Logic Notes

Jonathan S. Ostroff

October 3, 1994

1 Propositional/Sentential Logic - SL

Consider the following deductions (or arguments):

(D1)
If Alfred concentrates on his logic course then he enjoys it
Alfred is concentrating on his logic course
___
Therefore, Alfred is enjoying logic

(D2)
If Alfred concentrates on his logic course then he enjoys it
Alfred is not concentrating on his logic course
___
Therefore, Alfred is not enjoying logic

(D3)
If Prince Charles was married he had a wife
Prince Charles is married
___
Therefore, Prince Charles had a wife

In logic, deductions (or arguments) are often displayed in the above format. The premises (or hypotheses) are given in a column. The conclusion is written underneath seperated from them by a horizontal line.

Deduction D2 is an *incorrect deduction*. Even if the premises would be true the consequence is not necessarily true.

Deduction D1 is a *correct deduction*, i.e. if the hypotheses would be true, then the consequence must also be true. However, D1 may or may not be a *cogent deduction*.

A cogent deduction is a correct deduction that also has true premises. D3 is a cogent deduction, i.e. the first premise is true by the definition of "married", and the second premise is true as a matter of history.

Much of the study of logic is tied up with determining whether deductions are correct or not. Both D1 and D3 are instances of the *symbolic* deduction:

$$\text{(MP — } \textit{Modus Ponens}\text{)} \qquad \begin{array}{l} P \to Q \\ P \\ \hline Q \end{array}$$

It is important for you to know how to translate from an English argument into a symbolic argument, and vice versa — this kind of translation often involves subtle and sometimes quite complex thought processes. Try and translate the following argument into symbolic form:

> If I do not specify the initial conditions then my program will not begin. If I program an infinite loop then my program will not terminate. If the program does not begin or if it does not terminate, then the program will fail. Therefore, if the program fails then either I specified the initial conditions or I must have programmed an infinite loop.

Is the deduction (or argument) correct? Is it cogent?

We will learn how to (a) translate English sentences into symbolic notation, (b) translate Enlish arguments into symbolic deductions, and (c) check whether deductions are correct or not using a proof system.

We can already give some rules for (a):

- Simple statements are assigned uppercase Roman letters, e.g P stands for a particular simple statement such as "Plato is a man". Make sure that P symbolizes an assertion or statement that is either true or false, and not a command, question etc. that have no truth values.
- You must be consistent — P can only represent one statement in a given context.
- Two distinct letters can represent the same statement e.g. both P and Q can stand for "10 is greater than 11".

1.1 Notation

Definition 1 (Symbols of SL)

- *Atomic propositions* are denoted by $P, Q, R, \cdots, P_1, Q_1, \cdots$, i.e. by upper case Roman letters with or without positive numerical subscripts. An atomic proposition such as P_{11} stands for a particular simple statement such as "Plato is a man". Atomic propositions are also called simple sentences or sentence letters.
- *Truth-functional connectives:* $\neg, \wedge, \vee, \to, \equiv$.

- *Punctuation symbols:* (,),[,].

Definition 2 (Well-formed formulas of SL — wffs) We let $\alpha, \beta, \gamma, ...$ be metavariables that stand for any wff of SL. Wffs are defined inductively as follows:

1. All the atomic propositions are wffs of SL.

2. If α is a wff of SL then $\neg\alpha$ is a wff of SL.

3. If α, β are wffs of SL then $(\alpha \wedge \beta), (\alpha \vee \beta), (\alpha \rightarrow \beta), (\alpha \equiv \beta)$ are wffs of SL.

4. Anything that cannot be constructed by finitely many applications of of one or more of rules 1–3 is not a wff of SL.

We may relax the definition of a wff slightly by (a) dropping the outer brackets and (b) allowing the use of square brackets for greater readibility. The negation symbol $\neg$ binds tightest, then $\wedge, \vee$, and then $\rightarrow, \equiv$. Thus $P \wedge \neg Q \rightarrow R$ must be interpreted as $(P \wedge (\neg Q)) \rightarrow R$.

The *main connective* of a wff is the outermost connective linking the outermost components, e.g. the outermost connective of $\neg P \vee (P \rightarrow \neg Q)$ is $\vee$.

Sometimes authors use other symbols. The table below gives some equivalent uses of symbols.

Name	Read	Symbol	Gries	other
Negation	"not"	$\neg$	$\neg$	
Conjunction	"and"	$\wedge$	$\wedge$	&
Disjunction	"or"	$\vee$	$\vee$	
Conditional	"if...then.."	$\rightarrow$	$\Rightarrow$	$\supset$
Biconditional	"iff"	$\equiv$	$=$	$\leftrightarrow$, $\Leftrightarrow$

Gries' use of $=$ for $\equiv$ is particulary confusing. As we will see later, the equality sign is usually used for equality of *terms*. Terms refer to objects in the domain (e.g. $10 = 5+5$ means the object 10 is equal to the object 5+5, i.e. 10 and 5+5 refer to the same object, i.e. number) rather than to assertions (or relations) about objects.

Let H stand for any set of wffs such as $\{P, P \rightarrow Q\}$. Then $H \vdash Q$ means that Q can be *deduced* from the formulas in H, e.g. $\{P, P \rightarrow Q\} \vdash Q$ is an example of a correct deduction. Another way of writing $\{P, P \rightarrow Q\} \vdash Q$ is $P, P \rightarrow Q \vdash Q$. The rules for correct deductions are provided below.

If $H \vdash P$ and H is the empty set (no hypotheses) , then we write $\vdash P$ and we say P is a *theorem*.

The notation $H \vdash P, Q$ stands for "$H \vdash P$ and $H \vdash Q$".

1.2 Basic Rules of inference for SL

In the table below, H is a metavariable standing for any set of wffs. In addition to the rules in the table, there are three other rules that you can always use:

Premise: You can enter any wff you like anywhere in a proof except within a subderivation, with the justification PREMISE.

Reiteration - (R): You can always derive a sentence from itself with the justification (R).

Assumption: You can always "push" into a subderivation with any assumption you like. The wffs within a subderivation are not accessible from outside a subderivation. As soon as you "pop" out of a subderivation, we say that the derivation has been *discharged*.

Introduction Rules	Elimination Rules
$(\wedge I)$ $\begin{array}{l} H \vdash \alpha \\ H \vdash \beta \\ \hline H \vdash \alpha\wedge\beta \end{array}$	$(\wedge E)$ $\begin{array}{l} H \vdash \alpha\wedge\beta \\ \hline H \vdash \alpha \end{array}$ $\quad$ $\begin{array}{l} H \vdash \alpha\wedge\beta \\ \hline H \vdash \beta \end{array}$
$(\vee I)$ $\begin{array}{l} H \vdash \alpha \\ \hline H \vdash \alpha\vee\beta \end{array}$ $\quad$ $\begin{array}{l} H \vdash \alpha \\ \hline H \vdash \beta\vee\alpha \end{array}$ (for any β)	$(\vee E)$ $\begin{array}{l} H \vdash \alpha\vee\beta \\ H \vdash \alpha \to \delta \\ H \vdash \beta \to \delta \\ \hline H \vdash \delta \end{array}$
$(\to I)$ $\begin{array}{l} H, \alpha \vdash \beta \\ \hline H \vdash \alpha \to \beta \end{array}$ (Often called the DEDUCTION rule)	$(\to E)$ $\begin{array}{l} H \vdash \alpha \\ H \vdash \alpha \to \beta \\ \hline H \vdash \beta \end{array}$ (Often called Modus Ponens)
$(\neg I)$ $\begin{array}{l} H, \alpha \vdash \beta \\ H, \alpha \vdash \neg\beta \\ \hline H \vdash \neg\alpha \end{array}$	$(\neg E)$ $\begin{array}{l} H, \neg\alpha \vdash \beta \\ H, \neg\alpha \vdash \neg\beta \\ \hline H \vdash \alpha \end{array}$
$(\equiv I)$ $\begin{array}{l} H \vdash \alpha \to \beta \\ H \vdash \beta \to \alpha \\ \hline H \vdash \alpha \equiv \beta \end{array}$	$(\equiv E)$ $\begin{array}{l} H \vdash \alpha \equiv \beta \\ H \vdash \alpha \\ \hline H \vdash \beta \end{array}$ $\quad$ $\begin{array}{l} H \vdash \alpha \equiv \beta \\ H \vdash \beta \\ \hline H \vdash \alpha \end{array}$

The notation $\dfrac{\alpha}{\beta}$ (with a double line) is an abbreviated way of saying the two rules $\dfrac{\alpha}{\beta}$ and $\dfrac{\beta}{\alpha}$ i.e. you can derive either way.

The following rules can be derived from the already stated rules.

	Derived Rules of SD
Modus Tollens — (MT)	$\begin{array}{l} H \vdash \alpha \to \beta \\ H \vdash \neg\beta \\ \hline H \vdash \neg\alpha \end{array}$
Hypothetical Syllogism — (HS)	$\begin{array}{l} H \vdash \alpha \to \beta \\ H \vdash \beta \to \delta \\ \hline H \vdash \alpha \to \delta \end{array}$ $\quad \begin{array}{l} H \vdash \alpha \equiv \beta \\ H \vdash \beta \equiv \delta \\ \hline H \vdash \alpha \equiv \delta \end{array}$
Disjunctive Syllogism — (DS)	$\begin{array}{l} H \vdash \alpha \vee \beta \\ H \vdash \neg\alpha \\ \hline H \vdash \beta \end{array}$
Commutativity — (CM)	$\begin{array}{l} H \vdash \alpha \vee \beta \\ \hline\hline H \vdash \beta \vee \alpha \end{array}$ $\quad \begin{array}{l} H \vdash \alpha \equiv \beta \\ \hline\hline H \vdash \beta \equiv \alpha \end{array}$ $\quad \begin{array}{l} H \vdash \alpha \wedge \beta \\ \hline\hline H \vdash \beta \wedge \alpha \end{array}$
Associativity — (AS)	$\begin{array}{l} H \vdash \alpha \wedge (\beta \wedge \delta) \\ \hline\hline H \vdash (\alpha \wedge \beta) \wedge \delta \end{array}$ $\quad \begin{array}{l} H \vdash \alpha \vee (\beta \vee \delta) \\ \hline\hline H \vdash (\alpha \vee \beta) \vee \delta \end{array}$ $\quad \begin{array}{l} H \vdash \alpha \equiv (\beta \equiv \delta) \\ \hline\hline H \vdash (\alpha \equiv \beta) \equiv \delta \end{array}$
Implication — (IM)	$\begin{array}{l} H \vdash \alpha \to \beta \\ \hline\hline H \vdash \neg\alpha \vee \beta \end{array}$
Double Negation — (DN)	$\begin{array}{l} H \vdash \alpha \\ \hline\hline H \vdash \neg\neg\alpha \end{array}$
De Morgan — (DM)	$\begin{array}{l} H \vdash \neg(\alpha \wedge \beta) \\ \hline\hline H \vdash \neg\alpha \vee \neg\beta \end{array}$ $\quad \begin{array}{l} H \vdash \neg(\alpha \vee \beta) \\ \hline\hline H \vdash \neg\alpha \wedge \neg\beta \end{array}$
Idempotence — (ID)	$\begin{array}{l} H \vdash \alpha \\ \hline\hline H \vdash \alpha \wedge \alpha \end{array}$ $\quad \begin{array}{l} H \vdash \alpha \\ \hline\hline H \vdash \alpha \vee \alpha \end{array}$
Transposition — (TR)	$\begin{array}{l} H \vdash \alpha \to \beta \\ \hline\hline H \vdash \neg\beta \to \neg\alpha \end{array}$
Exportation — (EX)	$\begin{array}{l} H \vdash \alpha \to (\beta \to \delta) \\ \hline\hline H \vdash (\alpha \wedge \beta) \to \delta \end{array}$
Distribution — (DI)	$\begin{array}{l} H \vdash \alpha \wedge (\beta \vee \delta) \\ \hline\hline H \vdash (\alpha \wedge \beta) \vee (\alpha \vee \delta) \end{array}$ $\quad \begin{array}{l} H \vdash \alpha \vee (\beta \wedge \delta) \\ \hline\hline H \vdash (\alpha \vee \beta) \wedge (\alpha \vee \delta) \end{array}$
Equivalence — (EQ)	$\begin{array}{l} H \vdash \alpha \equiv \beta \\ \hline\hline H \vdash (\alpha \to \beta) \wedge (\beta \to \alpha) \end{array}$ $\quad \begin{array}{l} H \vdash \alpha \equiv \beta \\ \hline\hline H \vdash (\alpha \wedge \beta) \vee (\neg\alpha \wedge \neg\beta) \end{array}$
True and False $false \stackrel{\text{def}}{=} \alpha \wedge \neg\alpha$ $true \stackrel{\text{def}}{=} \alpha \vee \neg\alpha$	$\begin{array}{l} H, \alpha \vdash false \\ \hline H \vdash \neg\alpha \end{array}$ $\begin{array}{l} \\ \hline H \vdash \alpha \vee true \equiv true \end{array}$ $\quad \begin{array}{l} \\ \hline H \vdash \alpha \vee false \equiv \alpha \end{array}$ $\begin{array}{l} \\ \hline H \vdash \alpha \wedge true \equiv \alpha \end{array}$ $\quad \begin{array}{l} \\ \hline H \vdash \alpha \wedge false \equiv false \end{array}$

1.3 Example of using the proof system

Suppose we want to show that $\{D \equiv \neg B, A \rightarrow (\neg\neg R \wedge K), \neg B\} \vdash A \rightarrow [(R \vee H) \wedge D]$

The proof would work as follows (note that assumptions, unlike premises, must be discharged):

1	$D \equiv \neg B$	PREMISE
2	$A \rightarrow (\neg\neg R \wedge K$	PREMISE
3	$\neg B$	PREMISE
4	A	ASSUMPTION
5	$\neg R$	ASSUMPTION
6	$\neg R$	R 5
7	$\neg\neg R \wedge K$	$\rightarrow E$ 2,4
8	$\neg\neg R$	$\wedge E$ 7
9	R	$\neg E$ 5-6, 5-8
10	$R \vee H$	$\vee I$ 9
11	D	$\equiv E$ 1,3
12	$(R \vee H) \wedge D$	$\wedge I$ 10,11
13	$A \rightarrow [(R \vee H) \wedge D]$	$\rightarrow I$ 4-12

2 Predicate Logic — PL

Consider the following argument in which N stands for "the number 2 is even", and L stands for "there is at least one even number".

$$\frac{\text{The number two is even}}{\text{At least one number is even}} \qquad \frac{N}{L}$$

The above argument sounds intuitively correct, and yet we cannot prove it to be correct in SL. Note that if we assign *true* to N and *false* to L, then the premise is true whereas the conclusion is false.

We need a more expressive logic that deals with units smaller than atomic propositions N and L — we need to be able to deal with concrete objects such as the number 2 as well as express ideas such as "at least one object" (i.e. *there exits* at least one number). We need *terms* to denote objects such as numbers and *quantifiers* such as $\exists$ (there exists) which allow us to talk about some or all objects.

2.1 Terms and relations

In mathematics we will often see algebraic assertions such as

$$sin(90 - x) = cos(x)$$

This algebraic law is a complex statement that is composed of many different kinds of objects.

1. First there is the *constant* 90. The number 90 represents one of the objects about which the algebraic law above is expressing some truth.

2. Then there is the *variable* x. This too represents a number but not a fixed one like 90. The variable x stands for any number in the above assertion.

3. Next are the *functions* sin, cos and $-$. Note that $=$ is not a function. Functions take as their arguments numbers and return other numbers. The function sin is a function of arity one, i.e. it takes as its argument a single number (e.g. 90) and returns another number (1); given 30 it would return 0.5. The function $-$ has arity two, i.e. it takes as its arguments two numbers (e.g. 10 and 7) and returns another number (3).

4. Lastly is the *relation* $=$. This is similar to the functions, but instead of taking in numbers and returning numbers, it takes in numbers and returns a *truth value* true or false. Thus $7 = 8$ asserts that 7 and 8 are in the relation equality which is of course false.

   The key difference between a function and relation is that a function is used to refer to objects such as numbers, whereas relations assert something that is either true or false about those objects.

   Similarly, terms *represent objects in the universe of discourse, whereas* wffs *represent relationships that hold between those objects.*

Constants, variables and functions together with their arguments (items 1–3) will be denoted by *terms* of PL, and relations will be denoted by the *atomic predicates* of PL.

2.2 The status of variables

Variables in maths often have an ambiguous status whose resolution depends on the context. For instance the x in

$$(x + y) \cdot (x - y) = (x^2 - y^2) \tag{1}$$

is usually intended to stand for *any* x, and similarly for y. Thus what is really meant is:

$$\forall x \forall y[(x + y) \cdot (x - y) = (x^2 - y^2)]$$

The $\forall$ symbol is the universal quantifier (read:"for all").

The x in

$$\text{Solve} \quad x^2 + 2 \cdot x + 1 = 0 \;\; \text{for x} \tag{2}$$

stands for *some* particular number whose exact value is not yet revealed. If we want to assert: "(2) has a solution", we write

$$\exists x[x^2 + 2 \cdot x + 1 = 0]$$

The symbol $\exists$ is the existential quantifier (read:"there exists").

Consider now the mathematical statement

$$\text{Solve} \quad a \cdot x^2 + b \cdot x + c = 0 \;\; \text{for x} \tag{3}$$

Here a, b, c stand for any number whereas x stands for some particular number, whose precise value will depend on a, b, c. If we want to assert: "(3) has a solution", we write

$$\forall a \forall b \forall c \exists x[a \cdot x^2 + b \cdot x + c = 0]$$

Putting the existential quantifier before the universal quantifier would have a completely different meaning, viz. that all quadratic equations have the exact same solution!

Definition 3 (Symbols of PL) All the symbols of SL are also symbols of PL. However there will be some additional symbols not in SL.

- Constant symbols $a, b, c, \ldots$ (constant symbols can also be subscripted e.g. c_7.
- Variable symbols $v, w, x, \ldots$ (variable symbols can also be subscripted).
- Function symbols $f, g, h, \ldots$ (function symbols can be subscripted). A function symbol can have any arity (i.e. number of arguments), e.g. if f has arity 3, then f has 3 arguments and we write $f(\bullet, \bullet, \bullet)$.
- Relation symbols $p, q, r, \ldots$ (they can also be subscripted and have whatever arity you choose).

  When the arity of a relation symbol is zero (no arguments) then what we really have is a boolean variable (similar to the atomic propositions).

  There is also a distinguished[1] relation symbol $=$ with arity 2.
- Truth-functional connectives: $\neg, \wedge, \vee, \rightarrow, \equiv$.
- Quantifiers: $\forall$, $\exists$.

[1]In any interpretation, the symbol $=$ means equality of terms.

- Punctuation symbols: (,),[,].

We will find it useful to pay attention to the variables of formulas that have not been announced with qunatifiers: the *free-variables.* If t is a term then $FV(t)$ stands for the set of free variables of t. Similarly if α is a formula then $FV(\alpha)$ will denote the free-variables of α.

Definition 4 [Terms of PL].

- Any constant symbol c is a term of PL with $FV(c) = \{\}$.
- Any variable symbol v is a term of PL with $FV(v) = \{v\}$.
- If $t_1, t_2, \cdots, t_n$ are terms of PL and f is any n-ary function symbol then $f(t_1, t_2, \cdots, t_n)$ is a term of PL with $FV(f(t_1, t_2, \cdots, t_n)) = FV(t_1) \bigcup FV(t_2) \bigcup \cdots \bigcup FV(t_n)$.
- No other string of symbols is a term.

If Γ is a set of formulas then

$$FV(\Gamma) \stackrel{\text{def}}{=} \bigcup_{\alpha \in \Gamma} FV(\alpha)$$

Remark: The above definition of the free-variables of a term does not take into account the fact that in "relaxed" notation a *bound-variable* may occur in a term, e.g.

$$\sum_{x=1}^{20} x \cdot (x+1) \cdot y$$

is a term in which the variable x is bound — really x is a dummy variable as the above summation stands for

$$3 \cdot y + 4 \cdot y + \cdots + 460 \cdot y$$

The free-variables of the term $\sum_{x=1}^{20} x \cdot (x+1) \cdot y$ should thus be the singleton set $\{y\}$ (and not $\{y, x\}$).

Definition 5 [Well-formed formulas (wffs) of PL]

- If $t_1, t_2, \cdots, t_n$ are terms of PL and p is any n-ary relation symbol then $p(t_1, t_2, \cdots, t_n)$ is a wff of PL with $FV(p(t_1, t_2, \cdots, t_n)) = FV(t_1) \bigcup FV(t_2) \bigcup \cdots \bigcup FV(t_n)$.

  If p has arity zero, then $FV(p) = \{p\}$. [2]

  The expression $p(t_1, t_2, \cdots, t_n)$ is called an *atomic formula.*

[2]If $n > 0$ the proposition symbol is not included in the list of free-variables, whereas if p is boolean then its symbol is included in the list. The reason for this is that in providing a "state" for a wff (see semantics in later sections), the value of all the variables including boolean variables must be given. In programs, boolean variables have the same status as other variables.

- If α is a wff then $\neg\alpha$ is a wff with $FV(\neg\alpha) = FV(\alpha)$.
- If α, β are wffs of PL then $(\alpha \wedge \beta), (\alpha \vee \beta), (\alpha \to \beta), (\alpha \equiv \beta)$ are also wffs and $FV(\alpha \vee \beta) = FV(\alpha) \bigcup FV(\beta)$ (and similarly for the other connectives).
- If α is a wff and v a variable symbol then $\forall v \alpha$ and $\exists v \alpha$ are also wffs, and $FV(\forall v \alpha) = FV(\exists v \beta) = FV(\alpha) - \{v\}$.

2.3 Safe substitution

Consider the wff α given by $\exists z[y = p(z,z)]$. Let us interpret this wff as follows:

$$\begin{aligned} type(z) &:= \{0, 1, 2, 3, \cdots\} \\ p(x,y) &:= x + y \end{aligned}$$

In "relaxed" notation we can write α as $\exists z(y = z + z)$ which asserts that: "y is an even number". Note that $FV(\alpha) = \{y\}$, i.e. y is the only free-variable of α.

If in α we substitute every free occurrence of y by some other variable say v (this is denoted α^y_v) then we get a new formula (α^y_v) similar to α except y is replaced by v, i.e. we get $\exists z(v = z + z)$. The main point is that α^y_v asserts the same thing about v that $\exists y(y = z + z)$ asserts about y viz. that "v is even".

However, if we substitute y by z then we get a formula with a totally different meaning, viz. we get the formula $\exists z(z = z + z)$ which essentially asserts that "$0 = 0$" — i.e. there is in fact only one z with $z + z = z$ and that is the number zero. What has happened is that the z replacing the y was "captured" illegally by the existential quantifier $\exists z$, thereby changing the meaning of the sentence completely.

We want to have a definition of substitution that is "safe", i.e. avoids this illegal capture of substitued variables by quantifiers. If the quantified variable z is such that $z \in FV(t)$, where t is the replacement term (in this case t is also z — thus the condition $z \in FV(t)$ is satisfied in our example), then compute α^y_z is performed as follows:

- Drop the main outer quantifier of α — in this case $\exists z$ — to get $(y = z + z)$ (note: at this point z becomes free).
- Replace every free occurrence of z by a brand new variable u not occurring anywhere else — we thus get $(y = u + u)$.
- Now do the original replacement of y by z to get $(z = u + u)$.
- Now add back the original quantifier except over the new dummy variable u to get $\exists u(z = u + u)$.

To formally define safe substitution for an arbitrary wff, we first need to define safe subsitution on terms. Since terms are defined inductively, the definition of safe substition on them must also be dfined inductively on the structure of terms, i.e.

Definition 6 (Term Substitution) **Base Case** If the term t' is a constant symbol c then

$$c_t^x \stackrel{\text{def}}{=} c$$

If t' is a variable symbol v then

$$v_t^x \stackrel{\text{def}}{=} \begin{cases} t & \text{if } x = v \\ v & \text{otherwise} \end{cases}$$

Inductive Case If the term t' is $f(t_1, \cdots, t_n)$ then

$$f(t_1, \cdots, t_n)_t^x \stackrel{\text{def}}{=} f({t_1}_t^x, \cdots, {t_n}_t^x)$$

The formal definition of safe substitution for wffs is given inductively on the structure of wffs.

Definition 7 **[Safe substitution]**

Base Case If α is an atomic predicate $p(t_1, \cdots, t_n)$ then

$$p(t_1, \cdots, t_n)_t^x \stackrel{\text{def}}{=} p({t_1}_t^x, \cdots, {t_n}_t^x)$$

Inductive Case

1. If α is a wff then

$$(\neg\alpha)_t^x \stackrel{\text{def}}{=} \neg(\alpha_t^x)$$

2. If α and β are wffs and $\bullet$ is any of the truth functional connectives, then

$$(\alpha \bullet \beta)_t^x \stackrel{\text{def}}{=} \alpha_t^x \bullet \beta_t^x$$

3. Let $\alpha \stackrel{\text{def}}{=} \forall y \beta$ and let t be any term, and u is a new variable not occurring anywhere in either α or t. Then

$$\alpha_t^x \stackrel{\text{def}}{=} \begin{cases} \alpha & \text{if } x = y \\ \forall y(\beta_t^x) & \text{if } y \notin FV(t) \text{ and } x \neq y \\ \forall u[(\beta_u^y)_t^x] & \text{if } y \in FV(t) \text{ and } x \neq y \end{cases}$$

A similar definition of safe substitution can be given for existential operators.

Note that the definition is recursive and thus when computing inner expression e.g. β_t^x the same procedure must be used to avoid illegal capture of quantifiers.

Let α be $\neg[\forall z(y = z)]$. Then

$$\forall y \alpha \to \alpha_z^y \tag{4}$$

really means

$$\forall y[\neg \forall z(y = z)] \to \neg[\forall u(z = u)]$$

which is of course a valid formula. Thus safe substitution preserves the validity of the formula in (4). Intutively, (4) reads: "if α is true for all objects y in the domain of discourse then α is true for any particular object (e.g. z) you wish to choose". Without safe substition some instances of (4) would be invalid. Later we will see a rule of derivation called $\forall E$ whose correctness depends on the validity of (4).

2.4 PL is multi-typed

We assume that each variable v has a "type" (or range of values that it may assume) called $type(v)$. For example, in the wff

$$p \wedge \exists z(z = y + 4) \wedge x = \sqrt{2} \tag{5}$$

we have

$$\begin{aligned} type(p) &= \{true, false\} && (6)\\ type(z) &= \{0, 1, 2, \cdots\} && (7)\\ type(y) &= \{0, 1, 2, \cdots\} && (8)\\ type(x) &= \Re && (9) \end{aligned}$$

Any term t of PL must thus have an associated $type(t)$ which denotes the set of objects that the term may refer to. Thus $(y + 4)$ is a term with $type(y + 4) = \{0, 1, 2, \cdots\}$.

2.5 Simultaneous substitution

A formula such as $\alpha_{t_1,t_2,\cdots,t_n}^{v_1,v_2,\cdots,v_n}$ denotes the simultaneous replacement of each free occurrence of v_i by t_i for each i between 1 and n (making sure not to allow illegal capture of variables). It is important that $type(v_i) = type(t_i)$ when doing the substitution. For example, let α be the wff in (5). Then $\alpha_{x,x+1}^{y,x}$ is invalid because $type(y) \neq type(x)$, i.e. y is a natural number whereas x is real. On the other hand $\alpha_{y+2,x+1}^{y,x}$ is a valid substitution and evaluates to $p \wedge \exists z(z = y + 6) \wedge (x + 1 = \sqrt{2})$.

Note that

$$\begin{aligned} (x + x + y)_{x+y,z}^{x,y} &\equiv (x + y) + (x + y) + z \\ [(x + x + y)_{x+y}^{x}]_z^y &\equiv (x + z) + (x + z) + z \end{aligned}$$

2.6 Relativized quantifiers

$$
\begin{aligned}
(\forall x : \alpha : \beta) &\stackrel{\text{def}}{=} \forall x(\alpha \rightarrow \beta) \\
(\exists x : \alpha : \beta) &\stackrel{\text{def}}{=} \exists x(\alpha \wedge \beta)
\end{aligned}
$$

For example, consider a programming language in which we have the declaration

```
var s: array [1:3] of integer
```

Then to express the assertion: "every element in the array s is greater than 23", we could write:

$$(\forall x : q(1,x) \wedge q(x,3) : r(s(x), 23)) \tag{10}$$

where

$$
\begin{aligned}
type(x) &\;:-\; \text{the integers (1,3 and 23 are integer constants)} \\
q(x,y) &\;:-\; x \text{ is less than or equal to } y \text{ — } q \text{ is a two place relation symbol} \\
s(x) &\;:-\; \text{the } x\text{-th element of array } s \text{ — } s \text{ is a one place function symbol} \\
r(x,y) &\;:-\; x \text{ is greater than or equal to y — } r \text{ is a two-place relation symbol}
\end{aligned}
$$

In a more "relaxed" notation we could write:

$$(\forall x : 1 \leq x \leq 3 : s(x) \geq 23) \tag{11}$$

in which the fact that x is an integer is implicit. Note that $1 \leq x \leq 3$ expresses the range of the array pointer x and $s(x) \geq 23$ the assertion we want to make over that range.

Note that not even (10) is really a wff of PL, as it abbreviates

$$\forall x[q(1,x) \wedge q(x,3)) \rightarrow r(s(x), 23)] \tag{12}$$

which is a wff of PL.

When we write the relaxed statement (11) what we really mean formally is (12)

2.7 Inference Rules of PL

Introduction Rules	Elimination Rules
$(\forall I)$ $\dfrac{H \vdash \alpha}{H \vdash \forall x \alpha}$ Provided $x \notin FV(H)$	$(\forall E)$ $\dfrac{H \vdash \forall x \alpha}{H \vdash \alpha_t^x}$ α_t^x is safe substitution
$\dfrac{H \vdash \alpha_c^x}{H \vdash \forall x \alpha}$ Provided c does not occur in H and α	
$(\exists I)$ $\dfrac{H \vdash \alpha_t^x}{H \vdash \exists x \alpha}$ Partial substitution is allowed	$(\exists E)$ $\dfrac{H \vdash (\exists x)\alpha \quad H, \alpha_c^x \vdash \beta}{H \vdash \beta}$ Provided c does not occur in H, α and β

Derived Rules	
Quantifier Negation — (QN)	$\dfrac{H \vdash \forall x \neg \alpha}{H \vdash \neg \exists x \alpha}$ (double line) $\dfrac{H \vdash \neg \forall x \alpha}{H \vdash \exists x \neg \alpha}$ (double line)
Equivalence Substitution — (ES)	$\dfrac{H \vdash \forall x_1 \cdots \forall x_n [\alpha \equiv \beta]}{H \vdash \delta \equiv \delta_\beta^\alpha}$ where α is a sub-wff of δ and δ_β^α stands for the wff similar to δ except one or more occurrences of the sub-wff α in δ are replaced with β. The variables $x_1, \cdots, x_n$ must include all variables that occur bound in δ.

Rules for equality	
Reflexivity of equality	$\dfrac{}{t = t}$ for any term t
Term substitution — (TS)	$\dfrac{H \vdash t_1 = t_2}{H \vdash \alpha^x_{t_1} \equiv \alpha^x_{t_2}}$ Partial substitution is allowed.
Leibniz' Rule — (LL)	$\dfrac{H \vdash t_1 = t_2}{H \vdash \mathbf{f}(\bullet, \cdots, \bullet)^x_{t_1} = \mathbf{f}(\bullet, \cdots, \bullet)^x_{t_2}}$ for any function $\mathbf{f}(\bullet, \cdots, \bullet)$ with free-variable x. Partial substitution is allowed.
Derived rules for equality	
Variations on Leibniz — (LL)	$\dfrac{H \vdash \neg\alpha^x_{t_1} \quad H \vdash \alpha^x_{t_2}}{H \vdash t_1 \neq t_2}$ provided $x \in FV(\alpha)$, and where $t_1 \neq t_2 \stackrel{\text{def}}{=} \neg(t_1 = t_2)$ and partial substitution is allowed.
Symmetry — (Sm)	$\dfrac{H \vdash t_1 = t_2}{H \vdash t_2 = t_1}$ $\dfrac{H \vdash t_1 \neq t_2}{H \vdash t_2 \neq t_1}$
Transitivity	$\dfrac{H \vdash t_1 = t_2 \quad H \vdash t_2 = t_3}{H \vdash t_1 = t_3}$

Theorems derived from *no* premises	
(T1)	$\vdash \forall x(\alpha \wedge \beta) \equiv (\forall x\alpha) \wedge (\forall x\beta)$
(T2)	$\vdash \exists x(\alpha \vee \beta) \equiv \exists x\alpha) \vee (\exists x\beta)$
(T3)	$\vdash \forall x(\alpha \vee \beta) \equiv \alpha \vee (\forall x\beta)$ Provided that $x \notin FV(\alpha)$
(T4)	$\vdash \exists x(\alpha \wedge \beta) \equiv \alpha \wedge (\exists x\beta)$ Provided that $x \notin FV(\alpha)$
(T5)	$\vdash \forall x(\alpha \rightarrow \beta) \rightarrow [(\forall x\alpha) \rightarrow (\forall x\beta)]$
(T6)	$\vdash (\forall x : \mathit{false} : \alpha) \equiv \mathit{true}$
(T7)	$\vdash (\forall x : \alpha : \mathit{true}) \equiv \mathit{true}$
(T8)	$\vdash (\exists x : \mathit{false} : \alpha) \equiv \mathit{false}$
(T9)	$\vdash (\exists x : \alpha : \mathit{false}) \equiv \mathit{false}$
(T10)	$\vdash \neg(\forall x : \alpha : \beta) \equiv (\exists x : \alpha : \neg\beta)$
(T11)	$\vdash \neg(\exists x : \alpha : \beta) \equiv (\forall x : \alpha : \neg\beta)$
(T12)	$\vdash \exists x \forall y \alpha \rightarrow \forall y \exists x \alpha$ But not the converse

Theorems for equality	
(T15) Transitivity	$\forall x \forall y \forall z[x = y \wedge y = z \rightarrow x = z]$
(T16) Symmetry	$\forall x \forall y[x = y \rightarrow y = x]$
Theorems for integers — Domain Reasoning (DR)	
(T20) Transitivity	$\forall i \forall j \forall k[i \leq j \wedge j \leq k \rightarrow i \leq k]$
(T21) Symmetry	$\forall i \forall j[i \leq j \wedge j \leq i \rightarrow i = j]$
(T22)	$\forall i \forall j[i \leq j \wedge j \leq i \equiv \mathit{false}]$
(T23)	$\forall i \forall j \forall k[i \leq j \rightarrow i + k \leq j + k]$
(T24)	$\forall i \forall j \forall k[k > 0 \rightarrow (i \leq j \rightarrow k \cdot i \leq k \cdot j)]$
(T25)	$\forall i \forall j \forall k[k < 0 \rightarrow (i \leq j \rightarrow k \cdot i \geq k \cdot j)]$
(T26)	$\forall i \forall j[i < j \equiv i + 1 \leq j]$
(T27)	$\forall i \forall j[i \leq j + 1 \equiv i \leq j]$
(T28)	$\forall i \forall j[(k \cdot j) \div j = k]$ Where $\div$ is integer division
(T29)	$\forall i \forall j \forall k[j > 0 \wedge i \leq k \rightarrow (i \div j) \leq (k \div j)]$
(T30) Associativity	$\forall i \forall j \forall k[i + (k + j) = (i + k) + j)]$
(T31) Commutativity	$\forall i \forall j[i + j = j + 1]$
(T32)	$\forall i \forall j[i \neq j \rightarrow i < j \vee j < i]$
(T33)	$1 > 0$
(T34)	$\forall i \forall j[i < j \equiv (\exists z : z > 0 : i + z = j)]$

2.8 Relaxed proofs

We do not usually give a fully formal proof in which each line is justified by one of the rules of inference provided or one of the theorems. Usually we give a proof outline with enough “hints” to convince the reader that a fully formal proof could be constructed.

For example, to prove $\alpha \rightarrow \beta$, we might write

α
$\Rightarrow$ { Hint why $\alpha \rightarrow \delta_1$}
δ_1
$\Rightarrow$ { Hint why $\delta_1 \rightarrow \delta_2$}
δ_2
$\Rightarrow$ { Hint why $\delta_2 \rightarrow \beta$}
β

Example: Consider the following proof of (T6).

$(\forall x : \mathit{false} : \alpha)$
$\Leftrightarrow$ {Definition of relativized quantifiers}
$\forall x[\mathit{false} \rightarrow \alpha]$
$\Leftrightarrow$ {sentential reasoning, i.e. SD}
$\forall x[\neg \mathit{false} \vee \alpha]$
$\Leftrightarrow$ {SD}
$\forall x[\mathit{true} \vee \alpha]$
$\Leftrightarrow$ {SD}
$\forall x[\mathit{true}]$
$\Leftrightarrow$ {combination of $(\forall E)$ and $(\forall I)$, i.e. PD}
true

At each step you should say what kind of reasoning you are using (i.e. SD, PD or DR) and where necesary give a hint as to why the appropriate reaoning is in order. Until you become experienced rather fill in too much justification rather than too little.

3 Semantics of PL

Up until now we have been dealing with the *syntax* of SL and PL including the rules of how to form terms and wffs,the rules for free variables of formulas and the rules for doing derivations in SD and PD.

Now we want to discuss the *semantics* of PL (which includes SL as a subset). The semantics of a wff is its meaning. When we gave an interpretation or symbolization for translating a wff into English we were in fact doing an informal kind of semantics. For example, when we write

$$l(x, y) := \text{“x is less than y”}$$

we were really giving an *interpretation* or semantics for the wff $l(x, y)$.

It is important to realize that a wff such as $l(x, y)$ in and of itself has no meaning — it is merely a string of symbols of PL. The wff $l(x, y)$ only takes on a meaning once an interpretation is given. What is the significance of an interpretation? Given an interpretation for a wff one can evaluate whether the wff is true under that interpretation or not.

Example 1 Consider the wff $[p \vee l(x, f(a, y))]$. An interpretation $\mathcal{I}$ for the wff is made up of two kinds of information: a structure $\mathcal{S}$ and a state-assignment s, i.e. $\mathcal{I} = (\mathcal{S}, s)$.

The structure $\mathcal{S}$ consists of the universe of discourse (i.e. the type of each variable) and the meaning of each constant, function and relation symbol, e.g.

$$\begin{aligned} type(x) &:= \{0,1,2,\cdots\} \\ type(y) &:= \{0,1,2,\cdots\} \\ type(p) &:= \{true, false\} \\ a &:= \text{the number "5" in } type(x) \\ f(x,y) &:= x+y \\ l(x,y) &:= x<y \end{aligned}$$

In the sequel we will write $\mathcal{S}(a)$ to denote the value of a in the structure $\mathcal{S}$ — in this case $\mathcal{S}(a) = 5$. We write $\mathcal{S}(f)$ to denote the concrete function "+" for the nonnegative integers; f on its own is an uninterpreted function symbol having no meaning, it is only $\mathcal{S}(f)$ that has the meaning "plus". We write $\mathcal{S}(l)$ to denote the "less than" relation on the nonnegative integers. In this example, l is an uninterpreted relation symbol. $\mathcal{S}(l)$ stands for the concrete relation $<$, which is defined formally by a set of ordered pairs, i.e.

$$\begin{aligned} \mathcal{S}(l) &\stackrel{\text{def}}{=} \{(x,y)|x<y\} \\ &= \{(0,1),(0,2),\cdots,(1,2),(1,3),\cdots\} \end{aligned}$$

Thus, for example, $\mathcal{S}(l)(4,5)$ holds true precisely when $(4,5) \in \mathcal{S}(l)$.

The state-assignment s is a map that gives each free-variable of of the wff $[p \vee l(x, f(a,y))]$ a value in its corresponding type, e.g.

$$\begin{aligned} x &:= 8 \\ y &:= 4 \\ p &:= false \end{aligned}$$

Another way of denoting the state[3] s is to write

$$s = \{(x,8),(y,4),(p, false)\}$$

The state map gives values only to the free-variables of the wff as the bound occurrences of variables really act as "dummy" variables. If the dummy variable is universally quantified then it is a place holder that stands for all objects in the universe of discourse. If the dummy variable is existentially quantified then it stands for some object in the universe of discourse. Either way, a dummy variable does not need to be given a value by the state map in order for the truth value of the wff to be evaluated.

[3]We will often use the shorter word "state" instead of state-assignment

The wff $[p \vee l(x, f(a, y))]$ when evaluated in the *structure* given above becomes

$$\mathit{false} \text{ or } x < (5 + y)$$

which itself cannot be evaluated (to true or false) unless we also have the values of its free-variables. The value of the free-variables x, y, p is given by the *state* and thus we obtain:

$$\mathit{false} \text{ or } 8 < (5 + 4)$$

which is *true*.

The main point here is that both the structure and the state-assignment are needed to evaluate the truth value of the wff.

Of course, $[p \vee l(x, f(a, y))]$ evaluated in a different interpretation would mean something very different and hence have a different truth value.

3.1 Extension of the state map to a map on terms

A state map s for a wff α is an assignment of values from an appropriate universe of discourse to the set of all free-variables of α. Thus, in Example 1, $s(x) = 8$ where 8 is an object in $type(x)$, $s(y) = 4$ where 4 is an object in $type(y)$, and $s(p) = \mathit{false}$ where *false* is an object in $type(p)$.

Although the state map is defined for any free-variable of α, it is has not yet been defined for a given term t occurring in α. In Example 1, the term $f(a, y)$ under the state s has as yet no defined value — intuitively we would like to give it the value $(5 + 4)$, i.e. 9. What is needed is a definition of the value of an arbitrary term t for a given state s. Since terms were defined recursively in Definition 4, we must define the evalution of an arbitrary term in a state recursively using the recursive definition of terms.

Definition 8 [**Evaluation of a term t in a state s**] Let $\mathcal{I} = (S, s)$ be an interpretation.

Base Case 1. If the term t is a constant c then $s(t) = \mathcal{S}(c)$, where $\mathcal{S}(c)$ is the value of the constant c in the structure $\mathcal{S}$.

2. If the term t is a variable v then $s(t) = s(v)$.

Inductive Case If the term t is $f(t_1, \cdots, t_n)$, where f is any n-ary function symbol and $t_1, \cdots, t_n$ are any terms, then

$$s(t) = \mathcal{S}(f)(s(t_1), \cdots, s(t_n))$$

where $\mathcal{S}(f)$ is the concrete function assigned to the function symbol f by the structure $\mathcal{S}$.

For the term $f(a,y)$ in Example 1 we have by the above defintion

$$\begin{aligned} s(f(a,y)) &= \mathcal{S}(f)(s(a),s(y)) \\ &= s(a)+s(y) \\ &= \mathcal{S}(a)+4 \\ &= 5+4 \\ &= 9 \end{aligned}$$

The state $s' = \{(x,8),(y,7),(p,\mathit{false})\}$ is similar to the state $s = \{(x,8),(y,4),(p,\mathit{false})\}$ at all variables except at the variable y where it has the value 7 instead of 4. We then write

$$s' = (s;\ y\colon 7)$$

The formal definition of the state update is:

Definition 9 (State-update notation) Let s be any state, let x be any variable symbol, and let d be any object in the universe of discourse.

$$(s;\ x\colon d)(v) \stackrel{\text{def}}{=} \begin{cases} d & \text{if } v = x \\ s(v) & \text{otherwise, i.e. if } v \neq x \end{cases}$$

Note that x may not be in the domain of the state map s in which case case the map $(s;\ x\colon d)$ is extended to include x in its domain with $(s;\ x\colon d)(x) = d$. For example, if $s = \{(x,8),(y,4),(p,\mathit{false})\}$ then

$$(s;\ z\colon 19) = \{(x,8),(y,4),(p,\mathit{false}),(z,19)\}$$

Proposition 1 (Term substitution lemma) Let s be any state, let t, t_1 be any terms, and let x be any variable symbol. Then

$$(s;\ x\colon s(t_1))(t) = s(t^x_{t_1})$$

The proof of the above theorem must be done by induction on terms.

3.2 The satisfaction relation

Definition 10 Let $\mathcal{I} = (\mathcal{S}, s)$. The interpretation $(\mathcal{I};\ x\colon d)$ is a *variant-interpretation* of $\mathcal{I}$ which is defined as

$$(\mathcal{I};\ x\colon d) \stackrel{\text{def}}{=} (\mathcal{S}, (s;\ x\colon d))$$

i.e. $(\mathcal{I};\ x\colon d)$ is identical to $\mathcal{I}$ except at x where it has the value d.

We are now in a position to define the satisfaction relation. Informally we say that a wff α is satisfied in an interpretation $\mathcal{I}$ if α evaluates to true in the interpretation, in which case we write $\models^{\mathcal{I}}\alpha$.

Definition 11 (Satisfaction) Let α be any wff and let $\mathcal{I} = (\mathcal{S}, s)$ be an interpretation for the wff. The satisfaction relation $\models^{\mathcal{I}}\alpha$ is defined inductively on the structure of wffs as follows:

Base Case 1. If α is an atomic predicate p, where p has arity zero, then

$$\models^{\mathcal{I}} p \text{ iff } s(p) = \mathit{true}$$

2. If α is an atomic predicate $p(t_1, \cdots, t_n)$, for any terms $t_1, \cdots, t_n$, then

$$\models^{\mathcal{I}} p(t_1, \cdots, t_n) \text{ iff } (s(t_1), \cdots, s(t_n)) \in \mathcal{S}(p)$$

Inductive Case 1. If α is a wff then

$$\models^{\mathcal{I}} \neg\alpha \quad \text{iff} \quad \text{not} \models^{\mathcal{I}} \alpha$$

2. If α and β are wffs then

$$\begin{array}{rcl} \models^{\mathcal{I}}(\alpha\wedge\beta) & \text{iff} & \models^{\mathcal{I}}\alpha \ \text{ and } \ \models^{\mathcal{I}}\beta \\ \models^{\mathcal{I}}(\alpha\vee\beta) & \text{iff} & \models^{\mathcal{I}}\alpha \ \text{ or } \ \models^{\mathcal{I}}\beta \\ \models^{\mathcal{I}}(\alpha\rightarrow\beta) & \text{iff} & [\text{not} \models^{\mathcal{I}}\alpha] \ \text{ or } \ \models^{\mathcal{I}}\beta \\ \models^{\mathcal{I}}(\alpha\equiv\beta) & \text{iff} & \models^{\mathcal{I}}\alpha \ \text{ iff } \ \models^{\mathcal{I}}\beta \end{array}$$

3. If α is a wff then

$$\begin{array}{lcl} \models^{\mathcal{I}}\exists x\alpha & \text{iff} & (\text{there is some d in } \mathit{type}(x)) \models^{(\mathcal{I};x:d)}\alpha \\ \models^{\mathcal{I}}\forall x\alpha & \text{iff} & (\text{for each d in } \mathit{type}(x)) \models^{(\mathcal{I};x:d)}\alpha \end{array}$$

Example 2 Question: Let $\mathcal{I}$ be the interpretation given in Example 1. Check whether $p\vee\forall z[l(z,a)\wedge l(x, f(a,y))]$ is satisfied in $\mathcal{I}$, i.e. check for the truth of

$$\models^{\mathcal{I}} p\vee\forall z[l(z,a)\wedge l(x, f(a,y))]$$

Answer: Let $s' \stackrel{\text{def}}{=} (s;\ z\colon d)$. Then by the inductive definition of satisfaction we have:

$$
\begin{array}{lll}
\models^{\mathcal{I}} p \lor \forall z[l(z,a) \land l(x,f(a,y))] & \texttt{iff} & \models^{\mathcal{I}} p \ \text{ or } \ \models^{\mathcal{I}} \forall z[l(z,a) \land l(x,f(a,y))] \\
 & \texttt{iff} & s(p) \ \text{ or } \ \models^{\mathcal{I}} \forall z[l(z,a) \land l(x,f(a,y))] \\
 & \texttt{iff} & \mathit{true} \ \text{ or } \ \models^{\mathcal{I}} \forall z[l(z,a) \land l(x,f(a,y))] \\
 & \texttt{iff} & \models^{\mathcal{I}} \forall z[l(z,a) \land l(x,f(a,y))] \\
 & \texttt{iff} & (\text{for all d}) \models^{(\mathcal{I};\,z:d)} l(z,a) \land l(x,f(a,y)) \\
 & \texttt{iff} & (\text{for all d}) \ [\models^{(\mathcal{I};\,z:d)} l(z,a) \ \text{ and } \ \models^{(\mathcal{I};\,z:d)} l(x,f(a,y))] \\
 & \texttt{iff} & (\text{for all d}) \ [(s'(z), \mathcal{S}(a)) \in \mathcal{S}(l) \ \text{ and } \ (s'(x), s'(f(a,y))) \in \mathcal{S}(l)] \\
 & \texttt{iff} & (\text{for all d}) \ [(d,5) \in \mathcal{S}(l) \ \text{ and } \ (8, s'(a) + s'(y)) \in \mathcal{S}(l)] \\
 & \texttt{iff} & (\text{for all d}) \ [(d,5) \in \mathcal{S}(l) \ \text{ and } \ (8, s'(a) + s'(y)) \in \mathcal{S}(l)] \\
 & \texttt{iff} & (\text{for all d}) \ [(d,5) \in \mathcal{S}(l) \ \text{ and } \ (8, 5+4) \in \mathcal{S}(l)] \\
 & \texttt{iff} & (\text{for all d}) \ [(d,5) \in \mathcal{S}(l) \ \text{ and } \ (8,9) \in \mathcal{S}(l)] \\
 & \texttt{iff} & [(\text{for all d}) \ ((d,5) \in \mathcal{S}(l)) \ \text{ and } \ (8,9) \in \mathcal{S}(l)] \\
 & \texttt{iff} & [(\text{for all d}) \ ((d,5) \in \mathcal{S}(l)) \ \text{ and } \ \text{true}] \\
 & \texttt{iff} & (\text{for all d}) \ ((d,5) \in \mathcal{S}(l)) \\
 & \texttt{iff} & (\text{for all d}) \ (d < 5) \\
 & \texttt{iff} & \text{false, e.g. if } d = 6 \text{ then } d \not< 5.
\end{array}
$$

Thus, the wff $p \lor \forall z[l(z,a) \land l(x,f(a,y))]$ is *not* satisfied in $\mathcal{I}$.

Proposition 2 (Formula substitution lemma) Let α be any wff, x any variable, t any term, and let $\mathcal{I}$ any interpretation for α. Then

$$\models^{\mathcal{I}} \alpha^x_t \quad \Leftrightarrow \quad \models^{(\mathcal{I};\,x:s(t))} \alpha$$

The proof of the proposition must be done by induction on wffs α.

Definition 12 The following definitions all use the concept of satisfaction. Let Γ be any set off wffs. Then

$\models^{\mathcal{I}} \Gamma$ — **Γ is satisfied in $\mathcal{I}$**

$$\models^{\mathcal{I}} \Gamma \quad \Leftrightarrow \quad (\text{for all } \alpha \in \Gamma) \models^{\mathcal{I}} \alpha$$

$\Gamma \models^{\mathcal{I}} \alpha$ — **Γ entails α in $\mathcal{I}$**

$$\Gamma \models^{\mathcal{I}} \alpha \quad \Leftrightarrow \quad [\models^{\mathcal{I}} \Gamma \ \Rightarrow \ \models^{\mathcal{I}} \alpha]$$

$\models \alpha$ — α **is valid** [4]

$$\models \alpha \quad \Leftrightarrow \quad \text{(for any interpretation } \mathcal{I}) \models^{\mathcal{I}} \alpha$$

$\Gamma \models \alpha$ — Γ **entails** α

$$\Gamma \models \alpha \quad \Leftrightarrow \quad \text{(for any interpretation } \mathcal{I}) \;\; \Gamma \models^{\mathcal{I}} \alpha$$

3.3 Our proof system is sound

Consider the inference rule $(\forall E)$.

$$\frac{\Gamma \vdash \forall x \alpha}{\Gamma \vdash \alpha^x_t}$$

In the syntactic proof system PD we were able to make arguments of the form $\Gamma \vdash \alpha$ where Γ is the set of premises and α the last line of the proof. The proof was done using the rules of inference. How do we know that the rules of inference are correct. i.e. preserve truth?

We now have a way of checking that the rules of inference are indeed sound, i.e. preserve validity. For example, for the $(\forall E)$ rule it is sufficient to check that if Γ entails $\forall x \alpha$ then Γ must also entail α^x_t, i.e. we must check that

$$\Gamma \models \forall x \alpha \quad \Rightarrow \quad \Gamma \models \alpha^x_t \tag{13}$$

The reader may, as an exercise, prove that (13) is equivalent to

$$\forall x \alpha \models \alpha^x_t \tag{14}$$

Example 3 ($(\forall E)$ is sound) Prove that $(\forall E)$ is sound.

Proof: We must show the truth of (14), i.e. by definition we must show the truth of

$$\text{(for each interpretation } \mathcal{I}) \; [\models^{\mathcal{I}} \forall x \alpha \;\; \Rightarrow \;\; \models^{\mathcal{I}} \alpha^x_t]$$

Let $\mathcal{I} = (\mathcal{S}, s)$ be any interpretation, and note that $s(t)$ is some element in $type(x)$.

$\models^{\mathcal{I}} \forall x \alpha$
$\Leftrightarrow$ {By Definition 11 of satisfaction, Inductive Case 3 }
(for all d in $type(x)$) $\models^{(\mathcal{I}; x:d)} \alpha$
$\Rightarrow$ $\{ s(t) \in type(x) \}$
$\models^{(\mathcal{I}; x:s(t))} \alpha$
$\Leftrightarrow$ {Proposition 2: Formula substitution lemma}
$\models^{\mathcal{I}} \alpha^x_t$

[4] In Portararo, quantificationally true.

If we prove the soundness of each rule of inference of SD, then in effect we have proved the soundness of the SD proof system, i.e.

Proposition 3 (SD is sound) For any set off wffs Γ, and any wff α

$$\Gamma \vdash \alpha \;\Rightarrow\; \Gamma \models \alpha$$

3.4 Checking the validity of an argument

Suppose we have a symbolic argument such as

$$\begin{array}{l} \forall x[p(x) \rightarrow q(x)] \\ \underline{p(a) \qquad\qquad\qquad} \\ q(a) \end{array}$$

and we want to check the correctness of the argument, *irrespective of the interpretation chosen*. Obviously we would like to check if

$$\{\forall x[p(x) \rightarrow q(x)], p(a)\} \models q(a)$$

i.e.we would like to check that $\{\forall x[p(x) \rightarrow q(x)], p(a)\}$ entails $q(a)$. There are two ways to do this check:

Use the proof system SD Use the proof system to prove that $q(a)$ can be derived from $\{\forall x[p(x) \rightarrow q(x)], p(a)\}$, i.e. show that $\{\forall x[p(x) \rightarrow q(x)], p(a)\} \vdash q(a)$. Then use soundness of the proof system to argue that indeed $\{\forall x[p(x) \rightarrow q(x)], p(a)\} \models q(a)$ holds.

Find a counterexample Find an interpretation $\mathcal{I}$ in which $\{\forall x[p(x) \rightarrow q(x)], p(a)\} \overset{\mathcal{I}}{\models} q(a)$ *fails* to hold true. Finding just one such failing interpretation is sufficient to show that

$$\text{not} \quad \{\forall x[p(x) \rightarrow q(x)], p(a)\} \models q(a)$$

i.e. the argument is invalid. Note that, the argument might still be satisfiable in another interpretation; however, we can no longer claim that the argument is in general correct.

If you could find even one counterexample to any of the SD inference rules, then SD would be unsound.