**DEFINITION BY RECURSION**

Assuming that we have already defined the successor function $f$ on the natural numbers (so that, $f(0) = 1$, $f(1) = 2$, $f(2) = 3$, ...), consider how one might define the addition function in terms of $f$:

$$n + 0 = n$$

$$n + f(x) = f(n + x)$$

These equations uniquely characterize the function: given any pair of numbers $n, m$, one can easily compute their unique sum using these equations. Thus it qualifies as a definition by our criterion.

What ensures that the addition function is well defined is that the successor function is one-one. If $f$ were a many-one function, $n + m$ might have more than one value for some $m$. (For example, Let $g$ be just like $f$, except that $g(2) = g(3) = 4$, and $+$ be defined just like $+$ except with $g$ playing the role of $f$. Then, for example, $6 + 4 = g(6 + 3) = 10$ and $6 + 4 = g(6 + 2) = 9$)

A definition of this form, whose domain is an inductively defined set, is called a **recursive definition**. If we think of the domain as being generated over time—first we put in those members sanctioned by the base clause(s), and then successively generate new members using the inductive clause(s)—then the values of a recursively defined function are given in corresponding stages: first we assign images outright to those members of the domain that were included first (i.e. by virtue of the base clauses), and then explain the image of members of the domain included later in terms of the images of previously included ones. In the special case of a function on the natural numbers, this means assigning a value for the function at 0, and then explaining the value of the function at $n + 1$ in terms of its values for smaller arguments.

Here’s another well known example of a recursive definition that builds upon the earlier one:

$$n \times 0 = 0$$

$$n \times f(x) = (n \times x) + n$$

Again these equations uniquely characterize a function so that multiplication is well defined. What matters is not whether the addition function is one-one, but that the function in terms of which we generate the domain is, i.e. that every number can be uniquely represented in terms of successor.

Here is an example of a function defined by recursion whose domain is not the natural numbers, but another inductively defined set: the rank function $r: PL \Rightarrow N$.

**Definition**

We define the **rank** of a $PL$ sentence $\phi$, $r(\phi)$, as follows:

1) If $\phi$ is an atom, $r(\phi) = 1$.
2) If $\phi$ is of the form $\neg \alpha$, $r(\phi) = r(\alpha) + 1$
3) If $\phi$ is of the form $(\alpha \ast \beta)$ (where $\ast$ is one of $\land, \lor, \text{or } \rightarrow$), $r(\phi) = \max(r(\alpha), r(\beta)) + 1$

In this case, the function is well defined only because of the unique readability lemma proved earlier. (Zalabardo states what this requirement amounts to for inductively defined sets in general: the elements of the set must be freely generated from the base. This means, in effect, that there is only one way for each element to get into the set, i.e. it can be obtained from the atoms by applying the inductive clauses in only one way.) The rank function is useful because it is sometimes easier to use induction on rank, rather than induction on the definition of $PL$-sentence, to prove that every sentence of $PL$ has $P$. (Since
every sentence of $PL$ has a rank, it’s enough to prove:

For all $n$, if $r(\phi) = n$, then $\phi$ has $n$ by mathematical induction on the positive integers.)

Henceforth, we shall define functions by recursion, as needed, taking it for granted in each case that they are well defined.

**EXPRESSIVE COMPLETENESS**

**Definition:** If $A$ is a finite set of atoms, a **truth assignment** on $A$ is a function from $A$ to \{ $T$, $F$ \}, and a truth function on $A$ is a function from $T_A$ to \{ $T$, $F$ \}, where $T_A$ is the set of truth assignments on $A$.

For Example: \{ $<\alpha, T>$, $<\beta, F>$, $<\gamma, T>$ \} and \{ $<\alpha, F>$, $<\beta, F>$, $<\gamma, F>$ \} are truth assignments on the set \{ $\alpha$, $\beta$, $\gamma$ \}.

Furthermore, \{ $<<<<\rightarrow, T>$, $<<<<\rightarrow, F>$, $<<<<\leftrightarrow, T>$, $<<<<\leftrightarrow, F>$ \} is a truth function on \{ $\alpha$, $\beta$ \}.

**Definition:** A sentence $\phi$ of $PL$ represents the truth function $f$ on $A$ if, for all $v$, $v(\phi) = f(v \upharpoonright A)$.

For Example: $\neg (\alpha \lor \beta)$ represents the truth function in the preceding example.

**Definition:** A language is **expressively complete** if every truth function on a finite set of its atoms is represented by some sentence of the language.

**Theorem:**

1) $PL$ is expressively complete.

2) The fragments of $PL$ containing only negation and a single binary connective are expressively complete.

3) If * is the binary connective whose meaning is given by:

$\alpha$ is a sentence of the form $(\beta \ast \gamma)$, then

$v(\alpha) = T$ if $v(\beta) = v(\gamma) = F$

$v(\alpha) = F$ otherwise

then the language whose only connective is * is expressively complete.

4) If # is the binary connective whose meaning is given by:

$\alpha$ is a sentence of the form $(\beta \# \gamma)$, then

$v(\alpha) = F$ if $v(\beta) = v(\gamma) = T$

$v(\alpha) = T$ otherwise

then the language whose only connective is # is expressively complete.

**Proof:**

These results are proved in sections 2.7 and 2.8 of Zalabardo. Here we only sketch an informal argument for 1).

Let $A = \{ \alpha_1, ..., \alpha_n \}$ be a set of atoms and $f$ a truth function from $A$ onto \{ $T$, $F$ \}, we construct a sentence, $\phi$ of $PL$ which represents $f$.

For each truth assignment $a$ on $A$ such that $f(a) = T$, we construct the following conjunction:

$(\ldots (\pm \alpha_1 \land \ldots \land \pm \alpha_n))$

where, $\pm \alpha_i$ is $\alpha_i$ if $a(\alpha_i) = T$, and $\pm \alpha_i$ is $\neg \alpha_i$ if $a(\alpha_i) = F$.

Notice that, if $v$ is any admissible extension of $a$,

i) $v((\ldots (\pm \alpha_1 \land \ldots \land \pm \alpha_n)) = T$ (Why?)
and, if $a' \neq a$ and $v'$ is any admissible extension of $a'$, then

ii) $v'((\ldots(\pm \alpha_1 \land \ldots \land \pm \alpha_n)) = F$ (Why?)

Now, let $\phi$ be a disjunction formed from these conjunctions. We claim that $\phi$ represents $f$, i.e. we show that, for all $v$:

$$v(\phi) = T \iff f(v \uparrow A) = T$$

($\Rightarrow$) If $v(\phi) = T$, then $v$ assigns $T$ to exactly one disjunct of $\phi$ (by ii) and $f(v \uparrow A) = T$ by construction.

($\Leftarrow$) If $f(v \uparrow A) = T$, then $\phi$ contains a true disjunct (by i and construction); hence $v(\phi) = T$.

It remains to consider the case in which $f$ is a constant function. In this case, let $\phi$ be:

if $f$ always take the value $F$, and

$$(\ldots(\alpha_1 \land \neg \alpha_i) \lor \ldots \lor (\alpha_n \land \neg \alpha_n))$$

if $f$ always take the value $T$.

First Order Logic

SYNTAX

We describe the syntax of a family of languages $L$, the first order ones. All the languages of this type share a common logical vocabulary, comprising:

1) **Logical Operators**
   a) connectives: $\neg$, $\land$, $\lor$ and $\rightarrow$
   b) quantifiers: $\forall$ and $\exists$

2) **Identity**
   a two-place relation symbol $\approx$

3) **Auxiliary Symbols**
   a) an infinite list of individual variables
   b) three punctuation symbols: $(, )$ and ,

In addition, each member of the family may contain an extralogical vocabulary comprising symbols of one or more of the following types:

1) a list of individual constants

2) For each $n$, a list of $n$-place function symbols

3) For each $n$, a list of $n$-place predicate symbols

The difference between logical and non-logical vocabulary items is reflected in the semantics of languages of type $L$. The definition of what it means for an $L$-sentence to be true involves the meanings of the logical vocabulary, so that these meanings remain fixed when we consider different possible interpretations of a sentence. Our choice of logical vocabulary is motivated by intuitions about what ought to count as logical, but they are not sufficiently precise to determine uncontroversially how the distinction should be drawn. The decision to study first order languages commits us to one view of the domain of logic; others are possible.
We represent the extralogical vocabulary \( V \) of a language as an ordered triple
\[
\langle (P_n)_{n < \omega}, (f_n)_{n < \omega}, (c_n)_{n < \omega} \rangle
\]
where \((P_n)_{n < \omega}\) is a list of predicate symbols, \((f_n)_{n < \omega}\) is a list of function symbols and \((c_n)_{n < \omega}\) is a list of individual constants. Let \( V \) be some fixed extralogical vocabulary, then \( L_\cdot \), the first order language based upon \( V \), is given by the following rules of syntax:

**Definition**

The (individual) **terms** of \( L_\cdot \) are defined by induction as follows:

*Base clause*: each individual variable and each individual constant of \( V \) is a term of \( L_\cdot \).

*Inductive clause*: if \( f \) is an \( n \)-place function symbol of \( V \) and \( t_1, \ldots, t_n \) are terms of \( L_\cdot \), then \( f(t_1, \ldots, t_n) \) is a term of \( L_\cdot \).

*Exclusionary clause*: The only terms of \( L_\cdot \) are those given by the preceding clauses.

**Definition**

The (well-formed) **formulas** of \( L_\cdot \) are defined by induction as follows:

*Base clauses*: 1) If \( t_1, \ldots, t_n \) are terms of \( L_\cdot \) and \( P \) is an \( n \)-place \( P \) predicate symbol of \( V \), then \( P(t_1, \ldots, t_n) \) is a formula of \( L_\cdot \). 2) If \( t \) and \( u \) are terms of \( L_\cdot \), then \( t \neq u \) is a formula of \( L_\cdot \).

*Inductive clauses*: 1) If \( \phi \) is a formula of \( L_\cdot \), then so is \( \neg \phi \). 2) If \( \phi \) and \( \psi \) are formulas of \( L_\cdot \), then so is \( (\phi \land \psi) \). 3) If \( \phi \) and \( \psi \) are formulas of \( L_\cdot \), then so is \( (\phi \lor \psi) \). 4) If \( \phi \) and \( \psi \) are formulas of \( L_\cdot \), then so is \( (\phi \to \psi) \). 5) If \( x \) is a variable and \( \phi \) is a formula of \( L_\cdot \), then so is \( \forall x \phi \). 6) If \( x \) is a variable and \( \phi \) is a formula of \( L_\cdot \), then so is \( \exists x \phi \).

*Exclusionary clause*: The only formulas of \( L_\cdot \) are those given by the preceding clauses.

**Some Syntactic Lemmas**

1) (Unique Readability for Terms) For every term \( t \) of \( L_\cdot \), exactly one of the following holds:
   i) \( t \) is a variable
   ii) \( t \) is a constant
   iii) \( t \) is of the form \( f(t_1, \ldots, t_n) \) for exactly one function symbol \( f \) and terms \( t_1, \ldots, t_n \).

*Proof*  
By induction on terms, using the fact that no initial segment of a term can be a term.

2) (Unique Readability for Formulas) For every formula \( \phi \) of \( L_\cdot \), exactly one of the following holds:
   i) \( \phi \) is of the form \( P(t_1, \ldots, t_n) \), for unique \( P, t_1, \ldots, t_n \)
   ii) \( \phi \) is of the form \( t \neq u \), for unique \( t, u \)
   iii) \( \phi \) is of the form \( \neg \theta \), for unique \( \theta \)
   iv) \( \phi \) is of the form \( (\theta \land \psi) \), for unique \( \theta, \psi \)
   v) \( \phi \) is of the form \( (\theta \lor \psi) \), for unique \( \theta, \psi \)
   vi) \( \phi \) is of the form \( (\theta \to \psi) \), for unique \( \theta, \psi \)
   vii) \( \phi \) is of the form \( \forall x \theta \), for unique \( \theta \)
   viii) \( \phi \) is of the form \( \exists x \theta \), for unique \( \theta \)