Problem Set 6

Advanced Logic
9th October 2022

In the following problems, Arith stands for the ' 'signature of the language of arithmetic": recall that this has one individual constant 0 , one singulary function symbol suc, two binary function symbols $+$ and x, and one binary predicate $\leq$. See below for a reminder of the relevant definitions of FV and $[t/v]$.

Note: problems 2-4 ask you to prove something about all terms and formulae of an arbitrary first-order signature Σ. You may, if you wish, just prove these claims for the special case where Σ is Arith. This will lengthen the proofs a bit, but you may find it helpful if you're getting tripped up by notation like ' $f(t_1, \ldots, t_n)$ ' and ' $R(t_1, \ldots, t_n)$ '.

1. (a) (30%) Prove that no term of the language of arithmetic (i.e., no member of Terms(Arith)) contains the character a.

2. Answer a

(b) (30%) Using the result of part (a), prove that no formula of the language of arithmetic (i.e., no member of $\mathcal{L}($ Arith $))$ contains the character a.

3. (a) (15%) Prove that whenever $s, t \in$ Terms(Σ) and $v \neq FV(s), s[t/v] = s$.

(b) (10%) Using the result of part (a), prove that whenever $P \in \mathcal{L}(\Sigma), t \in$ Terms(Σ), and $v \neq FV(P), P[t/v] = P$

4. (a) (10%) Prove that whenever $s, t \in$ Terms(Σ) and $v \in FV(s), FV(s[t/v]) = (FV(s) \backslash \{v\}) \cup FV(t)$

(b) (5%) Using the result of part (a), prove that whenever $P \in \mathcal{L}(\Sigma), t \in$ Terms(Σ), and $v \in FV(P), FV(P[t/v]) = (FV(P) \backslash \{v\}) \cup FV(t)$.

Additional problems (10% extra credit for successfully solving any one of these.)

1. Suppose that v and v' are distinct variables and t and t' are terms in which neither v nor v' occur free. Show that (a) for every term $s, s[t/v]\left[t'/v'\right] = s\left[t'/v'\right][t/v]$, and (b) for every formula $P, P[t/v]\left[t'/v'\right] = P\left[t'/v'\right][t/v]$.

2. (Optional, ungraded) Define the "standard numeral" function $\langle\cdot\rangle : \mathbb{N} \rightarrow$ Terms(Arith) recursively as follows:

$$\langle 0\rangle = 0$$
$$\langle \operatorname{suc} n\rangle = \operatorname{suc}(\langle n\rangle)$$

Suppose we have some funtion $g : \mathrm{Var} \to \mathbb{N}$. Recursively define a function d : Terms (Arith) $\to \mathbb{N}$ such that $d(v) = g(v)$ for every variable v and $d(\langle n\rangle) = n$ for every $n \in \mathbb{N}$. Prove that the function you defined meets these requirements.

4. (Optional, ungraded) Suppose that we have a first-order signature Σ that is "well-behaved' in the following sense: no function symbol or predicate is an initial substring of any other funtion symbol or predicate, and no function symbol or predicate has a variable as an initial substring. Let PolishTerms (Σ) (the set of "terms in Polish notation" over Σ) be the closure of Var under the family of functions $\langle s_1, \ldots, s_n\rangle \mapsto f \oplus t_1 \oplus \cdots \oplus t_n$ where f is an n-ary function symbol of Σ. (Note that unlike our usual definition of term, this one does not use parentheses and commas.)

Prove unique readability for Polish terms: i.e. that whenever $s_1, \ldots, s_m, t_1, \ldots, t_n \in$ PolishTerms(Σ), f is an m-ary function symbol of Σ, and g is an n-ary function symbol of Σ, if $f s_1 \ldots s_m = g t_1 \ldots t_n$, then $f = g$ and $m = n$ and $s_1 = t_1$ and $\ldots$ and $s_m = t_m$.

Definitions The notations ' FV' and ' $[t/v]$ ' are used ambiguously for two different functions, one defined on Terms(Σ) and the other on $\mathcal{L}(\Sigma)$. (Or we could think of them as standing for the union of those two functions.)

Where Σ is a first-order signature with predicates R_Σ, function symbols F_Σ, and arity function a_Σ, the function FV : Terms$(\Sigma) \to \mathcal{P}$ (Var) is defined recursively by:

$$\begin{aligned}
FV(v) &= \{v\} && (\text{for } v \\
FV(c) &= \emptyset && (\text{for } c \in F_\Sigma \text{ with } a_\Sigma(\\
FV(f(t_1, \ldots, t_n)) &= FV(t_1) \cup \cdots \cup FV(t_n) && (\text{ for } f \in F_\Sigma \text{ with } a_\Sigma(f) = n > 0)
\end{aligned}$$

The companion function $FV : \mathcal{L}(\Sigma) \to \mathcal{P}(\mathrm{Var})$ is defined recursively as follows:

$$\begin{aligned}
FV(F(t_1, \ldots, t_n)) &= FV(t_1) \cup \cdots \cup FV(t_n) \quad (\text{ for } F \in R_\Sigma \text{ with } a_\Sigma(F) = n > 0) \\
FV(t_1 = t_2) &= FV(t_1) \cup FV(t_2) \\
FV(\neg P) &= FV(P) \\
FV(P \wedge Q) &= FV(P) \cup FV(Q) \\
FV(P \vee Q) &= FV(P) \cup FV(Q) \\
FV(P \to Q) &= FV(P) \cup FV(Q) \\
FV(\forall v P) &= FV(P) \backslash \{v\} \\
FV(\exists v P) &= FV(P) \backslash \{v\}
\end{aligned}$$

For any $t \in$ Terms(Σ) and $v \in$ Var, the function $[t/v]$: Terms$(\Sigma) \to$ Terms(Σ) (written in postfix position) is defined recursively by

$$u[t/v] \quad = \begin{cases} t & \text{if } u = v \\ u & \text{if } u \neq v \end{cases} \qquad (\text{for } u \in \mathrm{Var}\Big)$$

The companion function $[t/v] : \mathcal{L}(\Sigma) \to \mathrm{Terms}(\Sigma)$ is defined recursively as follows:

$$
\begin{aligned}
F\left(t_1, \ldots, t_n\right)[t/v] &= F\left(t_1[t/v], \ldots, t_n[t/v]\right) \\
(\neg P)[t/v] &= \neg(P[t/v]) \\
(P \wedge Q)[t/v] &= P[t/v] \wedge Q[t/v] \\
(P \vee Q)[t/v] &= P[t/v] \vee Q[t/v] \\
(P \to Q)[t/v] &= P[t/v] \to Q[t/v] \\
(\forall u P)[t/v] &= \begin{cases} \forall u P & \text{if } v = u \\ \forall u(P[t/v]) & \text{if } v \neq u \text{ and } (u \notin FV(t) \text{ or } v \notin FV(P)) \\ \text{ undefined} & \text{if } v \neq u \text{ and } u \in FV(t) \text{ and } v \in FV(P) \end{cases} \\
\{\exists(\exists u P)[t/v] &= \begin{cases} \exists u & \\ \exists u(P[t/v]) & \text{if } v \neq u \text{ and } (u \notin FV(t) \text{ or } v \notin FV(P)) \\ undefined & \text{if } v \neq u \text{ and } u \in FV(t) \text{ and } v \in FV(P) \end{cases}
\end{aligned}
$$