Laws of the Logical Calculi

by

Carroll Morgan and J. W. Sanders

Technical Monograph PRG-78
ISBN 0-902928-58-9

September 1989

Oxford University Computing Laboratory
Programming Research Group
8-11 Keble Road
Oxford OX1 3QD
England

Oxford University Computing Laboratory
Programming Research Group
8-11 Keble Road
Oxford OX1 3QD
England

Laws of the Logical Calculi

Carroll Morgan and J. W. Sanders

September 14, 1989

Abstract

This document records some important laws of classical predicate logic. It is designed as a reservoir to be tapped by *users* of logic, in system development. Though a systematic presentation is attempted, many of the laws appear just because they happen to be useful.

1 Introduction

The formulae of predicate logic are constructed from: *terms*, denoting values (for example, $x+1$); *predicate symbols* denoting properties (for example, *IsEven*); *logical operators* (negation $\neg$, conjunction $\wedge$, disjunction $\vee$, implication $\rightarrow$ and others); and *quantifiers* (*for all* $\forall$ and *there exists* $\exists$). This is such a formula

$$(\forall\ x \bullet IsEven(x) \rightarrow IsOdd(x+1))\,.$$

It happens to be true if we give *IsEven*, *IsOdd*, and $+$ their conventional meanings, and take the quantification $\forall$ to range over integers. The selection of such meanings — conventional or otherwise — is called an *interpretation*. Interpretations fix the meanings of predicate symbols (*IsEven*), function symbols ($+$), constants (1), and the range of quantifiers (x ranges over integers). However, the meaning of the logical operators is the same in every interpretation.

Most of the laws below are of the form $A = B$, where A and B stand for arbitrary formulae. These equalities state that A is true exactly when B is, no matter what the interpretation. Thus the equalities are useful whether reasoning about digital hardware or the Examination Decrees.

Note that $A = B$ is not *itself* a formula, even though A and B are; it is a statement *about* formulae, saying that they are equal — hence interchangeable. (The statement $1 + 1 = 2$ is not an integer, though both $1 + 1$ and 2 are.) So we can as usual in mathematics write chains of equalities $A = B = C = \cdots$ meaning "$A = B$ and $B = C$ and $C = \cdots$".

2 Some propositional laws

Throughout this section A, B and C denote formulae of predicate calculus. The laws are *propositional* because they do not deal with the quantifiers $\forall$ and $\exists$ or with substitution.

2.1 A distributive lattice

The propositional connectives for conjunction, $\wedge$, and disjunction, $\vee$, are idempotent, commutative, associative and absorptive, and they distribute through each other. Thus the formulae modulo $=$ form a what is known as a *distributive lattice*[1].

[1] But you do not need to know this in order to understand them.

2.1.1 Idempotence of $\wedge$ and $\vee$

Conjunction and disjunction are *idempotent* connectives

$$A \wedge A = A = A \vee A. \tag{1}$$

2.1.2 Commutativity of $\wedge$ and $\vee$

Conjunction and disjunction are *commutative* connectives. (Sometimes the adjective *symmetric* is used instead of commutative, but we follow mathematical tradition and reserve that for describing relations rather than operations.)

$$\begin{aligned} A \wedge B &= B \wedge A && (2) \\ A \vee B &= B \vee A. && (3) \end{aligned}$$

2.1.3 Associativity of $\wedge$ and $\vee$

Conjunction and disjunction are *associative* connectives

$$\begin{aligned} A \wedge (B \wedge C) &= (A \wedge B) \wedge C && (4) \\ A \vee (B \vee C) &= (A \vee B) \vee C. && (5) \end{aligned}$$

Laws 1 to 5 mean that we can ignore duplication, order and bracketing in conjunctions $A \wedge B \wedge \cdots \wedge C$ and disjunctions $A \vee B \vee \cdots \vee C$. This enables iterated conjunctions and disjunctions to be expressed in prefix notation indexed by a set[2].

[2] What happens to such notation for connectives which are merely commutative and associative?

2.1.4 Absorption laws

Sometimes terms can be removed immediately from expressions involving both conjunctions and disjunctions; this is ***absorption***

$$A \wedge (A \vee B) = A = A \vee (A \wedge B). \tag{6}$$

2.1.5 Distributive laws

The ***distribution*** of $\wedge$ through $\vee$ is reminiscent of the distribution of multiplication over addition in arithmetic. But in logic, distribution goes both ways: $\vee$ also distributes through $\wedge$

$$A \wedge (B \vee C) \;=\; (A \wedge B) \vee (A \wedge C) \tag{7}$$
$$A \vee (B \wedge C) \;=\; (A \vee B) \wedge (A \vee C). \tag{8}$$

These laws extend to distribution over any finite conjunction or disjunction.

2.2 A Boolean algebra

In the previous laws we can think of $A \wedge B$ as the ***smaller*** of A and B: that is, $\wedge$ yields the minimum of its arguments. Similarly $\vee$ yields the maximum. This intuition is relied upon in the next laws: the constants ***false*** and ***true*** act as minimum and maximum respectively of the distributive lattice, and negation $\neg$ acts as a complement. This converts the lattice into a Boolean algebra.

2.2.1 Maximum and minimum

The constant ***false*** is the minimum of the lattice and ***true*** is the maximum

$$A \wedge \textit{true} \;=\; A \tag{9}$$
$$A \vee \textit{true} \;=\; \textit{true} \tag{10}$$

$$A \wedge \mathit{false} = \mathit{false} \tag{11}$$

$$A \vee \mathit{false} = A. \tag{12}$$

2.2.2 Negation complements the lattice

Negation $\neg$ acts as a *complement* of the distributive lattice

$$A \wedge \neg A = \mathit{false} \tag{13}$$

$$A \vee \neg A = \mathit{true}. \tag{14}$$

Furthermore it is an *involution*

$$\neg\neg A = A \tag{15}$$

and it satisfies De Morgan's laws

$$\neg(A \wedge B) = \neg A \vee \neg B \tag{16}$$

$$\neg(A \vee B) = \neg A \wedge \neg B. \tag{17}$$

These laws extend by induction to negations of finite conjunctions and negations of finite disjunctions[3].

2.2.3 Further absorptive laws

With negation, we have two more absorptive laws

$$A \vee (\neg A \wedge B) = A \vee B \tag{18}$$

$$A \wedge (\neg A \vee B) = A \wedge B. \tag{19}$$

2.2.4 Normal forms

A formula is in *disjunctive normal form* if it is a finite disjunction of other formulae each of which is, in turn, a *conjunction* of single letters or their

[3]For "infinite" ones, we must wait for Laws 82 and 83 in section 4.

negations. (Although those letters may themselves stand for other formulae, we ignore that structure here.) *Conjunctive normal form* is defined complementarily.

Laws 7, 8, 16 and 17 serve to convert any proposition to either disjunctive conjunctive normal form, and Laws 13 and 14 serve to remove adjacent complementary terms. For example

$$\begin{array}{ll} A \wedge \neg(B \wedge C \wedge A) & \\ = \; A \wedge (\neg B \vee \neg C \vee \neg A) & \text{by Law 1} \\ = \; (A \wedge \neg B) \vee (A \wedge \neg C) \vee (A \wedge \neg A) & \text{by Law 7} \\ = \; (A \wedge \neg B) \vee (A \wedge \neg C) \vee \mathit{false} & \text{by Law 13} \\ = \; (A \wedge \neg B) \vee (A \wedge \neg C) & \text{by Law 12.} \end{array}$$

The second formula above is in conjunctive normal form and the third, fourth, and fifth are in disjunctive normal form.

2.3 Implication

Implication $\rightarrow$ can be defined by the law

$$A \rightarrow B \;=\; \neg A \vee B, \tag{20}$$

and this leads on to the laws below:

$$A \rightarrow A \;=\; \mathit{true} \tag{21}$$

$$A \rightarrow B \;=\; \neg(A \wedge \neg B) \tag{22}$$

$$\neg(A \rightarrow B) \;=\; A \wedge \neg B \tag{23}$$

$$A \rightarrow B \;=\; \neg B \rightarrow \neg A. \tag{24}$$

The last above is called the *contrapositive law*. Useful special cases of these are

$$A \rightarrow \mathit{true} \;=\; \mathit{true} \tag{25}$$

$$\mathit{true} \rightarrow A \;=\; A \tag{26}$$

$$A \rightarrow \mathit{false} \;=\; \neg A \tag{27}$$

$$\begin{aligned} \mathit{false} \rightarrow A &= \mathit{true} & (28) \\ A \rightarrow \neg A &= \neg A & (29) \\ \neg A \rightarrow A &= A. & (30) \end{aligned}$$

The next two laws distribute $\rightarrow$ through finite conjunction and disjunction[4].

$$\begin{aligned} C \rightarrow (A \wedge B) &= (C \rightarrow A) \wedge (C \rightarrow B) & (31) \\ (A \vee B) \rightarrow C &= (A \rightarrow C) \wedge (B \rightarrow C). & (32) \end{aligned}$$

The laws above are useful in negating an entire formula: that is, in moving an outer-most negation as far in as possible. For instance

$$\begin{aligned} & \neg(A \wedge (B \rightarrow (\neg A \vee C))) & \\ = \ & \neg A \vee \neg(B \rightarrow (\neg A \vee C)) & \text{by Law 16} \\ = \ & \neg A \vee (B \wedge \neg(\neg A \vee C)) & \text{by Law 23} \\ = \ & \neg A \vee (B \wedge A \wedge \neg C) & \text{by Law 17.} \end{aligned}$$

Here we have reached disjunctive normal form; but we can continue

$$\begin{aligned} = \ & (\neg A \vee B) \wedge (\neg A \vee A) \wedge (\neg A \vee \neg C) & \text{by Law 8} \\ = \ & (\neg A \vee B) \wedge (\neg A \vee \neg C) & \text{by Laws 14, 9} \\ = \ & \neg A \vee (B \wedge \neg C) & \text{by Law 8} \\ = \ & A \rightarrow (B \wedge \neg C) & \text{by Law 20.} \end{aligned}$$

Finally, we have these laws, true in any Boolean algebra

$$\begin{aligned} & A \wedge B = A \\ \text{iff} \ & A \vee B = B \\ \text{iff} \ & \neg A \vee B = \mathit{true}. \end{aligned} \qquad (33)$$

2.3.1 Extra laws of implication

The following laws are useful in showing that successive hypotheses may be conjoined or even reversed

$$\begin{aligned} A \rightarrow (B \rightarrow C) &= (A \wedge B) \rightarrow C & (34) \\ &= B \rightarrow (A \rightarrow C). & (35) \end{aligned}$$

[4] For their infinite counterparts, see Laws 111 and 112.

The next law is the basis of definition by cases:

$$(A \to B) \land (\neg A \to C) = (A \land B) \lor (\neg A \land C). \tag{36}$$

It extends to finitely many A_i, provided they are *pairwise exclusive* ($A_i \land A_j = \mathit{false}$ for $i \neq j$) and *overall exhaustive* ($(\bigvee_i A_i) = \mathit{true}$). Law 36 is often helpful when reasoning about specifications which are expressed like the right-hand side: the first disjunct describes the normal behaviour and the second describes the error case. Nonetheless it may well be easier to reason about the behaviour using implication, and so the law comes into play.

2.4 Other connectives

In this section we consider some other propositional connectives: equivalence, $\leftrightarrow$; exclusive or, $\bigtriangledown$; and the conditional connective with three arguments, **if _ then _ else _** .

2.4.1 Equivalence

Equivalence is defined from first equation below

$$\begin{aligned} A \leftrightarrow B &= (A \to B) \land (B \to A) & (37)\\ &= (A \land B) \lor \neg(A \lor B) & (38)\\ &= \neg A \leftrightarrow \neg B. & (39) \end{aligned}$$

Also

$$\begin{aligned} A \leftrightarrow A &= \mathit{true} & (40)\\ A \leftrightarrow \neg A &= \mathit{false} & (41)\\ A \leftrightarrow \mathit{true} &= A & (42)\\ A \leftrightarrow \mathit{false} &= \neg A & (43)\\ A \to B &= A \leftrightarrow (A \land B) & (44)\\ B \to A &= A \leftrightarrow (A \lor B) & (45)\\ A \lor (B \leftrightarrow C) &= (A \lor B) \leftrightarrow (A \lor C). & (46) \end{aligned}$$

Equivalence is commutative and associative

$$A \leftrightarrow B = B \leftrightarrow A \tag{47}$$

$$A \leftrightarrow (B \leftrightarrow C) = (A \leftrightarrow B) \leftrightarrow C \tag{48}$$

and, from Law 33, satisfies E. W. Dijkstra's *Golden Rule*

$$(A \wedge B \leftrightarrow A \leftrightarrow B \leftrightarrow A \vee B) = \mathit{true}. \tag{49}$$

2.4.2 Exclusive or

Exclusive or, ∇, is particularly useful in the description and development of digital systems. It is defined by one of the following three equivalent equations

$$\begin{aligned} A \nabla B &= \neg(A \leftrightarrow B) && (50)\\ &= (\neg A \wedge B) \vee (A \wedge \neg B) && (51)\\ &= (A \vee B) \wedge \neg(A \wedge B). && (52) \end{aligned}$$

Exclusive or is commutative and associative

$$A \nabla B = B \nabla A \tag{53}$$

$$(A \nabla B) \nabla C = A \nabla (B \nabla C), \tag{54}$$

and satisfies

$$\begin{aligned} A \nabla A &= \mathit{false} && (55)\\ A \nabla \neg A &= \mathit{true} && (56)\\ A \nabla \mathit{true} &= \neg A && (57)\\ A \nabla \mathit{false} &= A && (58)\\ A \wedge (B \nabla C) &= (A \wedge B) \nabla (A \wedge C). && (59) \end{aligned}$$

These laws can be used to prove, for instance, that the three assignments

$$x := x \nabla y; \quad y := x \nabla y; \quad x := x \nabla y$$

suffice to interchange the Boolean values of x and y without the luxury of an intermediate variable.

2.4.3 The conditional

The familiar *if* _ *then* _ *else* _ construct from programming we will write $A \lhd P \rhd B$; it is defined

$$A \lhd P \rhd B \;=\; (P \rightarrow A) \wedge (\neg P \rightarrow B). \tag{60}$$

Its main properties follow from this definition and the previous laws: the binary operator $\_ \lhd P \rhd \_$ is idempotent and associative

$$\begin{aligned} A \lhd P \rhd A &= A & (61)\\ A \lhd P \rhd (B \lhd P \rhd C) &= A \lhd P \rhd C & (62)\\ &= (A \lhd P \rhd B) \lhd P \rhd C & (63) \end{aligned}$$

and it distributes through $\_ \lhd Q \rhd \_$ in both directions

$$\begin{aligned} A \lhd P \rhd (B \lhd Q \rhd C) &= (A \lhd P \rhd B) \lhd Q \rhd (A \lhd P \rhd C) & (64)\\ (A \lhd P \rhd B) \lhd Q \rhd C &= (A \lhd Q \rhd C) \lhd P \rhd (B \lhd Q \rhd C). & (65) \end{aligned}$$

The next pair of laws enable a conditional to be expressed as a truth function of its components

$$\begin{aligned} A \lhd \mathit{true} \rhd B &= A & (66)\\ A \lhd \mathit{false} \rhd B &= B & (67) \end{aligned}$$

and the following laws assist in the simplification of conditional expressions

$$\begin{aligned} A \wedge B &= A \lhd B \rhd B & (68)\\ A \vee B &= A \lhd A \rhd B & (69)\\ \neg A &= \mathit{false} \lhd A \rhd \mathit{true} & (70)\\ \mathit{true} \lhd A \rhd \mathit{false} &= A & (71)\\ A \lhd (B \lhd P \rhd C) \rhd D &= (A \lhd B \rhd D) \lhd P \rhd (A \lhd C \rhd D). & (72) \end{aligned}$$

3 Equality and ordering

With the laws so far, we can show that $A = B$ iff $A \leftrightarrow B = \mathit{true}$. For *only if*, we use Law 40; for *if*, we proceed

$$\begin{array}{lll} & A & \\ = & A \leftrightarrow \mathit{true} & \text{by Law 42} \\ = & A \leftrightarrow (A \leftrightarrow B) & \text{by assumption} \\ = & (A \leftrightarrow A) \leftrightarrow B & \text{by Law 48} \\ = & B & \text{by Laws 40, 42.} \end{array}$$

By analogy, we define the relation $\Rightarrow$ between formulae so that $A \Rightarrow B$ iff $A \rightarrow B = \mathit{true}$. This allows us to write chains $A \Rightarrow B \Rightarrow C \Rightarrow \cdots$ meaning "$A \Rightarrow B$ and $B \Rightarrow C$ and $C \Rightarrow \cdots$".

We have that whenever $A \Rightarrow B$ and $B \Rightarrow A$ then $A = B$ also, and in fact it is easy to show that $\Rightarrow$ is a non-strict partial order over formulae. The next laws show that $A \vee B$ and $A \wedge B$ are upper and lower bounds respectively of A and B

$$A \Rightarrow (A \vee B) \quad \text{and} \quad B \Rightarrow (A \vee B) \tag{73}$$

$$(A \wedge B) \Rightarrow A \quad \text{and} \quad (A \wedge B) \Rightarrow B. \tag{74}$$

Finally, note that all the equalities (33) are equivalent to $A \Rightarrow B$, and thus that $\Rightarrow$ is the usual order on the Boolean algebra.

4 Some predicate laws

In this section we consider laws concerning the universal and existential quantifiers, $\forall$ and $\exists$. Although for most practical purposes we wish the quantification to be *typed*

$$\begin{array}{l} (\forall\, x{:}\, T \bullet P) \\ (\exists\, x{:}\, T \bullet P)\,, \end{array}$$

where T denotes a type and P a predicate, for theoretical purposes (e.g. statement of the completeness theorem) it is more convenient to insist that all quantification be *untyped*

$$(\forall x \bullet P)$$
$$(\exists x \bullet P).$$

We will adopt untyped quantification as our basis; the following laws enable us to convert between the two styles

$$(\forall x: T \bullet A) = (\forall x \bullet x: T \rightarrow A) \tag{75}$$
$$(\exists x: T \bullet A) = (\exists x \bullet x: T \wedge A), \tag{76}$$

where the expressions $x: T$ on the right-hand sides are interpreted as predicates meaning "x is of type T."

Before introducing further laws of the predicate calculus, we discuss *occurrence* and *substitution* of variables.

4.1 Occurrence of variables in formulae

We say informally that a variable occurs in a formula if we can see it written there: for example, x occurs in $x < y$ (we are regarding "$<$" as a two-place predicate). But we need to distinguish *free* and *bound* occurrences of variables in formulae; in the following formula, for example, x occurs free and y occurs bound

$$(\exists y \bullet x < y).$$

Informally, x occurs free in the formula because it is *saying something* about x: that "there is a number greater than it." But the formula does not say anything about y: y just a place-holder. Indeed, we could have used z, writing $(\exists z \bullet x < z)$, and the meaning would have been the same. This is why we say that y occurs bound.

The concepts of free and bound occurrence are found elsewhere in mathematics. In integrals, the "variable of integration" is bound; below we see that x is free and y is bound

$\int_0^1 xy\,dy$.

If we evaluate the integral

$$= \left.\frac{xy^2}{2}\right|_{y=0}^{y=1} = \frac{x}{2}$$

it's clear that the *meaning* of the expression depends only on x. In this sense, evaluating integrals corresponds to "eliminating quantifiers" (Laws 77).

4.2 Substitution

We write substitution of a term t for a variable x in a formula A as

$A[x\backslash t]$

and we write the multiple substitution of terms t and u for variables x and y respectively as

$A[x, y\backslash t, u]$.

In simple cases, such substitutions replace the variable by the term. In more complex cases, however, we must take account of whether variables are free or bound. Suppose, for example, that A is the formula $(\exists\, x \bullet x \neq y) \wedge x = y$; then

$$\begin{array}{rll} & A[x\backslash y] & \text{is} \quad (\exists\, x \bullet x \neq y) \wedge y = y \\ \text{and} & A[y\backslash x] & \text{is} \quad (\exists\, z \bullet z \neq x) \wedge x = x. \end{array}$$

The variable z is *fresh*, not appearing in A. In the first case, $x \neq y$ is unaffected because *that* occurrence of x is bound by $\exists\, x$. Indeed, since we could have used any other letter (except y) without affecting the meaning of the formula — and it would not have been replaced in that case — we do not replace it in this case either. The occurrence of x in $x = y$ is free, however, and the substitution occurs.

In the second case, since both occurrences of y are free, both are replaced by x. But on the left we must not "accidentally" quantify over the newly-introduced x — $(\exists\, x \bullet x \neq x)$ would be wrong — so we change (before the substitution) the bound x to a fresh variable z.

Finally, note that multiple substitution can differ from successive substitution

$$\begin{array}{rrcl} & A[y\backslash x][x\backslash y] & = & (\exists\, z \bullet z \neq y) \wedge y = y \\ \text{but} & A[y, x\backslash x, y] & = & (\exists\, z \bullet z \neq x) \wedge y = x. \end{array}$$

4.3 Eliminating quantifiers

The following *one-point* laws allow quantifiers to be eliminated from formulae

$$(\forall\, x \bullet x = t \rightarrow A) \;\; = A[x\backslash t] = \;\; (\exists\, x \bullet x = t \wedge A)\,. \tag{77}$$

If the type T in Laws 75 and 76 is finite, say $\{a, b\}$, we have similar laws:

$$(\forall\, x\colon \{a, b\} \bullet A) \;\; = \;\; A[x\backslash a] \wedge A[x\backslash b] \tag{78}$$

$$(\exists\, x\colon \{a, b\} \bullet A) \;\; = \;\; A[x\backslash a] \vee A[x\backslash b]. \tag{79}$$

These can be extended to larger (but still finite) types $\{a, b, \cdots, z\}$. We are led to think, informally, of universal and existential quantification as infinite conjunction and disjunction respectively over all the variables of our logic

$$\begin{array}{lll} (\forall\, x \bullet A) & \text{represents} & A(x_1) \wedge A(x_2) \wedge \cdots \wedge A(x_n) \cdots \\ (\exists\, x \bullet A) & \text{represents} & A(x_1) \vee A(x_2) \vee \cdots \vee A(x_n) \cdots. \end{array}$$

Still informally, many of the laws from the section 2 have *infinitary* counterparts, which we investigate in the present section. Throughout, A, B and C denote formulae of predicate calculus, x, y and z denote variables and t denotes a term.

4.4 Quantifiers alone

Quantification is idempotent

$$\begin{aligned}(\forall\, x \bullet (\forall\, x \bullet A)) &= (\forall\, x \bullet A) &(80)\\ (\exists\, x \bullet (\exists\, x \bullet A)) &= (\exists\, x \bullet A)\,. &(81)\end{aligned}$$

Extending De Morgans' laws (16 and 17)

$$\begin{aligned}\neg(\forall\, x \bullet A) &= (\exists\, x \bullet \neg A) &(82)\\ \neg(\exists\, x \bullet A) &= (\forall\, x \bullet \neg A)\,. &(83)\end{aligned}$$

With the laws so far, a formula can be negated. For example here is the negation of that cliché from analysis which expresses continuity of f at c

$$\begin{aligned}&\neg(\forall\, \epsilon > 0 \bullet (\exists\, \delta > 0 \bullet (\forall\, x \bullet \mid x - c \mid < \delta \rightarrow \mid f(x) - f(c) \mid < \epsilon)))\\ =\;& (\exists\, \epsilon > 0 \bullet \neg(\exists\, \delta > 0 \bullet (\forall\, x \bullet \mid x - c \mid < \delta \rightarrow \mid f(x) - f(c) \mid < \epsilon)))\\ =\;& (\exists\, \epsilon > 0 \bullet (\forall\, \delta > 0 \bullet \neg(\forall\, x \bullet \mid x - c \mid < \delta \rightarrow \mid f(x) - f(c) \mid < \epsilon)))\\ =\;& (\exists\, \epsilon > 0 \bullet (\forall\, \delta > 0 \bullet (\exists\, x \bullet \neg(\mid x - c \mid < \delta \rightarrow \mid f(x) - f(c) \mid < \epsilon))))\\ =\;& (\exists\, \epsilon > 0 \bullet (\forall\, \delta > 0 \bullet (\exists\, x \bullet \mid x - c \mid < \delta \wedge \neg(\mid f(x) - f(c) \mid < \epsilon))))\\ =\;& (\exists\, \epsilon > 0 \bullet (\forall\, \delta > 0 \bullet (\exists\, x \bullet \mid x - c \mid < \delta \wedge \mid f(x) - f(c) \mid \geq \epsilon)))\,.\end{aligned}$$

We have used Laws 82, 83, 15 and 23, and have followed standard mathematical practice in writing $\alpha > 0$ for the type statement

$$\alpha\colon \{x\colon \mathsf{R} \mid x > 0\}.$$

4.5 Extending the commutative laws

$$\begin{aligned}(\forall\, x \bullet (\forall\, y \bullet A)) &= (\forall\, y \bullet (\forall\, x \bullet A)) &(84)\\ (\exists\, x \bullet (\exists\, y \bullet A)) &= (\exists\, y \bullet (\exists\, x \bullet A)) &(85)\end{aligned}$$

4.6 Quantifiers accompanied

Extending the associative and previous laws,

$$(\forall x \bullet A \wedge B) = (\forall x \bullet A) \wedge (\forall x \bullet B) \tag{86}$$

$$(\exists x \bullet A \vee B) = (\exists x \bullet A) \vee (\exists x \bullet B) \tag{87}$$

$$(\exists x \bullet A \rightarrow B) = (\forall x \bullet A) \rightarrow (\exists x \bullet B). \tag{88}$$

Here are weaker laws — implications rather than equivalences (recall section 3) — which are nonetheless useful:

$$(\forall x \bullet A) \Rightarrow (\exists x \bullet A) \tag{89}$$

$$(\forall x \bullet A) \vee (\forall x \bullet B) \Rightarrow (\forall x \bullet A \vee B) \tag{90}$$

$$(\forall x \bullet A \rightarrow B) \Rightarrow (\forall x \bullet A) \rightarrow (\forall x \bullet B) \tag{91}$$

$$(\exists x \bullet A \wedge B) \Rightarrow (\exists x \bullet A) \wedge (\exists x \bullet B) \tag{92}$$

$$(\exists x \bullet A) \rightarrow (\exists x \bullet B) \Rightarrow (\exists x \bullet A \rightarrow B) \tag{93}$$

$$(\exists y \bullet (\forall x \bullet A)) \Rightarrow (\forall x \bullet (\exists y \bullet A)). \tag{94}$$

4.7 Manipulation of quantifiers

If a variable has no free occurrences, its quantification is superfluous

$$(\forall x \bullet A) = A \quad \text{if } x \text{ is not free in } A \tag{95}$$

$$(\exists x \bullet A) = A \quad \text{if } x \text{ is not free in } A. \tag{96}$$

Other useful laws of this kind are the following, many of which are specialisations of Laws 86 to 88. In each case, x must not be free in N.

$$(\forall x \bullet N \wedge B) = N \wedge (\forall x \bullet B) \tag{97}$$

$$(\forall x \bullet N \vee B) = N \vee (\forall x \bullet B) \tag{98}$$

$$(\forall x \bullet N \rightarrow B) = N \rightarrow (\forall x \bullet B) \tag{99}$$

$$(\forall x \bullet A \rightarrow N) = (\exists x \bullet A) \rightarrow N \tag{100}$$

$$(\forall x \bullet A \lhd N \rhd B) = (\forall x \bullet A) \lhd N \rhd (\forall x \bullet B) \tag{101}$$

$$(\exists\, x \bullet N \wedge B) = N \wedge (\exists\, x \bullet B) \tag{102}$$
$$(\exists\, x \bullet N \vee B) = N \vee (\exists\, x \bullet B) \tag{103}$$
$$(\exists\, x \bullet N \rightarrow B) = N \rightarrow (\exists\, x \bullet B) \tag{104}$$
$$(\exists\, x \bullet A \rightarrow N) = (\forall\, x \bullet A) \rightarrow N \tag{105}$$
$$(\exists\, x \bullet A \lhd N \rhd B) = (\exists\, x \bullet A) \lhd N \rhd (\exists\, x \bullet B)\,. \tag{106}$$

As mentioned in section 4.1, bound variables behave like dummy variables in mathematics

$$(\forall\, x \bullet A) = (\forall\, y \bullet A[x\backslash y]) \quad \text{if } y \text{ is not free in } A \tag{107}$$
$$(\exists\, x \bullet A) = (\exists\, y \bullet A[x\backslash y]) \quad \text{if } y \text{ not free in } A. \tag{108}$$

We can write these in the more general form

$$(\forall\, x \bullet A[z\backslash x]) = (\forall\, y \bullet A[z\backslash y]) \quad \text{if } x,\, y \text{ not free in } A \tag{109}$$
$$(\exists\, x \bullet A[z\backslash x]) = (\exists\, y \bullet A[z\backslash y]) \quad \text{if } x,\, y \text{ not free in } A. \tag{110}$$

These laws can be used to convert a formula to *prenex normal form.* For example if x and y occur free in A and C but x does not occur free in B,

$$\begin{array}{lll}
& ((\forall\, x \bullet A) \rightarrow \neg\,(\exists\, y \bullet B)) \rightarrow (\forall\, x \bullet (\forall\, y \bullet C)) & \\
= & ((\forall\, x \bullet A) \rightarrow \neg\,(\exists\, a \bullet B[y\backslash a])) \rightarrow (\forall\, b \bullet (\forall\, c \bullet C[x, y\backslash b, c])) & \text{by (108), (107)} \\
= & ((\forall\, x \bullet A) \rightarrow (\forall\, a \bullet \neg B[y\backslash a])) \rightarrow (\forall\, b \bullet (\forall\, c \bullet C[x, y\backslash b, c])) & \text{by Law 83} \\
= & (\forall\, a \bullet (\forall\, x \bullet A) \rightarrow \neg B[y\backslash a]) \rightarrow (\forall\, b \bullet (\forall\, c \bullet C[x, y\backslash b, c])) & \text{by Law 99} \\
= & (\forall\, a \bullet (\exists\, x \bullet A \rightarrow \neg B[y\backslash a])) \rightarrow (\forall\, b \bullet (\forall\, c \bullet C[x, y\backslash b, c])) & \text{by Law 105} \\
= & (\exists\, a \bullet (\forall\, x \bullet (A \rightarrow \neg B[y\backslash a]) \rightarrow (\forall\, b \bullet (\forall\, c \bullet C[x, y\backslash b, c])))) & \text{by (105), (100)} \\
= & (\exists\, a \bullet (\forall\, x \bullet (\forall\, b \bullet (\forall\, c \bullet (A \rightarrow \neg B[y\backslash a]) \rightarrow C[x, y\backslash b, c])))) & \text{by Law 99.}
\end{array}$$

where a, b, c are fresh variables.

Finally, extending the fact that $A \wedge B$ is a lower bound and $A \vee B$ an upper bound for A and B, we have

$$(\forall\, x \bullet A) \;\Rightarrow\; A[x\backslash t] \tag{111}$$
$$A[x\backslash t] \;\Rightarrow\; (\exists\, x \bullet A)\,. \tag{112}$$