_{[\![\mathsf{C}]\!]}([\![\mathsf{c}]\!], [\![\mathsf{t}]\!])$

where the hypotheses of the rules are as presented in Section 5.2.

We now propose our game-semantic unit type:

Lemma 5.4.29 (Unit type in $\mathcal{LPG}$)**.** *The CwF $\mathcal{LPG}$ supports unit type in the strict sense.*

Proof. Let $\Gamma, \Delta \in \mathcal{LPG}$ and $[\phi] : \Delta \to \Gamma$ in $\mathcal{LPG}$.

[13]Note that $\top$-Uniq implies Unit-Elim and Unit-Comp by defining $\mathcal{R}^{\mathbf{1}}_{A}(a, t) \overset{\text{df.}}{=} a$.

- (Unit-Form) We define $\mathbf{1}^{\Gamma}$ to be the constant dlp-game $\{\mathbf{1}\}_{\Gamma}$, where $\mathbf{1}$ is the unit p-game in Example 5.3.25.

- (Unit-Intro) We define $[\top_{\Gamma}] \in \mathit{Tm}(\Gamma, \{\mathbf{1}\}_{\Gamma})$ to be the morphism $[\&_{\otimes\gamma:!\Gamma}\top_{\otimes\gamma}] : \Gamma \to \mathbf{1}$ in $\mathcal{LPG}$, where $\top_{\otimes\gamma} \stackrel{\mathrm{df.}}{=} \top$ up to 'tags'.

  Remark. Again, we write $[\top_{\Gamma}]$ rather than $\top_{\Gamma}$ for consistency of the notation.

- (Unit-Elim) Given $A \in \mathcal{DLPG}(\widehat{\Sigma}(\Gamma, \{\mathbf{1}\}_{\Gamma}))$, $[\alpha] \in \mathit{Tm}(\Gamma, A\{\langle [der_{\Gamma}], [\top_{\Gamma}]\rangle\})$ and $[\iota] \in \mathit{Tm}(\Gamma, \{\mathbf{1}\}_{\Gamma})$ in $\mathcal{LPG}$, we clearly have $[\iota] = [\top_{\Gamma}]$, i.e., $\top$-Uniq is satisfied, and thus we define $\mathcal{R}^{\mathbf{1}}_{A}([\alpha], [\iota]) \stackrel{\mathrm{df.}}{=} [\alpha]$.

- (Unit-Comp) Under the same assumption, $\mathcal{R}^{\mathbf{1}}_{A}([\alpha], [\top_{\Gamma}]) = [\alpha]$.

- (Unit-Subst) By the definition, $\{\mathbf{1}\}_{\Gamma}\{[\phi]\} = \{\mathbf{1}\}_{\Delta}$.

- ($\top$-Subst) In the same way, $[\top_{\Gamma}] \bullet [\phi] = [\top_{\Delta}] : \Delta \to \mathbf{1}$

which completes the proof. □

5.4.6.6 Game-Semantic Empty Type

Finally, we interpret *empty type*. First, its categorical interpretation is as follows:

Definition 5.4.30 (CwFs with empty type [92])**.** A CwF $\mathcal{C}$ ***supports empty type*** if:

- (Empty-Form) Given $\Gamma \in \mathcal{C}$, there is a type

$$\mathbf{0}^{\Gamma} \in \mathit{Ty}(\Gamma)$$

  called the ***empty type*** (in Γ);

- (Empty-Elim) Given $\Gamma \in \mathcal{C}$, $A \in \mathit{Ty}(\Gamma.\mathbf{0}^{\Gamma})$ and $z \in \mathit{Tm}(\Gamma, \mathbf{0}^{\Gamma})$, there is a term

$$\mathcal{R}^{\mathbf{0}}_{A}(z) \in \mathit{Tm}(\Gamma, A\{\overline{z}\})$$

  where $\overline{z} \stackrel{\mathrm{df.}}{=} \langle id_{\Gamma}, z\rangle_{\mathbf{0}^{\Gamma}} : \Gamma \to \Gamma.\mathbf{0}^{\Gamma}$;

- (Empty-Subst) Given $f : \Delta \to \Gamma$ in $\mathcal{C}$,

$$\mathbf{0}^{\Gamma}\{f\} = \mathbf{0}^{\Delta} \in \mathit{Ty}(\Delta);$$

- ($\mathcal{R}^0$-Subst) Under the same assumption,

$$\mathcal{R}^{\mathbf{0}}_{A\{f^+\}}(z\{f\}) = \mathcal{R}^{\mathbf{0}}_{A}(z)\{f\}$$

where $f^+ \overset{\mathrm{df.}}{=} \langle f \bullet p(\mathbf{0}^\Delta), v_{\mathbf{0}^\Gamma} \rangle_{\mathbf{0}^\Gamma} : \Delta.\mathbf{0}^\Delta \to \Gamma.\mathbf{0}^\Gamma$.

Remark. Since there is no term of empty type, there is no notion of CwFs supporting empty type in the strict sense.

Definition 5.4.31 (Interpretation of empty type)**.** The interpretation $[\![\_]\!]$ of $\mathsf{0}$-type in a CwF $\mathcal{C}$ that supports empty type is given by:

- ($\mathsf{0}$-Form) $[\![\Gamma \vdash \mathsf{0}\ \mathsf{type}]\!] \overset{\mathrm{df.}}{=} \mathbf{0}^{[\![\Gamma]\!]}$;
- ($\mathsf{0}$-Elim) $[\![\Gamma \vdash \mathsf{R}^0(\mathsf{C},\mathsf{a}) : \mathsf{C}[\mathsf{a}/\mathsf{x}]]\!] \overset{\mathrm{df.}}{=} \mathcal{R}^{\mathbf{0}}_{[\![\mathsf{C}]\!]}([\![\mathsf{a}]\!])$

where the hypotheses of the rules are as presented in Section 5.2.

We now propose our game-semantic interpretation of empty type:

Lemma 5.4.32 (Empty type in $\mathcal{LPG}$)**.** *The CwF $\mathcal{LPG}$ supports empty type.*

Proof. Let $\Gamma \in \mathcal{LPG}$ and $[\phi] : \Delta \to \Gamma$ in $\mathcal{LPG}$.

- (Empty-Form) We define $\mathbf{0}^\Gamma$ to be the constant dlp-game $\{\mathbf{0}\}_\Gamma$, where $\mathbf{0}$ is the empty p-game defined in Example 5.3.25.
- (Empty-Elim) Let $A \in \mathcal{DLPG}(\widehat{\Sigma}(\Gamma, \{\mathbf{0}\}))$ and $[\zeta] \in \mathit{Tm}(\Gamma, \{\mathbf{0}\})$ in $\mathcal{LPG}$. For $[\zeta] \in \mathit{Tm}(\Gamma, \{\mathbf{0}\})$, we may obtain the term $\mathcal{R}^{\mathbf{0}}_A([\zeta]) \in \mathit{Tm}(\Gamma, A\{\overline{[\zeta]}\})$ by the case distinction on whether $A = \{T\}$, where $\overline{[\zeta]} \overset{\mathrm{df.}}{=} \langle [der_\Gamma], [\zeta] \rangle : \Gamma \to \widehat{\Sigma}(\Gamma, \{\mathbf{0}\})$

  Remark. It must be the case $A \neq \{I\}$, which is why we have excluded I from the CCC $\mathcal{LPG}$ at the end of Section 5.3.4.
- (Empty-Subst) We clearly have $\{\mathbf{0}\}_\Gamma\{[\phi]\} = \{\mathbf{0}\}_\Delta$.
- ($\mathcal{R}^{\mathbf{0}}$-Subst) By the definition of $\mathcal{R}^{\mathbf{0}}_{\_}(\_)$, we clearly have:

$$\begin{aligned}\mathcal{R}^{\mathbf{0}}_{A\{[\phi]^+\}}([\zeta]\{[\phi]\}) &= \mathcal{R}^{\mathbf{0}}_{A\{[\phi]^+\}}([\zeta] \bullet [\phi]) \\ &= \mathcal{R}^{\mathbf{0}}_{A}([\zeta]) \bullet \langle [\phi] \bullet p(\{\mathbf{0}\}_\Delta), v_{\{\mathbf{0}\}_\Delta} \rangle \\ &= \mathcal{R}^{\mathbf{0}}_{A}([\zeta])\{[\phi]^+\}\end{aligned}$$

which completes the proof. □

5.4.6.7 Game-Semantic Universes

At the end of the present section, let us describe why the CwF $\mathcal{LPG}$ cannot interpret a *cumulative hierarchy of universes*, giving the motivation for the next section.

Recall that in Section 5.2 we have adopted *Tarski-style* universes equipped with the dependent type El and the meta-theoretic operation *En*. They in particular allow us to formulate universes in such a way that follows the usual pattern of formation, introduction, elimination and computation rules:

Definition 5.4.33 (CwFs with universes)**.** A CwF $\mathcal{C}$ ***supports (a cumulative hierarchy of) universes*** if:

- (U-Form) Given $\Gamma \in \mathcal{C}$, there is a type

$$\mathcal{U}_k^\Gamma \in \mathit{Ty}(\Gamma)$$

  called the ***k^{th}-universe*** (in Γ) for each natural number $k \in \mathbb{N}$;

  Notation. We often omit the superscript Γ on $\mathcal{U}_k^\Gamma$, and even write $\mathcal{U}$ for $\mathcal{U}_k$ with some $k \in \mathbb{N}$ unspecified.

- (U-Intro) Given $A \in \mathit{Ty}(\Gamma)$ that is not a universe and a natural number $k \in \mathbb{N}$, there are terms

$$\begin{aligned} \mathit{En}(A) &\in \mathit{Tm}(\Gamma, \mathcal{U}); \\ \mathit{En}(\mathcal{U}_k) &\in \mathit{Tm}(\Gamma, \mathcal{U}_{k+1}); \end{aligned}$$

- (U-ElimComp) Any term $C \in \mathit{Tm}(\Gamma, \mathcal{U})$ induces a type $\mathit{El}(C) \in \mathit{Ty}(\Gamma)$ that satisfies $\mathit{El}(\mathit{En}(A)) = A$ for all $A \in \mathit{Ty}(\Gamma)$;

- (U-Cumul) If $C \in \mathit{Tm}(\Gamma, \mathcal{U}_k)$, then $C \in \mathit{Tm}(\Gamma, \mathcal{U}_{k+1})$ for all $k \in \mathbb{N}$;

- (U-Subst) Given $f : \Delta \to \Gamma$ in $\mathcal{C}$,

$$\mathcal{U}_k^\Gamma\{f\} = \mathcal{U}_k^\Delta \in \mathit{Ty}(\Delta)$$

  for all $k \in \mathbb{N}$.

Definition 5.4.34 (Interpretation of universes)**.** The interpretation $[\![\_]\!]$ of universes in a CwF $\mathcal{C}$ that supports universes is given by:

- (U-Form) $[\![\mathsf{\Gamma} \vdash \mathsf{U_i}\ \mathsf{type}]\!] \stackrel{\mathrm{df.}}{=} \mathcal{U}_i^{[\![\mathsf{\Gamma}]\!]}$;

- (U-Intro) $[\![\mathsf{\Gamma} \vdash \mathit{En}(\mathsf{A}) : \mathsf{U_{i-1}}]\!] \stackrel{\mathrm{df.}}{=} \mathit{En}([\![\mathsf{A}]\!])$;
- (U-Elim) $[\![\mathsf{\Gamma} \vdash \mathsf{El(c)\ type}]\!] \stackrel{\mathrm{df.}}{=} \mathit{El}([\![\mathsf{c}]\!])$

where the hypotheses of the rules are as presented in Section 5.2.

Nevertheless, we cannot interpret U-Intro in $\mathcal{LPG}$ for some dlp-games may not be encoded. For instance, consider a dlp-game $H \in \mathcal{DLPG}(N \Rightarrow N)$ such that $H[\phi] \neq H[\psi]$ if $[\phi] \neq [\psi]$ for all $[\phi], [\psi] \in \mathcal{LPG}(N \Rightarrow N)$. To interpret U-Intro and U-ElimComp, we need a term $[\mathit{En}(H)] \in \mathit{Tm}(N \Rightarrow N, \mathcal{U})$ such that $[\mathit{En}(H)] \bullet [\phi] = \underline{H[\phi]}$ for all $[\phi] \in \mathcal{LPG}(N \Rightarrow N)$ so that $\mathit{El}([\mathit{En}(H)]) = \{H[\phi] \,|\, [\phi] \in \mathcal{LPG}(N \Rightarrow N)\} = H$. Then, however, $\mathit{En}(H)$ would be able to identify each $[\phi] \in \mathcal{LPG}(N \Rightarrow N)$, which is impossible because $\mathit{En}(H)$ is noetherian. For this problem, we need to restrict dlp-games to what correspond to type-theoretic constructions in MLTT.

To such dlp-games we may inductively assign the corresponding terms of universes, which realizes our game-semantic interpretation of the encoding operation En:

Definition 5.4.35 (Constructions on codes of types)**.** Given $\Gamma, G \in \mathcal{LPG}$ and $[\phi] : \Gamma \to \mathcal{U}$, $[\psi] : \widehat{\Sigma}(\Gamma, \mathit{El}([\phi])) \to \mathcal{U}$, $[\sigma], [\sigma'] : \mathit{Tm}(\Gamma, \mathit{El}([\phi]))$ in $\mathcal{LPG}$, we define the morphisms $\underline{G}_\Gamma, \underline{\Pi}([\phi], [\psi]), \underline{\Sigma}([\phi], [\psi]), \underline{\mathsf{Id}}_{[\phi]}([\sigma], [\sigma']) : \Gamma \to \mathcal{U}$ in $\mathcal{LPG}$ as follows:

- We define $\underline{G}_\Gamma : \Gamma \to \mathcal{U}$ to be the obvious non-strict one at $\|G\|$;
- As ψ induces its p-subgame $\psi^{\otimes\gamma} : \mathit{El}([\phi] \circ [\otimes\gamma]) \to \mathcal{U}$ for each $\otimes\gamma : !\Gamma$ defined by $\psi^{\otimes\gamma} \stackrel{\mathrm{df.}}{=} \&_{\otimes\tau : !\mathit{El}([\phi] \circ [\otimes\gamma])} \psi_{(\otimes\gamma)\otimes(\otimes\tau)}$ (i.e., only $\otimes\tau$ varies), we may define $\underline{\Pi}([\phi], [\psi]) : \Gamma \to \mathcal{U}$ to be the morphism in $\mathcal{LPG}$ that plays as $[\phi]$ except that for each $\otimes\gamma : !\Gamma$ the last move $\|\mathit{El}([\phi] \circ [\otimes\gamma])\|$ is replaced by $\|\widehat{\Pi}(\mathit{El}([\phi] \circ [\otimes\gamma]), \mathit{El}([\psi^{\otimes\gamma}]))\|$;
- The morphism $\underline{\Sigma}([\phi], [\psi]) : \Gamma \to \mathcal{U}$ is defined in the same manner to $\underline{\Pi}([\phi], [\psi])$;
- We define $\underline{\mathsf{Id}}_{[\phi]}([\sigma], [\sigma'])$ to be $[\phi]$ except that for each $\otimes\gamma : !\Gamma$ the last move $\|\mathit{El}([\phi] \circ [\otimes\gamma])\|$ is replaced by $\|\widehat{\mathsf{Id}}_{\mathit{El}([\phi] \circ [\otimes\gamma])}([\sigma] \circ [\otimes\gamma], [\sigma'] \circ [\otimes\gamma])\|$.

With these constructions on wv-strategies into universe games, we shall interpret universes in the next section.

5.5 Effectivity and Bijectivity

We have interpreted MLTT in the CwF $\mathcal{LPG}$ in the last section, but the model in $\mathcal{LPG}$ is not *effective*, where a game G is ***effective*** if its components are representable by partial recursive functions, or more precisely:

- M_G is a recursively enumerable (r.e.) set of natural numbers;
- λ_G is a recursive function, whose labels are some fixed natural numbers;
- $\vdash_G$ is a semi-decidable relation;
- P_G is an r.e. set of finite sequences of natural numbers;
- $\simeq_G$ is a semi-decidable relation.

Note that fixing a recursive bijection $\langle \_ \rangle : \mathbb{N}^* \to \mathbb{N}$ whose inverse is also recursive, we may talk about r.e. sets of and semi-decidable relations on *finite sequences of natural numbers* (see, e.g., [46, 157]). Then, assuming that the set $\mathscr{B}$ in Definition 5.3.23 contains only a finite (or countable) number of elements other than natural numbers, this notion of effective games is clearly applicable to p-games as well.

Since our aim is to give an accurate explanation of MLTT, effectivity, full completeness and faithfulness (n.b., the model in $\mathcal{LPG}$ is already faithful for types built without N- and Id-types, which can be established by the same proof in this section) are desirable properties of an interpretation of MLTT. Therefore, the rest of the thesis is dedicated to carve out a subCwF of $\mathcal{LPG}$ that forms an *effective*, *faithful* (for types built without N- and Id-types) and *fully complete* model of MLTT; in fact, we shall establish an effective and *bijective* interpretation of MLTT. Moreover, it enables us to interpret the cumulative hierarchy of universes as promised before.

Remark. Below, we achieve the surjectivity result by an *inductive* construction for universes (n.b., non-inductive full completeness in the presence of universes would be quite hard). On the other hand, full completeness for types built without universes is left as future work. One may argue that such an inductive surjectivity is just too trivial or it says virtually nothing; however, we claim that it is not the case. First, although our inductive construction on elements in $\mathcal{LPG}$ *as a whole* corresponds to constructions of MLTT, it is not a *rule-wise* correspondence, i.e., we do not just mimic the syntactic induction. Also, our game-semantic construction does not correspond to constructions of CwFs in this sense either; some game-semantic phenomena do not make sense in an arbitrary CwF (see Definition 5.5.1 below). Thus, the surjectivity result in this section is not entirely trivial. Moreover, it clarifies our game-semantic interpretation of MLTT better than $\mathcal{LPG}$; it provides some insights both on the syntax and CwFs.

5.5.1 Elementary P-Games and V-Strategies

We now present the CwF $\mathcal{EPG}$ of *elementary* lp-games, wv-strategies and dlp-games:

Definition 5.5.1 (The CwF $\mathcal{EPG}$)**.** The subCwF $\mathcal{EPG}$ of the CwF $\mathcal{LPG}$ is constructed from $\mathcal{LPG}$ as follows:

1. Lp-games, wv-strategies and dlp-games in $\mathcal{LPG}$ are restricted respectively to ***elementary predicative games (ep-games)***, ***elementary v-strategies (ev-strategies)*** and ***dependent elementary predicative games (dep-games)*** defined below. Ep-games and equivalence classes of certain ev-strategies defined below, called ***contextual v-strategies (cv-strategies)***, form a subcategory $\mathcal{EPG}$ of the category $\mathcal{LPG}$. Given $\Gamma \in \mathcal{EPG}$, we write $\mathcal{DEPG}(\Gamma)$ for the set of all dep-games over Γ, and given $A \in \mathcal{DEPG}(\Gamma)$, we write $\mathcal{EVS}(\Gamma, A)$ for the set of all equivalence classes of ev-strategies on $\widehat{\Pi}(\Gamma, A)$.

   These elements of $\mathcal{EPG}$ are inductively defined as follows:

   - (Base case) $T \in \mathcal{EPG}$, $\{\mathbf{1}\}_T, \{\mathbf{0}\}_T, \{N\}_T, \{\mathcal{U}_k\}_T \in \mathcal{DEPG}(T)$, $[\mathcal{z}_T] \in \mathcal{EVS}(T, \{N\}_T)$, $[s_T] \in \mathcal{EVS}(\widehat{\Sigma}(T, \{N\}_T), \{N\}_{\widehat{\Sigma}(T,N)})$ (n.b., $\widehat{\Sigma}(T, \{N\}_T) \in \mathcal{EPG}$ and $\{N\}_{\widehat{\Sigma}(T,\{N\}_T)} \in \mathcal{DEPG}(\widehat{\Sigma}(T, \{N\}_T))$ by the constructions Σ and $\_\{\_\}$ below), $[\underline{\mathbf{1}}_T], [\underline{\mathbf{0}}_T], [\underline{N}_T] \in \mathcal{EVS}(T, \{\mathcal{U}_0\}_T)$, $[\underline{\mathcal{U}_k}_T] \in \mathcal{EVS}(T, \{\mathcal{U}_l\}_T)$, where $k, l \in \mathbb{N}$ with $l > k$;
   - (Inductive step) Ep-games, equivalence classes of ev-strategies and dep-games are inductively constructed via $\top_{\_} : \Gamma \mapsto \top_\Gamma \in \mathcal{EVS}(\Gamma, \{\mathbf{1}\}_\Gamma)$, $\widehat{\Sigma}$, $\langle \_, \_ \rangle$, $\_\{\_\} : (A, [\phi]) \mapsto A\{[\phi]\}$, $\bullet : ([\alpha], [\phi]) \mapsto [\alpha] \bullet [\phi]$, $[\mathit{fst}_{\widehat{\Sigma}(\_,\_)}]$, $[\mathit{snd}_{\widehat{\Sigma}(\_,\_)}]$, Π, $\underline{\Pi}$, λ, λ^{-1}, Σ, $\underline{\Sigma}$, $\mathsf{Id}_{\_}\{\_, \_\} : (A, [\alpha], [\alpha']) \mapsto \mathsf{Id}_A\{\langle\langle [\mathit{der}_\Gamma], [\alpha] \rangle, [\alpha'] \rangle\}$[14], $\underline{\mathsf{Id}}$, $\mathit{Refl}_{\_}$, $\mathit{Refl}_{\_}^{-1}$, $\mathcal{R}^N$, $\mathcal{R}^{\mathbf{0}}$, El, where $\Gamma, \Delta \in \mathcal{EPG}$, $A \in \mathcal{DEPG}(\Gamma)$, $[\phi] \in \mathcal{EVS}(\Delta, \{\Gamma\}_\Delta)$ and $[\alpha], [\alpha'] \in \mathcal{EVS}(\Gamma, A)$;
   - (Cv-strategies) Cv-strategies are ev-strategies generated by $\top_{\_}$ and $\langle \_, \_ \rangle$ (thus, each cv-strategy ϕ satisfies $[\phi] : \Delta \to \Gamma$ for some $\Gamma, \Delta \in \mathcal{EPG}$);

[14]This particularly means that we do not regard the construction $A \mapsto \mathsf{Id}_A$ as 'atomic'. This point is crucial to establish a correspondence between dep-games and ev-strategies into universe games (because it is not always possible to determine the equality of given two parallel morphisms in $\mathcal{EPG}$).

2. Fixing the order of generating elementary elements by these rules, which we call the ***canonical order***, we define the ***construction number*** $\sharp(G) \in \mathbb{N}$ of each ep-game G (resp. equivalence class of ev-strategies) G by $\sharp(G) \stackrel{\mathrm{df.}}{=} \langle \mathcal{C}_G, \mathcal{R}(G) \rangle \in \mathbb{N}$, where G is the $\mathcal{C}_G^{\mathrm{th}}$ element among ep-games (resp. equivalence classes of ev-strategies) of rank $\mathcal{R}(G)$ in the canonical order (n.b., $\mathcal{R}([\phi])$ is defined to be $\mathcal{R}(\phi)$, which clearly does not depend on the choice of a representative ϕ);

3. The name $\|G\|$ of an ep-game (resp. ev-strategy) G that occurs as a move in a position of another ep-game is replaced by the construction number $\sharp(G)$;

4. Each universe game $\mathcal{U}_k$ is modified into $\mathcal{U}_k \stackrel{\mathrm{df.}}{=} \{\underline{G} \mid G \in \mathcal{EPG}_{\leqslant k+1}\}$, where $\underline{G} \stackrel{\mathrm{df.}}{=} \mathit{flat}(\sharp(G))_1$.

Remark. V-strategies $\otimes\sigma$ for any ev-strategy of the form $\phi = \&_{\otimes\sigma:!\Gamma}\phi_{\otimes\sigma} : \widehat{\Pi}(\Gamma, A)$ range over *any* v-strategies on $!\Gamma$, not only elementary ones. Also, note that the 'tags' $[\_]^{\|\otimes\sigma\|}$ in ϕ are unchanged as they do not occur as moves in a position.

Remark. Two ep-games (resp. equivalence classes of ev-strategies) may get different construction numbers even if they are the same element of $\mathcal{EPG}$; e.g., consider p-games N and $\mathcal{L}(N)[\underline{1}]$. It just means that Player may have several different *programs* for the same algorithm, and such a choice does not affect plays in a p-game.

5.5.2 Effective, Bijective Game Semantics of MLTT

The finitary nature of the inductive construction of $\mathcal{EPG}$ establishes:

Corollary 5.5.2 (Effectivity and bijectivity)**.** *The structure $\mathcal{EPG}$ forms an effective CwF (in the sense that its objects and representatives of its morphisms are effective) which supports all the semantic type formers of $\mathcal{LPG}$ and a cumulative hierarchy of universes. Moreover, the induced interpretation of MLTT in $\mathcal{EPG}$ is injective (for types built without* N*- and* Id*-types) and surjective.*

Proof. First, it is straightforward to see that $\mathcal{EPG}$ has enough elements from $\mathcal{LPG}$ to form a subCwF of $\mathcal{LPG}$ equipped with $\mathbf{1}$-, $\mathbf{0}$-, N-, Π-, Σ- and Id-types. For instance, it has the identity $[\mathit{der}_\Gamma] : \Gamma \to \Gamma$ on every $\Gamma \in \mathcal{EPG}$: $\mathit{der}_T = \top_T : T \Rightarrow T$ if $\Gamma = T$ and $\mathit{der}_{\widehat{\Sigma}(\Delta,A)} = \langle \mathit{fst}_{\widehat{\Sigma}(\Delta,A)}, \mathit{snd}_{\widehat{\Sigma}(\Delta,A)} \rangle$ if $\Gamma = \widehat{\Sigma}(\Delta, A)$ for some $\Delta \in \mathcal{EPG}$ and $A \in \mathit{Ty}(\Delta)$.

Moreover, $\mathcal{EPG}$ supports a cumulative hierarchy of universes[15] since dlp-games are now restricted to elementary ones, so that it may interpret U-Intro:

[15]We emphasize here again that this point is the main motivation to carve out the subCwF $\mathcal{EPG}$ from the CwF $\mathcal{LPG}$.

- (U-Form) Given $\Gamma \in \mathcal{EPG}$, $\{\mathcal{U}_k\}_\Gamma = \{\mathcal{U}_k\}_T\{[\top_\Gamma]\} \in \mathcal{DEPG}(\Gamma)$ for all $k \in \mathbb{N}$.
- (U-Intro) By induction on $A \in \mathcal{DEPG}(\Gamma)$, it is easy to see that there is a term $\mathit{En}(A) \in \mathit{Tm}(\Gamma, \{\mathcal{U}_k\}_\Gamma)$ in $\mathcal{EPG}$, where $\mathcal{R}(A) = k+2$, such that $\mathit{En}(A) \bullet [\gamma] = \underline{A[\gamma]}$ for all $[\gamma] \in \mathcal{EPG}(\Gamma)$. Explicitly, we define the operation En by:
  - If $A = \{G\}_T$, where G is either $\mathbf{1}$, $\mathbf{0}$, N or $\mathcal{U}$, then:
$$\mathit{En}(\{G\}_T) \overset{\text{df.}}{=} [\underline{G}_T]$$
    where $\underline{G}_T : T \Rightarrow \mathcal{U}$ is $\underline{G}_T : \mathcal{U}$ up to 'tags';
  - If $A = A'\{[\phi]\}$, where $A' \in \mathcal{DEPG}(\Delta)$ and $[\phi] : \Gamma \to \Delta$ in $\mathcal{EPG}$, then:
$$\mathit{En}(A'\{[\phi]\}) \overset{\text{df.}}{=} \mathit{En}(A') \bullet [\phi];$$
  - If $A = \Pi(B, C)$ (resp. $A = \Sigma(B, C)$) for some $B \in \mathcal{DEPG}(\Gamma)$ and $C \in \mathcal{DEPG}(\widehat{\Sigma}(\Gamma, B))$, then:
$$\mathit{En}(\Pi(B,C)) \overset{\text{df.}}{=} \underline{\Pi}(\mathit{En}(B), \mathit{En}(C));$$
$$\mathit{En}(\Sigma(B,C)) \overset{\text{df.}}{=} \underline{\Sigma}(\mathit{En}(B), \mathit{En}(C));$$
  - If $A = \mathsf{Id}_B\{\langle\langle[\mathit{der}_\Gamma], [\beta]\rangle, [\beta']\rangle\}$ for some $B \in \mathcal{DEPG}(\Gamma)$ and $[\beta], [\beta'] \in \mathit{Tm}(\Gamma, B)$ in $\mathcal{EPG}$, then:
$$\mathit{En}(\mathsf{Id}_B\{\langle\langle[\mathit{der}_\Gamma], [\beta]\rangle, [\beta']\rangle\}) \overset{\text{df.}}{=} \underline{\mathsf{Id}}_{\mathit{En}(B)}([\beta], [\beta']);$$
    Remark. We cannot define En on Id_B since otherwise there would be some term in $\mathit{Tm}(\Gamma.B.B^+, \mathcal{U})$ that may decide whether given two terms in $\mathit{Tm}(\Gamma, B)$ are the same by a *finite* interaction with them, which is impossible in the presence of the p-game N. As already remarked before, this is why we have taken the operation $(B, [\beta], [\beta']) \mapsto \mathsf{Id}_B\{\langle\langle[\mathit{der}_\Gamma], [\beta]\rangle, [\beta']\rangle\}$, not $B \mapsto \mathsf{Id}_B$, as 'atomic' in $\mathcal{EPG}$.
  - If $A = \mathit{El}([\mu])$ for some $[\mu] \in \mathit{Tm}(\Gamma, \{\mathcal{U}\}_\Gamma)$ in $\mathcal{EPG}$, then:
$$\mathit{En}(\mathit{El}([\mu])) \overset{\text{df.}}{=} [\mu].$$
    It is well-defined since $\mathit{El}([\mu]) \neq \mathit{El}([\mu'])$ for any $[\mu] \neq [\mu'] \in \mathit{Tm}(\Gamma, \{\mathcal{U}\}_\Gamma)$ in $\mathcal{EPG}$, which is because each element of $\mathit{Tm}(\Gamma, \{\mathcal{U}\}_\Gamma)$ in $\mathcal{EPG}$ is either a non-strict one or the second projection (this fact is easily checked by induction on terms of this kind).

- (U-IntroElimComp) It is easy to see that $El(En(A)) = A$ holds for all $A \in \mathcal{DEPG}(\Gamma)$ (again by induction on A).
- (U-Cumul) $[\mu] \in Tm(\Gamma, \{\mathcal{U}_k\}_\Gamma)$ clearly implies $[\mu] \in Tm(\Gamma, \{\mathcal{U}_{k+1}\}_\Gamma)$.
- (U-Subst) Given $[\phi] : \Delta \to \Gamma$ in $\mathcal{EPG}$ and $k \in \mathbb{N}$, we clearly have $\{\mathcal{U}_k\}_\Gamma\{[\phi]\} = \{\mathcal{U}_k\}_\Delta \in \mathcal{DEPG}(\Delta)$.

We have shown that $\mathcal{EPG}$ is a CwF with the semantic type formers supported in $\mathcal{LPG}$ as well as a cumulative hierarchy of universes.

Now, recall the syntactic notion of *context morphisms* [92], which is *derived* rather than primitive. Formally, a ***context morphism*** from a context $\vdash \Gamma\ \mathsf{ctx}$ to another $\vdash \Delta\ \mathsf{ctx}$, where $\vdash \Delta \equiv \diamondsuit, \mathsf{x}_1 : \mathsf{D}_1, \mathsf{x}_2 : \mathsf{D}_2, \dots, \mathsf{x}_n : \mathsf{D}_n\ \mathsf{ctx}$, is a finite sequence $\mathbf{d} = (\mathsf{d}_1, \mathsf{d}_2, \dots, \mathsf{d}_n) : \Gamma \to \Delta$ of terms such that:

$$\begin{aligned}
&\Gamma \vdash \mathsf{d}_1 : \mathsf{D}_1;\\
&\Gamma \vdash \mathsf{d}_2 : \mathsf{D}_2[\mathsf{d}_1/\mathsf{x}_1];\\
&\quad\vdots\\
&\Gamma \vdash \mathsf{d}_n : \mathsf{D}_n[\mathsf{d}_1/\mathsf{x}_1, \mathsf{d}_2/\mathsf{x}_2, \dots, \mathsf{d}_{n-1}/\mathsf{x}_{n-1}].
\end{aligned}$$

Its interpretation in a CwF as a morphism $\boldsymbol{d} : [\![\Gamma]\!] \to [\![\Delta]\!]$ is defined by induction on $|\Delta|$: $[\![(\_)]\!] \stackrel{\mathrm{df.}}{=} !_\Gamma : [\![\Gamma]\!] \to T$ and $[\![(\mathsf{d}_1, \mathsf{d}_2, \dots, \mathsf{d}_n, \mathsf{d}_{n+1})]\!] \stackrel{\mathrm{df.}}{=} \langle [\![(\mathsf{d}_1, \mathsf{d}_2, \dots, \mathsf{d}_n)]\!], [\![\mathsf{d}_{n+1}]\!] \rangle_{[\![\mathsf{D}_{n+1}]\!]} : [\![\Gamma]\!] \to T.[\![\mathsf{D}_1]\!].[\![\mathsf{D}_2]\!] \dots [\![\mathsf{D}_n]\!].[\![\mathsf{D}_{n+1}]\!]$. Furthermore, given a syntactic expression E, we define the ***generalized substitution*** $\mathsf{E}[\mathbf{d}/\mathbf{x}]$ of $\mathbf{d}$ for $\mathbf{x}$ in E, where $\mathbf{x} = (\mathsf{x}_1, \mathsf{x}_2, \dots, \mathsf{x}_n)$, to be the expression

$$\mathsf{E}[\mathsf{d}_1/\mathsf{x}_1, \mathsf{d}_2/\mathsf{x}_2, \dots, \mathsf{d}_n/\mathsf{x}_n]$$

i.e., what is obtained from E by simultaneously substituting d_i for x_i in E for $i = 1, 2, \dots, n$. Then, it is shown in [92] that if $\Delta, \Theta \vdash \mathsf{J}$ is a judgement, then so is $\Gamma, \Theta[\mathbf{d}/\mathbf{x}] \vdash \mathsf{J}[\mathbf{d}/\mathbf{x}]$. Note that generalized substitution subsumes the rules Subst and Weak. It is an important theorem in [92] that generalized substitution corresponds to the *semantic substitution* $\_\{\_\}$ in a CwF. Although context morphisms and generalized substitution are implicit and rather derived in the syntax, it is one of the points that the categorical interpretation of type theories make elegant and useful to take them as primitive and interpret substitution in this manner.

Note, however, that morphisms in the category $\mathcal{EPG}$ are a particular kind of terms of the CwF $\mathcal{EPG}$, namely *contextual* ones. This suggests that context morphisms may be regarded *as terms*, which we call ***contextual terms***. It is in fact possible, and thus context morphisms are no longer an unofficial or auxiliary concept; let us define the contextual term $\Sigma(\mathbf{d})$ that corresponds to the context morphism $\mathbf{d}$ given above to be the term

$$\Gamma \vdash \Sigma(\mathbf{d}) \overset{\text{df.}}{\equiv} \langle \dots \langle\langle \star, \mathsf{d}_1\rangle, \mathsf{d}_2\rangle, \dots, \mathsf{d}_n\rangle : \Sigma(\Delta)$$

where the type $\Gamma \vdash \Sigma(\Delta)$ type (with no free variable) is defined by induction on $|\Delta|$:

$$\Sigma(\Delta) \overset{\text{df.}}{\equiv} \begin{cases} 1 & \text{if } \vdash \Delta \equiv \diamondsuit \ \mathsf{ctx}; \\ \Sigma_{\mathsf{s}:\Sigma(\Delta')} \mathsf{D}_{n+1}[\pi_1^{\Delta'}(\mathsf{s})/\mathsf{x}_1, \pi_2^{\Delta'}(\mathsf{s})/\mathsf{x}_2, \dots, \pi_n^{\Delta'}(\mathsf{s})/\mathsf{x}_n] & \text{if } \vdash \Delta \equiv \Delta', \mathsf{x}_{n+1} : \mathsf{D}_{n+1} \ \mathsf{ctx} \end{cases}$$

where $\vdash \Delta' \equiv \diamondsuit, \mathsf{x}_1 : \mathsf{D}_1, \mathsf{x}_2 : \mathsf{D}_2, \dots, \mathsf{x}_n : \mathsf{D}_n$ ctx, and the terms

$$\Gamma, \mathsf{s} : \Sigma(\Delta') \vdash \pi_0^{\Delta'}(\mathsf{s}) : 1;$$
$$\Gamma, \mathsf{s} : \Sigma(\Delta') \vdash \pi_i^{\Delta'}(\mathsf{s}) : \mathsf{D}_i^{\Delta'}(\mathsf{s}) \overset{\text{df.}}{\equiv} \mathsf{D}_i[\pi_1^{\Delta'}(\mathsf{s})/\mathsf{x}_1, \pi_2^{\Delta'}(\mathsf{s})/\mathsf{x}_2, \dots, \pi_{i-1}^{\Delta'}(\mathsf{s})/\mathsf{x}_{i-1}] \ (i = 1, 2, \dots, n)$$

are ***Δ'-projections*** defined simultaneously (in the sense that $\Sigma(\Delta')$ and Δ'-projections enable us to define $\Sigma(\Delta)$, which in turn defines Δ-projections) by induction on $|\Delta'|$:

$$\Gamma, \mathsf{s} : \Sigma(\diamondsuit) \vdash \pi_0^{\diamondsuit}(\mathsf{s}) \overset{\text{df.}}{\equiv} \star : 1;$$
$$\Gamma, \mathsf{s} : \Sigma(\Delta', \mathsf{x}_{n+1} : \mathsf{D}_{n+1}) \vdash \pi_i^{\Delta}(\mathsf{s}) \overset{\text{df.}}{\equiv} \pi_i^{\Delta'}(\pi_1^{\Sigma(\Delta'), \mathsf{D}_{n+1}}(\mathsf{s})) : \mathsf{D}_i^{\Delta}(\mathsf{s}) \ (i = 0, 1, \dots, n);$$
$$\Gamma, \mathsf{s} : \Sigma(\Delta', \mathsf{x}_{n+1} : \mathsf{D}_{n+1}) \vdash \pi_{n+1}^{\Delta}(\mathsf{s}) \overset{\text{df.}}{\equiv} \pi_2^{\Sigma(\Delta'), \mathsf{D}_{n+1}}(\mathsf{s}) : \mathsf{D}_{n+1}^{\Delta}(\mathsf{s}).$$

Clearly, we have:

$$\Gamma \vdash \pi_0^{\Delta}(\Sigma(\mathbf{d})) \equiv \star : 1;$$
$$\Gamma \vdash \pi_i^{\Delta}(\Sigma(\mathbf{d})) \equiv \mathsf{d}_i : \mathsf{D}_i^{\Delta}(\Sigma(\mathbf{d})) \ (i = 1, 2, \dots, n)$$

so that $\Gamma \vdash \Sigma(\mathbf{d}) : \Sigma(\Delta)$ holds.

Then, by induction on $|\Delta|$, it is easy to see that the interpretation of $\mathbf{d}$ in $\mathcal{EPG}$ coincides with that of $\Sigma(\mathbf{d})$, i.e.,

$$[\![(\mathsf{d}_1, \mathsf{d}_2, \dots, \mathsf{d}_n) : \Gamma \to \Delta]\!] = [\![\Gamma \vdash \langle \dots \langle\langle \star, \mathsf{d}_1\rangle, \mathsf{d}_2\rangle, \dots, \mathsf{d}_n\rangle : \Sigma(\Delta)]\!].$$

Notice that this equation does not necessarily hold (or it does not even make sense) in a CwF in general because terms may not be interpreted as morphisms in the underlying category. Moreover, given a context $\vdash \Delta$ ctx, a type $\Delta \vdash \mathsf{C}$ type and a term $\Delta \vdash \mathsf{c} : \mathsf{C}$, we may obtain a context $\vdash \diamondsuit, \mathsf{s} : \Sigma(\Delta)$ ctx, a type $\diamondsuit, \mathsf{s} : \Sigma(\Delta) \vdash \mathsf{C}^{\Delta}(\mathsf{s})$ type

and a term $\diamondsuit, \mathsf{s} : \Sigma(\Delta) \vdash \mathsf{c}^{\Delta}(\mathsf{s}) \overset{\mathrm{df.}}{\equiv} \mathsf{c}[\pi_1^{\Delta}(\mathsf{s})/\mathsf{x}_1, \pi_2^{\Delta}(\mathsf{s})/\mathsf{x}_2, \dots, \pi_{\mathsf{n}+1}^{\Delta}(\mathsf{s})/\mathsf{x}_{\mathsf{n}+1}] : \mathsf{C}^{\Delta}(\mathsf{s})$ such that the following five equations hold:

$$\begin{aligned}
[\![\vdash \diamondsuit, \mathsf{s} : \Sigma(\Delta) \ \mathsf{ctx}]\!] &= [\![\vdash \Delta \ \mathsf{ctx}]\!];\\
[\![\diamondsuit, \mathsf{s} : \Sigma(\Delta) \vdash \mathsf{C}^{\Delta}(\mathsf{s}) \ \mathsf{type}]\!] &= [\![\Delta \vdash \mathsf{C} \ \mathsf{type}]\!];\\
[\![\diamondsuit, \mathsf{s} : \Sigma(\Delta) \vdash \mathsf{c}^{\Delta}(\mathsf{s}) : \mathsf{C}^{\Delta}(\mathsf{s})]\!] &= [\![\Delta \vdash \mathsf{c} : \mathsf{C}]\!];\\
[\![\Gamma \vdash \mathsf{C}^{\Delta}[\Sigma(\mathbf{d})/\mathsf{s}] \ \mathsf{type}]\!] &= [\![\Gamma \vdash \mathsf{C}[\mathbf{d}/\mathbf{x}] \ \mathsf{type}]\!];\\
[\![\Gamma \vdash \mathsf{c}^{\Delta}[\Sigma(\mathbf{d})/\mathsf{s}] : \mathsf{C}^{\Delta}[\Sigma(\mathbf{d})/\mathsf{s}]]\!] &= [\![\mathsf{c}[\mathbf{d}/\mathbf{x}] : \mathsf{C}[\mathbf{d}/\mathbf{x}]]\!].
\end{aligned}$$

What these equations really mean is that our game semantics suggests that each context morphism $\mathbf{d} : \Gamma \to \Delta$ should be regarded as a term, namely the contextual term $\Gamma \vdash \Sigma(\mathbf{d}) : \Sigma(\Delta)$, and types $\Delta \vdash \mathsf{C}$ type and terms $\Delta \vdash \mathsf{c} : \mathsf{C}$ should be transformed into their ***contextual form*** $\vdash \diamondsuit, \mathsf{s} : \Sigma(\Delta)$ ctx and $\diamondsuit, \mathsf{s} : \Sigma(\Delta) \vdash \mathsf{C}^{\Delta}(\mathsf{s})$ type, respectively, where generalized substitution of context morphisms are replaced by that of context terms. This resolves the gap between syntactic (context) morphisms and game-semantic morphisms in $\mathcal{EPG}$. Nevertheless, for notational brevity, we shall not employ the transformation of context morphisms into contextual terms (resp. of types and terms into canonical form) in the rest of the thesis.

Next, it is not hard to see that ev-strategies are all effective if we ignore 'tags' in ev-strategies on linear implication[16], where note that the number of the 'tags' can be uncountable. Importantly, a v-strategy of the form $\phi : A \multimap B$ is representable as the union $\bigcup \phi \overset{\mathrm{df.}}{=} \bigcup_{\sigma:A} \phi_\sigma$ by its *uniformity*, and we may recover ϕ from $\bigcup \phi$ because $\phi = \&_{\sigma:A}\{\boldsymbol{s} \in P^{\mathsf{Even}}_{\bigcup \phi} \mid \boldsymbol{s} \upharpoonright A \in P_\sigma\}$. Thus, we may safely ignore the 'tags'.

In particular, we have shown that morphisms in $\mathcal{EPG}$ of the form $\mu : \Gamma \to \mathcal{U}$ are effective. Thus, given a dep-game $A \in \mathcal{DEPG}(\Gamma)$, we may obtain an 'effective algorithm' $En(A) : \Gamma \to \mathcal{U}$ that embodies A; thus, each dep-game is 'effective' in this sense. It then easily follows from this fact that each ep-game is also effective.

Below, we show that the interpretation $[\![\_]\!]$ is *fully complete* and *faithful*; in fact, it is *injective* (for types built without N- and Id-types) and *surjective*, i.e., the three maps (5.1), (5.2), (5.3) above are bijections. By induction on the construction of elementary elements, applying the *substitution lemma* [92] for the substitutions $\_\{\_\}$, it is straightforward to prove that the interpretation in $\mathcal{EPG}$ is *surjective*, i.e., given $\Gamma \in \mathcal{EPG}$, $A \in \mathcal{DEPG}(\Gamma)$ and $[\alpha] \in Tm(\Gamma, A)$ in $\mathcal{EPG}$, there are judgements $\vdash \Gamma$ ctx, $\Gamma \vdash \mathsf{A}$ $\mathsf{type_i}$ and $\Gamma \vdash \mathsf{a} : \mathsf{A}$ such that $[\![\vdash \Gamma \ \mathsf{ctx}]\!] = \Gamma$, $[\![\Gamma \vdash \mathsf{A} \ \mathsf{type_i}]\!] = A$, $[\![\Gamma \vdash \mathsf{a} : \mathsf{A}]\!] = [\alpha]$ and $\mathcal{R}(A) = i$. Just for clarity, we describe in detail the inductive argument below:

[16]It conceptually makes sense too because Player must be blind to the 'tags'.

- (Case $T \in \mathcal{EPG}$) We have $\vdash \diamondsuit\ \mathsf{ctx}$ by Ctx-Emp such that $[\![\vdash \diamondsuit\ \mathsf{ctx}]\!] = T$.

- (Case $\{\mathbf{1}\}_T, \{\mathbf{0}\}_T, \{N\}_T, \{\mathcal{U}\}_T \in \mathcal{DEPG}(T)$) Let A be $\{\mathbf{1}\}_T$, $\{\mathbf{0}\}_T$ or $\{N\}_T$. Then, $\diamondsuit \vdash \mathsf{A}\ \mathsf{type}_1$ by Ctx-Emp and A-Form such that $[\![\diamondsuit \vdash \mathsf{A}\ \mathsf{type}_1]\!] = A$ and $\mathcal{R}(A) = 1$. The case $A = \{\mathcal{U}\}_T$ is analogous.

- (Case $[z_T] \in \mathit{Tm}(T, \{N\}_T)$) We have $\diamondsuit \vdash \mathsf{zero} : \mathsf{N}$ by Ctx-Emp and N-IntroZ such that $[\![\diamondsuit \vdash \mathsf{zero} : \mathsf{N}]\!] = [z_T]$.

- (Case $[s_T] \in \mathit{Tm}(\widehat{\Sigma}(T, \{N\}_T), \{N\}_{\widehat{\Sigma}(T,\{N\}_T)})$) We have $\diamondsuit, \mathsf{x} : \mathsf{N} \vdash \mathsf{succ}(\mathsf{x}) : \mathsf{N}$ by Ctx-Emp, N-Form, Ctx-Ext, Var and N-IntroS such that

$$\begin{aligned}
[\![\diamondsuit, \mathsf{x} : \mathsf{N} \vdash \mathsf{succ}(\mathsf{x}) : \mathsf{N}]\!] &= v_{\{N\}_T}\{[\mathit{succ}_T] \bullet \langle \mathit{id}_{\widehat{\Sigma}(T,\{N\}_T)}, v_{\{N\}_T}\rangle\} \\
&= v_{\{N\}_T} \bullet \langle \mathit{id}_{\widehat{\Sigma}(T,\{N\}_T)}, v_{\{N\}_T}\{[\mathit{succ}_T]\}\rangle \\
&= v_{\{N\}_T} \bullet [\mathit{succ}_T] \\
&= v_{\{N\}_T} \bullet \langle p(\{N\}_T), [s_T]\rangle \\
&= [s_T].
\end{aligned}$$

- (Case $[\underline{\mathbf{1}}_T], [\underline{\mathbf{0}}_T], [\underline{N}_T], [\underline{\mathcal{U}}_T] \in \mathit{Tm}(T, \{\mathcal{U}\}_T)$) Let μ be either $\underline{\mathbf{1}}_I$, $\underline{\mathbf{0}}_T$, $\underline{N}_T$ or $\underline{\mathcal{U}}_T$. For $\{\mathbf{1}\}_T, \{\mathbf{0}\}_T, \{N\}_T, \{\mathcal{U}\}_T \in \mathcal{DEPG}(T)$, we have $\diamondsuit \vdash \mathsf{u} : \mathsf{U}$ by Ctx-Emp and U-Intro such that $[\![\diamondsuit \vdash \mathsf{u} : \mathsf{U}]\!] = [\mu]$.

- (Case $[\top_\Gamma] \in \mathcal{EPG}(\Gamma, T)$) Since $\Gamma \in \mathcal{EPG}$, we have $\vdash \mathsf{\Gamma}\ \mathsf{ctx}$ such that $[\![\vdash \mathsf{\Gamma}\ \mathsf{ctx}]\!] = \Gamma$ (by the induction hypothesis). Then, by $\mathsf{1}$-Intro, we have $\mathsf{\Gamma} \vdash \star : \mathsf{1}$ such that $[\![\mathsf{\Gamma} \vdash \star : \mathsf{1}]\!] = [\top_\Gamma]$.

- (Case $\widehat{\Sigma}(\Delta, A) \in \mathcal{EPG}$) Since $\Delta \in \mathcal{EPG}$, $A \in \mathcal{DEPG}(\Delta)$, we have $\vdash \mathsf{\Delta}\ \mathsf{ctx}$, $\mathsf{\Delta} \vdash \mathsf{A}\ \mathsf{type}$ such that $[\![\vdash \mathsf{\Delta}\ \mathsf{ctx}]\!] = \Delta$ and $[\![\mathsf{\Delta} \vdash \mathsf{A}\ \mathsf{type}]\!] = A$ (by the induction hypothesis). By Ctx-Ext, we obtain $\vdash \mathsf{\Delta}, \mathsf{x} : \mathsf{A}\ \mathsf{ctx}$ such that $[\![\vdash \mathsf{\Delta}, \mathsf{x} : \mathsf{A}\ \mathsf{ctx}]\!] = \widehat{\Sigma}([\![\vdash \mathsf{\Delta}\ \mathsf{ctx}]\!], [\![\mathsf{\Delta} \vdash \mathsf{A}\ \mathsf{type}]\!]) = \widehat{\Sigma}(\Delta, A)$.

- (Case $\langle [\phi], [\alpha] \rangle \in \mathcal{EPG}(\Delta, \widehat{\Sigma}(\Gamma, A))$) Since $\Gamma, \Delta \in \mathcal{EPG}$, $A \in \mathcal{DEPG}(\Gamma)$, $[\phi] : \Delta \to \Gamma$, $[\alpha] \in \mathit{Tm}(\Delta, A\{[\phi]\})$ in $\mathcal{EPG}$, we have terms $\mathsf{\Delta} \vdash \mathsf{\Sigma}(\mathbf{d}) : \mathsf{\Sigma}(\mathsf{\Gamma})$ and $\mathsf{\Delta} \vdash \mathsf{a} : \mathsf{A}[\mathsf{\Sigma}(\mathbf{d})/\mathbf{x}]$ such that $[\![\mathsf{\Delta} \vdash \mathsf{\Sigma}(\mathbf{d}) : \mathsf{\Sigma}(\mathsf{\Gamma})]\!] = [\phi]$ and $[\![\mathsf{\Delta} \vdash \mathsf{a} : \mathsf{A}[\mathsf{\Sigma}(\mathbf{d})/\mathbf{x}]]\!] = [\alpha]$ (by the induction hypothesis). We form another term $\mathsf{\Delta} \vdash \langle \mathsf{\Sigma}(\mathbf{d}), \mathsf{a} \rangle : \mathsf{\Sigma}(\mathsf{\Gamma}, \mathsf{y} : \mathsf{A})$ such that $[\![\mathsf{\Delta} \vdash \langle \mathsf{\Sigma}(\mathbf{d}), \mathsf{a} \rangle : \mathsf{\Sigma}(\mathsf{\Gamma}, \mathsf{y} : \mathsf{A})]\!] = \langle [\![\mathsf{\Sigma}(\mathbf{d})]\!], [\![\mathsf{a}]\!] \rangle = \langle [\phi], [\alpha] \rangle$.

- (CASE $A'\{[\phi]\} \in \mathcal{DEPG}(\Gamma)$, $[\alpha'] \bullet [\phi] \in \mathit{Tm}(\Gamma, A'\{[\phi]\})$ AND $[\phi] \in \mathcal{EPG}(\Gamma, \Delta)$) Since $A' \in \mathcal{DEPG}(\Delta)$ and $[\phi] : \Gamma \to \Delta$ in $\mathcal{EPG}$, we have a type $\mathsf{\Delta \vdash A'\ type}$ and a term $\mathsf{\Gamma \vdash \Sigma(\mathbf{d}) : \Sigma(\Delta)}$ such that $[\![\mathsf{\Delta \vdash A'\ type}]\!] = A'$ and $[\![\mathsf{\Gamma \vdash \Sigma(\mathbf{d}) : \Sigma(\Delta)}]\!] = [\phi]$ (by the induction hypothesis). Then, by the generalized substitution, we have the judgement $\mathsf{\Gamma \vdash A'[\Sigma(\mathbf{d})/\mathbf{x}]\ type}$ such that

$$[\![\mathsf{\Gamma \vdash A'[\Sigma(\mathbf{d})/\mathbf{x}]\ type}]\!] = [\![\mathsf{\Delta \vdash A'\ type}]\!]\{[\![\mathsf{\Gamma \vdash \Sigma(\mathbf{d}) : \Sigma(\Delta)}]\!]\} = A'\{[\phi]\}$$

by the substitution lemma. The case $[\alpha'] \bullet [\phi] \in \mathit{Tm}(\Gamma, A'\{[\phi]\})$ is analogous.

- (CASE $p(A) \in \mathcal{EPG}(\widehat{\Sigma}(\Gamma, A), \Gamma)$ AND $v_A \in \mathit{Tm}(\widehat{\Sigma}(\Gamma, A), A)$) Since $\Gamma \in \mathcal{EPG}$ and $A \in \mathcal{DEPG}(\Gamma)$, we have $\mathsf{\vdash \Gamma\ ctx}$ and $\mathsf{\Gamma \vdash A\ type}$ such that $[\![\mathsf{\vdash \Gamma\ ctx}]\!] = \Gamma$ and $[\![\mathsf{\Gamma \vdash A\ type}]\!] = A$ (by the induction hypothesis). Without loss of generality, we may assume $\mathsf{\vdash \Gamma \equiv \diamondsuit, x_1 : A_1, x_2 : A_2, \dots, x_n : A_n\ ctx}$. Then, we have:

$$\begin{aligned} &\mathsf{\Gamma, x : A \vdash x_1 : A_1} \\ &\mathsf{\Gamma, x : A \vdash x_2 : A_2[x_1/x_1]} \\ &\qquad\quad\vdots \\ &\mathsf{\Gamma, x : A \vdash x_n : A_n[x_1/x_1, x_2/x_2, \dots, x_{n-1}/x_{n-1}]} \end{aligned}$$

so that we may form the contextual term

$$\mathsf{\Gamma, x : A \vdash \langle \dots \langle\langle \star, x_1\rangle, x_2\rangle, \dots, x_n\rangle : \Sigma(\Gamma)}$$

that satisfies $[\![\mathsf{\Gamma, x : A \vdash \langle \dots \langle\langle \star, x_1\rangle, x_2\rangle, \dots, x_n\rangle : \Sigma(\Gamma)}]\!] = p(A)$. Also, by Ctx-Ext and Var, we obtain $\mathsf{\Gamma, x : A \vdash x : A}$ such that $[\![\mathsf{\Gamma, x : A \vdash x : A}]\!] = v_A$.

- (CASE $\Pi(A, B), \Sigma(A, B) \in \mathcal{DEPG}(\Gamma)$) For $A \in \mathcal{DEPG}(\Gamma)$, $B \in \mathcal{DEPG}(\widehat{\Sigma}(\Gamma, A))$, by the induction hypothesis, we have $\mathsf{\Gamma \vdash A\ type_i}$ and $\mathsf{\Gamma, x : A \vdash B\ type_j}$ such that $[\![\mathsf{\Gamma \vdash A\ type_i}]\!] = A$, $[\![\mathsf{\Gamma, x : A \vdash B\ type_j}]\!] = B$, $\mathcal{R}(A) = i$ and $\mathcal{R}(B) = j$. By Π-Form, we obtain $\mathsf{\Gamma \vdash \Pi_{x:A}B\ type_{max(i,j)}}$ such that $[\![\mathsf{\Pi_{x:A}B}]\!] = \Pi([\![\mathsf{A}]\!], [\![\mathsf{B}]\!]) = \Pi(A, B)$. Also, $\mathcal{R}(\Pi(A, B)) = \max(\mathcal{R}(A), \mathcal{R}(B)) = \max(i, j)$. It is completely analogous for the case $\Sigma(A, B) \in \mathcal{EPG}$.

- (CASE $\underline{\Pi}([\phi], [\psi]), \underline{\Sigma}([\phi], [\psi]) \in \mathit{Tm}(\Gamma, \{\mathcal{U}\}_\Gamma)$) Since we have $\phi \in \mathit{Tm}(\Gamma, \{\mathcal{U}\}_\Gamma)$ and $\psi \in \mathit{Tm}(\widehat{\Sigma}(\Gamma, \mathit{El}([\phi])), \{\mathcal{U}\}_{\widehat{\Sigma}(\Gamma, \mathit{El}([\phi]))})$ in $\mathcal{EPG}$, by the induction hypothesis, we have $\mathsf{\Gamma \vdash b : U}$ and $\mathsf{\Gamma, y : El(b) \vdash c : U}$ such that $[\![\mathsf{b}]\!] = [\phi]$ and $[\![\mathsf{c}]\!] = [\psi]$. We have $\mathsf{\Gamma \vdash El(b)\ type}$ and $\mathsf{\Gamma, y : El(b) \vdash El(c)\ type}$ by U-Elim such that $[\![\mathsf{El(b)}]\!] =$

$El(\llbracket \mathsf{b} \rrbracket) = El([\phi])$ and $\llbracket \mathsf{El(c)} \rrbracket = El(\llbracket \mathsf{c} \rrbracket) = El([\psi])$. Thus, by $\mathsf{\Pi}$-Form and U-Intro, we have $\mathsf{\Gamma \vdash En(\Pi_{y:El(b)}El(c)) : U}$ such that

$$\begin{aligned}\llbracket \mathsf{En(\Pi_{y:El(b)}El(c))} \rrbracket &= En(\Pi(\llbracket \mathsf{El(b)} \rrbracket, \llbracket \mathsf{El(c)} \rrbracket)) \\ &= En(\Pi(El([\phi]), El([\psi]))) \\ &= \underline{\Pi}(En(El([\phi])), En(El([\psi]))) \\ &= \underline{\Pi}([\phi], [\psi]).\end{aligned}$$

The case $\underline{\Sigma}([\phi], [\psi]) \in Tm(\Gamma, \{\mathcal{U}\}_\Gamma)$ is completely analogous.

- (Case $\lambda_{A,B}([\beta]) \in Tm(\Gamma, \Pi(A, B))$) For $A \in \mathcal{DEPG}(\Gamma)$, $B \in \mathcal{DEPG}(\widehat{\Sigma}(\Gamma, A))$ and $[\beta] \in Tm(\widehat{\Sigma}(\Gamma, A), B)$, by the induction hypothesis, we have $\mathsf{\Gamma \vdash A\ type}$, $\mathsf{\Gamma, x : A \vdash B\ type}$ and $\mathsf{\Gamma, x : A \vdash b : B}$ with $\llbracket \mathsf{A} \rrbracket = A$, $\llbracket \mathsf{B} \rrbracket = B$ and $\llbracket \mathsf{b} \rrbracket = [\beta]$. By $\mathsf{\Pi}$-Intro, we have $\mathsf{\Gamma \vdash \lambda x^A.b : \Pi_{x:A}B}$ such that $\llbracket \mathsf{\lambda x^A.b} \rrbracket = \lambda_{A,B}(\llbracket \mathsf{b} \rrbracket) = \lambda_{A,B}([\beta])$.

- (Case $\lambda^{-1}_{A,B}([\psi]) \in Tm(\widehat{\Sigma}(\Gamma, A), B)$) For $[\psi] \in Tm(\Gamma, \Pi(A, B))$, we have, by the induction hypothesis and $\mathsf{\Pi}$-Uniq, $\mathsf{\Gamma \vdash \lambda x.f(x) : \Pi_{x:A}B}$ such that $\llbracket \mathsf{\lambda x.f(x)} \rrbracket = [\psi]$. Thus, we have $\mathsf{\Gamma, x : A \vdash f(x) : B(x)}$ such that:

$$\begin{aligned}\llbracket \mathsf{\Gamma, x : A \vdash f(x) : B[x/x]} \rrbracket &= \lambda^{-1}_{A,B}(\llbracket \mathsf{\Gamma \vdash \lambda x.f(x) : \Pi_{x:A}B} \rrbracket) \\ &= \lambda^{-1}_{A,B}([\psi]).\end{aligned}$$

- (Case $\mathsf{Id}_A\{\langle\langle [der_\Gamma], [\alpha]\rangle, [\alpha']\rangle\} \in \mathcal{DEPG}(\Gamma)$) For $\Gamma \in \mathcal{EPG}$, $A \in \mathcal{DEPG}(\Gamma)$ and $[\alpha], [\alpha'] \in Tm(\Gamma, A)$ in $\mathcal{EPG}$, by the induction hypothesis, we have $\mathsf{\Gamma \vdash A\ type_i}$, $\mathsf{\Gamma \vdash a : A}$ and $\mathsf{\Gamma \vdash a' : A}$ such that $\llbracket \mathsf{A} \rrbracket = A$, $\mathcal{R}(A) = i$, $\llbracket \mathsf{a} \rrbracket = [\alpha]$ and $\llbracket \mathsf{a'} \rrbracket = [\alpha']$. Thus, by $=$-Form, we have $\mathsf{\Gamma \vdash a =_A a'\ type_i}$ such that

$$\begin{aligned}\llbracket \mathsf{a =_A a'} \rrbracket &= \mathsf{Id}_{\llbracket \mathsf{A} \rrbracket}\{\langle\langle der_{\llbracket \mathsf{\Gamma} \rrbracket}, \llbracket \mathsf{a} \rrbracket\rangle, \llbracket \mathsf{a'} \rrbracket\rangle\} \\ &= \mathsf{Id}_A\{\langle\langle [der_\Gamma], [\alpha]\rangle, [\alpha']\rangle\}.\end{aligned}$$

- (Case $\underline{\mathsf{Id}}_{[\mu]}([\tau], [\tau']) \in Tm(\Gamma, \mathcal{U})$) For $[\mu] \in Tm(\Gamma, \mathcal{U})$ and $[\tau], [\tau'] \in Tm(\Gamma, El([\mu]))$, we have $\mathsf{\Gamma \vdash b : U}$, $\mathsf{\Gamma \vdash t : El(b)}$ and $\mathsf{\Gamma \vdash t' : El(b)}$ (by the induction hypothesis) such that $\llbracket \mathsf{b} \rrbracket = [\mu]$, $\llbracket \mathsf{t} \rrbracket = [\tau]$ and $\llbracket \mathsf{t'} \rrbracket = [\tau']$. We then have $\mathsf{\Gamma \vdash En(t =_{El(b)} t') : U}$ by $=$-Form and U-Intro such that

$$\begin{aligned}\llbracket \mathsf{En(t =_{El(b)} t')} \rrbracket &= En(\llbracket \mathsf{t =_{El(b)} t'} \rrbracket) \\ &= En(\mathsf{Id}_{\llbracket \mathsf{El(b)} \rrbracket}\{\langle [der_\Gamma], \llbracket \mathsf{t} \rrbracket\rangle, \llbracket \mathsf{t'} \rrbracket\rangle\}) \\ &= En(\mathsf{Id}_{El([\mu])}\{\langle [der_\Gamma], [\tau]\rangle, [\tau']\rangle\}) \\ &= \underline{\mathsf{Id}}_{En(El([\mu]))}([\tau], [\tau']) \\ &= \underline{\mathsf{Id}}_{[\mu]}([\tau], [\tau']).\end{aligned}$$

- (Case $\mathit{Refl}_A : \widehat{\Sigma}(\Gamma, A_1) \leftrightarrows \widehat{\Sigma}(\widehat{\Sigma}(\widehat{\Sigma}(\Gamma, A_2), A_3^+), \mathsf{Id}_A) : \mathit{Refl}_A^{-1}$) For $\Gamma \in \mathcal{EPG}$ and $A \in \mathcal{DEPG}(\Gamma)$, by the induction hypothesis, $\vdash \mathsf{\Gamma\ ctx}$ and $\mathsf{\Gamma \vdash A\ type}$ such that $[\![\mathsf{\Gamma}]\!] = \Gamma$ and $[\![\mathsf{A}]\!] = A$. Let us write $\mathsf{\Gamma \vdash id_\Gamma : \Sigma(\Gamma)}$ for the obvious identity context term on $\mathsf{\Gamma}$. By $=$-Intro, $\mathsf{\Gamma, x : A \vdash \langle\langle\langle id_\Gamma, x\rangle, x\rangle, refl_x\rangle : \Sigma(\Gamma, y : A, z : A, y =_A z)}$ and $\mathsf{\Gamma, x : A, y : A, z : x =_A y \vdash \langle id_\Gamma, x\rangle : \Sigma(\Gamma, z : A)}$ such that $[\![\mathsf{\langle\langle\langle id_\Gamma, x\rangle, x\rangle, refl_x\rangle}]\!] = \mathit{Refl}_A$ and $[\![\mathsf{\langle id_\Gamma, x\rangle}]\!] = \mathit{Refl}_A^{-1}$ (n.b., the two terms are not inverses to each other, which indicates that our interpretation is not faithful for Id-types).

- (Case $\mathcal{R}^N_P([c_z],[c_s]) \in \mathit{Tm}(\widehat{\Sigma}(\Gamma, \{N\}_\Gamma), P)$ and $\mathcal{R}^{\mathbf{0}}([\zeta]) \in \mathit{Tm}(\Gamma, A\{\overline{[\zeta]}\})$) Since we have $P \in \mathcal{DEPG}(\widehat{\Sigma}(\Gamma, \{N\}_\Gamma))$, $[c_z] \in \mathit{Tm}(\Gamma, P\{\mathit{zero}\})$ and $[c_s] \in \mathit{Tm}(\widehat{\Sigma}(\widehat{\Sigma}(\Gamma, \{N\}_\Gamma), P), P\{\mathit{succ} \bullet p(P)\})$, by the induction hypothesis, we have $\mathsf{\Gamma, x : N \vdash P(x)\ type}$, $\mathsf{\Gamma \vdash c_z : P[0/x]}$, $\mathsf{\Gamma, x : N, y : P(x) \vdash c_s(x,y) : P[succ(x)/x]}$ such that $[\![\mathsf{P(x)}]\!] = P$, $[\![\mathsf{c_z}]\!] = [c_z]$ and $[\![\mathsf{c_s(x,y)}]\!] = [c_s]$. By Weak, Var and N-Elim, we obtain $\mathsf{\Gamma, x : N \vdash R^N(P, c_z, c_s, x) : P(x)}$ such that

$$\begin{aligned}[\![\mathsf{R^N(P, c_z, c_s, x)}]\!] &= \mathcal{R}^N_{[\![\mathsf{P}]\!]}([\![\mathsf{c_z}]\!], [\![\mathsf{c_s}]\!])\{\langle p(N), [\![\mathsf{\Gamma, x : N \vdash x : N}]\!]\rangle\} \\ &= \mathcal{R}^N_P([c_z],[c_s]) \bullet [\mathit{der}_{\widehat{\Sigma}(\Gamma,N)}] \\ &= \mathcal{R}^N_P([c_z],[c_s]).\end{aligned}$$

  It is even simpler for the case $\mathcal{R}^{\mathbf{0}}([\zeta]) : \widehat{\Pi}(\Gamma, A\{\overline{[\zeta]}\})$.

- (Case $\mathit{El}([\mu]) \in \mathcal{DEPG}(\Gamma)$) For $\Gamma \in \mathcal{EPG}$ and $[\mu] \in \mathit{Tm}(\Gamma, \{\mathcal{U}\}_\Gamma)$, by the induction hypothesis, we have $\mathsf{\Gamma \vdash u : U}$ such that $[\![\mathsf{\Gamma \vdash u : U}]\!] = [\mu]$. By U-Elim, we have $\mathsf{\Gamma \vdash El(u)\ type}$ such that $[\![\mathsf{\Gamma \vdash El(u)\ type}]\!] = \mathit{El}([\![\mathsf{\Gamma \vdash u : U}]\!]) = \mathit{El}([\mu])$.

It remains to show that the interpretation is injective (for types built without N- and Id-types). Let $\Gamma \in \mathcal{EPG}$ $A \in \mathcal{DEPG}(\Gamma)$ and $[\alpha] \in \mathit{Tm}(\Gamma, A)$ in $\mathcal{EPG}$. Exploiting the surjectivity just established, below we prove that a context $\vdash \mathsf{\Gamma\ ctx}$, a type $\mathsf{\Gamma \vdash A\ type}$ and a term $\mathsf{\Gamma \vdash a : A}$ that denote Γ, A and $[\alpha]$, respectively, are unique by induction on $\mathsf{\Gamma}$, A and a:

- If $\vdash \mathsf{\Gamma \equiv \diamondsuit\ ctx}$, then $\Gamma = T$. Clearly, only $\vdash \mathsf{\diamondsuit\ ctx}$ denotes T.

- If $\vdash \mathsf{\Gamma \equiv \Gamma', x : A\ ctx}$, then $\Gamma = \widehat{\Sigma}(\Gamma', A)$. Any context that denotes Γ must be of the form $\vdash \mathsf{\Delta', y : B\ ctx}$. Since $\widehat{\Sigma}(\Delta', B) = [\![\vdash \mathsf{\Delta', y : B\ ctx}]\!] = \Gamma = \widehat{\Sigma}(\Gamma', A)$, it follows that $\Delta' = \Gamma'$ and $B = A$. By the induction hypothesis, $\vdash \mathsf{\Delta' \equiv \Gamma'\ ctx}$ and $\mathsf{\Gamma' \vdash B \equiv A\ type}$, whence $\vdash \mathsf{\Delta', y : B \equiv \Gamma', x : A\ ctx}$ by Ctx-ExtEq.

- If either $\mathsf{\Gamma \vdash A \equiv 1\ type}$, $\mathsf{\Gamma \vdash A \equiv 0\ type}$, $\mathsf{\Gamma \vdash A \equiv N\ type}$ or $\mathsf{\Gamma \vdash A \equiv U\ type}$, then $A = \{G\}_\Gamma$, where G is either $\mathbf{1}$, $\mathbf{0}$, N or $\mathcal{U}$. Note that $\mathsf{\Gamma \vdash A\ type}$ cannot be a $\mathsf{\Pi}$-, $\mathsf{\Sigma}$- or Id-type (e.g., $T \Rightarrow G \neq G$ and $T \& G \neq G$ by 'tags' for disjoint union). Thus, $\mathsf{\Gamma \vdash A\ type}$ is the unique type that denotes A.

- By $\mathsf{\Pi}$-Uniq, $\mathsf{\Pi}$-Comp, $\mathsf{\Sigma}$-Uniq, $\mathsf{\Sigma}$-Comp, $\mathbf{1}$-Uniq and $\mathbf{1}$-Comp, we may exclude the cases where a is constructed by $\mathsf{\Pi}$-Elim, $\mathsf{\Sigma}$-Elim or $\mathbf{1}$-Elim.

  Remark. It is the lack of a uniqueness rule for the N-type that prevents us from obtaining an injective interpretation of MLTT in the presence of the N-type.

- If $\mathsf{\Gamma \vdash a \equiv x_i : A_i}$, where $\mathsf{\Gamma \equiv x_1 : A_1, \dots, x_n : A_n\ ctx}$, $i \in \{1, \dots, n\}$, then $[\alpha] = v_{A_i}$. It is then clear that $\mathsf{\Gamma \vdash x_i : A_i}$ is only the term that denotes $[\alpha]$. It is analogous for the cases where a is $\star$, zero, $\mathsf{succ(n)}$ or $\mathsf{R^0(C, a)}$. In the following, it is an important point that we may exclude these cases for a.

- If $\mathsf{\Gamma \vdash A \equiv El(a)\ type}$, where $\mathsf{\Gamma \vdash a : U}$, then $A = El([\alpha])$. Assume that there is another type $\mathsf{\Gamma \vdash A'\ type}$ such that $A' = El([\alpha])$. Then, since $En(A') = En(El([\alpha])) = [\alpha]$, by the induction hypothesis, we have $\mathsf{\Gamma \vdash En(A') \equiv a : U}$, whence $\mathsf{\Gamma \vdash A' \equiv El(En(A')) \equiv El(a) \equiv A\ type}$.

- If $\mathsf{\Gamma \vdash A \equiv \Pi_{x:C}D(x)\ type}$, then $A = \Pi(C, D)$. Let $\mathsf{\Gamma \vdash B\ type}$ be another type that denotes A. Then, we must have $\mathsf{\Gamma \vdash B \equiv \Pi_{x:C'}D'(x)\ type}$ for some $\mathsf{\Gamma \vdash C'\ type}$, $\mathsf{\Gamma, x : C' \vdash D'(x)\ type}$. However, since $\Pi(C, D) = \Pi(C', D')$ implies $C = C'$ and $D = D'$[17], by the induction hypothesis, we have $\mathsf{\Gamma \vdash C \equiv C'\ type}$ and $\mathsf{\Gamma, x : C \vdash D(x) \equiv D'(x)\ type}$. Finally, by the congruence for Π-types, we obtain $\mathsf{\Gamma \vdash A \equiv \Pi_{x:C}D(x) \equiv \Pi_{x:C'}D'(x) \equiv B\ type}$. It is analogous for $\mathsf{\Gamma \vdash A \equiv \Sigma_{x:C}D(x)\ type}$.

- If $\mathsf{\Gamma \vdash a \equiv \lambda x^C.d(x) : \Pi_{x:C}D(x)}$, then $[\alpha] = \lambda_{C,D}([\delta]) : \widehat{\Pi}(\Gamma, \Pi(C, D))$, where $\delta : \widehat{\Pi}(\widehat{\Sigma}(\Gamma, C), D)$ in $\mathcal{EPG}$. Let $\mathsf{\Gamma \vdash a' : \Pi_{x:C}D(x)}$ be a term that denotes $[\alpha]$. By $\mathsf{\Pi}$-Uniq, $\mathsf{a'}$ is of the form $\mathsf{\Gamma \vdash a' \equiv \lambda x^C.d'(x) : \Pi_{x:C}D(x)}$ for some $\mathsf{\Gamma, x : C \vdash d'(x) : D(x)}$. By the induction hypothesis, $\mathsf{\Gamma, x : C \vdash d(x) \equiv d'(x) : D(x)}$, whence, by the congruence for $\mathsf{\Pi}$-Intro, $\mathsf{\Gamma \vdash a \equiv \lambda x^C.d(x) \equiv \lambda x^C.d'(x) \equiv a' : \Pi_{x:C}D(x)}$.

- If $\mathsf{\Gamma \vdash a \equiv \langle c, d \rangle : \Sigma_{x:C}D(x)}$, then $[\alpha] = \langle [\vartheta], [\delta] \rangle : \Sigma(C, D)$ for some $[\vartheta]$ and $[\delta]$. Let $\mathsf{\Gamma \vdash a' : \Sigma_{x:C}D(x)}$ be another term that denotes $[\alpha]$, which is of the form $\mathsf{\Gamma \vdash a' \equiv \langle c', d' \rangle : \Sigma_{x:C}D(x)}$ by $\mathsf{\Sigma}$-Uniq. Let $[\vartheta']$ and $[\delta']$ be the denotations of $\mathsf{c'}$ and $\mathsf{d'}$, respectively. For $\langle [\vartheta], [\delta] \rangle = \langle [\vartheta'], [\delta'] \rangle$ implies $[\vartheta] = [\vartheta']$ and $[\delta] =$

[17]Note that $\Pi(C, \{\mathbf{1}\}_{\widehat{\Sigma}(\Gamma, C)}) \neq \Pi(C', \{\mathbf{1}\}_{\widehat{\Sigma}(\Gamma, C)})$ if $C \neq C'$ thanks to 'tags' for linear implication.

$[\delta']$, by the induction hypothesis, $\Gamma \vdash \mathsf{c} \equiv \mathsf{c}' : \mathsf{C}$ and $\Gamma \vdash \mathsf{d} \equiv \mathsf{d}' : \mathsf{D}[\mathsf{c}/\mathsf{x}]$, whence $\Gamma \vdash \mathsf{a} \equiv \langle \mathsf{c}, \mathsf{d} \rangle \equiv \langle \mathsf{c}', \mathsf{d}' \rangle \equiv \mathsf{a}' : \Sigma_{\mathsf{x}:\mathsf{C}} \mathsf{D}(\mathsf{x})$ by the congruence for Σ-Intro.

- If $\Gamma \vdash \mathsf{a} \equiv \mathsf{En}(\mathsf{A}) : \mathsf{U}$, then $[\alpha] = \mathit{En}(A)$. Let $\Gamma \vdash \mathsf{a}' : \mathsf{U}$ be a term that denotes $[\alpha]$. Then a' must be of the form $\Gamma \vdash \mathsf{a}' \equiv \mathsf{En}(\mathsf{A}') : \mathsf{U}$ for some $\Gamma \vdash \mathsf{A}'\ \mathsf{type}$. Let A' be the denotation of A'. Then $A = \mathit{El}(\mathit{En}(A)) = \mathit{El}([\alpha]) = \mathit{El}(\llbracket \mathsf{a}' \rrbracket) = \mathit{El}(\llbracket \mathsf{En}(\mathsf{A}') \rrbracket) = \mathit{El}(\mathit{En}(A')) = A'$. Thus, $\Gamma \vdash \mathsf{A} \equiv \mathsf{A}'\ \mathsf{type}$ by the induction hypothesis, whence $\Gamma \vdash \mathsf{a} \equiv \mathsf{En}(\mathsf{A}) \equiv \mathsf{En}(\mathsf{A}') \equiv \mathsf{a}' : \mathsf{U}$ by the congruence for U-Intro.

$\square$

5.6 Intensionality

We now investigate *how intensional the model of MLTT in $\mathcal{EPG}$ is* through some of the syntactic rules.

5.6.1 Equality Reflection

The axiom of *Equality reflection (EqRefl)*, which states that propositionally equal terms are are judgmentally equal too, i.e., $\Gamma \vdash \mathsf{p} : \mathsf{a} =_{\mathsf{A}} \mathsf{a}' \Rightarrow \Gamma \vdash \mathsf{a} \equiv \mathsf{a}' : \mathsf{A}$ for any type $\Gamma \vdash \mathsf{A}\ \mathsf{type}$ and terms $\Gamma \vdash \mathsf{a} : \mathsf{A}$, $\Gamma \vdash \mathsf{a}' : \mathsf{A}$ and $\Gamma \vdash \mathsf{p} : \mathsf{a} =_{\mathsf{A}} \mathsf{a}'$, is the difference between the *intensional* and *extensional* variants of MLTT: The extensional one is the extension of the intensional one by adding EqRefl.

EqRefl is not valid in $\mathcal{EPG}$ when we consider *open* terms. For example, consider terms $\mathsf{x} : \mathsf{N}, \mathsf{y} : \mathsf{0} \vdash \mathsf{x} : \mathsf{N}$ and $\mathsf{x} : \mathsf{N}, \mathsf{y} : \mathsf{0} \vdash \mathsf{succ}(\mathsf{x}) : \mathsf{N}$; they are interpreted as different v-strategies $T \& N \& \mathbf{0} \overset{[\mathit{fst}]}{\rightarrow} T \& N \overset{[\mathit{snd}]}{\rightarrow} N$ and $T \& N \& \mathbf{0} \overset{[\mathit{fst}]}{\rightarrow} T \& N \overset{[s_T]}{\rightarrow} N$ in $\mathcal{EPG}$, respectively. However, we may interpret a term $\mathsf{x} : \mathsf{N}, \mathsf{y} : \mathsf{0} \vdash \mathsf{p} : \mathsf{succ}(\mathsf{x}) =_{\mathsf{N}} \mathsf{x}$ as:

$$p : \widehat{\Pi}(\widehat{\Sigma}(\widehat{\Sigma}(T, \{N\}), \{\mathbf{0}\}), \mathsf{Id}_{\{N\}}\{\langle\langle [\mathit{der}_{\widehat{\Sigma}(\widehat{\Sigma}(T,\{N\}),\{\mathbf{0}\})}], [\mathit{snd}] \bullet [\mathit{fst}] \rangle, [s_T] \bullet [\mathit{fst}] \rangle\})$$

in $\mathcal{EPG}$ that just plays in the component game $\mathbf{0}$ in the domain at the second move, so that it is trivially a proof. Note that p is elementary because we may apply the construction $\mathcal{R}^{\mathbf{0}}$ for the dlp-game $\mathsf{Id}_{\{N\}}\{\langle\langle [\mathit{der}_{\widehat{\Sigma}(\widehat{\Sigma}(T,\{N\}),\{\mathbf{0}\})}], [\mathit{snd}] \bullet [\mathit{fst}] \rangle, [s_T] \bullet [\mathit{fst}] \rangle\} \in \mathcal{DEPG}(\widehat{\Sigma}(\widehat{\Sigma}(T, \{N\}), \{\mathbf{0}\}))$ and the projection $[\mathit{snd}] \in \mathit{Tm}(\widehat{\Sigma}(\widehat{\Sigma}(T, \{N\}), \{\mathbf{0}\}), \{\mathbf{0}\})$.

5.6.2 Function Extensionality

Next, we consider the principle of *function extensionality (FunExt)*, which states that for any types $\Gamma \vdash \mathsf{A}\ \mathsf{type}$ and $\Gamma, \mathsf{x} : \mathsf{A} \vdash \mathsf{B}\ \mathsf{type}$ and terms $\Gamma \vdash \mathsf{f} : \Pi_{\mathsf{x}:\mathsf{A}}\mathsf{B}(\mathsf{x})$ and $\Gamma \vdash \mathsf{g} : \Pi_{\mathsf{x}:\mathsf{A}}\mathsf{B}(\mathsf{x})$, we can inhabit the type

$$\Gamma \vdash \Pi_{\mathsf{x}:\mathsf{A}}\mathsf{f}(\mathsf{x}) = \mathsf{g}(\mathsf{x}) \to \mathsf{f} = \mathsf{g}\ \mathsf{type}.$$

Let us focus on the case where $\vdash \Gamma \equiv \diamondsuit\ \mathsf{ctx}$, and ignore $T = [\![\vdash \diamondsuit\ \mathsf{ctx}]\!]$ for simplicity. It is not hard to see that the model in $\mathcal{EPG}$ refutes this axiom for a v-strategy $\phi = \&_{\otimes\sigma:!A}\phi_\sigma : \widehat{\Pi}(A, B)$ is not completely specified by the function $\pi_\phi : \mathcal{VS}(!A) \to \mathcal{VS}(\int B)$ and its behavior in $\int B$. For instance, let us consider strategies $\underline{0}_N, \hat{\underline{0}}_N : N \to N$ defined by $P_{\underline{0}_N} \overset{\text{df.}}{=} \mathsf{Pref}(\{q.0\})$ and $P_{\hat{\underline{0}}_N} \overset{\text{df.}}{=} \mathsf{Pref}(\{q.q.n.0 \mid n \in \mathbb{N}\})$. Note that the former is clearly elementary, and so is the latter by the construction $\mathcal{R}^N$. Then, we do not have a strategy on the p-game $\widehat{\Pi}(N, \mathsf{Id}_{\{N\}}\{\langle\langle[der_N], \underline{0}_N\rangle, \hat{\underline{0}}_N\rangle\}) \Rightarrow \widehat{\mathsf{Id}}_{N \Rightarrow N}(\underline{0}_N, \hat{\underline{0}}_N)$ in $\mathcal{EPG}$ because the domain is true but the codomain is false.

5.6.3 Uniqueness of Identity Proofs

Next, we investigate the principle of *uniqueness of identity proofs (UIP)*, which states that for any type $\Gamma \vdash \mathsf{A}\ \mathsf{type}$, the following type can be inhabited:

$$\Gamma \vdash \Pi_{\mathsf{a}_1,\mathsf{a}_2:\mathsf{A}}\Pi_{\mathsf{p},\mathsf{q}:\mathsf{a}_1=\mathsf{a}_2}\mathsf{p} = \mathsf{q}\ \mathsf{type}.$$

If the empty type occurs in Γ, then the interpretation of this type has a proof in the same way as the case of EqRefl; so assume otherwise. Then, for any $\Gamma \in \mathcal{EPG}$, $A \in \mathcal{DEPG}(\Gamma)$, $a_1, a_2 : \widehat{\Pi}(\Gamma, A)$, any two v-strategies $p, q : \widehat{\Pi}(\Gamma, \mathsf{Id}_A\{\langle\langle[der_\Gamma], [a_1]\rangle, [a_2]\rangle\})$ must be the same, and so there must be the trivial v-strategy between them. It is now clear that there is a v-strategy on the game $\widehat{\Pi}([\![\Gamma]\!], [\![\Pi_{\mathsf{a}_1,\mathsf{a}_2:\mathsf{A}}\Pi_{\mathsf{p},\mathsf{q}:\mathsf{a}_1=\mathsf{a}_2}\mathsf{p} = \mathsf{q}]\!])$ in $\mathcal{EPG}$ (but it is a bit too tedious to write down what exactly the v-strategy is).

We regard this point as unsatisfactory since it has been shown by the classic groupoid model [93] by Hofmann and Streicher that UIP is not derivable in MLTT. Of course, one may regard that it is just a deficiency or incompleteness of the syntax (for instance, it is Streicher's motivation to introduce *Axiom K* [174]); however, our standpoint is that UIP should *not* be valid in MLTT for the following reasons:

- The recent active research on HoTT is stimulated by the infinite hierarchy of Id-types and its connection with higher categories and homotopy theory, which becomes trivial if UIP holds;

- Conceptually, proofs $\mathsf{\Gamma \vdash p : a_1 =_A a_2}$ and $\mathsf{\Gamma \vdash q : a_1 =_A a_2}$ are computations that witness an equality of $\mathsf{a_1}$ and $\mathsf{a_2}$, and thus p and q may be judgmentally different.

5.6.4 Criteria of Intensionality

There are Streicher's three *Criteria of Intensionality* [174]:

- (I) $\mathsf{A : U, x, y : A, z : x =_A y \nvdash x \equiv y : A}$.
- (II) $\mathsf{A : U, B : A \to U, x, y : A, z : x =_A y \nvdash B(x) \equiv B(y) : U}$.
- (III) If $\diamondsuit \vdash \mathsf{p : t =_A t'}$, then $\diamondsuit \vdash \mathsf{t \equiv t' : A}$.

The point here is that Opponent's anti-strategy does not have to be innocent, wb, total or noetherian; in particular, he does not need to provide a proof for an identity game. For the criterion I, we have the two derelictions on A as the interpretation of $\mathsf{A : U, x, y : A, z : x =_A y \vdash x : A}$ and $\mathsf{A : U, x, y : A, z : x =_A y \vdash y : A}$. These v-strategies are clearly different as Opponent may behave differently for x and y, so the model in $\mathcal{EPG}$ satisfies the criterion I. In a similar manner, it is straightforward to see that our model in $\mathcal{EPG}$ satisfies the criterion II as well. Finally, the criterion III is also satisfied in our model because the terms t and $\mathsf{t'}$ are *closed*.

5.6.5 Univalence Axiom

We finally analyze the *univalence axiom (UA)*, the heart of HoTT, which states that the type

$$\mathsf{\Gamma \vdash (En(A) =_U En(B)) \simeq (A \simeq B)\ type}$$

is inhabitable for all types $\mathsf{\Gamma \vdash A\ type}$ and $\mathsf{\Gamma \vdash B\ type}$. A term of an *equivalence* $\mathsf{\Gamma \vdash A \simeq B\ type}$ consists of functions $\mathsf{\Gamma \vdash f : A \to B}$, $\mathsf{\Gamma \vdash g : B \to A}$ and identity proofs $\mathsf{\Gamma \vdash p : f \circ g = id_B}$ and $\mathsf{\Gamma \vdash q : g \circ f = id_A}$; for the precise definition, see [184]. The essence of UA is the component

$$\mathsf{\Gamma \vdash (A \simeq B) \to En(A) =_U En(B)\ type.}$$

For simplicity, let us focus on the case where $\vdash \mathsf{\Gamma} \equiv \diamondsuit\ \mathsf{ctx}$. Its interpretation, if exists, must be a v-strategy $\phi : A \simeq B \to \widehat{\mathsf{Id}}_{\mathcal{U}}(En(A), En(B))$. The point is when $A = B$ but $\sharp(A) \neq \sharp(B)$; for instance consider the p-games N and $\mathcal{L}(N)[\underline{1}]$. They are clearly the same p-game, but $\sharp(N) \neq \sharp(\mathcal{L}(N)[\underline{1}])$, and so there is no proof for the implication

$$(N \simeq \mathcal{L}(N)[\underline{1}]) \Rightarrow \widehat{\mathsf{Id}}_{\mathcal{U}}(En(N), En(\mathcal{L}(N)[\underline{1}])).$$

Hence the model in $\mathcal{EPG}$ does not validate UA.

This suggests that to interpret UA we need to allow non-trivial proofs for identity spaces, so that we have proofs for $\widehat{\mathsf{Id}}_{\mathcal{U}}(En(A), En(B))$ when $\sharp(A) \neq \sharp(B)$ but A and B are *equivalent* p-games. For this aim, it seems promising to incorporate the groupoid structure of [93]; in fact, the work [190] formulates this idea precisely, and also it has improved the present work in the point that it refutes UIP.

Nevertheless, from a *computational* viewpoint, our game semantics suggests that the formula

$$(A \simeq B) \Rightarrow \widehat{\mathsf{Id}}_{\mathcal{U}}(En(A), En(B)).$$

is *not* computational or constructive as any proof of the formula, if exists, needs to determine whether or not $\sharp(A) = \sharp(B)$ by a *finite* interaction in the domain $A \simeq B$.

5.7 Conclusion and Future Work of the Chapter

In this chapter, we have presented a new variant of games that gives rise to an *effective* and *surjective* game semantics of MLTT with $\mathsf{1}$-, $\mathsf{0}$-, N-, $\mathsf{\Pi}$-, $\mathsf{\Sigma}$-, and Id-types as well as a *cumulative hierarchy of universes.* When Id-types are excluded, the interpretation is additionally *injective.* Notably, one may regard v-strategies as a mathematical and syntax-independent formulation of computation of MLTT, which is in accordance with the meaning explanation. Also, as shown in the previous section, our model is in fact *intensional*, and it is more abstract or 'high-level' than, e.g., TMs. Thus, we have achieved to some degree our research aim described in Section 1.3.

However, its interpretation of Id-types does not capture phenomena of HoTT very well as we have seen in the previous section (it validates UIP and refutes UA). In particular, it implies that our model is not faithful. Meanwhile, we have recognized that the CwF $\mathcal{LPG}$ looks quite similar to the CwF of *groupoids* in the classic paper [93]. Thus, it is future work to refine the model by equipping it with a groupoid structure to interpret Id-types better; see [190]. Also, it would be meaningful to strengthen our full completeness result in terms of some intrinsic constraint on games and strategies (perhaps for MLTT *without universes*). Moreover, our semantics provides some insights to reformulate CwFs *categorically* in the sense that types and terms are interpreted in the underlying category.

Chapter 6

Piecing Together

In the last three chapters, we have extended conventional game semantics reviewed in Chapter 2 to capture dynamics and intensionality of computation (Chapter 3), higher-order computability (Chapter 4) and dependent types (Chapter 5). As stated in the introduction, these three developments are essentially *orthogonal*, and thus it is possible to combine them to establish a single mathematical framework to encompass them all. The present chapter briefly summarizes the consequences of this unification.

6.1 Dynamic Game Semantics of MLTT

First, it is now clear how to incorporate the generalized structure of dynamic games and strategies in Chapter 3 into the game semantics of MLTT in Chapter 5.

From the view of dynamic game semantics, the reformulation of *games in terms of strategies* in Chapter 5 is pleasing. Recall that in the CCBoC $\mathcal{LDG}$ (Definition 3.4.1) a β-morphism $A \to B$ is a pair $(J, [\phi])$ of a dynamic game J such that $\mathcal{H}^\omega(J) \trianglelefteq A \Rightarrow B$ and $\mathcal{H}^\omega(J^\dagger) = \mathcal{H}^\omega(J)^\dagger$, and the equivalence class $[\phi] = \{\psi : J \mid \psi \text{ is winning}, \psi \simeq_J \phi\}$ of a valid, winning dynamic strategy $\phi : J$, where the underlying dynamic game J is mainly to take the equivalence class $[\phi]$ with respect to $\simeq_J$, but the reformulation of Chapter 5 allows us to dispense with it, resulting in a simpler semantics that ignores the inessential distinction on the underlying dynamic games.

From the point of game semantics of MLTT, the dynamic, intensional extension of Chapter 3 models the distinction between *canonical* and *non-canonical* terms, and the *normalization* of the meaning explanation of MLTT, respectively, by the distinction between normalized and non-normalized dynamic strategies, and the hiding operation $\mathcal{H}$. Therefore, we have achieved a mathematical, syntax-independent formulation of the meaning explanation. On the other hand, however, it remains to establish a DCP for MLTT (with respect to some small-step operational semantics).

6.2 Game-Semantic Realizability for MLTT

It is also straightforward to apply the definition of *viability* (Definition 4.4.7) or *JPA-computability* (Definition **??**) to the generalized strategies given in Chapter 5, and show that every dynamic strategy definable in MLTT is viable or JPA-computable, giving rise to an intrinsic model of computation for MLTT. Note in particular that fixed-point strategies introduced in Chapter 4 are not noetherian in general, but we may overcome this point just by restricting its use to primitive recursion.

This result is technically not surprising at all since MLTT is computationally weaker than PCF. However, it has a conceptually appealing point: The *low-level* computational processes of viable or JPA-computable dynamic strategies may be regarded reasonably as *realizers*, and thus this development seems suitable from the point of realizability interpretation [179, 181].

Moreover, one may say that we have established a mathematical model of higher-order computation that has enough structures to embody constructive logic and mathematics. In other words, our game-semantic approach might be useful for a semantic, analytic study of various principles in constructive logic and mathematics [181].

6.3 Classical, Intuitionistic and Linear Logics

Finally, we may certainly apply the recipe in Section 2.4 to the game semantics of MLTT in Chapter 5 in order to give a systematic account on classical, intuitionistic and linear *predicate* logics though it is still a rough idea, not an established theory. As already seen in Section 5.3.4, our game semantics of MLTT seems to be able to naturally model intuitionistic linear reasoning; let us here focus on classical reasoning.

Let us see reasonability of the *why not* construction ? (Definition 2.4.11) for the game semantics of MLTT by considering how it models a well-known phenomenon: the incompatibility of $\boldsymbol{\Sigma}$-types and classical reasoning [86]. The problem is the second projection $v_B \in \mathit{Tm}(?\Sigma(A, B), ?B\{p(B)\})$ since in general the first projection $p(A) : ?\Sigma(A, B) \Rightarrow ?A$ may consist of several different v-strategies on A, and thus we cannot select just one component lp-game of B as the codomain of the second projection v_B.

This problem does not occur if there is no non-constant dlp-game; however, even such a *propositional* case has another problem: An lp-game of the form $?(A \oplus B)$ does not allow Player to reselect a component game (i.e., A or B), and thus, e.g., the *law of excluded middle (LEM)* is not true in the BCC $\mathcal{RICPG}$ in Section 5.3.4. Thus, it seems that we have to adopt the *weak* sum $+$ to retain classical reasoning.

Chapter 7

Conclusion and Future Work

7.1 Conclusion of the Thesis

The present thesis has extended conventional game semantics to capture some central aspects of logic and computation in a natural, mathematical, syntax-independent manner. Therefore, to a certain degree, we may say that games may be regarded as *mathematics of logic and computation* as the title of the thesis indicates.

In Chapter 3, we have generalized the standard semantics of computation both in the categorical and the game-semantic levels to capture dynamics and intensionality of computation. Specifically, we have first studied in depth algebraic structures of dynamic games and strategies, and obtained a CCBoC of them. And then, we have given the first game semantics that satisfies the DCP for FPCF, capturing dynamics and intensionality of computation. In this manner, we have overcome the static, extensional nature of existing game semantics. On the other hand, it remains open how to extend this framework to more complex programming languages.

In Chapter 4, we have given an *intrinsic*, *non-axiomatic*, *non-inductive* notion of 'effective computability' in game semantics and proved that it is *Turing complete.* By the interactive nature of game semantics, this game-semantic approach naturally accommodates *higher-order computation*, which extends the fundamental analysis of classical computability by Turing to non-classical one. In this sense, our game-semantic approach can be seen as a mathematical model of computation in its own right, in contrast to conventional game semantics which has been primarily employed for full abstraction (i.e., to characterize programs up to observational equivalence).

In Chapter 5, we have developed a game semantics of MLTT equipped with $\mathsf{1}$-, $\mathsf{0}$-, N-, $\mathsf{\Pi}$-, $\mathsf{\Sigma}$- and Id-types as well as a cumulative hierarchy of universes. Notably, we have modeled the syntax by a novel variant of games, called *p-games*, not families or lists of games; in this sense, we have given a *truly game semantic* model of MLTT.

Finally in Chapter 6, we have combined these three developments into a single mathematical framework that captures dynamics and intensionality of computation, higher-order computability and dependent types. On the other hand, we have also witnessed that our framework is intrinsically intuitionistic, not classical.

7.2 Future Work of the Thesis

There are various directions for further work, and we have already mentioned them at the end of the main chapters. Finally, we repeat some of the most imminent ones.

First, it is definitely immediate future work to apply the framework of dynamic game semantics in Chapter 3 to other logics and computations. Note that the present formulation of the hiding operation is tailored to a specific operational semantics of FPCF; thus, this problem is not a trivial one. In particular, since the hiding operation can be refined to the 'move-wise' fashion, we are interested in modeling finer computational calculi such as *explicit substitution* [158] and the *differential λ-calculus* [55]. We believe that this direction would lead to a deeper understanding of dynamics and intensionality of logic and computation.

It is also important to extend the game-semantic model of PCF-computation. We are particularly concerned with extending it to *non-innocent* strategies so that we may cover computation with *states*. In addition, we are interested in employing the framework as a mathematical foundation to investigate the computational natures of various principles in constructive mathematics. Furthermore, it would be fruitful to employ the'low-level computational processes of viable and/or JPA-computable dynamic strategies as a measure of computational complexity for programming.

Finally, it remains open to improve our game-semantic interpretation of Id-types. Note that the result in Chapter 5 is unsatisfactory in the point that it validates UIP since the syntax does not [93]. The unpublished paper [190] has addressed this problem by incorporating the structure of *groupoinds* but based on the simpler version of our game semantics of MLTT. Thus, we have to see how this work can be adjusted to the game semantics of the present thesis. Moreover, it is a challenging, yet interesting, problem to generalize the groupoid structure to *ω-groupoids* in order to model the infinite hierarchy of Id-types in HoTT. This would in particular shed new light on connections between computation and topology via the common internal language, viz., HoTT; it would lead to new topological structures in computation as well as novel computational natures in topology.

Bibliography

[1] Samson Abramsky. Computational interpretations of linear logic. *Theor. Comput. Sci.*, 111(1&2):3–57, 1993.

[2] Samson Abramsky. Information, processes and games. *J. Benthem van & P. Adriaans (Eds.), Philosophy of Information*, pages 483–549, 2008.

[3] Samson Abramsky. Axioms for definability and full completeness. *arXiv preprint arXiv:1401.4735*, 2014.

[4] Samson Abramsky. Intensionality, Definability and Computation. In *Johan van Benthem on Logic and Information Dynamics*, pages 121–142. Springer, 2014.

[5] Samson Abramsky et al. Semantics of Interaction: An Introduction to Game Semantics. *Semantics and Logics of Computation, Publications of the Newton Institute*, pages 1–31, 1997.

[6] Samson Abramsky, Kohei Honda, and Guy McCusker. A fully abstract game semantics for general references. In *Logic in Computer Science, 1998. Proceedings. Thirteenth Annual IEEE Symposium on*, pages 334–344. IEEE, 1998.

[7] Samson Abramsky and Radha Jagadeesan. Games and Full Completeness for Multiplicative Linear Logic. *The Journal of Symbolic Logic*, 59(02):543–574, 1994.

[8] Samson Abramsky and Radha Jagadeesan. A Game Semantics for Generic Polymorphism. *Annals of Pure and Applied Logic*, 133(1):3–37, 2005.

[9] Samson Abramsky, Radha Jagadeesan, and Pasquale Malacaria. Full Abstraction for PCF. *Information and Computation*, 163(2):409–470, 2000.

[10] Samson Abramsky, Radha Jagadeesan, and Matthijs Vákár. Games for Dependent Types. In *Automata, Languages, and Programming*, pages 31–43. Springer, 2015.

[11] Samson Abramsky and Achim Jung. Domain theory. In *Handbook of logic in computer science*. Oxford University Press, 1994.

[12] Samson Abramsky and Guy McCusker. Linearity, sharing and state: a fully abstract game semantics for Idealized Algol with active expressions. In *Algol-like languages*, pages 297–329. Springer, 1997.

[13] Samson Abramsky and Guy McCusker. Call-by-value games. In *Computer Science Logic*, pages 1–17. Springer, 1998.

[14] Samson Abramsky and Guy McCusker. Game Semantics. In *Computational logic*, pages 1–55. Springer, 1999.

[15] Samson Abramsky and P-A Mellies. Concurrent games and full completeness. In *Logic in Computer Science, 1999. Proceedings. 14th Symposium on*, pages 431–442. IEEE, 1999.

[16] Peter Aczel. The type theoretic interpretation of constructive set theory. *Studies in Logic and the Foundations of Mathematics*, 96:55–66, 1978.

[17] Peter Aczel, Seppo Miettinen, and Jouko Vaananen. The strength of Martin-Löf's intuitionistic type theory with one universe. 1984.

[18] Roberto M Amadio and Pierre-Louis Curien. *Domains and lambda-calculi*. Number 46. Cambridge University Press, 1998.

[19] Hajnal Andréka, István Németi, and Ildikó Sain. Algebraic logic. In *Handbook of philosophical logic*, pages 133–247. Springer, 2001.

[20] Michael J Beeson. *Foundations of constructive mathematics: Metamathematical studies*, volume 6. Springer Science & Business Media, 2012.

[21] Stefano Berardi, Thierry Coquand, and Susumu Hayashi. Games with 1-backtracking. *Annals of Pure and Applied Logic*, 161(10):1254–1269, 2010.

[22] Gérard Berry and Pierre-Louis Curien. Sequential algorithms on concrete data structures. *Theoretical Computer Science*, 20(3):265–321, 1982.

[23] Gavin M Bierman. What is a categorical model of intuitionistic linear logic? In *International Conference on Typed Lambda Calculi and Applications*, pages 78–93. Springer, 1995.

[24] RS Bird. Introduction to functional programming in haskell. international series in computer science, 1998.

[25] Andreas Blass. A game semantics for linear logic. *Annals of Pure and Applied logic*, 56(1):183–220, 1992.

[26] Andreas Blass and Yuri Gurevich. Algorithms: a quest for absolute definitions. *Church?s Thesis After*, 70:24–57, 2006.

[27] Valentin Blot. *Game semantics and realizability for classical logic.* PhD thesis, Ecole normale supérieure de lyon-ENS LYON, 2014.

[28] Valentin Blot. Realizability for peano arithmetic with winning conditions in hon games. *Annals of Pure and Applied Logic*, 168(2):254–277, 2017.

[29] William Blum and CHL Ong. A Concrete Presentation of Game Semantics, 2008.

[30] Susanne Bobzien. Ancient logic. 2006.

[31] Antonio Bucciarelli. Another approach to sequentiality: Kleene's unimonotone functions. In *Mathematical Foundations of Programming Semantics*, pages 333–358. Springer, 1994.

[32] Chen Chung Chang and H Jerome Keisler. *Model theory*, volume 73. Elsevier, 1990.

[33] Noam Chomsky. Three models for the description of language. *IRE Transactions on information theory*, 2(3):113–124, 1956.

[34] Alonzo Church. An unsolvable problem of elementary number theory. *American journal of mathematics*, 58(2):345–363, 1936.

[35] Alonzo Church. Review of Turing 1936. *The journal of symbolic logic*, 2:42–43, 1937.

[36] Alonzo Church. A formulation of the simple theory of types. *The journal of symbolic logic*, 5(02):56–68, 1940.

[37] Pierre Clairambault and Russ Harmer. Totality in Arena Games. *Annals of pure and applied logic*, 161(5):673–689, 2010.

[38] RL Constable, SF Allen, HM Bromley, WR Cleaveland, JF Cremer, RW Harper, DJ Howe, TB Knoblock, NP Mendler, P Panangaden, et al. Implementing mathematics with the nuprl proof development system. 1986.

[39] Catarina Coquand. A realizability interpretation of martin-löf's type theory. *Twenty-Five Years of Constructive Type Theory*, 1998.

[40] Thierry Coquand. Computational content of classical logic. *Semantics and logics of computation*, 14:33, 1997.

[41] Roy L Crole. *Categories for Types.* Cambridge University Press, 1993.

[42] Pierre-Louis Curien. Abstract Böhm Trees. *Mathematical Structures in Computer Science*, 8(06):559–591, 1998.

[43] Pierre-Louis Curien. Notes on Game Semantics. *From the author?s web page*, 2006.

[44] Pierre-Louis Curien. Definability and full abstraction. *Electronic Notes in Theoretical Computer Science*, 172:301–310, 2007.

[45] Haskell B Curry. Grundlagen der kombinatorischen logik. *American journal of mathematics*, 52(4):789–834, 1930.

[46] Nigel Cutland. *Computability: An Introduction to Recursive Function Theory.* Cambridge university press, 1980.

[47] Vincent Danos, Hugo Herbelin, and Laurent Regnier. Game semantics and abstract machines. In *Logic in Computer Science, 1996. LICS'96. Proceedings., Eleventh Annual IEEE Symposium on*, pages 394–405. IEEE, 1996.

[48] Vincent Danos and Laurent Regnier. Head linear reduction. *Unpublished*, 2004.

[49] Antony JT Davie. *Introduction to Functional Programming Systems Using Haskell*, volume 27. Cambridge University Press, 1992.

[50] Richard Dedekind and Wooster Woodruff Beman. *Essays on the Theory of Numbers: I. Continuity and Irrational Numbers, II. The Nature and Meaning of Number.* Open court publishing Company, 1901.

[51] Peter J Denning. What is computation. *Ubiquity*, 2010.

[52] Aleksandar Dimovski, Dan R Ghica, and Ranko Lazić. Data-abstraction Refinement: A Game Semantic Approach. In *International Static Analysis Symposium*, pages 102–117. Springer, 2005.

[53] Peter Dybjer. Internal Type Theory. In *Types for Proofs and Programs*, pages 120–134. Springer, 1996.

[54] Peter Dybjer and Erik Palmgren. Intuitionistic type theory. *Stanford Encyclopedia of Philosophy*, 2016.

[55] Thomas Ehrhard and Laurent Regnier. The differential lambda-calculus. *Theoretical Computer Science*, 309(1):1–41, 2003.

[56] Herbert Enderton and Herbert B Enderton. *A mathematical introduction to logic*. Academic press, 2001.

[57] Solomon Feferman. A language and axioms for explicit mathematics. In *Algebra and logic*, pages 87–139. Springer, 1975.

[58] Solomon Feferman. Predicativity. 2005.

[59] Walter Felscher. Dialogues as a foundation for intuitionistic logic. In *Handbook of philosophical logic*, pages 341–372. Springer, 1986.

[60] Hugo Férée. Game semantics approach to higher-order complexity. *Journal of Computer and System Sciences*, 87:1–15, 2017.

[61] Gottlob Frege. Begriffsschrift, a formula language, modeled upon that of arithmetic, for pure thought. *From Frege to Gödel: A source book in mathematical logic*, 1931:1–82, 1879.

[62] RO Gandy. Dialogues, Blass games and sequentiality for objects of finite type. *Unpublished manuscript*, 1993.

[63] Robin Gandy. Church's thesis and principles for mechanisms. *Studies in Logic and the Foundations of Mathematics*, 101:123–148, 1980.

[64] Gerhard Gentzen. Untersuchungen über das logische schließen. i. *Mathematische zeitschrift*, 39(1):176–210, 1935.

[65] Gerhard Gentzen. Untersuchungen über das logische schließen. I. *Mathematische zeitschrift*, 39(1):176–210, 1935.

[66] Gerhard Gentzen. Untersuchungen über das logische schließen. II. *Mathematische Zeitschrift*, 39(1):405–431, 1935.

[67] Gerhard Gierz, Karl Heinrich Hofmann, Klaus Keimel, Jimmie D Lawson, Michael Mislove, and Dana S Scott. *Continuous lattices and domains*, volume 93. Cambridge University Press, 2003.

[68] Seymour Ginsburg, Sheila A Greibach, and Michael A Harrison. Stack automata and compiling. *Journal of the ACM (JACM)*, 14(1):172–201, 1967.

[69] Jean-Yves Girard. *Interprétation fonctionelle et élimination des coupures de l?arithmétique d?ordre supérieur*. PhD thesis, PhD thesis, Université Paris VII, 1972.

[70] Jean-Yves Girard. Linear logic. *Theoretical computer science*, 50(1):1–101, 1987.

[71] Jean-Yves Girard. Geometry of interaction I: Interpretation of System F. *Studies in Logic and the Foundations of Mathematics*, 127:221–260, 1989.

[72] Jean-Yves Girard. Geometry of interaction II: Deadlock-free algorithms. In *COLOG-88*, pages 76–93. Springer, 1990.

[73] Jean-Yves Girard. Geometry of interaction III: accommodating the additives. *London Mathematical Society Lecture Note Series*, pages 329–389, 1995.

[74] Jean-Yves Girard. Linear logic: its syntax and semantics. *London Mathematical Society Lecture Note Series*, pages 1–42, 1995.

[75] Jean-Yves Girard. Geometry of interaction IV: the feedback equation. In *Logic Colloquium*, volume 3, pages 76–117. Citeseer, 2003.

[76] Jean-Yves Girard. Geometry of interaction V: logic in the hyperfinite factor. *Theoretical Computer Science*, 412(20):1860–1883, 2011.

[77] Jean-Yves Girard. Geometry of interaction VI: a blueprint for transcendental syntax. *preprint*, 2013.

[78] Jean-Yves Girard, Paul Taylor, and Yves Lafont. *Proofs and Types*, volume 7. Cambridge University Press Cambridge, 1989.

[79] Kurt Gödel. *Uber die Vollstandigkeit des Logikkalkuls*. PhD thesis, PhD thesis, Vienna, 1929.

[80] Kurt Gödel. Über formal unentscheidbare sätze der principia mathematica und verwandter systeme i. *Monatshefte für mathematik und physik*, 38(1):173–198, 1931.

[81] William Edward Greenland. *Game Semantics for Region Analysis*. PhD thesis, University of Oxford, 2005.

[82] Carl A Gunter. *Semantics of Programming Languages: Structures and Techniques*. MIT press, 1992.

[83] Yuri Gurevich. Abstract state machines: An overview of the project. *Foundations of Information and Knowledge Systems*, pages 6–13, 2004.

[84] Matthew Hague, Andrzej S Murawski, C-HL Ong, and Olivier Serre. Collapsible pushdown automata and recursion schemes. In *Logic in Computer Science, 2008. LICS'08. 23rd Annual IEEE Symposium on*, pages 452–461. IEEE, 2008.

[85] Chris Hankin. Lambda calculi: A guide for the perplexed. 1994.

[86] Hugo Herbelin. On the degeneracy of sigma-types in presence of computational classical logic. In *TLCA*, pages 209–220. Springer, 2005.

[87] Arend Heyting. Die formalen regeln der intuitionistischen logik. *Sitzungsberichte der Preussischen Akademie der Wissenshaften, physikalisch-mathematische Klasse*, 42:158–169, 1930.

[88] David Hilbert. The foundations of mathematics. 1927.

[89] Barnaby P Hilken. Towards a proof theory of rewriting: the simply typed 2λ-calculus. *Theoretical Computer Science*, 170(1-2):407–444, 1996.

[90] Charles Antony Richard Hoare. Communicating sequential processes. In *The origin of concurrent programming*, pages 413–443. Springer, 1978.

[91] Wilfrid Hodges. *Model theory*, volume 42. Cambridge University Press, 1993.

[92] Martin Hofmann. Syntax and semantics of dependent types. In *Extensional Constructs in Intensional Type Theory*, pages 13–54. Springer, 1997.

[93] Martin Hofmann and Thomas Streicher. The groupoid interpretation of type theory. *Twenty-five years of constructive type theory (Venice, 1995)*, 36:83–111, 1998.

[94] Kohei Honda and Nobuko Yoshida. Game theoretic analysis of call-by-value computation. In *Automata, Languages and Programming*, pages 225–236. Springer, 1997.

[95] John E Hopcroft, Rajeev Motwani, and Jeffrey D Ullman. Introduction to automata theory, languages, and computation. *Acm Sigact News*, 32(1):60–65, 2001.

[96] John E. Hopcroft and Jeffrey D. Ullman. Nonerasing stack automata. *Journal of Computer and System Sciences*, 1(2):166–186, 1967.

[97] Dominic Hughes. *Hypergame semantics: full completeness for system F.* PhD thesis, D. Phil. thesis, Oxford University, 2000.

[98] Hagen Huwig and Axel Poigné. A note on inconsistencies caused by fixpoints in a cartesian closed category. *Theoretical Computer Science*, 73(1):101–112, 1990.

[99] J Martin E Hyland and C-H Luke Ong. Fair games and full completeness for multiplicative linear logic without the mix-rule. *preprint*, 190, 1993.

[100] J Martin E Hyland and C-HL Ong. On full abstraction for PCF: I, II, and III. *Information and computation*, 163(2):285–408, 2000.

[101] Martin Hyland. Game Semantics. *Semantics and logics of computation*, 14:131, 1997.

[102] Bart Jacobs. *Categorical Logic and Type Theory*, volume 141. Elsevier, 1999.

[103] Giorgi Japaridze. Introduction to computability logic. *Annals of Pure and Applied Logic*, 123(1-3):1–99, 2003.

[104] Peter T Johnstone. *Sketches of an elephant: A topos theory compendium*, volume 2. Oxford University Press, 2002.

[105] Peter T Johnstone. *Topos theory.* Courier Corporation, 2014.

[106] SC Kleene. Unimonotone functions of finite types (recursive functionals and quantifiers of finite types revisited IV). *Recursion Theory*, 42:119–138, 1985.

[107] Stephen Cole Kleene. Introduction to metamathematics. 1952.

[108] Stephen Cole Kleene. Recursive functionals and quantifiers of finite types I. *Transactions of the American Mathematical Society*, 91(1):1–52, 1959.

[109] Stephen Cole Kleene. Recursive functionals and quantifiers of finite types II. *Transactions of the American Mathematical Society*, 108(1):106–142, 1963.

[110] Stephen Cole Kleene. Recursive functionals and quantifiers of finite types revisited I. *Studies in Logic and the Foundations of Mathematics*, 94:185–222, 1978.

[111] Stephen Cole Kleene. Recursive functionals and quantifiers of finite types revisited II. *Studies in Logic and the Foundations of Mathematics*, 101:1–29, 1980.

[112] Stephen Cole Kleene. Recursive functionals and quantifiers of finite types revisited III. *Studies in Logic and the Foundations of Mathematics*, 109:1–40, 1982.

[113] Stephen Cole Kleene. Recursive functionals and quantifiers of finite types revisited. V. *Transactions of the American Mathematical Society*, 325(2):593–630, 1991.

[114] Teodor Knapik, Damian Niwiński, and Paweł Urzyczyn. Higher-order pushdown trees are easy. In *International Conference on Foundations of Software Science and Computation Structures*, pages 205–222. Springer, 2002.

[115] Dexter C Kozen. *Theory of Computation*. Springer Science & Business Media, 2006.

[116] Dexter C Kozen. *Automata and Computability*. Springer Science & Business Media, 2012.

[117] James Laird. Full Abstraction for Functional Languages with Control. In *Logic in Computer Science, 1997. LICS'97. Proceedings., 12th Annual IEEE Symposium on*, pages 58–67. IEEE, 1997.

[118] Joachim Lambek and Philip J Scott. *Introduction to Higher-order Categorical Logic*, volume 7. Cambridge University Press, 1988.

[119] Olivier Laurent. Polarized games. *Annals of Pure and Applied Logic*, 130(1-3):79–123, 2004.

[120] Olivier Laurent. Game semantics for first-order logic. *arXiv preprint arXiv:1009.4400*, 2010.

[121] Øystein Linnebo. Platonism in the philosophy of mathematics. 2009.

[122] John Longley and Dag Normann. *Higher-Order Computability.* Springer, 2015.

[123] Saunders MacLane and Ieke Moerdijk. *Sheaves in geometry and logic: A first introduction to topos theory.* Springer Science & Business Media, 2012.

[124] Per Martin-Löf. An intuitionistic theory of types: Predicative part. *Studies in Logic and the Foundations of Mathematics*, 80:73–118, 1975.

[125] Per Martin-Löf. Constructive mathematics and computer programming. *Studies in Logic and the Foundations of Mathematics*, 104:153–175, 1982.

[126] Per Martin-Löf. *Intuitionistic Type Theory: Notes by Giovanni Sambin of a series of lectures given in Padova, June 1980.* 1984.

[127] Per Martin-Löf. An intuitionistic theory of types. *Twenty-five years of constructive type theory*, 36:127–172, 1998.

[128] GUY McCUSKER. Games and definability for fpc. *Bulletin of Symbolic Logic*, 3(3):347–362, 1997.

[129] Guy McCusker. *Games and Full Abstraction for a Functional Metalanguage with Recursive Types.* Springer Science & Business Media, 1998.

[130] Colin McLarty. *Elementary categories, elementary toposes.* Clarendon Press, 1992.

[131] Paul-André Mellies. Asynchronous games 4: A fully complete model of propositional linear logic. In *LiCS 2005–Logic in Computer Science*, pages 386–395. IEEE, 2005.

[132] Paul-André Mellies. Axiomatic rewriting theory I: A diagrammatic standardization theorem. In *Processes, Terms and Cycles: steps on the road to infinity*, pages 554–638. Springer, 2005.

[133] Robin Milner. Software science: From virtual to reality. *Bulletin of EATCS*, 87.

[134] Robin Milner. A calculus of communicating systems. *Lecture Notes in Comput. Sci. 92*, 1980.

[135] Gerd Mitschke. The standardization theorem for λ-calculus. *Mathematical Logic Quarterly*, 25(1-2):29–31, 1979.

[136] Eugenio Moggi. Notions of computation and monads. *Information and computation*, 93(1):55–92, 1991.

[137] Yiannis N Moschovakis. On founding the theory of algorithms. *Truth in mathematics*, pages 71–104, 1998.

[138] John Myhill. Constructive set theory. *The Journal of Symbolic Logic*, 40(3):347–382, 1975.

[139] Hanno Nickau. Hereditarily Sequential Functionals. In *Logical Foundations of Computer Science*, pages 253–264. Springer, 1994.

[140] Bengt Nordström, Kent Petersson, and Jan M Smith. Programming in martin-löfs type theory, volume 7 of international series of monographs on computer science, 1990.

[141] Ulf Norell. *Towards a Practical Programming Language Based on Dependent Type Theory*, volume 32. Citeseer, 2007.

[142] Piergiorgio Odifreddi. *Classical Recursion Theory: The Theory of Functions and Sets of Natural Numbers*, volume 125. Elsevier, 1992.

[143] C-H Luke Ong. Normalisation by traversals. *arXiv preprint arXiv:1511.02629*, 2015.

[144] C-HL Ong. On Model-checking Trees Generated by Higher-order Recursion Schemes. In *21st Annual IEEE Symposium on Logic in Computer Science (LICS'06)*, pages 81–90. IEEE, 2006.

[145] Joël Ouaknine. *A Two-Dimensional Extension of Lambek's Categorical Proof Theory.* PhD thesis, McGill University, Montréal, 1997.

[146] Erik Palmgren. On universes in type theory. In *Twenty Five Years of Constructive Type Theory*, volume 36, pages 191–204. Oxford University Press, 1998.

[147] Erik Palmgren and Viggo Stoltenberg-Hansen. Domain interpretations of martin-löf's partial type theory. *Annals of Pure and Applied Logic*, 48(2):135–196, 1990.

[148] Giuseppe Peano. Arithmetices principia, nova methodo exposita, 1899. *English translation in [51]*, pages 83–97, 1879.

[149] Giuseppe Peano. *Arithmetices principia: nova methodo exposita*. Fratres Bocca, 1889.

[150] Andrew M Pitts. Categorical logic. In *Handbook of logic in computer science*, pages 39–123. Oxford University Press, 2001.

[151] Gordon D. Plotkin. LCF considered as a programming language. *Theoretical computer science*, 5(3):223–255, 1977.

[152] Gordon D Plotkin. A structural approach to operational semantics. 1981.

[153] Emil L Post. Formal reductions of the general combinatorial decision problem. *American journal of mathematics*, 65(2):197–215, 1943.

[154] Michael Rathjen. The constructive hilbert program and the limits of martin-löf type theory. In *Logicism, Intuitionism, and Formalism*, pages 397–433. Springer, 2009.

[155] Estéban Requena, Paul LORENZEN, and Kuno LORENZ. Dialogische logik, 1980.

[156] Bernhard Reus. Realizability models for type theories. *Electronic Notes in Theoretical Computer Science*, 23(1):128–158, 1999.

[157] Hartley Rogers and H Rogers. *Theory of Recursive Functions and Effective Computability*, volume 5. McGraw-Hill New York, 1967.

[158] Kristoffer Høgsbro Rose. *Explicit Substitution: Tutorial & Survey.* Computer Science Department, 1996.

[159] Moses Schönfinkel. Über die bausteine der mathematischen logik. *Mathematische Annalen*, 92(3):305–316, 1924.

[160] Helmut Schwichtenberg and Stanley S Wainer. *Proofs and computations*. Cambridge University Press, 2011.

[161] Dana Scott. *Outline of a mathematical theory of computation*. Oxford University Computing Laboratory, Programming Research Group, 1970.

[162] Dana Scott. Data types as lattices. *SIAM Journal on computing*, 5(3):522–587, 1976.

[163] Dana S Scott. A type-theoretical alternative to ISWIM, CUCH, OWHY. *Theoretical Computer Science*, 121(1):411–440, 1993.

[164] Dana S Scott and Christopher Strachey. *Toward a mathematical semantics for computer languages*, volume 1. Oxford University Computing Laboratory, Programming Research Group, 1971.

[165] Robert AG Seely. *Linear logic, *-autonomous categories and cofree coalgebras*. Ste. Anne de Bellevue, Quebec: CEGEP John Abbott College, 1987.

[166] Robert AG Seely. Modelling computations: A 2-categorical framework. In *LICS*, pages 65–71, 1987.

[167] Stewart Shapiro. Classical logic. 2000.

[168] Joseph R Shoenfield. *Mathematical Logic*, volume 21. Addison-Wesley Reading, 1967.

[169] Michael Sipser. *Introduction to the Theory of Computation*. Cengage Learning, 2012.

[170] Jan Smith. An interpretation of martin-löf's type theory in a type-free theory of propositions. *The Journal of symbolic logic*, 49(03):730–753, 1984.

[171] Robert I Soare. The history and concept of computability. *Handbook of computability theory*, 140:3–36, 1999.

[172] Morten Heine Sørensen and Pawel Urzyczyn. *Lectures on the Curry-Howard isomorphism*, volume 149. Elsevier, 2006.

[173] Viggo Stoltenberg-Hansen, Ingrid Lindström, and Edward R Griffor. *Mathematical theory of domains*, volume 22. Cambridge University Press, 1994.

[174] Thomas Streicher. *Investigations into Intensional Type Theory*. 1993.

[175] Alfred Tarski. The concept of truth in the languages of the deductive sciences. *Prace Towarzystwa Naukowego Warszawskiego, Wydzial III Nauk Matematyczno-Fizycznych*, 34(13-172):198, 1933.

[176] Alfred Tarski. *Pojecie prawdy w jezykach nauk dedukcyjnych: la notion de la vérité dans les langages des sciences déductives*, volume 2. Nakladem Towarzystwa Naukowego Warszawskiego, 1933.

[177] COQ DEVELOPMENT TEAM et al. The Coq proof assistant reference manual. *TypiCal Project (formerly LogiCal)*, 2012.

[178] Anne S Troelstra. *Metamathematical investigation of intuitionistic arithmetic and analysis*, volume 344. Springer Science & Business Media, 1973.

[179] Anne Sjerp Troelstra et al. Realizability. 1998.

[180] Anne Sjerp Troelstra and Helmut Schwichtenberg. *Basic proof theory*. Number 43. Cambridge University Press, 2000.

[181] Anne Sjerp Troelstra and Dirk van Dalen. Constructivism in mathematics. Vol. I, volume 121 of. *Studies in Logic and the Foundations of Mathematics*, page 26, 1988.

[182] Anne Sjerp Troelstra and Dirk Van Dalen. *Constructivism in Mathematics*, volume 2. Elsevier, 2014.

[183] Alan Mathison Turing. On computable numbers, with an application to the entscheidungsproblem. *J. of Math*, 58(345-363):5, 1936.

[184] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. `https://homotopytypetheory.org/book`, Institute for Advanced Study, 2013.

[185] Jean Van Heijenoort. *From Frege to Gödel: a source book in mathematical logic, 1879-1931*, volume 9. Harvard University Press, 1967.

[186] Jaap Van Oosten. *Realizability: an introduction to its categorical side*, volume 152. Elsevier, 2008.

[187] Alfred North Whitehead and Bertrand Russell. *Principia mathematica*, volume 1. University Press, 1910.

[188] Glynn Winskel. *The Formal Semantics of Programming Languages: An Introduction.* MIT press, 1993.

[189] Norihiro Yamada. Game Semantics for Martin-Löf Type Theory. *arXiv preprint arXiv:1610.01669*, 2016.

[190] Norihiro Yamada. Game-theoretic investigation of intensional equalities. *arXiv preprint arXiv:1703.02015*, 2017.