

THE LANGUAGE AND SYSTEM

FOEA uses a modification of a natural-language deduction system, like that found in Lemmon's *Beginning Logic*. It is different, first, in that it is dynamic. The language changes as FOEA progresses; some symbols are original (introduced at the beginning), but others are defined later over the course of time, according to rules prescribed here. Secondly, although this is relatively simple, comments are permitted and used to make any appropriate remarks parenthetically. So for instance, it may be noted that a particular theorem is called the Commutative Law of Addition. There are other borrowings from computer science, most especially the notion of an environment, which is described in further detail below.

Symbols

As in any written language, the fundamental atom is a symbol. Here we stipulate that a symbol must be horizontally connected; that is, it is impossible to draw a vertical line through white which divides a symbol in two. The contrary holds as well, and any marking which is horizontally connected is a candidate to be a symbol. For example, "w" is a symbol since it is connected. "i" is a symbol, since only a horizontal line separates the dot from the body. However, "..." is not a (one) symbol, since vertical lines separate the dots.

To be clear, a different font, size, or style (e.g. bold or italic) differentiates symbols. For example, "w", "**w**", "*w*", and "**W**" are all different. "a" (Courier 12 pt plain), "**a**" (Courier 12 pt bold), , "a" (Geneva 12 pt plain), and "a" (Courier 14 pt plain) are all different as well. Case also distinguishes, of course: "a" and "A" are different. Color has not been used in FOEA, but evidently it could also be used to distinguish symbols (or it could be used as a stylistic, clarifying device and *not* distinguish symbols). Position, such as a sub- or super-script, will also (though more rarely) be used to differentiate symbols.

A complete list of symbols, their meanings (very briefly), and their first appearance, is given in Appendices 1 and 2 of this chapter.

Lines

FOEA is divided into lines. Or, in other words, FOEA is a concatenation of lines. Lines are not the same as physical lines, in that a FOEA line may extend over several physical lines (what you see on a page). Each FOEA line is one of five sorts:

a comment,

a definition,

the proposition header (where the proposition which needs to be proved, is stated),

a statement in a proof (statements in proofs are separated by a comma), or

the proof-termination symbol (\square).

A **comment** line begins with ! and ends with ;. The following physical line is an example of a comment line:

! This is a comment. ;

Normally, but not always, the ! symbol is in the first column, and the ; symbol is in the last column. Again, a comment may extend over several physical lines.

A **definition** is one of three sorts: **S**, **D**, or **T**. Each variety will be explained in more detail below.

The **proposition header** asserts a proposition, prior to its proof. For instance, the first proposition header of FOEA is:

$\vdash \forall x \forall y (x = y \Rightarrow y = x)$;

A proposition header begins with the "turnstile" symbol \vdash , which is always in the first column, and terminates with ;. In between there lies a proposition, which has a definite syntax, described below.

A **statement in a proof** usually has the same syntax as found in statements belonging to a proposition header, with one addition. Again, this is described in detail below.

The Environment and Language

An environment is a 4-tuple, which may be written $\mathcal{E} = (\mathcal{L}, \mathcal{P}, \mathcal{W}, \mathcal{A})$. It consists of:

- (1) the language,
- (2) the set of all propositions which have been proven,
- (3) a position marker, specifying where one is; and
- (4) if one is within a proof, a stack.

When one is within a proof, the position marker is the proposition to be proved. Otherwise, it is any symbol representing "out-of-proof."

"Stack" has been chosen because of its computer-science connotations. Over the course of a proof, things may be popped onto the stack and then later popped off, on a last-in, first-out basis. Technically, the stack is a list (t_1, t_2, \dots, t_n) of 2-tuples, i.e. each $t_i = (\mathcal{A}_i, C_i)$. The first tuple \mathcal{A}_i is itself a list, of statements. The second tuple C_i is a set of constants. Intuitively, the each element of the stack represents an environment, and as i increases, the environment gets more and more restricted. \mathcal{A}_i represents the statements which have been proven up to that point in the environment, and C_i the constants which have been introduced.

In order to improve readability, we will write the stack list and the list of statements making up a stack element using " $\langle \dots \rangle$ " instead of " (\dots) ". So a stack list will be written $\langle t_1, t_2, \dots, t_n \rangle$, where each $t_i = (\mathcal{A}_i, C_i)$, and each $\mathcal{A}_i = \langle s_1, s_2, \dots, s_{m_i} \rangle$.

A proof can only terminate when the stack is null, defined to mean the list (of length one) of the 2-tuple, where the first tuple is the empty list (the list of length 0) and the second

tuple is the empty set. This will be written $\langle \langle \rangle, \{\} \rangle$.

The environment is dynamic and will change with each additional line. In particular, FOEA's language can be modified over the course of its derivations, so it is dynamic as well. The initial language is fixed, but subsequent evolutions are not (although the rules of evolution are nonetheless stated in advance, of course).

A language \mathfrak{L} may be thought of as a 6-tuple: of symbols, constants, variables, and three lists of definitions (what we will call \mathfrak{S} -, \mathfrak{D} -, and \mathfrak{T} -definitions). These will be represented by: $\text{Symbol}(\mathfrak{L})$, $\text{Const}(\mathfrak{L})$, $\text{Var}(\mathfrak{L})$, $\mathfrak{S}(\mathfrak{L})$, $\mathfrak{D}(\mathfrak{L})$, and $\mathfrak{T}(\mathfrak{L})$.

Now all well-behaved predicates are things, but not all things are well-behaved predicates. As upper-case letters represent well-behaved predicates and lower-case things, upper-case may be substituted for lower-, but not the reverse, and in a formal characterization of FOEA's language and grammar, which is after all the purpose of this chapter, one should strictly speaking differentiate lower- and upper-case constants, variables, and defined terms. Nonetheless, the formalization of this distinction is at the price of greater complexity. As the existing development is likely to prove already somewhat difficult by its novelty, this additional complication to the formalism has been rejected, although from time to time there may be comments, exterior to the actual technical characterization, about the distinction. Instead, according to the actual formalization appearing in this chapter, FOEA is a standard first-order system, which has numerous advantages, not least of which is that it is the most familiar to logicians and students alike. In any case, the FOEA derivations never substitute small-letters for big-, which can be readily verified by anyone willing to inspect all its proofs.

The initial symbols are listed in Appendix 1 to this chapter. As already stated, new symbols may be introduced, so long as they are horizontally connected. They are listed in Appendix 2.

The initial (lower-case) constants are:

0 1 2 3 4

The initial (upper-case) constants are:

$\mathfrak{N}, \omega, \sigma, \oplus, \ominus, \Lambda, \otimes$

Initial constants are always subsequently constants.

The initial variables are letters, either big or little, and either normal or bold, in Courier 12 point. Big letters are, of course, upper-case, and little lower-case. Subsequently, from time to time, and temporarily, letters may become constants and cease to be variables. They keep their case, e.g. a lower-case variable can only become a lower-case constant. This is what happens when one says "Suppose \mathbf{x} exists." \mathbf{x} , which was up to this point a lower-case variable, ceases to be one and now becomes a lower-case constant. It retains this new character until the " \forall I" rule is invoked, after which it again becomes a variable.

The syntax of \mathfrak{S} -, \mathfrak{D} -, and \mathfrak{T} - definitions are given in Appendix 3. There are no initial definitions.

Logical Rules of Inference

The initial environment consists of: (1) the initial language, (2) the set of axioms, which will be described herein, (3) a marker indicating one is out-of-proof, and (4) the null stack, since one is not within a proof.

As already stated, FOEA is a concatenation of lines, which may be of five sorts. Not all concatenations of lines are permitted, and this section will describe which are.

A comment line may appear at any time, either in- or out-of-proof.

Definition lines are only allowed out-of-proof. The environment changes (it is different before and after the definition line), in that the language changes. One remains out-of-proof.

A proposition header may only be concatenated if one is positioned out-of-proof. It must consist of a statement in the current environment's language. The environment changes, in that one is now in-proof, and the proposition in the header becomes the marker, to show that one is currently trying to prove it.

A statement in a proof is only permitted in-proof and if it follows according to the rules of inference stated herein. It changes the stack and may change the language.

A proof-termination symbol is only allowed if all of the following hold: the stack is a list of length one; the first tuple of this list's one element is itself a list, whose last element is identical to the proposition in the header (which is the position marker); and the second tuple of this list's one element is the empty set. I.e. the stack must be of the form $\langle (\langle s_1, s_2, \dots, s_m \rangle, \{\}) \rangle$, where s_m is the proposition which was to be proved. The stack becomes null, the set of proved propositions adds the proposition in the header, and the position marker indicates that one is now out-of-proof.

Throughout the statement of the rules of inference and the definitions, it will be supposed that the current environment is $\mathcal{E} = (\mathcal{L}, \mathcal{P}, \mathcal{W}, \mathcal{A})$.

Let $\mathcal{P}^* =$ all the propositions in \mathcal{P} plus all the statements which appear in the lists of the stack.

When X is a subset of $\text{Var}(\mathcal{L})$ for some language \mathcal{L} , then \mathcal{L}_X refers to the language exactly like \mathcal{L} , except that $\text{Var}(\mathcal{L}_X) = \text{Var}(\mathcal{L}) \setminus X$ and $\text{Const}(\mathcal{L}_X) = \text{Const}(\mathcal{L}) \cup X$. When \mathcal{L} is a language and X is a subset of $\text{Const}(\mathcal{L})$, then ${}_X\mathcal{L}$ refers to the language exactly like \mathcal{L} , except that $\text{Var}({}_X\mathcal{L}) = \text{Var}(\mathcal{L}) \cup X$ and $\text{Const}({}_X\mathcal{L}) = \text{Const}(\mathcal{L}) \setminus X$. When X is the singleton set $\{y\}$, we may simply write \mathcal{L}_y instead of $\mathcal{L}_{\{y\}}$ and ${}_y\mathcal{L}$ instead of $\{y\}\mathcal{L}$.

Recall that the stack is a (finite) list of 2-tuples, $\langle t_1, t_2, \dots, t_n \rangle$, where $t_i = (\mathcal{A}_i, C_i)$, \mathcal{A}_i being a list of statements

and C_i being a set of constants. We will say a statement S is *added* to a stack $\langle t_1, t_2, \dots, t_{n-1}, t_n \rangle$ to mean that the new stack is $\langle t_1, t_2, \dots, t_{n-1}, t_n^* \rangle$, where $t_n = (\Delta_n, C_n)$ and $t_n^* = (\Delta_n^{\langle S \rangle}, C_n)$. That is, the new stack is identical to the original stack, *and in particular keeps the same length*; the only difference is that the terminal list of statements Δ_n now has added on, as a last element, the statement S . Intuitively, a statement is added to the stack when it has just been proved.

Saying a constant is *added* to the stack has a similar sense, where everything stays the same except that now C_n contains the constant in question. Intuitively, a constant is added to the stack when it has just been introduced.

We will say that a tuple t is *popped onto* a stack to mean that the stack changes, with the new stack identical to the original stack, except there is now an additional element on the list t_{n+1} (indeed, it is the terminal element), with $t_{n+1} = t$. Intuitively, a tuple is popped onto a stack when the environment changes. E.g. when we assume a proposition ("Suppose x is non-zero"), so that we can deduce its consequences, then the environment has changed, and the stack's length increases by 1, to represent this new environment. If the stack is a list of length two or greater, we will say that a tuple t is *popped off* a stack, in order to mean that new stack is the same as the original stack, except it is missing the terminal element, i.e. it equals $(t_1, t_2, \dots, t_{n-1})$. So, the length of the stack decreases by one. E.g. suppose again that we have assumed a proposition "Suppose x is non-zero", and suppose that we have concluded that " x has a predecessor". Then we can infer that "If x is non-zero, then x has a predecessor," and this inference is correct in the *previous* environment, i.e. it does not need the assumption that "Suppose x is non-zero."

Rules of inference impact always the stack and sometimes the language. The only impact of definitions is on the language.

$\&$ Statements, \forall Statements, and the like, are defined in

Appendix 3. The reader can probably nonetheless understand the subsequent without learning the particularities of the definitions.

$[x\backslash a, y\backslash b, \dots\backslash c]$ is used to mean that all instances of x are replaced by a , all instances of y by b , ..., and all instances of z by c . The replacements are simultaneous, so e.g. if b contains instances of z , these z -instances are not replaced by c . Although lower-case letters are used, upper-case may also appear. In order to differentiate lower- and upper-case, it would be necessary to stipulate that any pair around the " \backslash " must be either both lower-case or both upper-case. E.g. either x and a must be both lower-case or both upper-case.

" $\backslash\backslash$ " is used instead of " \backslash " to mean that some but not all substitutions are made. E.g. $[x\backslash\backslash c]$ is used to mean that some (perhaps not all and perhaps no) instances of x are replaced by c . Again, both letters may be upper-case, rather than lower-.

ASSUMPTION

(Prem)

Any statement may be asserted (as a premise).

Remark that a statement may either be a proposition (something which could be proved) or a PremEx statement, i.e. of the sort "Suppose \mathbf{x} exists."

Suppose the statement S is a proposition. Then it, or more precisely $((S), \text{empty set})$ is popped onto the stack. The language remains the same.

On the other hand, suppose S is a PremEx statement, i.e. S is \mathbf{x} for some variable \mathbf{x} . Then the language changes, with the new language being $\mathfrak{L}_{\mathbf{x}}$, where the old (current) language was \mathfrak{L} .

Moreover, $(\langle S \rangle, \{S\})$ is popped onto the stack.

As a convention, Prem statements will be indented two spaces more

than the previous statement (indented in column three if the first statement). Nonetheless, successive PremEx statements can be grouped on one line, like

$\mathbf{x}, \mathbf{y}, \mathbf{z},$! Suppose $\mathbf{x}, \mathbf{y},$ and \mathbf{z} exists. i

rather than like

$\mathbf{x},$! Suppose \mathbf{x} exists. i
 $\mathbf{y},$! Suppose \mathbf{y} exists. i
 $\mathbf{z},$! Suppose \mathbf{z} exists. i

AND INTRODUCTION

(&I)

ϕ
 $\psi \ \& \ \dots \ \& \ \zeta$

 $\psi \ \& \ \dots \ \& \ \phi \ \& \ \dots \ \& \ \zeta$ (note: ϕ may be at one of the borders)

Precisely, if ϕ and $\psi \ \& \ \dots \ \& \ \zeta \in \mathbb{P}^*$ and ϕ, ψ, \dots, ζ are &Statements, then the line

$\psi \ \& \ \dots \ \& \ \phi \ \& \ \dots \ \& \ \zeta$

can be deduced. It is added to the stack.

AND ELIMINATION

(&E)

$\psi \ \& \ \dots \ \& \ \phi \ \& \ \dots \ \& \ \zeta$ (note: ϕ may be at one of the borders)

 ϕ

Precisely, if $\psi \ \& \ \dots \ \& \ \phi \ \& \ \dots \ \& \ \zeta \in \mathbb{P}^*$ and ϕ, ψ, \dots, ζ are &Statements, then the line

ϕ

can be deduced. It is added to the stack.

OR INTRODUCTION

(\vee I)

ϕ

 $\phi \ \vee \ \psi$ or $\psi \ \vee \ \phi$

Precisely, if $\phi \in \mathbb{P}^*$ and ϕ and ψ are \vee Statements, then either the line

$$\phi \vee \psi$$

or the line

$$\psi \vee \phi$$

can be deduced. It is added to the stack.

OR ELIMINATION

(\vee E)

$$\begin{array}{l} \phi \vee \psi \\ \phi \Rightarrow \zeta \\ \psi \Rightarrow \zeta \\ \hline \zeta \end{array}$$

Precisely, suppose $\phi \vee \psi$ and $\phi \Rightarrow \zeta$ and $\psi \Rightarrow \zeta \in \mathbb{P}^*$. Further suppose that ϕ and ψ are \vee Statements and ζ is an $\&$ Statement. Then the line

$$\zeta$$

can be deduced. It is added to the stack.

EQUIVALENCE INTRODUCTION

(\Leftrightarrow I)

$$\begin{array}{l} \phi \Rightarrow \psi \\ \psi \Rightarrow \phi \\ \hline \phi \Leftrightarrow \psi \end{array}$$

Precisely, if $\phi \Rightarrow \psi$ and $\psi \Rightarrow \phi \in \mathbb{P}^*$ and ϕ and ψ are $\&$ Statements, then the line

$$\phi \Leftrightarrow \psi$$

can be deduced. It is added to the stack.

EQUIVALENCE ELIMINATION

(\Leftrightarrow E)

$$\begin{array}{c} \phi \Leftrightarrow \psi \\ \hline \phi \Rightarrow \psi \text{ or } \psi \Rightarrow \phi \end{array}$$

Precisely, if $\phi \Leftrightarrow \psi \in \mathbb{P}^*$ and ϕ and ψ are &Statements, then either the line

$$\phi \Rightarrow \psi$$

or the line

$$\psi \Rightarrow \phi$$

can be deduced. It is added to the stack.

IMPLICATION INTRODUCTION

(\Rightarrow I)

Assume ϕ and ψ are &Statements. Suppose ϕ is the first element and ψ is the last element of the first tuple of the stack's last element, i.e. $\phi = s_1$ and $\psi = s_m$, where $\mathcal{A}_n = (s_1, s_2, \dots, s_m)$, where n is the length of the stack list. (It is possible that $m = 1$ and that $\phi = \psi$.) Further suppose that no constant appearing in ψ belongs to C_n . Then $\phi \Rightarrow \psi$ can be deduced. (\mathcal{A}_n, C_n) is popped off the stack, and $\phi \Rightarrow \psi$ is added. The language changes, with the new language being $C_n \mathcal{L}$, where the old (current) language was \mathcal{L} .

Remark that the condition about C_n (about no constant appearing in ψ belonging to C_n) is to prevent the following reasoning:

$$\begin{array}{l} \exists y \ x = y \quad , ! \quad \text{Prem} \quad ; \\ x = \mathbf{y} \quad , ! \quad \exists E \quad ; \\ \exists y \ x = y \Rightarrow x = \mathbf{y} \\ \quad , ! \quad \Rightarrow I \quad ; \end{array}$$

As will be noted below, \mathbf{y} is added to the stack (i.e. it is added to C_n), in order to forestall this erroneous conclusion.

As a convention, statements justified by Implication Introduction will be indented two spaces less than the previous statement.

MODUS PONENS/IMPLICATION ELIMINATION

(\Rightarrow E)

$$\begin{array}{c} \phi \Rightarrow \psi \\ \phi \\ \hline \psi \end{array}$$

Precisely, if $\phi \Rightarrow \psi$ and $\phi \in \mathcal{P}^*$ and ϕ and ψ are $\&$ Statements, then the line

ψ

can be deduced. It is added to the stack.

ABSURDITY INTRODUCTION

(\mathfrak{F} I)

$$\begin{array}{c} \phi \\ \neg\phi \\ \hline \mathfrak{F} \end{array}$$

Precisely, if ϕ and $\neg\phi \in \mathcal{P}^*$ and ϕ is a \neg Statement, then the line

\mathfrak{F}

can be deduced. It is added to the stack.

NEGATION INTRODUCTION

(\neg I)

$$\begin{array}{c} \phi \Rightarrow \mathfrak{F} \\ \hline \neg\phi \end{array}$$

Precisely, if $\phi \Rightarrow \mathfrak{F} \in \mathbb{P}^*$ and ϕ is an &Statement, then the line

$\neg\phi$

can be deduced. It is added to the stack.

NEGATION ELIMINATION

(\neg E)

$\neg\neg\phi$

 ϕ

Precisely, if $\neg\neg\phi \in \mathbb{P}^*$ and ϕ is a \neg Statement, then the line

ϕ

can be deduced. It is added to the stack.

UNIVERSAL QUANTIFICATION INTRODUCTION

(\forall I)

If ϕ has been deduced from the assumption that a, b, \dots, c exist, then

$\forall x \forall y \dots \forall z \phi [a \setminus x, b \setminus y, \dots, c \setminus z]$

can be deduced.

Precisely, suppose the last elements of the stack list are t_1, \dots, t_n , where $n-i+1$ is the number of variables appearing in the list x, y, \dots, z or the number of constants in the list

a, b, \dots, c . Let $t_j = (\mathcal{A}_j, C_j)$ and $\mathcal{A}_j = \langle s_1^j, s_2^j, \dots, s_{m_j}^j \rangle$ for

$i \leq j \leq n$. And suppose $s_1^i = a, s_1^{i+1} = b, \dots, s_1^n = c,$

$s_{m_n}^n = \phi$, which is assumed to be a BlockStatement. And suppose

$C_i \cup C_{i+1} \cup \dots \cup C_n$ does not include any symbol which appears

in $\phi[a \setminus x, b \setminus y, \dots, c \setminus z]$. Finally, suppose that x, y, \dots, z are variables in the current language and do not appear in ϕ . Then (call this the statement S)

$$\forall x \forall y \dots \forall z \phi[a \setminus x, b \setminus y, \dots, c \setminus z]$$

can be deduced.

The elements t_n, \dots, t_i are popped off the stack, and the proposition S is added. The language changes, with the new language being $C \mathfrak{L}$, where the old (current) language is \mathfrak{L} , and C being $C_i \cup C_{i+1} \cup \dots \cup C_n$.

The above description allows for single or multiple generalizations. Here is an example of two single generalizations:

\mathbf{P}	,!	1 Prem	;
\mathbf{x}	,!	2 Prem	;
$\mathbf{P}[\mathbf{x}]$,!	3 Prem	;
$\mathbf{P}[\mathbf{x}] \Rightarrow \mathbf{P}[\mathbf{x}]$,!	4 \Rightarrow I	;
$(\mathbf{P}[\mathbf{x}] \Rightarrow \mathbf{P}[\mathbf{x}])$,!	5 $()$ I	;
$\forall \mathbf{x}(\mathbf{P}[\mathbf{x}] \Rightarrow \mathbf{P}[\mathbf{x}])$,!	6 \forall I	;
$\mathbf{P} \subseteq \mathbf{P}$,!	7 \mathfrak{S} I	;
$\forall \mathbf{P} \mathbf{P} \subseteq \mathbf{P}$!	8 \forall I	;

The stack behaves as follows, supposing the initial stack to be $\langle \langle \rangle, \{ \} \rangle$, and indicating the stack existing at the end of each statement. Quotation marks are used around statements to increase readability.

- 1: $\langle \langle \mathbf{P} \rangle, \{ \mathbf{P} \} \rangle$
- 2: $\langle \langle \mathbf{P} \rangle, \{ \mathbf{P} \} \rangle, \langle \langle \mathbf{x} \rangle, \{ \mathbf{x} \} \rangle$
- 3: $\langle \langle \mathbf{P} \rangle, \{ \mathbf{P} \} \rangle, \langle \langle \mathbf{x} \rangle, \{ \mathbf{x} \} \rangle, \langle \text{"P[x]"}, \{ \} \rangle$
- 4: $\langle \langle \mathbf{P} \rangle, \{ \mathbf{P} \} \rangle, \langle \langle \mathbf{x}, \text{"P[x] } \Rightarrow \text{P[x]}" \rangle, \{ \mathbf{x} \} \rangle$
- 5: $\langle \langle \mathbf{P} \rangle, \{ \mathbf{P} \} \rangle, \langle \langle \mathbf{x}, \text{"P[x] } \Rightarrow \text{P[x]"}, \text{"(P[x] } \Rightarrow \text{P[x])"} \rangle, \{ \mathbf{x} \} \rangle$

6: $\langle \langle \mathbf{P}, "\forall \mathbf{x}(\mathbf{P}[\mathbf{x}] \Rightarrow \mathbf{P}[\mathbf{x}])" \rangle, \{\mathbf{P}\} \rangle$

7: $\langle \langle \mathbf{P}, "\forall \mathbf{x}(\mathbf{P}[\mathbf{x}] \Rightarrow \mathbf{P}[\mathbf{x}])", "\mathbf{P} \subseteq \mathbf{P}" \rangle, \{\mathbf{P}\} \rangle$

8: $\langle \langle "\forall \mathbf{P} \mathbf{P} \subseteq \mathbf{P}" \rangle, \{\} \rangle$

Here is an example of a multiple generalization. Notice that the first physical line in fact includes two lines, with one line separator (i.e. the comma) between the " \mathbf{x} " and the " \mathbf{y} ".

\mathbf{x}, \mathbf{y}	,!	1 Prem	;
$\mathbf{x} = \mathbf{y}$,!	2 Prem	;
$\mathbf{x} = \mathbf{x}$,!	3 =I	;
$\mathbf{y} = \mathbf{x}$,!	4 =E	;
$\mathbf{x} = \mathbf{y} \Rightarrow \mathbf{y} = \mathbf{x}$,!	5 \Rightarrow I	;
$(\mathbf{x} = \mathbf{y} \Rightarrow \mathbf{y} = \mathbf{x})$,!	6 ()I	;
$\forall \mathbf{x} \forall \mathbf{y} (\mathbf{x} = \mathbf{y} \Rightarrow \mathbf{y} = \mathbf{x})$!	7 \forall I	;

The stack behaves as follows, supposing the initial stack to be $\langle \langle \rangle, \{\} \rangle$, and indicating the stack existing at the end of each statement.

1: $\langle \langle \mathbf{x} \rangle, \{\mathbf{x}\} \rangle, \langle \langle \mathbf{y} \rangle, \{\mathbf{y}\} \rangle$

2: $\langle \langle \mathbf{x} \rangle, \{\mathbf{x}\} \rangle, \langle \langle \mathbf{y} \rangle, \{\mathbf{y}\} \rangle, \langle \langle \mathbf{x} = \mathbf{y} \rangle, \{\} \rangle$

3: $\langle \langle \mathbf{x} \rangle, \{\mathbf{x}\} \rangle, \langle \langle \mathbf{y} \rangle, \{\mathbf{y}\} \rangle, \langle \langle \mathbf{x} = \mathbf{y}, \mathbf{x} = \mathbf{x} \rangle, \{\} \rangle$

4: $\langle \langle \mathbf{x} \rangle, \{\mathbf{x}\} \rangle, \langle \langle \mathbf{y} \rangle, \{\mathbf{y}\} \rangle, \langle \langle \mathbf{x} = \mathbf{y}, \mathbf{x} = \mathbf{x}, \mathbf{y} = \mathbf{x} \rangle, \{\} \rangle$

5: $\langle \langle \mathbf{x} \rangle, \{\mathbf{x}\} \rangle, \langle \langle \mathbf{y}, \mathbf{x} = \mathbf{y} \Rightarrow \mathbf{y} = \mathbf{x} \rangle, \{\mathbf{y}\} \rangle$

6: $\langle \langle \mathbf{x} \rangle, \{\mathbf{x}\} \rangle, \langle \langle \mathbf{y}, \mathbf{x} = \mathbf{y} \Rightarrow \mathbf{y} = \mathbf{x},$
 $\mathbf{x} = \mathbf{y} \Rightarrow \mathbf{y} = \mathbf{x} \rangle, \{\mathbf{y}\} \rangle$

7: $\langle \langle \langle \forall \mathbf{x} \forall \mathbf{y} (\mathbf{x} = \mathbf{y} \Rightarrow \mathbf{y} = \mathbf{x}) \rangle, \{\} \rangle$

Note that the restriction about C_n not including any other constants contained in ϕ is necessary, to prevent the following reasoning:

$\forall \mathbf{x} \exists \mathbf{y} \mathbf{x} = \mathbf{y}$,!	Prem	;
\mathbf{x}	,!	Prem	;
$\exists \mathbf{y} \mathbf{x} = \mathbf{y}$,!	\forall E	;
$\mathbf{x} = \mathbf{y}$,!	\exists E	;
$\forall \mathbf{x} \mathbf{x} = \mathbf{y}$,!	\forall I	;

$$\begin{array}{l} \exists y \forall x x = y \quad ,! \quad \exists I \quad ; \\ \forall x \exists y x = y \Rightarrow \exists y \forall x x = y \quad ,! \quad \text{Prem} \quad ; \end{array}$$

As will be noted below, y is added to the stack (C_n), in order to forestall this erroneous conclusion.

As a convention, statements justified by Universal Quantification Introduction will be indented two spaces less than the previous statement.

UNIVERSAL QUANTIFICATION ELIMINATION

($\forall E$)

Suppose a, b, \dots, c are terms and x, y, \dots are variables. Further suppose a, b, \dots, c are on the list of successfully referring terms in the current environment. Then:

$$\frac{\forall x \forall y \dots \forall z \phi}{\phi [x \setminus a, y \setminus b, \dots, z \setminus c]}$$

Precisely, suppose that $\forall x \forall y \dots \forall z \phi \in \mathbb{P}^*$, that ϕ is a BlockStatement, and that a, b, \dots, c are proven to be successfully referring terms (see below). Then $\phi [x \setminus a, y \setminus b, \dots, z \setminus c]$ may be deduced. It is added to the stack.

We will define what it means to prove a term is successfully referring much later. Suffice to say here that most terms are inherently successfully referring, so do not have to be proved such. Those which are not, are terms defined with a supposition, and these terms are proven to be successfully referring when the supposition is proven. For example, if in order to write (3') we must know that 3 is a natural number, we must have proven that 3 is a natural number before we can substitute the term (3') for a universally quantified variable.

EXISTENTIAL QUANTIFICATION INTRODUCTION

(\exists I)

ϕ where ϕ contains constant c

 $\exists x\phi [c \setminus x]$

Precisely, suppose constant c is successfully referring and appears in $\phi \in \mathcal{P}^*$, which is a BlockStatement. Further suppose that x is a variable. Then

$\exists x\phi [c \setminus x]$

can be deduced. It is added to the stack.

EXISTENTIAL QUANTIFICATION ELIMINATION

(\exists E)

$\exists x\phi$

 $\phi [x \setminus c]$ for some available c .

Precisely, suppose $\exists x\phi \in \mathcal{P}^*$, and that ϕ is a BlockStatement. Further suppose that c is a variable. Then

$\phi [x \setminus c]$

can be deduced. It is added to the stack. The variable c is also added to the stack. The language changes, with the new language being $\mathcal{L}\{c\}$, where the old (current) language is \mathcal{L} .

IDENTITY INTRODUCTION

(=I)

 $c = c$ where c is a successfully referring term

Precisely, suppose c is a successfully referring term in the current environment. Then $c = c$ can be deduced. It is added to the stack.

IDENTITY ELIMINATION

(=E)

$a = b$

ϕ

$\phi[a \setminus b]$ or $\phi[b \setminus a]$

Precisely, suppose a and b are terms in the current language, and that proposition $\phi \in \mathcal{P}^*$. Then either $\phi[a \setminus b]$ or $\phi[b \setminus a]$ can be deduced. It is added to the stack.

PARENTHESSES INTRODUCTION

(()I)

ϕ

(ϕ)

Precisely, if proposition $\phi \in \mathcal{P}^*$, then (ϕ) can be deduced. It is added to the stack.

PARENTHESSES ELIMINATION

(()E)

(ϕ)

ϕ

Precisely, if proposition $(\phi) \in \mathcal{P}^*$, then ϕ can be deduced. It is added to the stack.

EXCHANGE

(Exch)

A dummy variable in a term $\{x, y, \dots, z : \psi\}$ can be replaced by any other variable which does not appear in ψ .

Precisely, suppose that term $t = \{x, y, \dots, z : \psi\}$ appears in proposition $\phi \in \mathbb{P}^*$. (y may be the first or last of the variable list.) Also suppose that variable v does not appear in ϕ , and let s be $\{x, v, \dots, z : \psi[y \setminus v]\}$. Then $\phi[t \setminus s]$ can be deduced. It is added to the stack.

PREDICATES

(Pred)

$$\text{-----}$$

$$\forall a \forall b \dots \forall c (\{x, y, \dots, z : \phi\}[a, b, \dots, c] \Leftrightarrow \phi[a \setminus x, b \setminus y, \dots, c \setminus z])$$

where ϕ is a \Rightarrow PStatement($\mathfrak{L}, (x, y, \dots, z)$), \mathfrak{L} being the current language.

(A \Rightarrow PStatement has no variables in a predicate position. A precise definition is given in Appendix 3.)

Precisely, let \mathfrak{L} be the current language, and let x, y, \dots, z and a, b, \dots, c be disjoint lists of variables (of the same length and with no duplicates). Suppose ϕ is a \Rightarrow PStatement($\mathfrak{L}, (x, y, \dots, z)$), which does not contain any of the a, b, \dots, c . Then

$$\forall a \forall b \dots \forall c (\{x, y, \dots, z : \phi\}[a, b, \dots, c] \Leftrightarrow \phi[a \setminus x, b \setminus y, \dots, c \setminus z])$$

can be deduced. It is added to the stack.

Mathematical Axioms

(Count0)

$$\forall n \forall C \forall x \forall y \forall z (\mathfrak{C}[n, C] \ \& \ C[x, y] \ \& \ C[z, y] \Rightarrow x = z)$$

(Count1a)

$$\forall n \forall C (\mathfrak{C}[n, C] \ \& \ \neg n = 0 \Rightarrow \exists a \ C[n, a])$$

(Count1b)

$$\begin{aligned} \forall n \forall m \forall k \forall C \forall a (\ \omega[n] \ \& \ \omega[m] \ \& \ \mathfrak{C}[n, C] \ \& \ \neg \exists x \ C[x, a] \\ \ \& \ \mathfrak{C}[m, \{x, y : C[x, y] \vee (x = m \ \& \ y = a)\}] \\ \ \& \ \mathfrak{C}[k, \{x, y : C[x, y] \vee (x = k \ \& \ y = a)\}] \\ \Rightarrow m = k) \end{aligned}$$

(Count2)

$$\forall C (\mathfrak{C}[0, C] \Leftrightarrow \neg \exists x \exists y \ C[x, y])$$

(Count3)

$$\begin{aligned} \forall n \forall m \forall C \forall D \forall a \quad & (\omega[n] \ \& \ \sigma[n,m] \ \& \ \neg \exists x \ C[x,a] \\ & \ \& \ \forall x \forall y \ (D[x,y] \Leftrightarrow C[x,y] \vee (x = m \ \& \ y = a)) \\ & \Rightarrow (\mathfrak{C}[n,C] \Leftrightarrow \mathfrak{C}[m,D])) \end{aligned}$$

($\omega 0$)

$\omega[0]$

(CountExist)

$$\begin{aligned} \forall n \forall C \forall a \quad & (\omega[n] \ \& \ \mathfrak{C}[n,C] \ \& \ \neg \exists x \ C[x,a] \\ & \Rightarrow \exists m \ (\omega[m] \ \& \ \mathfrak{C}[m, \{x,y : C[x,y] \vee (x = m \ \& \ y = a)\}])) \end{aligned}$$

(Numb)

$$\forall n \forall P \ (\mathfrak{N}[n,P] \Leftrightarrow \exists C \ (\mathfrak{C}[n,C] \ \& \ \forall y \ (\exists x \ C[x,y] \Leftrightarrow P[y])))$$

ONE, TWO, THREE, AND FOUR

(One)

$\sigma[0,1]$

(Two)

$\sigma[1,2]$

(Three)

$\sigma[2,3]$

(Four)

$\sigma[3,4]$

ADDITION

(Add)

$$\begin{aligned} \forall n \forall m \forall k \forall A \forall B \quad & (\omega[n] \ \& \ \omega[m] \ \& \ \omega[k] \ \& \ \mathfrak{N}[n,A] \ \& \ \mathfrak{N}[m,B] \\ & \ \& \ \neg \exists x \ (A[x] \ \& \ B[x]) \\ & \Rightarrow (\oplus[n,m,k] \Leftrightarrow \mathfrak{N}[k, \{x : A[x] \vee B[x]\}])) \end{aligned}$$

INEQUALITY

(Ineq1)

$$\begin{aligned} \forall n \forall m \forall A \forall B \quad & (\mathfrak{N}[n,A] \ \& \ \mathfrak{N}[m,B] \ \& \ \forall x \ (A[x] \Rightarrow B[x]) \\ & \Rightarrow \Lambda[n,m]) \end{aligned}$$

(Ineq2)

$$\begin{aligned} \forall n \forall m \forall A \forall B \quad & (\Lambda[n,m] \ \& \ \mathfrak{N}[n,A] \ \& \ \mathfrak{N}[m,B] \ \& \ \forall x \ (B[x] \Rightarrow A[x]) \\ & \Rightarrow \forall x \ (A[x] \Leftrightarrow B[x])) \end{aligned}$$

SUBTRACTION

(Subtr)

$$\begin{aligned} \forall n \forall m \forall k \forall A \forall B \quad & (\omega[n] \ \& \ \omega[m] \ \& \ \omega[k] \ \& \ \mathfrak{N}[n,A] \ \& \ \mathfrak{N}[m,B] \\ & \ \& \ \forall x \ (A[x] \Rightarrow B[x]) \\ & \Rightarrow (\ominus[m,n,k] \Leftrightarrow \mathfrak{N}[k, \{x : B[x] \ \& \ \neg A[x]\}])) \end{aligned}$$

MULTIPLICATION

(Mult)

$$\begin{aligned} & \forall n \forall m \forall k \forall P \forall R (\omega[n] \ \& \ \omega[m] \ \& \ \omega[k] \ \& \ \mathfrak{N}[n,P] \\ & \quad \& \ \forall x (P[x] \Rightarrow \mathfrak{N}[m, \{y : R[x,y]\}]) \\ & \quad \& \ \forall x \forall y \forall z (R[x,y] \ \& \ R[z,y] \Rightarrow x = z) \\ & \Rightarrow (\otimes[n,m,k] \Leftrightarrow \mathfrak{N}[k, \{y : \exists z (R[z,y] \ \& \ P[z]) \}])) \end{aligned}$$

Mathematical Rule of Inference

INDUCTION

(Induct)

$$\begin{array}{l} \phi[0 \setminus n] \\ \forall n \forall m (\omega[n] \ \& \ \sigma[n,m] \ \& \ \phi \Rightarrow \phi[m \setminus n]) \\ \hline \forall n (\omega[n] \Rightarrow \phi) \end{array}$$

Precisely, suppose ϕ is an &Statement which does not contain m , and that

$$\phi[0 \setminus n] \in \mathfrak{P}^* \text{ and}$$

$$\forall n \forall m (\omega[n] \ \& \ \sigma[n,m] \ \& \ \phi \Rightarrow \phi[m \setminus n]) \in \mathfrak{P}^*.$$

Then

$$\forall n (\omega[n] \Rightarrow \phi)$$

can be deduced. It is added to the stack.

Definitions

There are three kinds of definitions: \mathfrak{S} -, \mathfrak{D} -, and \mathfrak{T} -. The first two are abbreviations, \mathfrak{S} - for propositions, \mathfrak{D} - for terms. \mathfrak{T} -definitions are baptisms, referring to the one and only one thing which has been shown to exist.

Definitions themselves are only allowed out-of-proof. That is, it is a historical fact that the proofs did not use them, and so when it came time to characterize the system and language (which, historically, was done only after the proofs were first completed), they were not permitted in-proof. Nonetheless, obviously logically this limitation could easily be reversed.

In any case the various rules, such as Introduction and

Elimination, involving definitions, are allowed in-proof and only in-proof.

Suppose D is a definition. Then define:

$\mathcal{D}_N(D)$ = the *new symbol* of the definition

$\mathcal{D}_M(D)$ = the *definiendum* part of the definition

$\mathcal{D}_S(D)$ = the *definiens* part of the definition

$\mathcal{D}_P(D)$ = the *pre-supposition* part of the definition

$\mathcal{D}_V(D)$ = the set of *variables* used in the definition

For instance, a definition D might be:

$$\mathbb{D} \Delta ; (\mathbf{n} \Delta \mathbf{m}) ; \omega[\mathbf{n}] \ \& \ \omega[\mathbf{m}] \ \& \ (\neg \mathbf{n} = 0 \vee \neg \mathbf{m} = 0) ;$$

$$(\chi(\delta \mathbf{n} \mathbf{m}))$$

Then

$\mathcal{D}_N(D)$ = Δ

$\mathcal{D}_M(D)$ = $(\mathbf{n} \Delta \mathbf{m})$

$\mathcal{D}_S(D)$ = $(\chi(\delta \mathbf{n} \mathbf{m}))$

$\mathcal{D}_P(D)$ = $\omega[\mathbf{n}] \ \& \ \omega[\mathbf{m}] \ \& \ (\neg \mathbf{n} = 0 \vee \neg \mathbf{m} = 0)$

$\mathcal{D}_V(D)$ = $\{\mathbf{n}, \mathbf{m}\}$

\mathbb{S} -Definitions

An \mathbb{S} -definition never makes a pre-supposition. So as long as it is part of the language, any \mathbb{S} -definition is allowed out-of-proof.

With \mathbb{S} -definitions a proposition (with certain variables) can abbreviated by an expression which is a concatenation of those same variables and a new symbol. When either the abbreviation or the abbreviated is asserted, with constants now replacing the variables, so may the other.

For instance, here is an example of an \mathbb{S} definition:

$\mathbb{S} \subseteq ; \mathbf{P} \subseteq \mathbf{Q} ; \forall x(\mathbf{P}[x] \Rightarrow \mathbf{Q}[x])$

The new symbol, in this case " \subseteq ", follows directly after " $\$$ " and is followed by a semi-colon. Next is a concatenation involving the symbol and certain variables, followed by another semi-colon. Last is a statement containing the same certain variables, followed by an upside-down exclamation point.

The sense of this definition, of course, is that " $P \subseteq Q$ " may be used to abbreviate " $\forall x(P[x] \Rightarrow Q[x])$ ". Any term in the language may replace P or Q .

In a proof (and only in a proof) the $\$$ symbol may be introduced or eliminated. An example where it is introduced:

$\forall x(P[x] \Rightarrow P[x])$, ! ($\forall I$)	i
$P \subseteq P$, ! ($\$I$)	i

And a case where it is eliminated:

$Q \subseteq R$, ! ($\&E$)	i
$\forall x(Q[x] \Rightarrow R[x])$, ! ($\E)	i

Needless to say, except for different terms replacing P or Q , and differences in whitespace, the statements must be exactly as appear in the $\$$ definition for the introduction and elimination rules to be used. For instance, the appearance of parentheses or indeed any additions would prevent the use of elimination. The following is incorrect:

($Q \subseteq R$)	, ! ($\&E$)	i
! INCORRECT USE		i
($\forall x(Q[x] \Rightarrow R[x])$)	, ! ($\$E$)	i

So is the following (the rule does not apply on sub-statements):

$Q \subseteq R \ \& \ R \subseteq Q$, ! ($\&E$)	i
! INCORRECT USE		i
$\forall x(Q[x] \Rightarrow R[x]) \ \& \ \forall x(R[x] \Rightarrow Q[x])$, ! ($\E)	i

§ DEFINITION

Out-of-proof one may always state a definition $D \in \mathcal{S}\text{Def}(\mathcal{L})$. It changes the language to \mathcal{L}' , where \mathcal{L}' is exactly like \mathcal{L} , except that $\mathcal{S}(\mathcal{L}')$ contains D and $\text{Symbol}(\mathcal{L}')$ contains $\mathcal{D}_N(D)$.

§ INTRODUCTION

(§I)

Suppose $D \in \mathcal{S}\text{Def}(\mathcal{L})$ and that x, y, \dots, z lists the variables in $\mathcal{D}_V(D)$. Suppose $a, b, \dots, c \in \text{Term}(\mathcal{L})$ and that

$$\phi = (\mathcal{D}_S(D))[x \setminus a, y \setminus b, \dots, z \setminus c] \in \mathcal{P}^*.$$

Then $(\mathcal{D}_M(D))[x \setminus a, y \setminus b, \dots, z \setminus c]$ may be deduced. It is added to the stack.

§ ELIMINATION

(§E)

Suppose $D \in \mathcal{S}\text{Def}(\mathcal{L})$ and that x, y, \dots, z lists the variables in $\mathcal{D}_V(D)$. Suppose $a, b, \dots, c \in \text{Term}(\mathcal{L})$ and that

$$\phi = (\mathcal{D}_M(D))[x \setminus a, y \setminus b, \dots, z \setminus c] \in \mathcal{P}^*.$$

Then $(\mathcal{D}_S(D))[x \setminus a, y \setminus b, \dots, z \setminus c]$ may be deduced. It is added to the stack.

T-Definitions

The **T**-definition is akin to a baptism: by proving that one and only thing exists, we introduce a term to refer to it. In FOEA all instances are such that a condition must be premised in order for the one and only one thing to exist, so all uses involve presuppositions. So not all **T**-definitions are permitted, but only those for which the proper pre-suppositions have been proven.

For instance, it turns out that we are able to prove the following two propositions:

! 18. i

$\vdash \forall R \forall x (\mathbf{f} R \ \& \ (R^D)[x] \Rightarrow \exists a R[x, a])$ i

and

! 19. i

$\vdash \forall R \forall x (\mathbf{f} R \ \& \ (R^D)[x] \Rightarrow \forall y \forall z (R[x, y] \ \& \ R[x, z] \Rightarrow y = z))$ i

The first says that, "if R is a function and x is in its domain, then there exists an a such that $R[x, a]$ ", the second that, "if R is a function and x is in its domain, then whenever $R[x, y]$ and $R[x, z]$, y and

z must be the same thing." Thus, when $f R \ \& \ (R^D)[x]$ holds, we can be sure there is one and only one thing a such that $R[x,a]$. We therefore are permitted to introduce a term which refers to this, as such:

! 20. $(R'x)$ refers to the thing y to which x bears R . It pre-supposes that R is functional and x is in the domain of R , which ensures that y exists (P18) and is unique (P19). ;

$\mathbb{T} \ ' \ ; \ (R'x) \ ; \ f R \ \& \ (R^D)[x] \ ; \ R[x,y] \ \ ; \ ! \ (\mathbb{T}D: P18,P19) \ ;$

A \mathbb{T} definition has to be justified, and the comment at the end explains that the " $\mathbb{T}D$ " rule is being used (in fact, it is the only rule which justifies a \mathbb{T} definition), based on the two preceding propositions, P18 and P19.

There are the usual associated rules. First, introduction:

$f R \ \& \ (R^D)[x] \ \ \ \ \ \ , \ ! \ 8 \ (\&I: 3,7) \ \ \ \ \ \ ;$

$R[x, (R'x)] \ \ \ \ \ \ , \ ! \ 9 \ (\mathbb{T}I: P20,8) \ \ \ \ \ \ ;$

Since $(R'x)$ is the unique thing y such that $R[x,y]$ when $f R \ \& \ (R^D)[x]$, then $R[x, (R'x)]$. Notice that, previously, the pre-supposition $f R \ \& \ (R^D)[x]$ has been asserted.

Secondly, elimination:

$(R'x) = y \ \ \ \ \ \ , \ ! \ 4 \ (\&E: 2) \ \ \ \ \ \ ;$

$f R \ \& \ (R^D)[x] \ \ \ \ \ \ , \ ! \ 5 \ (\mathbb{T}E: P20,4) \ \ \ \ \ \ ;$

When an atomic statement has been asserted with $(R'x)$ in an argument place, then the pre-supposition holds and may be asserted. Remark the requirement about the argument place is strict: $(R'x)$ may not be a sub-term in an argument place, so for instance $\{a : a = (R'x)\}$ appearing in an argument place does not allow one to assert $f R \ \& \ (R^D)[x]$.

TERM DEFINITION

($\mathbb{T}D$)

Suppose one is out-of-proof, with $D \in \mathbb{T}\text{Def}(\mathfrak{L})$. Set $\psi = \mathfrak{O}_P(D)$, $\phi = \mathfrak{O}_S(D)$. Let x, y, \dots, z be the elements of $\mathfrak{O}_V(D)$ (again, they may be upper-case), where x is the variable not in ψ . Let a, b, \dots, c be (perhaps upper-case) variables of \mathfrak{L} , and let d be another variable distinct from these. Suppose

$$\begin{aligned} & \forall b \dots \forall c (\psi[y \setminus b, \dots, z \setminus c] \\ & \Rightarrow \exists a \phi[x \setminus a, y \setminus b, \dots, z \setminus c]) \in \mathfrak{P}^* \end{aligned}$$

and

$$\begin{aligned} & \forall b \dots \forall c (\psi[y \setminus b, \dots, z \setminus c] \Rightarrow \\ & \forall a \forall d (\phi[x \setminus a, y \setminus b, \dots, z \setminus c] \ \& \ \phi[x \setminus d, y \setminus b, \dots, z \setminus c] \\ & \Rightarrow a = d)) \in \mathfrak{P}^*. \end{aligned}$$

Then at that point one may state the definition D . It changes the language to \mathfrak{L}' , where \mathfrak{L}' is exactly like \mathfrak{L} , except that $\mathbb{T}(\mathfrak{L}')$ contains D and $\text{Symbol}(\mathfrak{L}')$ contains $\mathfrak{O}_N(D)$.

TERM INTRODUCTION

(**TI**)

Suppose $D \in \mathbb{T}(\mathfrak{L})$. Set $\psi = \mathfrak{O}_P(D)$, $\phi = \mathfrak{O}_S(D)$, $t = \mathfrak{O}_M(D)$. Let x, y, \dots, z be the elements of $\mathfrak{O}_V(D)$, where x is the variable not in ψ (and so not in t) And let a, b, \dots, c be terms of \mathfrak{L} . Set

$$s = t[y \setminus b, \dots, z \setminus c]$$

If

$$\psi[y \setminus b, \dots, z \setminus c] \in \mathfrak{P}^*,$$

then one may deduce

$$\phi[x \setminus s, y \setminus b, \dots, z \setminus c].$$

It is added to the stack.

TERM ELIMINATION

(**TE**)

Suppose $D \in \mathbb{T}(\mathfrak{L})$. Set $\psi = \mathfrak{O}_P(D)$, $t = \mathfrak{O}_M(D)$. Let x, y, \dots, z be the elements of $\mathfrak{O}_V(D)$, where x is the variable not in ψ (and so not in t). Suppose

$$\phi \in \mathfrak{P}^*,$$

where ϕ is an atomic formula with s in one of its argument, where

$$s = t[y \setminus b, \dots, z \setminus c]$$

for some terms b, \dots, c . Then one may deduce

$$\psi[y \setminus b, \dots, z \setminus c].$$

D-Definitions

D-definitions are abbreviations, but since they may abbreviate for **T**-definitions, they may also make pre-suppositions.

Here are two examples of a \mathbb{D} definition without pre-suppositions:

! 1. $(\mathbf{b} _ \mathbf{c})$ represents the finite interval between \mathbf{b} and \mathbf{c}
(inclusive). i

$\mathbb{D} _ ; (\mathbf{b} _ \mathbf{c}) ; ; \{a : \leq[\mathbf{b},a] \ \& \ \leq[a,\mathbf{c}]\}$ i

and

! 1. ϕ represents the predicate of non-identical things, which of
course is empty (nothing satisfies it). i

$\mathbb{D} \ \phi ; \phi ; ; \{a : \neg a = a\}$ i

The new symbol--in the two examples " $_$ " or " ϕ "-- follows " \mathbb{D} " and is followed by a semi-colon. Next comes a concatenation involving the symbol and perhaps certain variables. If there are variables, then the term must be enclosed in parentheses, otherwise no. Because neither example involves a pre-supposition, two semi-colons follow. Finally comes a term, which must contain the same variables as appeared earlier in the line.

The evident meaning of the first definition is that " $(\mathbf{b} _ \mathbf{c})$ " is to abbreviate " $\{a : \leq[\mathbf{b},a] \ \& \ \leq[a,\mathbf{c}]\}$ ".

These \mathbb{D} -defined terms may be introduced or eliminated.

An instance of $\mathbb{D}\mathbb{I}$:

$\forall x (\{a : \leq[\mathbf{b},a] \ \vee \ \leq[a,\mathbf{c}]\}[x] \Leftrightarrow \leq[\mathbf{b},x] \ \& \ \leq[x,\mathbf{c}])$
, ! 2 (Pred) i

$\forall x ((\mathbf{b} _ \mathbf{c})[x] \Leftrightarrow \leq[\mathbf{b},x] \ \& \ \leq[x,\mathbf{c}])$, ! 3 ($\mathbb{D}\mathbb{I}$: P1,2) i

And of $\mathbb{D}\mathbb{E}$:

$((\mathbf{b}+1) _ \mathbf{c})[\mathbf{b}]$, ! 2 (Prem) i

$\{a : \leq[(\mathbf{b}+1),a] \ \& \ \leq[a,\mathbf{c}]\}[\mathbf{b}]$, ! 3 ($\mathbb{D}\mathbb{E}$: P1,2) i

\mathbb{D} definitions also allow presuppositions. For instance, " (\mathbf{n}') " is introduced to abbreviate the term

$((\sigma \lceil \omega)' \mathbf{n})$.

Now " $((\sigma \lceil \omega)' \mathbf{n})$ " pre-supposes that

$\mathbf{f} (\sigma \lceil \omega) \ \& \ ((\sigma \lceil \omega)^{\mathbb{D}})[\mathbf{n}]$

So it must first be proven that

! 11. i
 $\vdash \forall n (\omega[n] \Rightarrow \mathbf{f} (\sigma \lceil \omega) \ \& \ ((\sigma \lceil \omega)^D) [n])$ i

to make the following definition:

! 12. i
 $\mathbb{D} \quad ' ; (\mathbf{n}') ; \omega[n] ; ((\sigma \lceil \omega)' \mathbf{n})$; ! (DD: III8.20, P11)
i

Here $\omega[n]$ is the pre-supposition of (\mathbf{n}') . Obviously, if $\omega[n]$ holds, then $\mathbf{f} (\sigma \lceil \omega) \ \& \ ((\sigma \lceil \omega)^D) [n]$, and so $((\sigma \lceil \omega)' \mathbf{n})$ makes sense.

The $\mathbb{D}\mathbb{I}$ and $\mathbb{D}\mathbb{E}$ rules remains the same, with the exception that for the $\mathbb{D}\mathbb{I}$ rule, the pre-supposition must be proven before the term is substituted:

$\omega[n]$, ! 2 (Prem)	i
$(\omega[n] \Rightarrow \mathbf{f} (\sigma \lceil \omega) \ \& \ ((\sigma \lceil \omega)^D) [n])$, ! 3 ($\forall\mathbb{E}$: P11)	i
$\omega[n] \Rightarrow \mathbf{f} (\sigma \lceil \omega) \ \& \ ((\sigma \lceil \omega)^D) [n]$, ! 4 ($(\)\mathbb{E}$: 3)	i
$\mathbf{f} (\sigma \lceil \omega) \ \& \ ((\sigma \lceil \omega)^D) [n]$, ! 5 ($\Rightarrow\mathbb{E}$: 2,4)	i
$(\sigma \lceil \omega) [n, ((\sigma \lceil \omega)' \mathbf{n})]$, ! 6 ($\mathbb{T}\mathbb{I}$: III8.20,5)	i
$(\sigma \lceil \omega) [n, (\mathbf{n}')]$, ! 7 ($\mathbb{D}\mathbb{I}$: P12,2,6)	i

If the \mathbb{D} term appears in the argument place of an atomic proposition, then the pre-supposition obtains, using the $\mathbb{D}\mathbb{P}$ rule:

$(\mathbf{n}') = \mathbf{m}$, ! 2 (Prem)	i
$\omega[n]$, ! 3 ($\mathbb{D}\mathbb{P}$: P12,2)	i

As before, (\mathbf{n}') may be substituted for a universally quantified variable, only when $\omega[n]$ has already be asserted.

In practice, these definition rules involving pre-suppositions are seldom used in the proofs, except for when terms with pre-suppositions

are introduced for a universally quantified variable, which happens frequently from Section IV on.

TERM DEFINITION WITHOUT PRE-SUPPOSITION

Out-of-proof one may always state a definition $D \in \mathbb{D}\text{Def}(\mathfrak{K})$. It changes the language to \mathfrak{K}' , where \mathfrak{K}' is exactly like \mathfrak{K} , except that $\mathbb{D}(\mathfrak{K}')$ contains D and $\text{Symbol}(\mathfrak{K}')$ contains $\mathcal{O}_N(D)$.

TERM DEFINITION WITH PRE-SUPPOSITION

(\mathbb{D})

Suppose one is out-of-proof, with $D \in \mathbb{D}\text{Def}(\mathfrak{K})$. Set $\psi_1 = \mathcal{O}_P(D)$. And suppose $\mathcal{O}_S(D)$ is a defined term, with corresponding pre-supposition ψ_2 . Let x, y, \dots, z be the elements of $\mathcal{O}_V(D)$ (again, they may be upper-case). Suppose a, b, \dots, c are variables of \mathfrak{K} , and that

$$\begin{aligned} \forall a \forall b \dots \forall c (\psi_1[x \setminus a, y \setminus b, \dots, z \setminus c] \\ \Rightarrow \psi_2[x \setminus a, y \setminus b, \dots, z \setminus c]) \in \mathbb{P}^*. \end{aligned}$$

Then at that point one may state the definition D . It changes the language to \mathfrak{K}' , where \mathfrak{K}' is exactly like \mathfrak{K} , except that $\mathbb{D}(\mathfrak{K}')$ contains D and $\text{Symbol}(\mathfrak{K}')$ contains $\mathcal{O}_N(D)$.

\mathbb{D} INTRODUCTION

(\mathbb{D} I)

Suppose $D \in \mathbb{D}(\mathfrak{K})$. Set $\psi = \mathcal{O}_P(D)$, $\sigma = \mathcal{O}_S(D)$, $\tau = \mathcal{O}_M(D)$. Let x, y, \dots, z be the elements of $\mathcal{O}_V(D)$. And let a, b, \dots, c be terms of \mathfrak{K} . Set

$$\begin{aligned} s &= \sigma[x \setminus a, y \setminus b, \dots, z \setminus c] \\ t &= \tau[x \setminus a, y \setminus b, \dots, z \setminus c] \end{aligned}$$

If

$$\begin{aligned} \psi[x \setminus a, y \setminus b, \dots, z \setminus c] \in \mathbb{P}^* \text{ and} \\ \phi \in \mathbb{P}^* \end{aligned}$$

then one may deduce

$$\phi[\sigma \setminus \tau].$$

It is added to the stack. Evidently, if $\psi = \mathcal{O}_P(D)$ is null (i.e. there is no pre-supposition), then one may simply deduce

$$\phi[\sigma \setminus \tau]$$

if $\phi \in \mathbb{P}^*$.

\mathbb{D} ELIMINATION

(\mathbb{D} E)

Suppose $D \in \mathbb{D}(\mathfrak{K})$. Set $\psi = \mathcal{O}_P(D)$, $\sigma = \mathcal{O}_S(D)$, $\tau = \mathcal{O}_M(D)$. Let x, y, \dots, z be the elements of $\mathcal{O}_V(D)$. And let a, b, \dots, c be terms of \mathfrak{K} . Set

$$s = \sigma[x \setminus a, y \setminus b, \dots, z \setminus c]$$

$$t = \tau[x\backslash a, y\backslash b, \dots, z\backslash c]$$

If

$$\psi[x\backslash a, y\backslash b, \dots, z\backslash c] \in \mathcal{P}^*$$

then one may deduce

$$\phi[\tau\backslash\sigma].$$

It is added to the stack.

D PRE-CONDITION

(**D**P)

Suppose $D \in \mathbb{D}(\mathcal{L})$. Set $\psi = \mathcal{D}_P(D)$, $t = \mathcal{D}_M(D)$. Let x, y, \dots, z be the elements of $\mathcal{D}_V(D)$. Suppose

$$\phi \in \mathcal{P}^*,$$

where ϕ is an atomic formula with s in one of its argument, where

$$s = t[x\backslash a, y\backslash b, \dots, z\backslash c]$$

for some terms a, b, \dots, c . Then one may deduce

$$\psi[x\backslash a, y\backslash b, \dots, z\backslash c].$$

Successfully Referring Terms

A term is always successfully referring if is not a **D**- or a **T**-term defined with a supposition. (Since all **T** terms are in fact defined with a supposition, this means that no **T**-defined term is automatically successfully referring.)

On the other hand, suppose t is a **D**- or a **T**-defined term with supposition ψ using variables p, q, \dots, r . Then t is successfully referring in an environment if there exist constants d, e, \dots, f and $D \in \mathbb{D}(\mathcal{L}) \cup \mathbb{T}(\mathcal{L})$ such that

$$\mathcal{D}_V(D) = \{p, q, \dots, r\} \text{ (all distinct)}$$

$$t = (\mathcal{D}_M(D)) [p\backslash d, q\backslash e, \dots, r\backslash f]$$

and

$$(\mathcal{D}_P(D)) [p\backslash d, q\backslash e, \dots, r\backslash f] \in \mathcal{P}^*.$$

Appendix 1: Original symbols

- (Beginning parenthesis
-) End parenthesis
- , Separates entries in a list (variables or lines of a proof)
- ! Begins a comment
- ! Ends a comment or proposition

\vdash	Begins a proposition (introduces what is to be proved)
\square	Ends a proof
[Begins list of variables after a predicate symbol
]	Ends list of variables after a predicate symbol
$\$$	Introduces $\$$ -type definition
\mathbb{D}	Introduces \mathbb{D} -type definition
\mathbb{T}	Introduces \mathbb{T} -type definition
;	Separates parts of definition
Upper-case Courier 12 pt plain and bold alphabet	UC (upper-case) Variables
Lower-case Courier 12 pt plain and bold alphabet	LC (lower-case) Variables
\mathfrak{F}	contradiction
\neg	not
$\&$	and
\vee	or
\Rightarrow	implies
\Leftrightarrow	if and only if
\forall	for all
\exists	there is
=	is equal to
\mathfrak{N}	numbers (UC constant)
ω	is a natural number (UC constant)
σ	succeeds (UC constant)
0	zero (LC constant)
1	one (LC constant)
2	two (LC constant)
3	three (LC constant)
4	four (LC constant)
\oplus	addition predicate (triadic) (UC constant)
\ominus	subtraction predicate (triadic) (UC constant)
\wedge	inequality predicate (UC constant)
\otimes	multiplication predicate (triadic) (UC constant)

Appendix 2: Defined symbols

\subseteq	is contained in (for monadic predicates) (CII.1)
\equiv	is contained in and is contained by (ditto) (CII.1)
\cup	union (ditto) (CII.2)
\cap	intersection (CII.3)
c (Courier 12 pt superscript)	complement (CII.4)
ϕ	the (monadic) predicate of those things not equal to themselves (CII.5)
\mathbb{U}	the (monadic) predicate of those things equal to themselves (CII.6)
\setminus	difference (CII.7)
\bullet	singleton (CII.8)
\ddagger	doublet (CII.9)
\vee	triplet (CII.9)
\forall	quadruplet (CII.9)
\subseteq	is contained in (for dyadic predicates) (CIII.1)
\equiv	is contained in and is contained by (ditto) (CIII.1)
\sqcup	union (ditto) (CIII.2)
*	inverse (CIII.3)

Φ	the (dyadic) predicate of those things a,b such that a does not equal itself (CIII.4)
D (Courier 12 pt superscript)	domain of (CIII.5)
\mathcal{D}	has domain (CIII.5)
I (Courier 12 pt superscript)	image of (CIII.6)
\mathcal{I}	has image (CIII.6)
[restricted to (first co-ordinate) (CIII.7)
]	restricted to (second co-ordinate) (CIII.7)
f	is a function (CIII.8)
F	is a function with domain (CIII.8)
'	the value of (CIII.8)
1	is one-to-one (CIII.9)
1l	is one-to-one with image (CIII.9)
'	the argument (inverse value) of (CIII.9)
o	composition (CIII.10)
E	equality predicate (CIII.11)
I	restricted equality predicate (CIII.11)
▪	pairing predicate (CIII.12)
~	is in one-to-one correspondance with (CIII.13)
ι	pair removal (CIII.14)
α	pair switching (CIII.15)
X	simple Cartesian product (CIII.16)
◇	projection (of triadic onto dyadic predicates) (CIII.17)
f	is finite (CIV.5)
ι	is infinite (CIV.6)
\mathcal{H}	is a hierarchal predicate (CIV.7)
'	the successor of (CIV.8)
υ	uniform predicate (CIV.11)
+	plus (CV.1)
≤	is less than or equal to (CV.3)
<	is less than (CV.4)
-	minus (CV.5)
x	times (CV.7)
∞	infinite interval (CV.9)
⊕	addition predicate (dyadic) (CV.10)
⊖	subtraction predicate (dyadic) (CV.10)
⊗	multiplication predicate (dyadic) (CV.10)
	divides (CVI.1)
-	finite interval (CVI.2)
μ	the least natural number (CVI.3)
χ	the greatest natural number (CVI.4)
δ	divisors of (CVI.5)
Δ	the greatest common divisor (CVI.5)
θ	finite sequence (CVII.1)
λ	the length of (a finite sequence) (CVII.1)
^	concatenation (CVII.2)
π	prime number predicate (CVII.4)

First, remark that whitespace may always be added between symbols (for instance, to improve reading clarity), and the resulting formula is to be considered equivalent to the one where all whitespace has been removed and the order of all the symbols remains unchanged.

The grammar is defined using Backus-Naur form (BNF). Because BNF is in general not able to describe a grammar where conditions are placed on the interactions of different elements, it has been modified. For instance, in order to express that " $\forall zP$ " is grammatical only if the Block " P " contains the variable " z ", we write:

$\langle \text{Var} : x \rangle \langle \text{Block} : y \rangle \langle ! y \text{ contains } x \rangle$.

That is " $\langle ! \dots \rangle$ " expresses a condition that the preceding must obey, and " $: x$ " and the like are used to name elements so that conditions can subsequently be placed on them.

When \mathcal{L} is a language and x is a subset of $\text{Var}(\mathcal{L})$, then \mathcal{L}_x refers to the language exactly like \mathcal{L} , except that $\text{Var}(\mathcal{L}_x) = \text{Var}(\mathcal{L}) \setminus x$ and $\text{Const}(\mathcal{L}_x) = \text{Const}(\mathcal{L}) \cup x$. When \mathcal{L} is a language and x is a subset of $\text{Const}(\mathcal{L})$, then ${}_x\mathcal{L}$ refers to the language exactly like \mathcal{L} , except that $\text{Var}({}_x\mathcal{L}) = \text{Var}(\mathcal{L}) \cup x$ and $\text{Const}({}_x\mathcal{L}) = \text{Const}(\mathcal{L}) \setminus x$. When x is the singleton set $\{y\}$, we may simply write \mathcal{L}_y instead of $\mathcal{L}_{\{y\}}$ and ${}_y\mathcal{L}$ instead of $\{y\}\mathcal{L}$.

When A is a set of symbols, $\text{Concat}(A)$ refers to all (finite) concatenations of the symbols A .

When z is a finite list and x is an element, $(z \wedge (x))$ means the list where x is concatenated onto the end of z .

$\text{NewSymbol}(\mathcal{L}) ::= \text{any symbol which does not appear in } \text{Symbol}(\mathcal{L})$

$\neg\text{Statement}(\mathcal{L}) ::=$

$\langle \text{Block}(\mathcal{L}) \rangle$

$| \neg \langle \neg\text{Statement}(\mathcal{L}) \rangle$

$$\begin{aligned}
\text{vStatement}(\mathcal{L}) & ::= \\
& \langle \neg \text{Statement}(\mathcal{L}) \rangle \\
& | \langle \text{vStatement}(\mathcal{L}) \rangle \vee \langle \text{vStatement}(\mathcal{L}) \rangle \\
\\
\&\text{Statement}(\mathcal{L}) & ::= \\
& \langle \text{vStatement}(\mathcal{L}) \rangle \\
& | \langle \&\text{Statement}(\mathcal{L}) \rangle \& \langle \&\text{Statement}(\mathcal{L}) \rangle \\
\\
\Rightarrow \text{Statement}(\mathcal{L}) & ::= \\
& \langle \&\text{Statement}(\mathcal{L}) \rangle \\
& | \langle \&\text{Statement}(\mathcal{L}) \rangle \Rightarrow \langle \&\text{Statement}(\mathcal{L}) \rangle \\
\\
\text{Block}(\mathcal{L}) & ::= \\
& (\langle \Rightarrow \text{Statement}(\mathcal{L}) \rangle) \\
& | \langle \text{Atomic}(\mathcal{L}) \rangle \\
& | \forall \langle \text{Var}(\mathcal{L}) : x \rangle \langle \text{Block}(\mathcal{L}_x) \rangle \\
& | \exists \langle \text{Var}(\mathcal{L}) : x \rangle \langle \text{Block}(\mathcal{L}_x) \rangle \\
\\
\text{Atomic}(\mathcal{L}) & ::= \\
& \langle \text{Term}(\mathcal{L}) \rangle = \langle \text{Term}(\mathcal{L}) \rangle \\
& | \langle \text{Term}(\mathcal{L}) \rangle [\langle \text{NonEmptyTermList}(\mathcal{L}) \rangle] \\
& | \mathfrak{F} \\
& | \langle \mathfrak{S}^*(\mathcal{L}, \langle \text{Term}(\mathcal{L}) \rangle) \rangle
\end{aligned}$$

\mathfrak{F} represents a contradictory statement. If it can be proved, then a contradiction has been demonstrated, implying the contrary of the most recent premise.

$\mathfrak{S}^*(\mathcal{L}, \langle \text{Term}(\mathcal{L}) \rangle)$ is the set of all \mathfrak{S} rules in $\mathfrak{S}(\mathcal{L})$, where $\langle \text{Term}(\mathcal{L}) \rangle$ replaces the variables used in the \mathfrak{S} definition. For

example, an \mathfrak{S} -definition in a language \mathfrak{L} , i.e. an element of $\mathfrak{S}(\mathfrak{L})$, could be:

$$\mathfrak{S} \subseteq ; \quad \mathbf{P} \subseteq \mathbf{Q} ; \quad \forall x(\mathbf{P}[x] \Rightarrow \mathbf{Q}[x])$$

Then $\{x : x = 1\} \subseteq \{x : x = x\}$ is an example of an element of $\mathfrak{S}^*(\mathfrak{L}, \langle \text{Term}(\mathfrak{L}) \rangle)$, since $\{x : x = 1\}$ and $\{x : x = x\}$ are terms in the initial language and so forcibly in \mathfrak{L} .

Proposition(\mathfrak{L}) ::=

$$\langle \Rightarrow \text{Statement}(\mathfrak{L}, ()) \rangle$$

A proposition is what may appear in the proposition header, i.e. is something which may be proved. " $()$ " represents the empty list.

PremEx(\mathfrak{L}) ::=

$$\langle \text{Var}(\mathfrak{L}) \rangle$$

ex **x**

The preceding statement says "Suppose **x** exists."

Statement(\mathfrak{L}) ::=

$$\langle \text{Proposition}(\mathfrak{L}) \rangle$$

$$| \langle \text{PremEx}(\mathfrak{L}) \rangle$$

A statement is what may appear in a proof: it is either something which can be proved, or a supposition that something exists.

The definition of a term requires that the notion of predicativity be introduced. A predicative proposition for a (finite) set z of constants, is one which does not have in a predicate position either (1) any quantified variables or (2) any constants which are elements of z . In order that all predicates are well-behaved, only predicative propositions for z will be allowed in terms $\{z \mid \dots\}$.

\neg PStatement(\mathfrak{L}, z) ::=

$$\langle \text{PBlock}(\mathcal{I}, z) \rangle$$

$$| \neg \langle \neg \text{PStatement}(\mathcal{I}, z) \rangle$$

$$\text{VPStatement}(\mathcal{I}, z) ::=$$

$$\langle \neg \text{PStatement}(\mathcal{I}, z) \rangle$$

$$| \langle \text{VPStatement}(\mathcal{I}, z) \rangle \vee \langle \text{VPStatement}(\mathcal{I}, z) \rangle$$

$$\&\text{PStatement}(\mathcal{I}, z) ::=$$

$$\langle \text{VPStatement}(\mathcal{I}, z) \rangle$$

$$| \langle \&\text{PStatement}(\mathcal{I}, z) \rangle \& \langle \&\text{PStatement}(\mathcal{I}, z) \rangle$$

$$\Rightarrow \text{PStatement}(\mathcal{I}, z) ::=$$

$$\langle \&\text{PStatement}(\mathcal{I}, z) \rangle$$

$$| \langle \&\text{PStatement}(\mathcal{I}, z) \rangle \Rightarrow \langle \&\text{PStatement}(\mathcal{I}, z) \rangle$$

$$\text{PBlock}(\mathcal{I}, z) ::=$$

$$(\langle \Rightarrow \text{PStatement}(\mathcal{I}, z) \rangle)$$

$$| \langle \text{PAtomic}(\mathcal{I}, z) \rangle$$

$$| \forall \langle \text{Var}(\mathcal{I}) : x \rangle \langle \text{PBlock}(\mathcal{I}_x, (z \wedge (x))) \rangle$$

$$| \exists \langle \text{Var}(\mathcal{I}) : x \rangle \langle \text{PBlock}(\mathcal{I}_x, (z \wedge (x))) \rangle$$

Note that, unlike a Block, quantified variables of PBlocks may not appear in the predicate place.

$$\text{PAtomic}(\mathcal{I}, z) ::=$$

$$\langle \text{NoDefTerm}(\mathcal{I}) \rangle = \langle \text{NoDefTerm}(\mathcal{I}) \rangle$$

$$| \langle \text{NoDefTerm}(z\mathcal{I}) \rangle [\langle \text{NonEmptyNoDefTermList}(\mathcal{I}) \rangle]$$

$$| \mathcal{F}$$

Evidently, the appearance of "z" in " $z\mathcal{I}$ " is crucial, since it prevents z-constants from appearing in the predicate place.

NoDefTerm(\mathfrak{L}) ::=

<Const(\mathfrak{L})>

| { <NonEmptyVarList(\mathfrak{L}) : x> | <=>PStatement(\mathfrak{L}_y, y)> }

<! y is the (finite) set of variables in the (finite) list x>

As a convention, only lower-case variables will be used in the NonEmptyVarList, since the mentioned predicate is for all *things* which satisfy, etc.

Term(\mathfrak{L}) ::=

<NoDefTerm(\mathfrak{L})>

| < $\mathbb{D}^*(\mathfrak{L}, \langle \text{Term}(\mathfrak{L}) \rangle)$ >

| < $\mathbb{T}^*(\mathfrak{L}, \langle \text{Term}(\mathfrak{L}) \rangle)$ >

$\mathbb{D}^*(\mathfrak{L}, \langle \text{Term}(\mathfrak{L}) \rangle)$ is the set of all \mathbb{D} rules in $\mathbb{D}(\mathfrak{L})$, where

$\langle \text{Term}(\mathfrak{L}) \rangle$ replaces the variables used in the \mathbb{D} definition

(similarly for \mathbb{T}). For example, a \mathbb{D} -rule in a language \mathfrak{L} , i.e.

an element of $\mathbb{D}(\mathfrak{L})$, could be:

$\mathbb{D} \ \Delta \ ; \ (\mathbf{n} \ \Delta \ \mathbf{m}) \ ; \ \omega[\mathbf{n}] \ \& \ \omega[\mathbf{m}] \ \& \ (\neg \ \mathbf{n} = 0 \ \vee \ \neg \ \mathbf{m} = 0) \ ;$

$(\chi(\delta \ \mathbf{n} \ \mathbf{m}))$

Then $(1 \ \Delta \ 2)$ is an example of an element of $\mathbb{D}^*(\mathfrak{L}, \langle \text{Term}(\mathfrak{L}) \rangle)$,

since 1 and 2 are initial constants, so are forcibly constants and so terms in \mathfrak{L} .

Because of the uses to which we put them, NonEmptyTermList allows repetitions, while NonEmptyVarList does not.

NonEmptyVarList(\mathfrak{L})

::= <Var(\mathfrak{L})>

| <Var(\mathfrak{L}) : x> , <NonEmptyVarList(\mathfrak{L}) : y>

<! x does not appear in y>

NonEmptyTermList(\mathcal{I})

::= <Term(\mathcal{I})>

| <Term(\mathcal{I})> , <NonEmptyTermList(\mathcal{I})>

NonEmptyNoDefTermList(\mathcal{I})

::= <NoDefTerm(\mathcal{I})>

| <Term(\mathcal{I})> , <NonEmptyNoDefTermList(\mathcal{I})>

\mathcal{S} Def(\mathcal{I}) ::=

\mathcal{S} <NewSymbol(\mathcal{I}) : x> ;

<Concat($y \cup \{x\}$)> ;

< \Rightarrow Statement(\mathcal{I}_y) : z>

<! y a finite subset of Var(\mathcal{I})>

<! all elements of y appear in z>

ex. $\mathcal{S} \subseteq \mathcal{I}$; $\mathcal{P} \subseteq \mathcal{Q}$; $\forall x(\mathcal{P}[x] \Rightarrow \mathcal{Q}[x])$

\mathcal{D} Def(\mathcal{I}) ::=

\mathcal{D} <NewSymbol(\mathcal{I}) : x> ;

(<Concat($y \cup \{x\}$)>) ;

< \Rightarrow Statement(\mathcal{I}_y)> ;

<Term(\mathcal{I}_y) : z> <! y a finite set of variables in \mathcal{I} >

<! all elements of y appear in z>

| \mathcal{D} <NewSymbol(\mathcal{I}) : x> ;

(<Concat($y \cup \{x\}$)>) ;

;

<Term(\mathcal{I}_y): z> <! y a finite set of variables in \mathcal{I} >

<! all elements of y appear in z>

| \mathcal{D} <NewSymbol(\mathcal{I}) : x> ;

x ;

$\langle \Rightarrow \text{Statement}(\mathfrak{L}) \rangle ;$
 $\langle \text{Term}(\mathfrak{L}) \rangle$
 $| \mathbb{D} \langle \text{NewSymbol}(\mathfrak{L}) : x \rangle ;$
 $x ;$
 $;$
 $\langle \text{Term}(\mathfrak{L}) \rangle$

The third and fourth alternatives are allowed so that we don't have to write x in parenthesis.

ex. $\mathbb{D} \cup ; (\mathbf{P} \cup \mathbf{Q}) ; ; \{a : \mathbf{P}[a] \vee \mathbf{Q}[a]\}$

or

$\mathbb{D} \phi ; \phi ; ; \{a : \neg a = a\}$

or

$\mathbb{D} \Delta ; (\mathbf{n} \Delta \mathbf{m}) ; \omega[\mathbf{n}] \& \omega[\mathbf{m}] \& (\neg \mathbf{n} = 0 \vee \neg \mathbf{m} = 0) ;$

$(\chi(\delta \mathbf{n} \mathbf{m}))$

Replacing the variables of a \mathbb{D} -definition with constant terms results in a \mathbb{D} -term. For instance, $(1 \Delta 2)$ is a \mathbb{D} -term. Indeed, it is said to be a \mathbb{D} -term defined as $(\mathbf{n} \Delta \mathbf{m})$ with supposition $\omega[\mathbf{n}] \& \omega[\mathbf{m}] \& (\neg \mathbf{n} = 0 \vee \neg \mathbf{m} = 0)$ using variables \mathbf{n} and \mathbf{m} .

$\mathbb{T}\text{Def}(\mathfrak{L}) ::= \mathbb{T} \langle \text{NewSymbol}(\mathfrak{L}) : x \rangle ;$

$(\langle \text{Concat}(y \cup \{x\}) \rangle) ;$

$\langle \Rightarrow \text{Statement}(\mathfrak{L}_y) \rangle ;$

$\langle \Rightarrow \text{Statement}(\mathfrak{L}_{y \cup \{v\}}) : z \rangle$

$\langle ! y \cup \{v\} \text{ a finite set of variables in } \mathfrak{L}, v \text{ not in } y \rangle$

$\langle ! \text{ all elements of } y \cup \{v\} \text{ appear in } z \rangle$

ex. $\mathbb{T} ' ; (\mathbf{R}'a) ; \mathbf{f} \mathbf{R} \& (\mathbf{R}^{\mathbb{D}})[a] ; \mathbf{R}[a,b]$

(Here, $y = \{R,a\}$, $x = '$, and $v = b$.)

Replacing the variables of a \mathbb{T} -definition with constant terms

results in a \mathbb{T} -term. For instance, $(\sigma'1)$ is a \mathbb{T} -term (which is

the successor of 1, i.e. 2). As above, it is said to be a \mathbb{T} -term defined as $(R'a)$ with supposition $\mathbf{f} R \ \& \ (R^D)[a]$ using variables R and a .