Computing Laboratory

QUANTITATIVE MULTI-OBJECTIVE VERIFICATION FOR PROBABILISTIC SYSTEMS

Vojtěch Forejt
Marta Kwiatkowska
Gethin Norman
David Parker
Hongyang Qu

CL-RR-10-26

Oxford University Computing Laboratory
Wolfson Building, Parks Road, Oxford OX1 3QD

Quantitative Multi-Objective Verification for Probabilistic Systems

Vojtěch Forejt[1], Marta Kwiatkowska[1],
Gethin Norman[2], David Parker[1], and Hongyang Qu[1]

[1] Oxford University Computing Laboratory, Parks Road, Oxford, OX1 3QD, UK
[2] School of Computing Science, University of Glasgow, Glasgow, G12 8RZ, UK

Abstract. We present a verification framework for analysing multiple quantitative objectives of systems that exhibit both nondeterministic and stochastic behaviour. These systems are modelled as probabilistic automata, enriched with cost or reward structures that capture, for example, energy usage or performance metrics. Quantitative properties of these models are expressed in a specification language that incorporates probabilistic safety and liveness properties, expected total cost or reward, and supports multiple objectives of these types. We propose and implement an efficient verification framework for such properties and then present two distinct applications of it: firstly, *controller synthesis* subject to multiple quantitative objectives; and, secondly, quantitative *compositional verification*. The practical applicability of both approaches is illustrated with experimental results from several large case studies.

1 Introduction

Automated formal verification techniques such as model checking have proved to be an effective way of establishing rigorous guarantees about the correctness of real-life systems. In many instances, though, it is important to also take stochastic behaviour of these systems into account. This might be because of the presence of components that are prone to failure, because of unpredictable behaviour, e.g. of lossy communication media, or due to the use of randomisation, e.g. in distributed communication protocols such as Bluetooth.

Probabilistic verification offers techniques to automatically check quantitative properties of such systems. Models, typically labelled transition systems augmented with probabilistic information, are verified against properties specified in probabilistic extensions of temporal logics. Examples of such properties include "the probability of both devices failing within 24 hours is less than 0.001" or "with probability at least 0.99, all message packets are sent successfully".

In this paper, we focus on verification techniques for *probabilistic automata* (PAs) [23], which model both nondeterministic and probabilistic behaviour. We augment these models with one or more *reward* structures that assign real values to certain transitions of the model. In fact, these can associate a notion of either *cost* or *reward* with the executions of the model and capture a wide range of quantitive measures of system behaviour, for example "number of time steps", "energy usage" or "number of messages successfully sent".

Properties of PAs can be specified using well-known temporal logics such as PCTL, LTL or PCTL* [6] and extensions for reward-based properties [1]. The corresponding verification problems can be executed reasonably efficiently and are implemented in tools such as PRISM, LiQuor and RAPTURE.

A natural extension of these techniques is to consider *multiple objectives*. For example, rather than verifying two separate properties such as "message loss occurs with probability at most 0.001" and "the expected total energy consumption is below 50 units", we might ask whether it is possible to satisfy both properties *simultaneously*, or to investigate the *trade-off* between the two objectives as some parameters of the system are varied.

In this paper, we consider verification problems for probabilistic automata on properties with *multiple*, *quantitative* objectives. We define a language that expresses Boolean combinations of probabilistic ω-regular properties (which subsumes e.g. LTL) and expected total reward measures. We then present, for properties expressed in this language, techniques both to *verify* that a property holds for all adversaries (strategies) of a PA and to *synthesise* an adversary of a PA under which a property holds. We also consider *numerical* queries, which yield an optimal value for one objective, subject to constraints imposed on one or more other objectives. This is done via reduction to a linear programming problem, which can be solved efficiently. It takes time polynomial in the size of the model and doubly exponential in the size of the property (for LTL objectives), i.e. the same as for the single-objective case [14].

Multi-criteria optimisation for PAs or, equivalently, Markov decision processes (MDPs) is well studied in operations research [13]. More recently, the topic has also been considered from a probabilistic verification point of view [12,16,9]. In [16], ω-regular properties are considered, but not rewards which, as illustrated by the examples above, offer an additional range of useful properties. In, [12] *discounted* reward properties are used. In practice, though, a large class of properties, such as "expected total time for algorithm completion" are not accurately captured when using discounting. Finally, [9] handles a complementary class of long-run average reward properties. All of [12,16,9] present algorithms and complexity results for verifying properties and approximating Pareto curves; however, unlike this paper, they do not consider implementations.

We implement our multi-objective verification techniques and present two distinct applications. Firstly, we illustrate the feasibility of performing *controller synthesis*. Secondly, we develop *compositional verification* methods based on assume-guarantee reasoning and quantitative multi-objective properties.

Controller synthesis. Synthesis, which aims to build correct-by-construction systems from formal specifications of their intended behaviour, represents a long-standing and challenging goal in the field of formal methods. One area where progress has been made is *controller synthesis*, a classic problem in control engineering which devises a strategy to control a system such that it meets its specification. We demonstrate the application of our techniques to synthesising controllers under multiple quantitative objectives, illustrating this with experimental results from a realistic model of a disk driver controller.

Compositional verification. Perhaps the biggest challenge to the practical applicability of formal verification is scalability. *Compositional verification* offers a powerful means to address this challenge. It works by breaking a verification problem down into manageable sub-tasks, based on the structure of the system being analysed. One particularly successful approach is the *assume-guarantee* paradigm, in which properties (*guarantees*) of individual system components are verified under *assumptions* about their environment. Desired properties of the combined system, which is typically too large to verify, are then obtained by combining separate verification results using proof rules. Compositional analysis techniques are of particular importance for probabilistic systems because verification is often more expensive than for non-probabilistic models.

Recent work in [21] presents an assume-guarantee framework for probabilistic automata, based on a reduction to the multi-objective techniques of [16]. However, the assumptions and guarantees in this framework are restricted to probabilistic safety properties. This limits the range of properties that can be verified and, more importantly, can be too restrictive to express assumptions of the environment. We use our techniques to introduce an alternative framework where assumptions and guarantees are the *quantitative multi-objective properties* defined in this paper. This adds the ability to reason compositionally about, for example, probabilistic liveness or expected rewards. To facilitate this, we also incorporate a notion of *fairness* into the framework. We have implemented the techniques and present results from compositional verification of several large case studies, including instances where it is infeasible non-compositionally.

Related work. Existing research on *multi-objective* analysis of MDPs and its relationship with this work has been discussed above. On the topic of *controller synthesis*, the problem of synthesising MDP adversaries to satisfy a temporal logic specification has been addressed several times, e.g. [3,7]. Also relevant is [11], which synthesises non-probabilistic automata based on quantitative measures. In terms of *compositional verification*, the results in this paper significantly extend the recent work in [21]. Other related approaches include: [8,15], which present specification theories for compositional reasoning about probabilistic systems; and [10], which presents a theoretical framework for compositional verification of quantitative (but not probabilistic) properties. None of [8,15,10], however, consider practical implementations of their techniques.

Contributions. In summary, the contributions of this paper are as follows:

- novel *multi-objective* verification techniques for probabilistic automata (and MDPs) that include both ω-regular and *expected total reward* properties;
- a corresponding method to generate optimal adversaries, with direct applicability to the problem of *controller synthesis* for these models;
- new *compositional verification* techniques for probabilistic automata using expressive quantitative properties for assumptions and guarantees.

This paper is an extended version, with proofs, of [18].

2 Background

We use $Dist(S)$ for the set of all discrete probability distributions over a set S, η_s for the point distribution on $s \in S$, and $\mu_1 \times \mu_2$ for the product distribution of $\mu_1 \in Dist(S_1)$ and $\mu_2 \in Dist(S_2)$, defined by $\mu_1 \times \mu_2((s_1, s_2)) = \mu_1(s_1) \cdot \mu_2(s_2)$.

2.1 Probabilistic automata (PAs)

Probabilistic automata [23] are a commonly used model for systems that exhibit both probabilistic and nondeterministic behaviour. PAs are very similar to Markov decision processes (MDPs).[1] For the purposes of verification (as in Section 3), they can often be treated identically; however, for compositional analysis (as in Section 4), the distinction becomes important.

Definition 1 (Probabilistic automata). *A* probabilistic automaton *(PA) is a tuple* $\mathcal{M}=(S, \overline{s}, \alpha_{\mathcal{M}}, \delta_{\mathcal{M}})$ *where* S *is a set of states,* $\overline{s} \in S$ *is an initial state,* $\alpha_{\mathcal{M}}$ *is an alphabet and* $\delta_{\mathcal{M}} \subseteq S \times \alpha_{\mathcal{M}} \times Dist(S)$ *is a probabilistic transition relation.*

In a state s of a PA $\mathcal{M}$, a *transition* $s \xrightarrow{a} \mu$, where a is an *action* and μ is a distribution over states, is available if $(s, a, \mu) \in \delta$. The selection of an available transition is nondeterministic and the subsequent choice of successor state is probabilistic, according to the distribution of the chosen transition.

A *path* is a sequence $\omega = s_0 \xrightarrow{a_0,\mu_0} s_1 \xrightarrow{a_1,\mu_1} \cdots$ where $s_0 = \overline{s}$, $s_i \xrightarrow{a_i} \mu_i$ is an available transition and $\mu_i(s_{i+1}) > 0$ for all $i \in \mathbb{N}$. We denote by $IPaths$ ($FPaths$) the set of all infinite (finite) paths. If ω is finite, $|\omega|$ denotes its length and $last(\omega)$ its last state. The trace, $tr(\omega)$, of ω is the sequence of actions $a_0 a_1 \ldots$ and we use $tr(\omega){\restriction_\alpha}$ to indicate the projection of such a trace onto an alphabet $\alpha \subseteq \alpha_{\mathcal{M}}$.

A *reward structure* for $\mathcal{M}$ is a mapping $\rho : \alpha_\rho \to \mathbb{R}_{>0}$ from some alphabet $\alpha_\rho \subseteq \alpha_{\mathcal{M}}$ to the positive reals. We sometimes write $\rho(a) = 0$ to indicate that $a \notin \alpha_\rho$. For an infinite path $\omega = s_0 \xrightarrow{a_0,\mu_0} s_1 \xrightarrow{a_1,\mu_1} \cdots$, the *total reward* for ω over ρ is $\rho(\omega) = \sum_{i \in \mathbb{N}, a_i \in \alpha_\rho} \rho(a_i)$.

An *adversary* of $\mathcal{M}$ is a function $\sigma : FPaths \to Dist(\alpha_{\mathcal{M}} \times Dist(S))$ such that, for a finite path ω, $\sigma(\omega)$ only assigns non-zero probabilities to action-distribution pairs (a, μ) for which $(last(\omega), a, \mu) \in \delta$. Employing standard techniques [20], an adversary σ induces a probability measure $Pr^\sigma_{\mathcal{M}}$ over $IPaths$. An adversary σ is *deterministic* if $\sigma(\omega)$ is a point distribution for all ω, *memoryless* if $\sigma(\omega)$ depends only on $last(\omega)$, and *finite-memory* if there are a finite number of memory configurations such that $\sigma(\omega)$ depends only on $last(\omega)$ and the current memory configuration, which is updated (possibly stochastically) when an action is performed. We let $Adv_{\mathcal{M}}$ denote the set of all adversaries for $\mathcal{M}$.

If $\mathcal{M}_i = (S_i, \overline{s}_i, \alpha_{\mathcal{M}_i}, \delta_{\mathcal{M}_i})$ for i=1, 2, then their *parallel composition*, denoted $\mathcal{M}_1 \| \mathcal{M}_2$, is given by the PA $(S_1 \times S_2, (\overline{s}_1, \overline{s}_2), \alpha_{M_1} \cup \alpha_{M_2}, \delta_{M_1 \| M_2})$ where $\delta_{M_1 \| M_2}$ is defined such that $(s_1, s_2) \xrightarrow{a} \mu_1 \times \mu_2$ if and only if one of the following holds: (i) $s_1 \xrightarrow{a} \mu_1$, $s_2 \xrightarrow{a} \mu_2$ and $a \in \alpha_{M_1} \cap \alpha_{M_2}$; (ii) $s_1 \xrightarrow{a} \mu_1$, $\mu_2 = \eta_{s_2}$ and $a \in \alpha_{M_1} \backslash \alpha_{M_2}$; or (iii) $s_2 \xrightarrow{a} \mu_2$, $\mu_1 = \eta_{s_1}$ and $a \in \alpha_{M_2} \backslash \alpha_{M_1}$.

[1] For MDPs, $\delta_{\mathcal{M}}$ in Definition 1 becomes a partial function $S \times \alpha_{\mathcal{M}} \to Dist(S)$.

When verifying systems of PAs composed in parallel, it is often essential to consider *fairness*. In this paper, we use a simple but effective notion of fairness called *unconditional fairness*, in which it is required that each process makes a transition infinitely often. For probabilistic automata, a natural approach to incorporating fairness (as taken in, e.g., [4,2]) is to restrict analysis of the system to a class of adversaries in which fair behaviour occurs with probability 1.

If $\mathcal{M} = \mathcal{M}_1 \| \dots \| \mathcal{M}_n$ is a PA comprising n components, then an (unconditionally) *fair path* of $\mathcal{M}$ is an infinite path $\omega \in \mathit{IPaths}$ in which, for each component M_i, there exists an action $a \in \alpha_{M_i}$ that appears infinitely often. A *fair adversary* σ of $\mathcal{M}$ is an adversary for which $\mathit{Pr}^{\sigma}_{\mathcal{M}}\{\omega \in \mathit{IPaths} \mid \omega \text{ is fair}\} = 1$. We let $\mathit{Adv}^{\text{fair}}_{\mathcal{M}}$ denote the set of fair adversaries of $\mathcal{M}$.

2.2 Verification of PAs

Throughout this section, let $\mathcal{M} = (S, \overline{s}, \alpha_{\mathcal{M}}, \delta_{\mathcal{M}})$ be a PA.

Definition 2 (Probabilistic predicates). *A* probabilistic predicate $[\phi]_{\sim p}$ *comprises an ω-regular property $\phi \subseteq (\alpha_\phi)^\omega$ over some alphabet $\alpha_\phi \subseteq \alpha_{\mathcal{M}}$, a relational operator $\sim \in \{<, \leqslant, >, \geqslant\}$ and a rational probability bound p. Satisfaction of $[\phi]_{\sim p}$ by $\mathcal{M}$, under adversary σ, denoted $\mathcal{M}, \sigma \models [\phi]_{\sim p}$, is defined as follows:*

$$\mathcal{M}, \sigma \models [\phi]_{\sim p} \Leftrightarrow \mathit{Pr}^{\sigma}_{\mathcal{M}}(\phi) \sim p \text{ where } \mathit{Pr}^{\sigma}_{\mathcal{M}}(\phi) \stackrel{\text{def}}{=} \mathit{Pr}^{\sigma}_{\mathcal{M}}(\{\omega \in \mathit{IPaths} \mid \mathit{tr}(\omega){\upharpoonright_{\alpha_\phi}} \in \phi\}).$$

Definition 3 (Reward predicates). *A* reward predicate $[\rho]_{\sim r}$ *comprises a reward structure $\rho : \alpha_\rho \to \mathbb{R}_{>0}$ over some alphabet $\alpha_\rho \subseteq \alpha_{\mathcal{M}}$, a relational operator $\sim \in \{<, \leqslant, >, \geqslant\}$ and a rational reward bound r. Satisfaction of $[\rho]_{\sim r}$ by $\mathcal{M}$, under adversary σ, denoted $\mathcal{M}, \sigma \models [\rho]_{\sim r}$, is defined as follows:*

$$\mathcal{M}, \sigma \models [\rho]_{\sim r} \;\Leftrightarrow\; \mathit{ExpTot}^{\sigma}_{\mathcal{M}}(\rho) \sim r \;\; \text{where} \;\; \mathit{ExpTot}^{\sigma}_{\mathcal{M}}(\rho) \;\stackrel{\text{def}}{=}\; \int_\omega \rho(\omega)\, d\mathit{Pr}^{\sigma}_{\mathcal{M}}.$$

Verification of PAs is based on quantifying over all adversaries. For example, we define satisfaction of probabilistic predicate $[\phi]_{\sim p}$ by $\mathcal{M}$, denoted $\mathcal{M} \models [\phi]_{\sim p}$, as:

$$\mathcal{M} \models [\phi]_{\sim p} \;\Leftrightarrow\; \forall \sigma \in \mathit{Adv}_{\mathcal{M}} \,.\, \mathcal{M}, \sigma \models [\phi]_{\sim p}\,.$$

In similar fashion, we can verify a multi-component PA $\mathcal{M}_1 \| \dots \| \mathcal{M}_n$ *under fairness* by quantifying only over fair adversaries:

$$\mathcal{M}_1 \| \dots \| \mathcal{M}_n \models_{\text{fair}} [\phi]_{\sim p} \;\Leftrightarrow\; \forall \sigma \in \mathit{Adv}^{\text{fair}}_{\mathcal{M}_1 \| \dots \| \mathcal{M}_n} \,.\, \mathcal{M}_1 \| \dots \| \mathcal{M}_n, \sigma \models [\phi]_{\sim p}\,.$$

Verifying whether $\mathcal{M}$ satisfies a probabilistic predicate $[\phi]_{\sim p}$ or reward predicate $[\rho]_{\sim r}$ can be done with, for example, the techniques in [14,1]. In the remainder of this section, we give further details of the case for ω-regular properties, since we need these later in the paper. We follow the approach of [1], which is based on the use of *deterministic Rabin automata* and *end components*.

An *end component* (EC) of $\mathcal{M}$ is a pair (S', δ') comprising a subset $S' \subseteq S$ of states and a probabilistic transition relation $\delta' \subseteq \delta$ that is strongly connected when restricted to S' and closed under probabilistic branching, i.e., $\{s \in S \mid \exists (s, a, \mu) \in \delta'\} \subseteq S'$ and $\{s' \in S \mid \mu(s') > 0 \text{ for some } (s, a, \mu) \in \delta\} \subseteq S'$. An EC (S', δ') is *maximal* if there is no EC (S'', δ'') such that $\delta' \subsetneq \delta''$.

A *deterministic Rabin automaton* (DRA) is a tuple $\mathcal{A} = (Q, \overline{q}, \alpha, \delta, Acc)$ of states Q, initial state $\overline{q}$, alphabet α, transition function $\delta : Q \times \alpha \to Q$, and acceptance condition $Acc = \{(L_i, K_i)\}_{i=1}^{k}$ with $L_i, K_i \subseteq Q$. Any infinite word $w \in (\alpha)^\omega$ has a unique corresponding run $\overline{q}\, q_1 q_2 \ldots$ through $\mathcal{A}$ and we say that $\mathcal{A}$ accepts w if the run contains, for some $1 \leqslant i \leqslant k$, finitely many states from L_i and infinitely many from K_i. For any ω-regular property $\phi \subseteq (\alpha_\phi)^\omega$ we can construct a DRA, say $\mathcal{A}_\phi$, over α_ϕ that accepts precisely ϕ.

Verification of $[\phi]_{\sim p}$ on $\mathcal{M}$ is done by constructing the *product* of $\mathcal{M}$ and $\mathcal{A}_\phi$, and then identifying *accepting end components*. The *product* $\mathcal{M} \otimes \mathcal{A}_\phi$ of $\mathcal{M}$ and DRA $\mathcal{A}_\phi = (Q, \overline{q}, \alpha_\mathcal{M}, \delta, \{(L_i, K_i)\}_{i=1}^{k})$ is the PA $(S \times Q, (\overline{s}, \overline{q}), \alpha_\mathcal{M}, \delta'_\mathcal{M})$ where for all $(s, a, \mu) \in \delta_M$ there is $((s, q), a, \mu') \in \delta'_\mathcal{M}$ such that $\mu'(s', q') = \mu(s')$ for $q' = \delta(q, a)$ and all $s' \in S$. An *accepting EC* for ϕ in $\mathcal{M} \otimes \mathcal{A}_\phi$ is an EC (S', δ') for which there exists an $1 \leqslant i \leqslant k$ such that the set of states S', when projected onto Q, contains some state from K_i, but no states from L_i. Verifying, for example, that $\mathcal{M} \models [\phi]_{\sim p}$, when $\sim \in \{<, \leqslant\}$, reduces to checking that $\mathcal{M} \otimes \mathcal{A}_\phi \models [\Diamond T]_{\sim p}$, where T is the union of states of accepting ECs for ϕ in $\mathcal{M} \otimes \mathcal{A}_\phi$.

Verification of such properties under fairness, e.g. checking $\mathcal{M} \models_{\text{fair}} [\phi]_{\sim p}$, can be done by further restricting the set of accepting ECs. For details, see [2], which describes verification of PAs under strong and weak fairness conditions, of which unconditional fairness is a special case.

3 Quantitative Multi-Objective Verification

In this section, we define a language for expressing multiple quantitative objectives of a probabilistic automaton. We then describe, for properties expressed in this language, techniques both to *verify* that the property holds for all adversaries of a PA and to *synthesise* an adversary of a PA under which the property holds. We also consider *numerical* queries, which yield an optimal value for one objective, subject to constraints imposed on one or more other objectives.

Definition 4 (Quantitative multi-objective properties). *A* quantitative multi-objective property *(qmo-property) for a PA $\mathcal{M}$ is a Boolean combination of probabilistic and reward predicates, i.e. an expression produced by the grammar:*

$$\Psi ::= \texttt{true} \mid \Psi \wedge \Psi \mid \Psi \vee \Psi \mid \neg\Psi \mid [\phi]_{\sim p} \mid [\rho]_{\sim r}$$

where $[\phi]_{\sim p}$ and $[\rho]_{\sim r}$ are probabilistic and reward predicates for $\mathcal{M}$, respectively. A simple *qmo-property comprises a single conjunction of predicates, i.e. is of the form $(\wedge_{i=1}^{n} [\phi_i]_{\sim_i p_i}) \wedge \big(\wedge_{j=1}^{m} [\rho_j]_{\sim_j r_j}\big)$. We refer to the predicates occurring in a formula as* objectives*. For property Ψ_P, we use α_P to denote the set of actions used in Ψ_P, i.e. the union of α_ϕ and α_ρ over $[\phi]_{\sim p}$ and $[\rho]_{\sim r}$ occurring in Ψ_P.*

A quantitative multi-objective property Ψ is evaluated over a PA $\mathcal{M}$ and an adversary σ of $\mathcal{M}$. We say that $\mathcal{M}$ satisfies Ψ under σ, denoted $\mathcal{M}, \sigma \models \Psi$, if Ψ evaluates to `true` when substituting each predicate x with the result of $\mathcal{M}, \sigma \models x$. *Verification* of Ψ over a PA $\mathcal{M}$ is defined as follows.

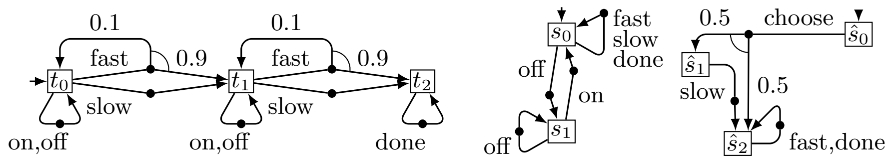


Fig. 1. PAs for a machine $\mathcal{M}_m$ (left) and two controllers, $\mathcal{M}_{c_1}$ (centre) and $\mathcal{M}_{c_2}$(right)

Definition 5 (Verification queries). *For a PA $\mathcal{M}$ and a qmo-property Ψ, a* verification query *asks whether Ψ is satisfied under all adversaries of $\mathcal{M}$:*

$$\mathcal{M} \models \Psi \;\Leftrightarrow\; \forall \sigma \in Adv_{\mathcal{M}} \,.\, \mathcal{M}, \sigma \models \Psi.$$

For a *simple* qmo-property Ψ, we can verify whether $\mathcal{M} \models \Psi$ using standard techniques [14,1] (since each conjunct can be verified separately). To treat the general case, we will use multi-objective model checking, proceeding via a reduction to the dual notion of *achievability queries*.

Definition 6 (Achievability queries). *For a PA $\mathcal{M}$ and qmo-property Ψ, an* achievability query *asks if there exists a satisfying adversary of $\mathcal{M}$, i.e. whether there exists $\sigma \in Adv_{\mathcal{M}}$ such that $\mathcal{M}, \sigma \models \Psi$.*

Remark. Since qmo-properties are closed under negation, we can convert any *verification* query into an equivalent (negated) *achievability* query. Furthermore, any qmo-property can be translated to an equivalent disjunction of *simple* qmo-properties (obtained by converting to disjunctive normal form and pushing negation into predicates, e.g. $\neg([\phi]_{>p}) \equiv [\phi]_{\leqslant p}$).

In practice, it is also often useful to obtain the minimum/maximum *value* of an objective, subject to constraints on others. For this, we use *numerical queries*.

Definition 7 (Numerical queries). *For a PA $\mathcal{M}$, qmo-property Ψ and ω-regular property ϕ or reward structure ρ, a* (maximising) numerical query *is:*

$$Pr^{\max}_{\mathcal{M}}(\phi \,|\, \Psi) \stackrel{\text{def}}{=} \sup\{Pr^{\sigma}_{\mathcal{M}}(\phi) \mid \sigma \in Adv_{\mathcal{M}} \wedge \mathcal{M}, \sigma \models \Psi\},$$
$$\text{or } ExpTot^{\max}_{\mathcal{M}}(\rho \,|\, \Psi) \stackrel{\text{def}}{=} \sup\{ExpTot^{\sigma}_{\mathcal{M}}(\rho) \mid \sigma \in Adv_{\mathcal{M}} \wedge \mathcal{M}, \sigma \models \Psi\}.$$

If the property Ψ is not satisfied by any adversary of $\mathcal{M}$, these queries return $\perp$. A minimising numerical query *is defined similarly.*

Example 1. Figure 1 shows the PAs we use as a running example. A machine, $\mathcal{M}_m$, executes 2 consecutive jobs, each in 1 of 2 ways: *fast*, which requires 1 time unit and 20 units of energy, but fails with probability 0.1; or *slow*, which requires 3 time units and 10 units of energy, and never fails. The reward structures $\rho_{time}=\{fast\mapsto 1, slow\mapsto 3\}$ and $\rho_{pow}=\{fast\mapsto 20, slow\mapsto 10\}$ capture the time elapse and power consumption of the system. The controllers, $\mathcal{M}_{c_1}$ and $\mathcal{M}_{c_2}$, can (each) control the machine, when composed in parallel with $\mathcal{M}_m$. Using qmo-property $\Psi = [\Diamond done]_{\geq 1} \wedge [\rho_{pow}]_{\leq 20}$, we can write a verification query $\mathcal{M}_m \models \Psi$

(which is false) or a numerical query $\mathit{ExpTot}^{\min}_{\mathcal{M}_m}(\rho_{time} \,|\, \Psi)$ (which yields 6). Before describing our techniques to check verification, achievability and numerical queries, we first need to discuss some *assumptions* made about PAs. One of the main complications when introducing rewards into multi-objective queries is the possibility of *infinite* expected total rewards. For the classical, single-objective case (see e.g. [1]), it is usual to impose assumptions so that such behaviour does not occur. For the multi-objective case, the situation is more subtle, and requires careful treatment. We now outline what assumptions should be imposed; later we describe how they can be checked algorithmically.

A key observation is that, if we allow arbitrary reward structures, situations may occur where extremely improbable (but non-zero probability) behaviour still yields infinite expected reward. Consider e.g. the PA $(\{s_0, s_1, s_2\}, s_0, \{a, b\}, \delta)$ with $\delta = \{(s_0, b, \eta_{s_1}), (s_0, b, \eta_{s_2}), (s_1, a, \eta_{s_1}), (s_2, b, \eta_{s_2})\}$, reward structure $\rho = \{a \mapsto 1\}$, and the qmo-property $\Psi = [\Box b]_{\geq p} \wedge [\rho]_{\geq r}$. For any p, including values arbitrarily close to 1, there is an adversary satisfying Ψ for any $r \in \mathbb{R}_{\geqslant 0}$, because it suffices to take action a with non-zero probability. This rather unnatural behaviour would lead to misleading verification results, masking possible errors in the model design.

Motivated by such problems, we enforce the restriction below on multi-objective queries. To match the contents of the next section, we state this for a maximising numerical query on rewards. We describe *how* to check the restriction holds in the next section.

Assumption 1 *Let $\mathit{ExpTot}^{\max}_{\mathcal{M}}(\rho \,|\, \Psi)$ be a numerical query for a PA $\mathcal{M}$ and qmo-property Ψ which is a disjunction*[2] *of simple qmo-properties $\Psi_1, \ldots, \Psi_l$. For each $\Psi_k = (\wedge_{i=1}^n [\phi_i]_{\sim_i p_i}) \wedge \big(\wedge_{j=1}^m [\rho_j]_{\sim_j r_j}\big)$, we require that:*

$$\sup\{\mathit{ExpTot}^{\sigma}_{\mathcal{M}}(\zeta) \mid \mathcal{M}, \sigma \models \textstyle\bigwedge_{i=1}^n [\phi_i]_{\sim_i p_i}\} < \infty$$

for all $\zeta \in \{\rho\} \cup \{\rho_j \,|\, 1 \leqslant j \leqslant m \ \wedge\ \sim_j \in \{>, \geq\}\}$.

3.1 Checking Multi-Objective Queries

We now describe techniques for checking the multi-objective queries described previously. For presentational purposes, we focus on *numerical* queries. It is straightforward to adapt this to *achievability* queries by introducing, and then ignoring, a dummy property to maximise (with no loss in complexity). As mentioned earlier, *verification* queries are directly reducible to achievability queries.

Let $\mathcal{M}$ be a PA and $\mathit{ExpTot}^{\max}_{\mathcal{M}}(\rho \,|\, \Psi)$ be a maximising numerical query for reward structure ρ (the cases for minimising queries and ω-regular properties are analogous). As discussed earlier, we can convert Ψ to a disjunction of simple qmo-properties. Clearly, we can treat each element of the disjunction separately and then take the maximum. So, without loss of generality, we assume that Ψ is simple, i.e. $\Psi = (\wedge_{i=1}^n [\phi_i]_{\sim_i p_i}) \wedge \big(\wedge_{j=1}^m [\rho_j]_{\sim_j r_j}\big)$. Furthermore, we assume that each $\sim_i$ is $\geqslant$ or $>$ (which we can do by changing e.g. $[\phi]_{<p}$ to $[\neg\phi]_{>1-p}$).

[2] This assumption extends to arbitrary properties Ψ by, as described earlier, first reducing to disjunctive normal form.

Our technique to compute $ExpTot^{\max}_{\mathcal{M}}(\rho \,|\, \Psi)$ proceeds via a sequence of modifications to $\mathcal{M}$, producing a PA $\hat{\mathcal{M}}$. From this, we construct a linear program $L(\hat{\mathcal{M}})$, whose solution yields both the desired numerical result and a corresponding adversary $\hat{\sigma}$ of $\hat{\mathcal{M}}$. Crucially, $\hat{\sigma}$ is *memoryless* and can thus be mapped to a matching *finite-memory* adversary of $\mathcal{M}$. The structure of $L(\hat{\mathcal{M}})$ is very similar to the one used in [16], but many of the steps to construct $\hat{\mathcal{M}}$ and the techniques to establish a memoryless adversary are substantially different. We also remark that, although not discussed here, $L(\hat{\mathcal{M}})$ can be adapted to a multi-objective linear program, or used to approximate the Pareto curve between objectives.

In the remainder of this section, we describe the process in detail, which comprises 4 steps: **1.** checking Assumption 1; **2.** building a PA $\bar{\mathcal{M}}$ in which unneccessary actions are removed; **3.** converting $\bar{\mathcal{M}}$ to a PA $\hat{\mathcal{M}}$; **4.** building and solving the linear program $L(\hat{\mathcal{M}})$. The correctness of the procedure is formalised with a corresponding sequence of propositions (proved in the appendix).

Step 1. We start by constructing a PA $\mathcal{M}^{\phi} = \mathcal{M} \otimes \mathcal{A}_{\phi_1} \otimes \cdots \otimes \mathcal{A}_{\phi_n}$ which is the product of $\mathcal{M}$ and a DRA $\mathcal{A}_{\phi_i}$ for each ω-regular property ϕ_i appearing in Ψ. We check Assumption 1 by analysing $\mathcal{M}^{\phi}$: for each *maximising* reward structure ζ (i.e. letting $\zeta{=}\rho$ or $\zeta{=}\rho_j$ when $\sim_j \in \{>, \geq\}$) we use the proposition below. This requires a simpler multi-objective achievability query on probabilistic predicates only. In fact, this can be done with the techniques of [16].

Proposition 1. *We have* $\sup\{ExpTot^{\sigma}_{\mathcal{M}}(\zeta) \mid \mathcal{M}, \sigma \models \bigwedge_{i=1}^{n} [\phi_i]_{\sim_i p_i}\} = \infty$ *for a reward structure* ζ *of* $\mathcal{M}$ *iff there is an adversary* σ *of* $\mathcal{M}^{\phi}$ *such that* $\mathcal{M}^{\phi}, \sigma \models [\Diamond pos]_{>0} \wedge \bigwedge_{i=1}^{n} [\phi_i]_{\sim_i p_i}$ *where "pos" labels any transition* (s, a, μ) *that satisfies* $\zeta(a){>}0$ *and is contained in an EC.*

Step 2. Next, we build the PA $\bar{\mathcal{M}}$ from $\mathcal{M}^{\phi}$ by removing actions that, thanks to Assumption 1, will not be used by any adversary which satisfies Ψ and maximises the expected value for the reward ρ. Let $\mathcal{M}^{\phi} = (S^{\phi}, \overline{s}, \alpha_{\mathcal{M}}, \delta^{\phi}_{\mathcal{M}})$. Then $\bar{\mathcal{M}} = (\bar{S}, \overline{s}, \alpha_{\mathcal{M}}, \bar{\delta}_{\mathcal{M}})$ is the PA obtained from $\mathcal{M}^{\phi}$ as follows. First, we remove (s, a, μ) from $\delta^{\phi}_{\mathcal{M}}$ if it is contained in an EC and $\zeta(a){>}0$ for some *maximising* reward structure ζ. Second, we repeatedly remove states with no outgoing transitions and transitions that lead to non-existent states, until a fixpoint is reached. The following proposition holds whenever Assumption 1 is satisfied.

Proposition 2. *There is an adversary* σ *of* $\mathcal{M}^{\phi}$ *where* $ExpTot^{\sigma}_{\mathcal{M}^{\phi}}(\rho){=}x$ *and* $\mathcal{M}^{\phi}, \sigma \models \Psi$ *iff there is an adversary* $\bar{\sigma}$ *of* $\bar{\mathcal{M}}$ *where* $ExpTot^{\bar{\sigma}}_{\bar{\mathcal{M}}}(\rho){=}x$ *and* $\bar{\mathcal{M}}, \bar{\sigma} \models \Psi$.

Step 3. Then, we construct PA $\hat{\mathcal{M}}$ from $\bar{\mathcal{M}}$, by converting the n probabilistic predicates $[\phi_i]_{\sim_i p_i}$ into n reward predicates $[\lambda_i]_{\sim_i p_i}$. For each $R \subseteq \{1, \ldots, n\}$, we let S_R denote the set of states that are contained in an EC (S', δ') that: (i) is accepting for all $\{\phi_i \mid i \in R\}$; (ii) satisfies $\rho_j(a) = 0$ for all $1 \leq j \leq m$ and $(s, a, \mu) \in \delta'$. Thus, in each S_R, no reward is gained and almost all paths satisfy the ω-regular properties ϕ_i for $i \in R$. Note that identifying the sets S_R can be done in time polynomial in the size of $\bar{\mathcal{M}}$ (see Appendix A.3 for clarification).

We then construct $\hat{\mathcal{M}}$ by adding a new terminal state s_{dead} and adding transitions from states in each S_R to s_{dead}, labelled with a new action a^R. Intuitively, *Maximise* $\sum_{(s,a,\mu)\in\hat{\delta}_{\mathcal{M}},s\neq s_{dead}} \rho(a)\cdot y_{(s,a,\mu)}$ *subject to:*

$$\begin{array}{rll}
\sum_{(s,a^R,\mu)\in\hat{\delta}_{\mathcal{M}},s\neq s_{dead}} y_{(s,a^R,\mu)} & = 1 & \\
\sum_{(s,a,\mu)\in\hat{\delta}_{\mathcal{M}},s\neq s_{dead}} \lambda_i(a)\cdot y_{(s,a,\mu)} & \sim_i p_i & \text{for all } 1\leq i\leq n \\
\sum_{(s,a,\mu)\in\hat{\delta}_{\mathcal{M}},s\neq s_{dead}} \rho_j(a)\cdot y_{(s,a,\mu)} & \sim_j r_j & \text{for all } 1\leq j\leq m \\
\sum_{(s,a,\mu)\in\hat{\delta}_{\mathcal{M}}} y_{(s,a,\mu)} - \sum_{(\hat{s},\hat{a},\hat{\mu})\in\hat{\delta}_{\mathcal{M}}} \mu'(s)\cdot y_{(\hat{s},\hat{a},\hat{\mu})} & = init(s) & \text{for all } s\in\hat{S}\setminus\{s_{dead}\} \\
y_{(s,a,\mu)} & \geq 0 & \text{for all } (s,a,\mu)\in\hat{\delta}_{\mathcal{M}}
\end{array}$$

where $init(s)$ *is* 1 *if* $s=\overline{s}$ *and* 0 *otherwise.*

Fig. 2. The linear program $L(\hat{\mathcal{M}})$

taking an action a^R in $\hat{\mathcal{M}}$ corresponds to electing to remain forever in the corresponding EC of $\bar{\mathcal{M}}$. Formally, $\hat{\mathcal{M}} = (\hat{S},\overline{s},\hat{\alpha}_{\mathcal{M}},\hat{\delta}_{\mathcal{M}})$ where $\hat{S} = \bar{S}\cup\{s_{dead}\}$, $\hat{\alpha}_{\mathcal{M}} = \alpha_{\mathcal{M}}\cup\{a^R \mid R\subseteq\{1,\dots,n\}\}$, and $\hat{\delta}_{\mathcal{M}} = \bar{\delta}_{\mathcal{M}}\cup\{(s,a^R,\eta_{s_{dead}}) \mid s\in S_R\}$. Finally, we create, for each $1\leq i\leq n$, a reward structure $\lambda_i : \{a^R \mid i\in R\}\to\mathbb{R}_{>0}$ with $\lambda_i(a^R)=1$ whenever λ_i is defined.

Proposition 3. *There is an adversary* $\bar{\sigma}$ *of* $\bar{\mathcal{M}}$ *such that* $ExpTot^{\bar{\sigma}}_{\bar{\mathcal{M}}}(\rho)=x$ *and* $\bar{\mathcal{M}},\bar{\sigma}\models\Psi$ *iff there is a memoryless adversary* $\hat{\sigma}$ *of* $\hat{\mathcal{M}}$ *such that* $ExpTot^{\hat{\sigma}}_{\hat{\mathcal{M}}}(\rho)=x$ *and* $\hat{\mathcal{M}},\hat{\sigma}\models(\wedge_{i=1}^n[\lambda_i]_{\sim_i p_i})\wedge\left(\wedge_{j=1}^m[\rho_j]_{\sim_j r_j}\right)\wedge([\Diamond s_{dead}]_{\geqslant 1})$.

Step 4. Finally, we create a linear program $L(\hat{\mathcal{M}})$, given in Figure 2, which encodes the structure of $\hat{\mathcal{M}}$ as well as the objectives from Ψ. Intuitively, in a solution of $L(\hat{\mathcal{M}})$, the variables $y_{(s,a,\mu)}$ express the expected number of times that state s is visited and transition $s\xrightarrow{a}\mu$ is taken subsequently. The expected total reward w.r.t. ρ_i is then captured by $\sum_{(s,a,\mu)\in\hat{\delta}_{\mathcal{M}},s\neq s_{dead}}\rho_i(a)\cdot y_{(s,a,\mu)}$. The result of $L(\hat{\mathcal{M}})$ yields the desired value for our numerical query.

Proposition 4. *For* $x\in\mathbb{R}_{\geqslant 0}$*, there is a memoryless adversary* $\hat{\sigma}$ *of* $\hat{\mathcal{M}}$ *where* $ExpTot^{\hat{\sigma}}_{\hat{\mathcal{M}}}(\rho)=x$ *and* $\hat{\mathcal{M}},\hat{\sigma}\models(\wedge_{i=1}^n[\lambda_i]_{\sim_i p_i})\wedge\left(\wedge_{j=1}^m[\rho_j]_{\sim_j r_j}\right)\wedge([\Diamond s_{dead}]_{\geq 1})$ *iff there is a feasible solution* $(y^\star_{(s,a,\mu)})_{(s,a,\mu)\in\hat{\delta}_{\mathcal{M}}}$ *of the linear program* $L(\hat{\mathcal{M}})$ *such that* $\sum_{(s,a,\mu)\in\hat{\delta}_{\mathcal{M}},s\neq s_{dead}}\rho_i(a)\cdot y^\star_{(s,a,\mu)}=x$.

In addition, a solution to $L(\hat{\mathcal{M}})$ gives a memoryless adversary σ_{prod} defined by $\sigma_{prod}(s)(a,\mu) = \frac{y_{(s,a,\mu)}}{\sum_{a',\mu'}y_{(s,a',\mu')}}$ if the denominator is nonzero (and defined arbitrarily otherwise). This can be converted into a finite memory adversary σ' for $\mathcal{M}^\phi$ by combining decisions of σ on actions in $\alpha_{\mathcal{M}}$ and, instead of taking actions a^R, mimicking adversaries witnessing that the state which precedes a^R in the history is in S_R. Adversary σ' can be translated into an adversary σ of $\mathcal{M}$ in standard fashion (see Appendix A.5 for clarification) using the fact that every finite path in $\mathcal{M}^\phi$ has a counterpart in $\mathcal{M}$ given by projecting states of $\mathcal{M}^\phi$ to their first components.
The following is then a direct consequence of Propositions 2, 3 and 4.

Theorem 1. *Given a PA* $\mathcal{M}$ *and numerical query* $ExpTot^{\max}_{\mathcal{M}}(\rho\,|\,\Psi)$ *satisfying Assumption 1, the result of the query is equal to the solution of the linear program*

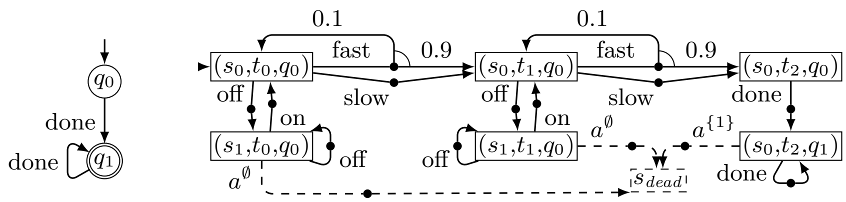


Fig. 3. A DRA $\mathcal{A}_\phi$ for the property $\phi = \Diamond done$ and the PA $\hat{\mathcal{M}}$ for $\mathcal{M} = \mathcal{M}_m \| \mathcal{M}_{c_1}$

$L(\hat{\mathcal{M}})$ *(see Figure 2). Furthermore, this requires time polynomial in the size of* $\mathcal{M}$ *and doubly exponential in the size of the property (for LTL objectives).*

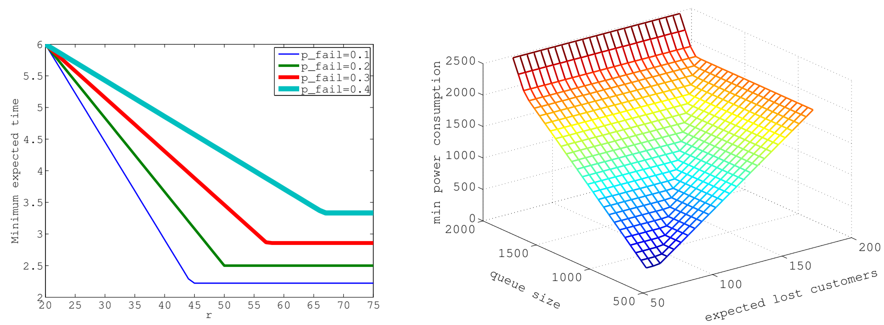
An analogous result holds for numerical queries of the form $ExpTot^{\min}_{\mathcal{M}}(\rho \,|\, \Psi)$, $Pr^{\max}_{\mathcal{M}}(\phi \,|\, \Psi)$ or $Pr^{\min}_{\mathcal{M}}(\phi \,|\, \Psi)$. As discussed previously, this also yields a technique to solve both achievability and verification queries in the same manner.

3.2 Controller Synthesis

The achievability and numerical queries presented in the previous section are directly applicable to the problem of *controller synthesis*. We first illustrate these ideas on our simple example, and then apply them to a large case study.

Example 2. Consider the composition $\mathcal{M} = \mathcal{M}_{c_1} \| \mathcal{M}_m$ of PAs from Figure 1; $\mathcal{M}_{c_1}$ can be seen as a template for a controller of $\mathcal{M}_m$. We synthesise an adversary for $\mathcal{M}$ that minimises the expected execution time under the constraints that the machine completes both jobs and the expected power consumption is below some bound r. Thus, we use the minimising numerical query $ExpTot^{\min}_{\mathcal{M}}(\rho_{time} \,|\, [\rho_{power}]_{\leqslant r} \wedge [\Diamond done]_{\geqslant 1})$. Figure 3 shows the corresponding PA $\hat{\mathcal{M}}$, dashed lines indicating additions to construct $\hat{\mathcal{M}}$ from $\bar{\mathcal{M}}$. Solving the LP problem $L(\hat{\mathcal{M}})$ yields the minimum expected time under these constraints. If r=30, for example, the result is $\frac{49}{11}$. Examining the choices made in the corresponding (memoryless) adversary, we find that, to obtain this time, a controller could schedule the first job *fast* with probability $\frac{5}{6}$ and *slow* with $\frac{1}{6}$, and the second job *slow*. Figure 4(a) shows how the result changes as we vary the bound r and use different values for the failure probability of *fast* (0.1 in Figure 1).

Case study. We have implemented the techniques of Section 3 as an extension of PRISM [19] and using the ECLiPSe LP solver. We applied them to perform controller synthesis on a realistic case study: we build a *power manager* for an IBM TravelStar VP disk-drive [5]. Specific (randomised) power management policies can already be analysed in PRISM [22]; here, we *synthesise* such policies, subject to constraints specified as qmo-properties. More precisely, we minimise the expected power consumption under restrictions on, for example, the expected job-queue size, expected number of lost jobs, probability that a request waits more than K steps, or probability that N requests are lost. Further details are available from [24]. As an illustration, Figure 4(b) plots the minimal power consumption under restrictions on both the expected queue size and number of lost customers. This shows the familiar power-versus-performance trade-off: policies can offer improved performance, but at the expense of using more power.



(a) Running example (see Ex. 2) (b) IBM TravelStar disk-drive controller

Fig. 4. Experimental results illustrating controller synthesis

4 Quantitative Assume-Guarantee Verification

We now present novel compositional verification techniques for probabilistic automata, based on the quantitative multi-objective properties defined in Section 3. The key ingredient of this approach is the *assume-guarantee triple*, whose definition, like in [21], is based on quantification over adversaries. However, whereas [21] uses a single *probabilistic safety property* as an assumption or guarantee, we permit *quantitative multi-objective* properties. Another key factor is the incorporation of *fairness*.

Definition 8 (Assume-guarantee triples). *If $\mathcal{M} = (S, \overline{s}, \alpha_{\mathcal{M}}, \delta_{\mathcal{M}})$ is a PA and Ψ_A, Ψ_G are qmo-properties such that $\alpha_G \subseteq \alpha_A \cup \alpha_M$, then $\langle \Psi_A \rangle \, \mathcal{M} \, \langle \Psi_G \rangle$ is an* assume-guarantee triple *with the following semantics:*

$$\langle \Psi_A \rangle \, \mathcal{M} \, \langle \Psi_G \rangle \;\Leftrightarrow\; \forall \sigma \in Adv_{\mathcal{M}[\alpha_A]} \, . \; (\mathcal{M}, \sigma \models \Psi_A \rightarrow \mathcal{M}, \sigma \models \Psi_G) \, .$$

where $\mathcal{M}[\alpha_A]$ denotes the alphabet extension *[21] of $\mathcal{M}$, which adds a-labelled self-loops to all states of $\mathcal{M}$ for each $a \in \alpha_A \backslash \alpha_{\mathcal{M}}$.*

Informally, an assume-guarantee triple $\langle \Psi_A \rangle \, \mathcal{M} \, \langle \Psi_G \rangle$, means "if $\mathcal{M}$ is a component of a system such that the environment of $\mathcal{M}$ satisfies Ψ_A, then the combined system (under fairness) satisfies Ψ_G".

Verification of an assume guarantee triple, i.e. checking whether $\langle \Psi_A \rangle \, \mathcal{M} \, \langle \Psi_G \rangle$ holds, reduces directly to the verification of a qmo-property since:

$$(\mathcal{M}, \sigma \models \Psi_A \rightarrow \mathcal{M}, \sigma \models \Psi_G) \;\Leftrightarrow\; \mathcal{M}, \sigma \models (\neg \Psi_A \vee \Psi_G) \, .$$

Thus, using the techniques of Section 3, we can reduce this to an achievability query, solvable via linear programming. Using these assume-guarantee triples as a basis, we can now formulate several proof rules that permit compositional verification of probabilistic automata. We first state two such rules, then explain their usage, illustrating with an example.

Theorem 2. *If $\mathcal{M}_1$ and $\mathcal{M}_2$ are PAs, and Ψ_{A_1}, Ψ_{A_2} and Ψ_G are quantitative multi-objective properties, then the following proof rules hold:*

$$\frac{\begin{array}{c}\mathcal{M}_1 \models_{\text{fair}} \Psi_{A_1} \\ \langle \Psi_{A_1} \rangle\, \mathcal{M}_2\, \langle \Psi_G \rangle\end{array}}{\mathcal{M}_1 \| \mathcal{M}_2 \models_{\text{fair}} \Psi_G} \quad (\textsc{ASym}) \qquad \frac{\begin{array}{c}\mathcal{M}_2 \models_{\text{fair}} \Psi_{A_2} \\ \langle \Psi_{A_2} \rangle\, \mathcal{M}_1\, \langle \Psi_{A_1} \rangle \\ \langle \Psi_{A_1} \rangle\, \mathcal{M}_2\, \langle \Psi_G \rangle\end{array}}{\mathcal{M}_1 \parallel \mathcal{M}_2 \models_{\text{fair}} \Psi_G} \quad (\textsc{Circ})$$

where, for well-formedness, we assume that, if a rule contains an occurrence of the triple $\langle \Psi_A \rangle\, \mathcal{M}\, \langle \Psi_G \rangle$ in a premise, then $\alpha_G \subseteq \alpha_A \cup \alpha_{\mathcal{M}}$; similarly, for a premise that checks Ψ_A against $\mathcal{M}$, we assume that $\alpha_A \subseteq \alpha_{\mathcal{M}}$

Theorem 2 presents two assume-guarantee rules. The simpler, (ASym), uses a single assumption Ψ_{A_1} about $\mathcal{M}_1$ to prove a property Ψ_G on $\mathcal{M}_1 \| \mathcal{M}_2$. This is done compositionally, in two steps. First, we verify $\mathcal{M}_1 \models_{\text{fair}} \Psi_{A_1}$. If $\mathcal{M}_1$ comprises just a single PA, the stronger (but easier) check $\mathcal{M}_1 \models \Psi_{A_1}$ suffices; the use of fairness in the first premise is to permit recursive application of the rule. Second, we check that $\langle \Psi_{A_1} \rangle\, \mathcal{M}_2\, \langle \Psi_G \rangle$ holds. Again, optionally, we can consider fairness here.[3] In total, these two steps have the potential to be significantly cheaper than verifying $\mathcal{M}_1 \| \mathcal{M}_2$. The other rule, (Circ), operates similarly, but using assumptions about both $\mathcal{M}_1$ and $\mathcal{M}_2$.

Example 3. We illustrate assume-guarantee verification using the PAs $\mathcal{M}_m$ and $\mathcal{M}_{c_2}$ from Figure 1. Our aim is to verify that $\mathcal{M}_{c_2} \| \mathcal{M}_m \models_{\text{fair}} [\rho_{time}]_{\leqslant \frac{19}{6}}$, which does indeed hold. We do so using the proof rule (ASym) of Theorem 2, with $\mathcal{M}_1 = \mathcal{M}_{c_2}$ and $\mathcal{M}_2 = \mathcal{M}_m$. We use the assumption $\mathcal{A}_1 = [\rho_{slow}]_{\leqslant \frac{1}{2}}$ where $\rho_{slow} = \{slow \mapsto 1\}$, i.e. we assume the expected number of *slow* jobs requested is at most 0.5. We verify $\mathcal{M}_{c_2} \models [\rho_{slow}]_{\leqslant \frac{1}{2}}$ and the triple $\langle [\rho_{slow}]_{\leqslant \frac{1}{2}} \rangle\, \mathcal{M}_m\, \langle [\rho_{time}]_{\leqslant \frac{19}{6}} \rangle$. The triple is checked by verifying $\mathcal{M}_m \models \neg [\rho_{slow}]_{\leqslant \frac{1}{2}} \vee [\rho_{time}]_{\leqslant \frac{19}{6}}$ or, equivalently, that no adversary of $\mathcal{M}_m$ satisfies $[\rho_{slow}]_{\leqslant \frac{1}{2}} \wedge [\rho_{time}]_{> \frac{19}{6}}$.

Experimental Results. Using our implementation of the techniques in Section 3, we now demonstrate the application of our quantitative assume-guarantee verification framework to two large case studies: Aspnes & Herlihy's randomised *consensus* algorithm and the *Zeroconf* network configuration protocol. For *consensus*, we check the maximum expected number of steps required in the first R rounds; for *zeroconf*, we verify that the protocol terminates with probability 1 and the minimum/maximum expected time to do so. In each case, we use the (Circ) rule, with a combination of probabilistic safety and liveness properties for assumptions. All models and properties are available from [24]. In fact, we execute numerical queries to obtain lower/upper bounds for system properties, rather than just verifying a specific bound.

Table 1 summarises the experiments on these case studies, which were run on a 2.66GHz PC with 8GB of RAM, using a time-out of 1 hour. The table

[3] Adding fairness to checks of both qmo-properties and assume-guarantee triples is achieved by encoding the unconditional fairness constraint as additional objectives.

Case study [parameters]		Non-compositional States	Time (s)	Result	Compositional LP size	Time (s)	Result
consensus *(2 processes)* *(max. steps)* $[R\ K]$	3 2	1,806	0.4	89.00	1,565	1.8	89.65
	3 20	11,598	27.8	5,057	6,749	10.8	5,057
	4 2	7,478	1.3	89.00	5,368	3.9	98.42
	4 20	51,830	155.0	5,057	15,160	16.2	5,120
	5 2	30,166	3.1	89.00	10,327	6.5	100.1
	5 20	212,758	552.8	5,057	24,727	21.9	5,121
consensus *(3 processes)* *(max. steps)* $[R\ K]$	3 2	114,559	20.5	212.0	43,712	12.1	214.3
	3 12	507,919	1,361.6	4,352	92,672	284.9	4,352
	3 20	822,607	time-out	-	131,840	901.8	11,552
	4 2	3,669,649	728.1	212.0	260,254	118.9	260.3
	4 12	29,797,249	mem-out	-	351,694	642.2	4,533
	4 20	65,629,249	mem-out	-	424,846	1,697.0	11,840
zeroconf *(termination)* $[K]$	4	57,960	8.7	1.0	155,458	23.8	1.0
	6	125,697	16.6	1.0	156,690	24.5	1.0
	8	163,229	19.4	1.0	157,922	25.5	1.0
zeroconf *(min. time)* $[K]$	4	57,960	6.7	13.49	155,600	23.0	16.90
	6	125,697	15.7	17.49	154,632	23.1	12.90
	8	163,229	22.2	21.49	156,568	23.9	20.90
zeroconf *(max. time)* $[K]$	4	57,960	5.8	14.28	154,632	23.7	17.33
	6	125,697	13.3	18.28	155,600	24.2	22.67
	8	163,229	18.9	22.28	156,568	25.1	28.00

Table 1. Experimental results for compositional verification

shows the (numerical) result obtained and the time taken for verification done both compositionally and non-compositionally (with PRISM). As an indication of problem size, we give the size of the (non-compositional) PA, and the number of variables in the linear programs for multi-objective model checking.

Compositional verification performs very well. For the *consensus* models, it is almost always faster than the non-compositional case, often significantly so, and is able to scale up to larger models. For *zeroconf*, times are similar. Encouragingly, though, times for compositional verification grow much more slowly with model size. We therefore anticipate better scalability through improvements to the underlying LP solver. Finally, we note that the numerical results obtained compositionally are very close to the true results (where obtainable).

5 Conclusions

We have presented techniques for studying multi-objective properties of PAs, using a language that combines ω-regular properties, expected total reward and multiple objectives. We described how to verify a property over all adversaries of a PA, synthesise an adversary that satisfies and/or optimises objectives, and compute the minimum or maximum value of an objective, subject to constraints. We demonstrated direct applicability to controller synthesis, illustrated with a realistic disk-drive controller case study. Finally, we proposed an assume-guarantee framework for PAs that significantly improves existing ones [21], and demonstrated successful compositional verification on several large case studies.

Possible directions for future work include extending our compositional verification approach with learning-based assumption generation, as [17] does for the simpler framework of [21], and investigation of continuous-time models.

Acknowledgments. The authors are part supported by ERC Advanced Grant VERIWARE, EU FP7 project CONNECT and EPSRC grant EP/D07956X. Vojtěch Forejt is also supported by a Royal Society Newton Fellowship and the Institute for Theoretical Computer Science, project no. 1M0545.

References

1. de Alfaro, L.: Formal Verification of Probabilistic Systems. Ph.D. thesis, Stanford University (1997)
2. Baier, C., Größer, M., Ciesinski, F.: Quantitative analysis under fairness constraints. In: Liu, Z., Ravn, A. (eds.) Proc. ATVA'09. LNCS, vol. 5799, pp. 135–150. Springer (2009)
3. Baier, C., Größer, M., Leucker, M., Bollig, B., Ciesinski, F.: Controller synthesis for probabilistic systems. In: Levy, J., Mayr, E., Mitchell, J. (eds.) Exploring New Frontiers of Theoretical Informatics. IFIP International Federation for Information Processing, vol. 155, pp. 493–506. Springer (2004)
4. Baier, C., Kwiatkowska, M.: Model checking for a probabilistic branching time logic with fairness. Distributed Computing 11(3), 125–155 (1998)
5. Benini, L., Bogliolo, A., Paleologo, G., Micheli, G.D.: Policy optimization for dynamic power management. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 8(3), 299–316 (2000)
6. Bianco, A., de Alfaro, L.: Model checking of probabilistic and nondeterministic systems. In: Thiagarajan, P. (ed.) Proc. FSTTCS'95. LNCS, vol. 1026, pp. 499–513. Springer (1995)
7. Brázdil, T., Forejt, V., Kučera, A.: Controller synthesis and verification for Markov decision processes with qualitative branching time objectives. In: Aceto, L., Damgård, I., Goldberg, L., Halldórsson, M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) Proc. ICALP'08. LNCS, vol. 5126, pp. 148–159. Springer (2008)
8. Caillaud, B., Delahaye, B., Larsen, K., Legay, A., Pedersen, M., Wasowski, A.: Compositional design methodology with constraint Markov chains. In: Proc. QEST'10. pp. 123–132. IEEE CS Press (2010)
9. Chatterjee, K.: Markov decision processes with multiple long-run average objectives. In: Arvind, V., Prasad, S. (eds.) Proc. FSTTCS'07. LNCS, vol. 4855, pp. 473–484. Springer (2007)
10. Chatterjee, K., de Alfaro, L., Faella, M., Henzinger, T., Majumdar, R., Stoelinga, M.: Compositional quantitative reasoning. In: Proc. QEST'06. pp. 179–188. IEEE CS Press (2006)
11. Chatterjee, K., Henzinger, T., Jobstmann, B., Singh, R.: Measuring and synthesizing systems in probabilistic environments. In: Touili, T., Cook, B., Jackson, P. (eds.) Proc. CAV'10. LNCS, vol. 6174, pp. 380–395. Springer (2010)
12. Chatterjee, K., Majumdar, R., Henzinger, T.: Markov decision processes with multiple objectives. In: Durand, B., Thomas, W. (eds.) Proc. STACS'06. LNCS, vol. 3884, pp. 325–336. Springer (2006)
13. Clímaco, J. (ed.): Multicriteria Analysis. Springer (1997)
14. Courcoubetis, C., Yannakakis, M.: Markov decision processes and regular events. IEEE Transactions on Automatic Control 43(10), 1399–1418 (1998)
15. Delahaye, B., Caillaud, B., Legay, A.: Probabilistic contracts: A compositional reasoning methodology for the design of stochastic systems. In: Proc. ACSD'10. pp. 223–232. IEEE CS Press (2010)
16. Etessami, K., Kwiatkowska, M., Vardi, M., Yannakakis, M.: Multi-objective model checking of Markov decision processes. Logical Methods in Computer Science 4(4), 1–21 (2008)
17. Feng, L., Kwiatkowska, M., Parker, D.: Compositional verification of probabilistic systems using learning. In: Proc. QEST'10. pp. 133–142. IEEE CS Press (2010)

18. Forejt, V., Kwiatkowska, M., Norman, G., Parker, D., Qu, H.: Quantitative multi-objective verification for probabilistic systems. In: Abdulla, P., K. Rustan M. Leino (eds.) Proc. TACAS'11. LNCS, Springer (2011), to appear
19. Hinton, A., Kwiatkowska, M., Norman, G., Parker, D.: PRISM: A tool for automatic verification of probabilistic systems. In: Hermanns, H., Palsberg, J. (eds.) Proc. TACAS'06. LNCS, vol. 3920, pp. 441–444. Springer (2006)
20. Kemeny, J., Snell, J., Knapp, A.: Denumerable Markov Chains. Springer, 2nd edn. (1976)
21. Kwiatkowska, M., Norman, G., Parker, D., Qu, H.: Assume-guarantee verification for probabilistic systems. In: Esparza, J., Majumdar, R. (eds.) Proc. TACAS'10. LNCS, vol. 6105, pp. 23–37. Springer (2010)
22. Norman, G., Parker, D., Kwiatkowska, M., Shukla, S., Gupta, R.: Using probabilistic model checking for dynamic power management. Formal Aspects of Computing 17(2), 160–176 (2005)
23. Segala, R.: Modelling and Verification of Randomized Distributed Real Time Systems. Ph.D. thesis, Massachusetts Institute of Technology (1995)
24. `http://www.prismmodelchecker.org/files/tacas11/`

A Appendix

Below, we include the proofs omitted from the main text. The results and proofs presented for Section 3.1 concern the case for maximising a numerical query on rewards. For the case of minimising rewards, we would first decide whether the minimal reward is infinite and, if yes, remove it from the query and then continue following the methods presented.

For the remainder of the section, we fix a PA $\mathcal{M}$ and qmo-property Ψ. Before we give the proofs, we require the following definitions. For a state s, let $\mathcal{M}_s$ be the PA obtained from $\mathcal{M}$ by changing the initial state to s. For a finite path $\omega_1 = s_0 \xrightarrow{a_0,\mu_0} s_1 \xrightarrow{a_1,\mu_1} \cdots \xrightarrow{a_{n-1},\mu_{n-1}} s_n$ and path $\omega_2 = s_n \xrightarrow{a_n,\mu_n} s_{n+1} \xrightarrow{a_{n+1},\mu_{n+1}} \cdots$, let

$$\omega_1 \odot \omega_2 = \omega_1 = s_0 \xrightarrow{a_0,\mu_0} s_1 \xrightarrow{a_1,\mu_1} \cdots \xrightarrow{a_{n-1},\mu_{n-1}} s_n \xrightarrow{a_n,\mu_n} s_{n+1} \xrightarrow{a_{n+1},\mu_{n+1}} \cdots$$

be the concatenation of ω_1 and ω_2. For a state s, we let $init_{\mathcal{M}}(s) = 1$ if s is the initial state of $\mathcal{M}$ and $init_{\mathcal{M}}(s) = 0$ otherwise. For any adversary σ and triple $(s, a, \mu) \in S \times \alpha_{\mathcal{M}} \times Dist(S)$ we let:

$$vis^{\sigma}(s, a, \mu) \stackrel{\text{def}}{=} \sum_{j=0}^{\infty} \left(\sum_{\substack{\omega \in FPaths, |\omega| = j+1\, \& \\ j\text{th trans. of } \omega \text{ is } (s,a,\mu)}} Pr^{\sigma}_{\mathcal{M}}(\omega) \right)$$

i.e. the expected number of times the transition $s \xrightarrow{a} \mu$ is taken by σ. Similarly, let $vis^{\sigma}(s)$ equal the expected number of times the state s is visited by σ.

A.1 Proof of Propositions 1 and 2

We first require the following lemma.

Lemma 1. *If (s,a,μ) is a transition that is not contained in any end component (EC) of $\mathcal{M}$, then $\sup_{\sigma\in Adv_{\mathcal{M}}} vis^{\sigma}(s,a,\mu)$ is finite.*

Proof. If (s,a,μ) is not in an EC of $\mathcal{M}$, then there exists $\varepsilon>0$ such that, starting from (s,a,μ), the probability of returning to s is at most $1-\varepsilon$, independent of the adversary. The proof then follows from basic results of probability theory. $\square$

Proposition 1, direction $\Rightarrow$. Suppose

$$\sup\{ExpTot^{\sigma}_{\mathcal{M}}(\zeta) \mid \sigma\in Adv_{\mathcal{M}} \wedge \mathcal{M},\sigma\models \textstyle\bigwedge_{i=1}^{n}[\phi]_{\sim_i p_i}\}=\infty$$

for a reward structure ζ of $\mathcal{M}$. A direct consequence of Lemma 1 is that, if a transition (s,a,μ) appears infinitely often (i.e. $\sup_{\sigma\in Adv_{\mathcal{M}}} vis^{\sigma}(s,a,\mu)$ is infinite), then it is contained in an EC. Therefore, for the total reward $ExpTot^{\sigma}_{\mathcal{M}}(\zeta)$ to be unbounded there must exist some adversary σ, such that $\sigma\models\bigwedge_{i=1}^{n}[\phi]_{\sim_i p_i}$ and under σ we enter some EC with non-zero probability in which some transition is assigned a positive reward. More precisely, we have that, under σ, the formula $[\Diamond pos]_{>0}$ is satisfied, and hence the result follows. $\square$

Proposition 1, direction $\Leftarrow$. Suppose there is an adversary σ of $\mathcal{M}^{\phi}$ such that $\mathcal{M}^{\phi},\sigma\models[\Diamond pos]_{>0}\wedge\bigwedge_{i=1}^{n}[\phi_i]_{\sim_i p_i}$. Since the formula $[\Diamond pos]_{>0}$ is satisfied by σ, under σ we reach a transition (s,a,μ) contained in an EC such that $\zeta(a)>0$. Therefore there exists a finite path $\omega = s_0 \xrightarrow{a_0,\mu_0} s_1 \xrightarrow{a_1,\mu_1} \cdots s_m$ that has non-zero probability under σ and $(s_{m-1},a_{m-1},\mu_{m-1})=(s,a,\mu)$.

To complete the proof it is sufficient to create a sequence of adversaries $\langle\sigma_k\rangle_{k\in\mathbb{N}}$ under which the probabilistic predicates $[\phi_1]_{\sim_1 p_1},\dots,[\phi_n]_{\sim_n p_n}$ are satisfied and $\lim_{k\to\infty} ExpTot^{\sigma_k}_{\mathcal{M}^{\phi}}(\zeta)=\infty$. For $k\in\mathbb{N}$, we construct σ_k as follows.

- For all paths that do not start with ω, the adversary σ_k mimics σ.
- When the path ω is performed, σ_k stays in the EC containing (s,a,μ) for k steps, picking uniformly all transitions of the EC. After k steps, on the next visit to s_m, i.e. when the history is of the form $\omega\odot\omega'$ where ω' starts and ends in s_m, the adversary σ_k mimics σ assuming that the part of the path after ω was never performed.

To see that the probabilistic predicates $[\phi_i]_{\sim_i p_i}$ are satisfied under σ_k for any $k\in\mathbb{N}$, consider the function θ_k that maps each path ω of σ to the paths of σ_k that differ only in the part where σ_k forcingly stays in the component containing (s,a,μ) (if such a part exists). Now, $\theta(\omega)\cap\theta(\omega')=\emptyset$ for all $\omega\neq\omega'$ and for all sets Ω we have $Pr^{\sigma}_{\mathcal{M}^{\phi}}(\Omega) = Pr^{\sigma_k}_{\mathcal{M}^{\phi}}(\theta(\Omega))$. The satisifaction of the predicates under each σ_k then follows from the fact that, for any path ω of σ, we have ω satisfies a ω-regular property ϕ if and only if each path in $\theta(\Omega)$ satisfies ϕ.

To complete the proof it remains to show that $\lim_{k\to\infty} ExpTot^{\sigma_k}_{\mathcal{M}^{\phi}}(\zeta)=\infty$. From the hypothesis, under σ, the probability of performing the path ω is equal to some positive value p, and therefore by construction, for any $k \in \mathbb{N}$, under σ_k the probability of ω is also at least p. From basic results of end components, there exists a positive value q such that when starting from any state contained in the EC containing s and uniformly picking the next transition, we reach s with probability at least q within ℓ steps, where ℓ is the number of states of $\mathcal{M}^{\varphi}$. Now, considering any $k \in \mathbb{N}$, it follows that $ExpTot^{\sigma_k}_{\mathcal{M}^{\phi}}(\zeta)$ at least $p{\cdot}q{\cdot}\zeta(a){\cdot}\lfloor k/\ell \rfloor$ and the result follows. □

The proof of Proposition 2 follows straightforwardly from Proposition 1.

A.2 Proof of Proposition 3

Proposition 3, direction $\Leftarrow$. Suppose there exists a memoryless adversary $\hat{\sigma}$ of $\hat{\mathcal{M}}$ such that $ExpTot^{\hat{\sigma}}_{\hat{\mathcal{M}}}(\rho){=}x$ and $\hat{\mathcal{M}}, \hat{\sigma} \models (\wedge_{i=1}^{n}[\lambda_i]_{\sim_i p_i}) \wedge \left(\wedge_{j=1}^{m}[\rho_j]_{\sim_j r_j}\right) \wedge ([\Diamond s_{dead}]_{\geqslant 1})$. The result follows by constructing an adversary $\bar{\sigma}$ on $\bar{\mathcal{M}}$ which makes choices according to $\hat{\sigma}$ until an action a^R for some $R \subseteq \{1, \ldots, n\}$ is taken, and then we switch to an adversary that stays in the end component witnessing that the last state belongs to S_R.

Proposition 3, direction $\Rightarrow$. Suppose there exists an adversary $\bar{\sigma}$ of $\bar{\mathcal{M}}$ such that $ExpTot^{\bar{\sigma}}_{\bar{\mathcal{M}}}(\rho){=}x$ and $\bar{\mathcal{M}}, \bar{\sigma} \models \Psi$. We first prove a weaker form of the proposition by removing the requirement on the adversary being memoryless and then extend the result to memoryless adversaries.

History-dependent adversaries. In this section we prove that the following proposition holds.

Proposition 5. *If there is an adversary $\bar{\sigma}$ of $\bar{\mathcal{M}}$ such that $ExpTot^{\bar{\sigma}}_{\bar{\mathcal{M}}}(\rho){=}x$ and $\bar{\mathcal{M}}, \bar{\sigma} \models \Psi$ for some $x \in \mathbb{R}_{\geqslant 0}$, then there is an adversary $\hat{\sigma}$ of $\hat{\mathcal{M}}$ such that $ExpTot^{\hat{\sigma}}_{\hat{\mathcal{M}}}(\rho){=}x$ and $\hat{\mathcal{M}}, \hat{\sigma} \models (\wedge_{i=1}^{n}[\lambda_i]_{\sim_i p_i}) \wedge \left(\wedge_{j=1}^{m}[\rho_j]_{\sim_j r_j}\right) \wedge ([\Diamond s_{dead}]_{\geqslant 1})$.*

To prove this proposition, we first require the following lemmas.

Lemma 2. *There exists $r \in \mathbb{R}_{\geqslant 0}$ such that $ExpTot^{\bar{\sigma}}_{\bar{\mathcal{M}}}(\rho_j){\leq}r$ for all adversaries $\bar{\sigma}$ of $\bar{\mathcal{M}}$ and reward predicates $[\rho_j]_{\sim_j r_j}$ where $\sim_i \in \{>, \geq\}$ appearing in Ψ.*

Proof. The proof follows directly from Lemma 1 and the fact that by construction, no EC of $\bar{\mathcal{M}}$ contains a transition (s, a, μ) such that $\rho_j(a){>}0$. □

Lemma 3. *For any $\bar{\sigma} \in Adv_{\bar{\mathcal{M}}}$ and $\varepsilon{>}0$, if $\mathcal{M}, \bar{\sigma} {\models} \Psi$, then there exists $k \in \mathbb{N}$ such that, under $\bar{\sigma}$, the expected reward accumulated after k steps is at most ε.*

Proof. The proof is by contradiction. Suppose that there exists $\bar{\sigma}' \in Adv_{\bar{\mathcal{M}}}$ and $\varepsilon'{>}0$ such that $\mathcal{M}, \bar{\sigma}' {\models} \Psi$ and, under $\bar{\sigma}'$, the expected reward accumulated after k steps is at least ε' for all $k \in \mathbb{N}$. It then follows that, for any $k \in \mathbb{N}$, there exists $l{>}k$ such that the expected reward accumulated between the kth and lth step is at least $\varepsilon'/2$. However, this means that the expected reward is infinite for $\bar{\sigma}'$ which yields a contradiction since from Lemma 2 and the fact that $\bar{\mathcal{M}}, \bar{\sigma}' \models \Psi$ it follows that the reward is bounded for $\bar{\sigma}'$. □

We now proceed with the proof of Proposition 5 by constructing a sequence of adversaries $\langle\sigma_i\rangle_{i\in\mathbb{N}}$ and sets $\Theta_R \subseteq FPaths$ for $R \subseteq \{1,\ldots,n\}$. The limit of this sequence yields an adversary σ_∞ which we will show satisfies the following properties:

- $\bar{\mathcal{M}}, \sigma_\infty \models \Psi$ and $ExpTot^{\sigma_\infty}_{\bar{\mathcal{M}}}(\rho) = x$;
- under σ_∞, almost surely we take a path ω ending in a state s in S_R for some R, such that almost surely the infinite paths initiated in ω stay in the EC which witnesses that s is in S_R.

The second property will allow us to take actions a^R after ω, turning probabilistic properties into reward properties. To keep track of the paths ω after which we switch to a^R, we create the set Θ_R that contains these paths.

The construction of the adversaries $\langle\sigma_i\rangle_{i\in\mathbb{N}}$ is by induction on $i \in \mathbb{N}$. For the base case, let $\sigma_0 = \bar{\sigma}$. For the inductive step, suppose that σ_{i-1} has been constructed and $k \in \mathbb{N}$ is obtained from Lemma 3 when applied to the adversary σ_{i-1} for $\varepsilon{=}\frac{1}{i}$. Given a finite path ω, we denote by σ_ω the adversary given by $\sigma_\omega(\omega'){=}\sigma_{i-1}(\omega \odot \omega')$. Let Pos be the set of all transitions (s,a,μ) of $\bar{\mathcal{M}}$ with $\rho_j(a){>}0$. Now, for an ω-regular property ϕ and finite path ω, let $\phi[\omega]$ be the property which holds on a path ω' if and only if ω' does not contain a transition of Pos and ϕ holds on $\omega \odot \omega'$.

By [16], we can suppose that there is a finite-memory adversary σ'_ω satisfying:

- $Pr^{\sigma'_\omega}_{\bar{\mathcal{M}}_{s_\omega}}(\varphi_i[\omega]) = Pr^{\sigma_\omega}_{\bar{\mathcal{M}}_{s_\omega}}(\varphi_i[\omega])$ for all $1 \le i \le n$;
- $Pr^{\sigma'_\omega}_{\bar{\mathcal{M}}_{s_\omega}}(\xi_{(s,a,\mu)}) = Pr^{\sigma_\omega}_{\bar{\mathcal{M}}_{s_\omega}}(\xi_{(s,a,\mu)})$ for all $(s,a,\mu) \in Pos$

where $s_\omega{=}last(\omega)$ and $\xi_{(s,a,\mu)}$ is the set of paths that contain the transition (s,a,μ) and do not contain any other transition in Pos before the first occurence of (s,a,μ).

Because σ' is a finite-memory adversary, almost surely the path of $\bar{\mathcal{M}}_{last(\omega)}$ has a finite prefix (i.e. starts with a finite path) ω' satisfying the following:

- there is an end component (S',δ') such that almost surely the paths starting with ω' will visit all transitions of δ' infinitely often;
- for all $\mathcal{A}_{\phi_i}$ (for $1 \le i \le n$), either almost surely the trace the path starting with ω' is accepted by $\mathcal{A}_{\phi_i}$, or almost surely the trace of no path starting with ω' is accepted by $\mathcal{A}_{\phi_i}$.

For all such paths ω', let us put ω' to a set T_R, where R contains all i such that the EC (S',δ') above is accepting for $\mathcal{A}_{\phi_i}$.

Now let us fix $(s,a,\mu) \in Pos$. Let $\omega_1, \omega_2, \ldots$ be all finite paths that start with ω and end with $s \xrightarrow{a,\mu} s'$ for some s' and do not contain any transition in Pos before the last position (we only handle the case when there is infinitely many such paths, the finite case is similar). For any finite path $\bar{\omega}$ and $(last(\bar{\omega}), a', \mu') \in \bar{\delta}_{\mathcal{M}}$, we let π_s be the adversary of $\bar{\mathcal{M}}$ such that:

$$\pi_{s'}(\bar{\omega})(a',\mu') = \frac{\sum_{j\in\mathbb{N}} Pr^{(\sigma_{i-1})_\omega}_{\bar{\mathcal{M}}_{last(\omega)}}(\omega_j \odot \bar{\omega} \xrightarrow{a'} \mu)}{\sum_{j\in\mathbb{N}} Pr^{(\sigma_{i-1})_\omega}_{\bar{\mathcal{M}}_{last(\omega)}}(\omega_j \odot \bar{\omega})}$$

if the denominator is positive, and arbitrarily otherwise.

Lemma 4. *For every measurable set Ω of paths of $\bar{\mathcal{M}}$:*

$$Pr^{\pi_{s'}}_{\bar{\mathcal{M}}_{s'}}(\Omega) = \sum_{j\in\mathbb{N}} Pr^{(\sigma_{i-1})_{\omega_j}}_{\bar{\mathcal{M}}_{s'}}(\Omega)\,.$$

Proof. The proof follows from the construction of the probability measure over infinite paths for an adversary of a PA [20], first proving by induction that the result holds for basic cylinders, i.e. proving by induction on the length of a finite path ω_{fin} of $\bar{\mathcal{M}}_{s'}$ that the lemma holds for the measurable set $\{\omega \in \mathit{IPaths} \,|\, \omega = \omega_{\mathit{fin}} \odot \omega' \text{ for some } \omega' \in \mathit{IPaths}\}$. □

Next, we let $\hat{\sigma}_\omega$ be the adversary such that for any finite path ω':

$$\hat{\sigma}_\omega(\omega') = \begin{cases} \sigma_\omega(\omega') & \text{if } \omega' \text{ does not contain any transition in } \mathit{Pos} \\ \pi_s(s \xrightarrow{a,\mu} \bar{\omega}) & \text{otherwise} \end{cases}$$

where $\bar{\omega}$ is the finite path such that $\omega' = \hat{\omega} \odot s \xrightarrow{a,\mu} \bar{\omega}$ and $\hat{\omega}$ contains no transition in Pos.

We now construct the adversary σ_i such that for any finite path ω:

- $\sigma_i(\omega) = \sigma_{i-1}(\omega)$ if $|\omega| < k$ or if no transitions (s, a, μ) where a has positive reward are reachable from ω under σ_{i-1};
- and otherwise $\sigma_i(\omega) = \hat{\sigma}_{\omega'}(\omega'')$ where $\omega = \omega' \odot \omega''$ and $|\omega'| = k$.

For all $\omega' \in T_R$ and finite paths ω of length k, we add $\omega \odot \omega'$ to Θ_R, whenever $\omega \odot \omega'$ is defined.

This completes the construction of the sequence of adversaries $\langle\sigma_i\rangle_{i\in\mathbb{N}}$ and sets $\Theta_R \subseteq \mathit{FPaths}$ for $R \subseteq \{1, \ldots, n\}$ and let $\sigma_\infty = \lim_{i\to\infty} \sigma_i$.

Lemma 5. *The adversaries σ_i and σ_∞ preserve the probabilities that the ω-regular properties in Ψ are satisfied, and the expected rewards achieved. Moreover, under σ_∞, almost surely the path is initiated with a path in Θ_R for some R.*

Proof. The fact that probabilities are preserved in σ_i follows from its construction via $\hat{\sigma}_\omega$ and σ_{i-1}. That this result holds for σ_∞ follows from the fact that σ_∞ is obtained as a limit of the sequence $\langle\sigma_i\rangle_{i\in\mathbb{N}}$. To see that almost surely the path is initiated with a path in Θ_R, suppose for a contradiction this does not hold. Hence, a non-zero measure of paths must contain infinitely many actions with non-zero reward. However, for $\varepsilon{>}0$, the probability of such paths in σ_i (where $\varepsilon{>}\frac{1}{i}$) is lower than ε, yielding a contradiction. □

Finally, the proof of Proposition 5 follows by constructing the adversary $\hat{\sigma}$ from σ_∞, with the change that, after a finite path ω in Θ_R is taken, $\hat{\sigma}$ chooses the action a^R. Note that such an action is indeed available.

Memoryless adversaries. We now prove the following proposition, which, together with Proposition 5, proves the $\Rightarrow$ direction of Proposition 3.

Proposition 6. *If there is an adversary σ of $\hat{\mathcal{M}}$ such that $ExpTot^{\sigma}_{\hat{\mathcal{M}}}(\rho)=x$ and $\hat{\mathcal{M}}, \hat{\sigma} \models (\wedge_{i=1}^{n}[\lambda_i]_{\sim_i p_i}) \wedge \big(\wedge_{j=1}^{m}[\rho_j]_{\sim_j r_j}\big) \wedge ([\Diamond s_{dead}]_{\geqslant 1})$, then there is a memoryless adversary σ' of $\hat{\mathcal{M}}$ with the same properties.*

In the remainder of this section we suppose σ is an adversary of $\hat{\mathcal{M}}$ such that $ExpTot^{\sigma}_{\hat{\mathcal{M}}}(\rho)=x$ and $\hat{\mathcal{M}}, \hat{\sigma} \models (\wedge_{i=1}^{n}[\lambda_i]_{\sim_i p_i}) \wedge \big(\wedge_{j=1}^{m}[\rho_j]_{\sim_j r_j}\big) \wedge ([\Diamond s_{dead}]_{\geqslant 1})$. By the definitions of *vis* and *ExpTot*, to complete the proof of Proposition 6 it is sufficient to find a memoryless adversary σ' of $\hat{\mathcal{M}}$ for which $vis^{\sigma'}(s,a,\mu)=vis^{\sigma}(s,a,\mu)$ for all $(s,a,\mu) \in \hat{\delta}_{\mathcal{M}}$ such that $\rho(a)>0$, $\rho_j(a)>0$ and $\lambda_i(a)>0$ for $1 \le j \le m$ and $1 \le i \le n$.

Note that, for all (s,a,μ), whenever $vis^{\sigma}(s,a,\mu)=\infty$, we have $\rho(a)=0$, $\rho_i(a)=0$ for all $1 \le i \le m$ and $\lambda_i(a)=0$ for all $1 \le i \le n$. For reward structures related to a reward predicate with an operator $<$ or $\le$, this holds because the predicate is satisfied (and hence the expected total reward with respect to the reward structure is finite, meaning that no transition whose action has nonzero reward is taken). For other reward structures, this follows from the fact that no EC contains any transition (s',a',μ') with a' having nonzero reward under the reward structure, and from Lemma 1.

The remainder of this section is therefore concerned with constructing a memoryless adversary σ' of $\hat{\mathcal{M}}$ for which $vis^{\sigma'}(s,a,\mu)=vis^{\sigma}(s,a,\mu)$ whenever $vis^{\sigma}(s,a,\mu)<\infty$.

Let $InfA \stackrel{\text{def}}{=} \{(s,a,\mu) \mid vis^{\sigma}(s,a,\mu)=\infty\}$ and $InfS \stackrel{\text{def}}{=} \{s \mid (s,a,\mu) \in InfS\}$. Obviously, $s \in InfS$ if and only if $vis^{\sigma}(s)=\infty$.

Lemma 6. *The sets of states $(InfS, InfA)$ are unions of disjoint end-components.*

Proof. The result follows from basic results of probability theory. □

Lemma 7. *If (S',δ') is an EC which is maximal with respect to the PA induced by $(InfS, InfA)$, then there is a memoryless adversary $\pi_{(S',\delta')}$ such that for all $\hat{s}, s \in S'$ and (s,a,μ) satisfying $vis^{\sigma}(s,a,\mu) < \infty$:*

$$\frac{vis^{\sigma}(s,a,\mu)}{\sum_{s' \in S', vis^{\sigma}(s',a',\mu')<\infty} vis^{\sigma}(s',a',\mu')} = Pr^{\pi_{(S',\delta')}}_{\mathcal{M}_{\hat{s}}}(\{\omega \xrightarrow{a,\mu} s'' \mid \omega \text{ stays in } S'\})$$

Proof. We can see the lemma as a multi-objective reachability query where each of the states to be reached is terminal. For such a problem, memoryless adversaries are sufficient [16]. Moreover, because the end component is left eventually, the expected numbers of times each of its transitions will be taken are finite. □

Let σ' be the memoryless adversary where

$$\sigma'(\omega)(a,\mu) = \begin{cases} \frac{vis^{\sigma}(s,a,\mu)}{\sum_{a'} vis^{\sigma}(s,a',\mu')} & \text{if } last(\omega) \notin InfS \\ \pi_{(S',\delta')}(\omega) & \text{otherwise} \end{cases}$$

where (S',δ') is the unique (maximal with respect to the PA induced by $(InfS, InfA)$) end component such that $last(\omega) \in S'$.

To complete the proof it therefore remains to show $vis^{\sigma'}(s,a,\mu)$=$vis^{\sigma}(s,a,\mu)$ for all $(s,a,\mu) \in \hat{\delta}_{\mathcal{M}}$ such that $vis^{\sigma}(s,a,\mu)$<∞. Suppose for a contradiction this is not the case, and let (s,a,μ) be any transition such that $d_{(s,a,\mu)} = \frac{vis^{\sigma}(s,a,\mu)}{vis^{\sigma'}(s,a,\mu)}$ is maximal and denote this maximal value by d. However, if d≤1, then instead choose (s,a,μ) such that $d_{(s,a,\mu)}$ is minimal, let d equal this minimal value and proceed similarly.

Let $\omega = s_0 \xrightarrow{a_0,\mu_0} \cdots s_{n+1}$ be a finite path where (s_n,a_n,μ_n)=(s,a,μ) and $Pr^{\sigma}_{\hat{\mathcal{M}}}(\omega)$>0. We show that the case when n=0 can not occur, and if n⩾1, then there exists $i < n$ such that $d = \frac{vis^{\sigma}(s_i,a_i,\mu_i)}{vis^{\sigma'}(s_i,a_i,\mu_i)}$ which together yields the desired contradiction. There are two cases to consider.

Case 1. If $s \notin \mathit{InfS}$, then $\frac{vis^{\sigma}(s,a,\mu)}{vis^{\sigma}(s)} = \sigma(s)(a,\mu) = \frac{vis^{\sigma'}(s,a,\mu)}{vis^{\sigma'}(s)}$ and thus:

$$\begin{aligned}
vis^{\sigma}(s) &= d{\cdot}vis^{\sigma'}(s) \\
&= d{\cdot}\left(init(s) + \sum_{(s',a',\mu')\in\hat{\delta}_{\mathcal{M}}\wedge\mu(s')>0} vis^{\sigma'}(s',a',\mu'){\cdot}\mu'(s)\right) \\
&= init(s) + d'{\cdot}\left(\sum_{(s',a',\mu')\in\hat{\delta}_{\mathcal{M}}\wedge\mu(s')>0} vis^{\sigma'}(s',a',\mu'){\cdot}\mu'(s)\right) \\
&= init(s) + \left(\sum_{(s',a',\mu')\in\hat{\delta}_{\mathcal{M}}\wedge\mu(s')>0} d_{(s',a',\mu')}{\cdot}vis^{\sigma'}(s',a',\mu'){\cdot}\mu'(s)\right).
\end{aligned}$$

for some $d' \geq d$. Suppose $init(s)$=1 (which is the case n=0), then it follows that:

$$vis^{\sigma'}(s) = 1 + \sum_{(s',a',\mu')\in\hat{\delta}_{\mathcal{M}}\wedge\mu(s')>0} vis^{\sigma'}(s',a',\mu'){\cdot}\mu'(s)$$

which means some $d_{(s',a',\mu')}$ is greater than d, which is a contradiction. On the other hand, suppose $init(s) = 0$, it then follows that:

$$vis^{\sigma'}(s) = \sum_{(s',a',\mu')\in\hat{\delta}_{\mathcal{M}}\wedge\mu(s')>0} vis^{\sigma'}(s',a',\mu'){\cdot}\mu'(s)$$

which by maximality of d implies $d_{(s',a',\mu')}$=d for all $\mu'(s)$>0 and thus we can choose i=n−1.

Case 2. If $s \in \mathit{InfS}$, let (S',δ') be the EC that contains s and is maximal in $(\mathit{InfS}, \mathit{InfA})$,

$$\begin{aligned}
E_{out} &= \{(s',a',\mu') \mid s' \in S' \wedge vis^{\sigma'}(s',a',\mu')<\infty\} \\
E_{in} &= \{(s',a',\mu') \mid vis^{\sigma'}(s',a',\mu')<\infty \wedge \exists s'' \in S' \,.\, \mu'(s'')>0\}\,.
\end{aligned}$$

By definition of $\pi_{(S',\delta')}$:

$$\sum_{(s',a',\mu')\in E_{out}} vis^{\sigma}(s',a',\mu') = d{\cdot}\left(\sum_{(s',a',\mu')\in E_{out}} vis^{\sigma'}(s',a',\mu')\right)$$

and because the flow into (S', δ') is equal to the flow out of (S', δ'), we have:

$$init(S') + \sum_{(s',a',\mu')\in E_{in}} \left(\sum_{s''\in S'} vis^{\sigma}(s',a',\mu')\cdot\mu'(s'') \right) = \sum_{(s',a',\mu')\in E_{out}} vis^{\sigma}(s',a',\mu')$$

where $init(S')$ is 1 if S' contains the initial state and 0 otherwise. Suppose $init(S') = 1$ (which includes the case n=0), then both the following must hold:

$$1 + \sum_{(s',a',\mu')\in E_{in}} \left(\sum_{s''\in S'} vis^{\sigma'}(s',a',\mu')\cdot\mu(s'') \right) = \sum_{(s',a',\mu')\in E_{out}} vis^{\sigma'}(s',a',\mu')$$

$$1 + \sum_{(s',a',\mu')\in E_{in}} \left(\sum_{s''\in S'} d_{(s',a',\mu')}\cdot vis^{\sigma'}(s',a',\mu')\cdot\mu(s'') \right) = d\cdot \left(\sum_{(s',a',\mu')\in E_{out}} vis^{\sigma'}(s',a',\mu') \right)$$

which is not possible by the maximality of d. On the other hand, suppose $init(S') = 0$, then both the following must hold:

$$\sum_{(s',a',\mu')\in E_{in}} \left(\sum_{s''\in S'} vis^{\sigma'}(s',a',\mu')\cdot\mu(s'') \right) = \sum_{(s',a',\mu')\in E_{out}} vis^{\sigma'}(s',a',\mu')$$

$$\sum_{(s',a',\mu')\in E_{in}} \left(\sum_{s''\in S'} d_{(s',a',\mu')}\cdot vis^{\sigma'}(s',a',\mu')\cdot\mu(s'') \right) = d\cdot \left(\sum_{(s',a',\mu')\in E_{out}} vis^{\sigma'}(s',a',\mu') \right)$$

which necessarily means that for all $(s', a', \mu') \in E_{out}$ we have $d_{(s',a',\mu')}$=d, and we can thus pick some i<n (since there must be i where $(s_i, a_i, \mu_i) \in E_{in}$).

A.3 Computing the Sets S_R

In this section, we describe the computation of the sets S_R from page 9 and analyse its complexity. For each $R \subseteq \{1, \ldots, n\}$, and for every possible choice of pairs $\{(L_i, K_i)\}_{i\in R}$ where (L_i, K_i) is an accepting pair of $\mathcal{A}_{\phi_i}$, we perform the following procedure: repeatedly remove until a fixpoint is reached:

- states whose (i+1)th component is in L_i for some i;
- transitions that lead to non-existent states;
- states that have no outgoing transitions.

In the resulting PA, we find all maximal ECs that for each K_i contain a state whose (i+1)th component is in K_i.

For a fixed R and set of pairs, we can perform the procedure in time polynomial in the size of the model. We need to perform this procedure $\mathcal{O}(2^n \cdot \ell^n)$-times where ℓ is the maximal number of accepting pairs in the automata. Because ℓ is at most exponential in the size of the original (LTL) property, we obtain that the whole procedure can be performed in time polynomial in the size of the model and doubly exponential in the size of the (LTL) properties.

A.4 Proof of Proposition 4

Proposition 4, direction ⇒. We begin by defining the values $y'_{(s,a,\mu)}$ as in [16], i.e. we set $y'_{(s,a,\mu)} = vis^{\sigma}_{(s,a,\mu)}$. The following lemma is then the analogue of [16, Lemma 3.3].

Lemma 8. *The vector y' is well-defined and is a feasible solution of $L(\hat{\mathcal{M}})$.*

Proof. The proof follows similarly to [16, Lemma 3.3] except we also need to show:

$$\textstyle\sum_{(s,a,\mu)\in\hat{\delta}_{\mathcal{M}},s\neq s_{dead}} \rho(a)\cdot y'_{(s,a,\mu)} = \mathit{ExpTot}^{\sigma}_{\mathcal{M}}(\rho) \quad (1)$$

$$\textstyle\sum_{(s,a,\mu)\in\hat{\delta}_{\mathcal{M}},s\neq s_{dead}} \rho_i(a)\cdot y'_{(s,a,\mu)} = \mathit{ExpTot}^{\sigma}_{\mathcal{M}}(\rho_i) \quad (2)$$

$$\textstyle\sum_{(s,a,\mu)\in\hat{\delta}_{\mathcal{M}},s\neq s_{dead}} \lambda_j(a)\cdot y'_{(s,a,\mu)} = \mathit{ExpTot}^{\sigma}_{\mathcal{M}}(\lambda_j) \quad (3)$$

for $1\leq i\leq m$ and $1\leq j\leq n$. Below we consider only (1) as the other cases follow similarly. For any infinite path $\omega = s_0 \xrightarrow{a_0,\mu_0} s_1 \xrightarrow{a_1,\mu_1} \cdots$, we let $tr(\omega)_j = a_j$. Now by definition we have:

$$\begin{aligned}
&\mathit{ExpTot}^{\sigma}_{\mathcal{M}}(\rho) = \textstyle\int_{\omega\in IPaths} Pr^{\sigma}_{\mathcal{M}}(\omega)\cdot\left(\sum_{j=0}^{\infty}\rho(tr(\omega)_j)\,dPr^{\sigma}_{\mathcal{M}}\right) \\
&= \textstyle\sum_{j=0}^{\infty}\left(\int_{\omega\in IPaths} Pr^{\sigma}_{\mathcal{M}}(\omega)\cdot\rho(tr(\omega)_j)\,dPr^{\sigma}_{\mathcal{M}}\right) && \text{rearranging} \\
&= \textstyle\sum_{j=0}^{\infty}\left(\sum_{\substack{\omega\in FPaths\\ \wedge|\omega|=j+1}} Pr^{\sigma}_{\mathcal{M}}(\omega)\cdot\rho(tr(\omega)_{j-1})\right) && \text{by construction of } Pr^{\sigma}_{\mathcal{M}} \\
&= \textstyle\sum_{(s,a,\mu)\in\delta}\rho(a)\cdot\left(\sum_{j=0}^{\infty}\left(\sum_{\substack{\omega\in FPaths,|\omega|=j+1\wedge\\ j\text{th trans. of }\omega\text{ is }(s,a,\mu)}} Pr^{\sigma}_{\mathcal{M}}(\omega)\right)\right) && \text{rearranging} \\
&= \textstyle\sum_{(s,a,\mu)\in\delta}\rho(a)\cdot y'_{(s,a,\mu)} && \text{by definition of } y'_{(s,a,\mu)}
\end{aligned}$$

and hence (1) holds as required. □

Proposition 4, direction ⇐. Suppose y'' is a solution of $L(\hat{\mathcal{M}})$. We define an adversary σ identically to [16, Theorem 3.2, (2.) ⇒ (1.)], i.e. we let $\sigma(s)(a,\mu) = \frac{y_{(s,a,\mu)}}{\sum_{a,\mu}(s,a,\mu)}$. From the results of [16] we have $y''_{(s,a,\mu)} = y'_{(s,a,\mu)}$ for any state reachable under σ from the initial state. For the unreachable states, we have that if $y''_{(s,a,\mu)}>0$, then (s,a,μ) is contained in some end component (this follows from the structure of the equations in $L(\hat{\mathcal{M}})$). Thus, for such transitions, we have $\rho_i(a)>0$ only if the reward predicate containing ρ_i is of the form $[\rho_i]_{<r_i}$ or $[\rho_i]_{\leq r_i}$ (and similarly for λ_i). This means we can distinguish between 2 cases.

- If the reward predicate containing ρ_i is of the form $[\rho_i]_{>r_i}$ or $[\rho_i]_{\geq r_i}$, then we obtain the same equations as above.
- If the reward predicate containing ρ_i is of the form $[\rho_i]_{<r_i}$ or $[\rho_i]_{\leq r_i}$, then we obtain the same equations as above, changing the last = to ≤.

A.5 Construction of adversaries

From $\mathcal{M}^\phi$ to $\mathcal{M}$. To construct an adversary of $\mathcal{M}$ from an adversary of $\mathcal{M}^\phi$ that preserves the rewards gained and probabilities of satisfying ϕ_i for $1{\leq}i{\leq}n$, we require the following function θ that maps finite paths on $\mathcal{M}$ to finite paths on $\mathcal{M}^\phi$. For paths of length 0, we let $\theta(s) \stackrel{\text{def}}{=} (s, q_1, \ldots, q_i)$ where q_i is the initial state of $\mathcal{A}_{\phi_i}$ for $1{\leq}i{\leq}n$. Supposing θ has been defined for paths of length k and ω is a finite path of length $k{+}1$, then ω is of the form $\omega' \xrightarrow{a,\mu} s$ for some path ω' of length k and we let $\theta(\omega) \stackrel{\text{def}}{=} \theta(\omega') \xrightarrow{a,\mu'} s'$ where μ' is obtained from μ following the definition of the product automaton $\mathcal{M}^\phi$, and s' is the unique state that is chosen by μ' with nonzero probability and has s in the first component.

Now given any adversary σ' of $\mathcal{M}^\phi$, we define an adversary σ of $\mathcal{M}$ as follows: for any finite path ω and action-distribution pair (a, μ) of $\mathcal{M}$ let $\sigma(\omega)(a, \mu) = \sigma(\theta(\omega))(a, \mu')$ where μ' is obtained from μ following the definition of the product automaton $\mathcal{M}^\phi$. It is straightforward to see that σ preserves the rewards gained and the probabilities of satisfying ϕ_i for $1{\leq}i{\leq}n$ of σ'. Also, if σ' is a finite memory adversary, then σ is also a finite memory strategy.

A.6 Proof of Theorem 2

Let M_1 and M_2 be PAs. Before we give the proof, we require the following definitions and lemmas. We first define *projections*. For a state $s = (s_1, s_2)$ of $M_1 \| M_2$, the projection of s onto M_i, denoted by $s{\upharpoonright}_{M_i}$, is s_i. We extend this notation to distributions over $S_1{\times}S_2$ of $M_1\|M_2$ in the standard manner. For any path ω of $M_1\|M_2$, the projection of ω onto M_i, denoted $\omega{\upharpoonright}_{M_i}$, is the path obtained from ω by projecting each state of ω onto M_i and removing all the actions not in α_{M_i} together with subsequent states. For any fair adversary σ of $M_1\|M_2$, the *projection* of σ onto M_i, denoted $\sigma{\upharpoonright}_{M_i}$, is the adversary of M_i where, for any finite path ω_i of M_i, $\sigma{\upharpoonright}_{M_i}(\omega)(a, \mu_i)$ equals

$$\frac{\sum\{\!| Pr^\sigma(\omega){\cdot}\sigma(\omega)(a,\mu) \mid \omega \in Path^\sigma_{M_1\|M_2} \wedge \omega{\upharpoonright}_{M_i}{=}\omega_i \wedge \mu{\upharpoonright}_{M_i}{=}\mu_i |\!\}}{Pr^{\sigma{\upharpoonright}_{M_i}}(\omega_i)}$$

Note that, in the lemmas below, the projections onto $M_i[\alpha]$ are well defined since the condition $\alpha \subseteq \alpha_{M_1\|M_2}$ implies that $M_1\|M_2 = M_1[\alpha]\|M_2 = M_1\|M_2[\alpha]$.

Lemma 9. *Let $\mathcal{M}_1, \mathcal{M}_2$ be PAs, $\sigma \in Adv^{\text{fair}}_{\mathcal{M}_1\|\mathcal{M}_2}$, $\alpha \subseteq \alpha_{\mathcal{M}_1\|\mathcal{M}_2}$ and $i = 1, 2$. If ϕ_i and ϕ_i^α are ω-regular properties over $\alpha_{\mathcal{M}_i}$ and $\alpha_{\mathcal{M}_i[\alpha]}$, then:*

$$Pr^\sigma_{\mathcal{M}_1\|\mathcal{M}_2}(\phi_i) = Pr^{\sigma{\upharpoonright}_{\mathcal{M}_i}}_{\mathcal{M}_i}(\phi_i) \quad \text{and} \quad Pr^\sigma_{\mathcal{M}_1\|\mathcal{M}_2}(\phi_i^\alpha) = Pr^{\sigma{\upharpoonright}_{\mathcal{M}_i[\alpha]}}_{\mathcal{M}_i[\alpha]}(\phi_i^\alpha)\,.$$

Furthermore, if ρ and ρ_i^α are reward structures over $\alpha_{\mathcal{M}_i}$ and $\alpha_{\mathcal{M}_i[\alpha]}$, then:

$$ExpTot^\sigma_{\mathcal{M}_1\|\mathcal{M}_2}(\rho_i){=}ExpTot^{\sigma{\upharpoonright}_{\mathcal{M}_i}}_{\mathcal{M}_i}(\rho_i) \ \text{ and } \ ExpTot^\sigma_{\mathcal{M}_1\|\mathcal{M}_2}(\rho_i^\alpha){=}ExpTot^{\sigma{\upharpoonright}_{\mathcal{M}_i[\alpha]}}_{\mathcal{M}_i[\Sigma]}(\rho_i^\alpha).$$

Proof. The result is a simple extension of [23, Lemma 7.2.6, page 141]. □

Lemma 10. *Let $\mathcal{M}_1, \mathcal{M}_2$ be PAs, $\sigma \in Adv^{\text{fair}}_{\mathcal{M}_1\|\mathcal{M}_2}$, $\alpha \subseteq \alpha_{\mathcal{M}_1\|\mathcal{M}_2}$ and $i = 1, 2$. If Ψ_i and Ψ_i^α are qmo-properties for $\mathcal{M}_i$ and $\mathcal{M}_i[\alpha]$, then:*

(a) $\mathcal{M}_1\|\mathcal{M}_2, \sigma \models \Psi_i$ *if and only if* $\mathcal{M}_i, \sigma{\restriction}_{\mathcal{M}_i} \models \Psi_i$*;*
(b) $\mathcal{M}_1\|\mathcal{M}_2, \sigma \models \Psi_i^\alpha$ *if and only if* $\mathcal{M}_i[\alpha], \sigma{\restriction}_{\mathcal{M}_i[\alpha]} \models \Psi_i^\alpha$*.*

Proof. The proof follows from Lemma 9 above. □

Proof (of Theorem 2). We consider only the ASym proof rule, since the Circ rule follows similarly. Therefore suppose $\mathcal{M}_1$ and $\mathcal{M}_2$ are PAs and Ψ_{A_1} and Ψ_G are quantitative multi-objective properties, such that both $M_1 \models_{\text{fair}} \Psi_{A_1}$ and $\langle\Psi_{A_1}\rangle\, M_2\, \langle\Psi_G\rangle$ hold. Now, consider an arbitrary fair adversary σ of $\mathcal{M}_1\|\mathcal{M}_2$. Since $M_1 \models_{\text{fair}} \Psi_{A_1}$, by definition, we have $\mathcal{M}_1, \sigma{\restriction}_{\mathcal{M}_1} \models \Psi_{A_1}$ and therefore, using the assumption $\alpha_{A_1} \subseteq \alpha_{\mathcal{M}_1}$ and applying Lemma 10(a), we have:

$$
\begin{aligned}
\mathcal{M}_1, \sigma{\restriction}_{\mathcal{M}_1} \Psi_A \Rightarrow\;& \mathcal{M}_1\|\mathcal{M}_2, \sigma \models \Psi_A \\
\Rightarrow\;& \mathcal{M}_2[\alpha_A], \sigma{\restriction}_{\mathcal{M}_2[\alpha_A]} \models \Psi_A && \text{by Lemma 10(b) since } \alpha_A \subseteq \alpha_{\mathcal{M}_2[\alpha_A]} \\
\Rightarrow\;& \mathcal{M}_2[\alpha_A], \sigma{\restriction}_{\mathcal{M}_2[\alpha_A]} \models \Psi_G && \text{since } \langle\Psi_A\rangle\, M_2\, \langle\Psi_G\rangle \\
\Rightarrow\;& \mathcal{M}_1\|\mathcal{M}_2, \sigma \models \Psi_G && \text{by Lemma 10(b) since } \alpha_G \subseteq \alpha_{\mathcal{M}_2[\alpha_A]}.
\end{aligned}
$$

Now, since σ was an arbitrary fair adversary of $\mathcal{M}_1\|\mathcal{M}_2$, by definition we have $\mathcal{M}_1\|\mathcal{M}_2 \models_{\text{fair}} \Psi_G$ as required. □