Capítulo udf

Syntax and Semantics

Basic syntax and semantics for SOL covered so far. As a chapter it's too short. Substitution for second-order variables has to be covered to be able to talk about derivação systems for SOL, and there's some subtle issues there.

-NoValue-

syn.1 Introduction

sol:syn:int:sec

In first-order logic, we combine the non-logical symbols of a given language, O.s., its símbolos de constante, símbolos de função, and símbolos de predicado, with the logical symbols to express things about first-order estruturas. This is done using the notion of satisfaction, which relates a estrutura $\mathfrak{M}$, together with a variable assignment s, and a fórmula φ: $\mathfrak{M}, s \vDash \varphi$ holds iff what φ expresses when its símbolos de constante, símbolos de função, and símbolos de predicado are interpreted as $\mathfrak{M}$ says, and its free variables are interpreted as s says, is true. The interpretation of the predicado de identidade $=$ is built into the definition of $\mathfrak{M}, s \vDash \varphi$, as is the interpretation of $\forall$ and $\exists$. The former is always interpreted as the identity relation on the domínio $|\mathfrak{M}|$ of the structure, and the quantifiers are always interpreted as ranging over the entire domínio. But, crucially, quantification is only allowed over elements of the domínio, and so only object variáveis are allowed to follow a quantifier.

In second-order logic, both the language and the definition of satisfaction are extended to include free and bound function and predicate variables, and quantification over them. These variables are related to símbolos de função and símbolos de predicado the same way that object variables are related to símbolos de constante. They play the same role in the formation of terms and fórmulas of second-order logic, and quantification over them is handled in a similar way. In the *standard* semantics, the second-order quantifiers range

over all possible objects of the right type (n-place functions from $|\mathfrak{M}|$ to $|\mathfrak{M}|$ for function variables, n-place relations for predicate variables). For instance, while $\forall v_0\,(P^1_0(v_0) \vee \neg P^1_0(v_0))$ is a formula in both first- and second-order logic, in the latter we can also consider $\forall V^1_0\,\forall v_0\,(V^1_0(v_0) \vee \neg V^1_0(v_0))$ and $\exists V^1_0\,\forall v_0\,(V^1_0(v_0) \vee \neg V^1_0(v_0))$. Since these contain no free variables, they are sentenças of second-order logic. Here, V^1_0 is a second-order 1-place predicate variable. The allowable interpretations of V^1_0 are the same that we can assign to a 1-place símbolo de predicado like P^1_0, O.s., subsets of $|\mathfrak{M}|$. Quantification over them then amounts to saying that $\forall v_0\,(V^1_0(v_0) \vee \neg V^1_0(v_0))$ holds for all ways of assigning a subset of $|\mathfrak{M}|$ as the value of V^1_0, or for at least one. Since every set either contains or fails to contain a given object, both are true in any estrutura.

-NoValue-

syn.2 Terms and Fórmulas

sol:syn:frm:sec

Like in first-order logic, expressions of second-order logic are built up from a basic vocabulary containing *variáveis*, *símbolos de constante*, *símbolos de predicado* and sometimes *símbolos de função*. From them, together with logical connectives, quantifiers, and punctuation symbols such as parentheses and commas, *terms* and *fórmulas* are formed. The difference is that in addition to variables for objects, second-order logic also contains variables for relations and functions, and allows quantification over them. So the logical symbols of second-order logic are those of first-order logic, plus:

1. A infinitos enumeráveis set of second-order relation variáveis of every arity n: V^n_0, V^n_1, V^n_2, ...
2. A infinitos enumeráveis set of second-order function variáveis: u^n_0, u^n_1, u^n_2, ...

Just as we use x, y, z as meta-variables for first-order variables v_i, we'll use X, Y, Z, etc., as metavariables for V^n_i and u, v, etc., as meta-variables for u^n_i.

explanation

The non-logical symbols of a second-order language are specified the same way a first-order language is: by listing its símbolos de constante, símbolos de função, and símbolos de predicado.

In first-order logic, the predicado de identidade $=$ is usually included. In first-order logic, the non-logical symbols of a language $\mathcal{L}$ are crucial to allow us to express anything interesting. There are of course sentenças that use no non-logical symbols, but with only $=$ it is hard to say anything interesting. In second-order logic, since we have an unlimited supply of relation and function variables, we can say anything we can say in a first-order language even without a special supply of non-logical symbols.

Definição syn.1 (Second-order Terms). The set of *second-order terms* of $\mathcal{L}$, $\mathrm{Trm}^2(\mathcal{L})$, is defined by adding to **??** the clause

1. If u is an n-place function variable and $t_1, \ldots, t_n$ are terms, then $u(t_1, \ldots, t_n)$ is a term.

So, a second-order term looks just like a first-order term, except that where a first-order term contains a símbolo de função f_i^n, a second-order term may contain a function variable u_i^n in its place. explanation

Definição syn.2 (Second-order fórmula). The set of *second-order fórmulas* $\mathrm{Frm}^2(\mathcal{L})$ of the language $\mathcal{L}$ is defined by adding to **??** the clauses

1. If X is an n-place predicate variable and $t_1, \ldots, t_n$ are second-order terms of $\mathcal{L}$, then $X(t_1, \ldots, t_n)$ is an atomic fórmula.
2. If φ is a fórmula and u is a function variable, then $\forall u\, \varphi$ is a fórmula.
3. If φ is a fórmula and X is a predicate variable, then $\forall X\, \varphi$ is a fórmula.
4. If φ is a fórmula and u is a function variable, then $\exists u\, \varphi$ is a fórmula.
5. If φ is a fórmula and X is a predicate variable, then $\exists X\, \varphi$ is a fórmula.

-NoValue-

syn.3 Satisfaction

sol:syn:sat:sec To define the satisfaction relation $\mathfrak{M}, s \vDash \varphi$ for second-order fórmulas, we have to extend the definitions to cover second-order variáveis. The notion of a estrutura is the same for second-order logic as it is for first-order logic. There is only a difference for variable assignments s: these now must not just provide values for the first-order variáveis, but also for the second-order variáveis. explanation

Definição syn.3 (Variable Assignment). A *variable assignment* s for a estrutura $\mathfrak{M}$ is a function which maps each

1. object variável v_i to an element of $|\mathfrak{M}|$, O.s., $s(v_i) \in |\mathfrak{M}|$
2. n-place relation variable V_i^n to an n-place relation on $|\mathfrak{M}|$, O.s., $s(V_i^n) \subseteq |\mathfrak{M}|^n$;
3. n-place function variable u_i^n to an n-place function from $|\mathfrak{M}|$ to $|\mathfrak{M}|$, O.s., $s(u_i^n)\colon |\mathfrak{M}|^n \to |\mathfrak{M}|$;

A estrutura assigns a valor to each símbolo de constante and símbolo de função, and a second-order variable assignment assigns objects and functions to each object and function variable. Together, they let us assign a value to every term. explanation

Definição syn.4 (Valor of a Term). If t is a term of the language $\mathcal{L}$, $\mathfrak{M}$ is a estrutura for $\mathcal{L}$, and s is a variável assignment for $\mathfrak{M}$, the *valor* $\mathrm{Val}_s^{\mathfrak{M}}(t)$ is defined as for first-order terms, plus the following clause:

$t \equiv u(t_1, \ldots, t_n)$:

$$\mathrm{Val}_s^{\mathfrak{M}}(t) = s(u)(\mathrm{Val}_s^{\mathfrak{M}}(t_1), \ldots, \mathrm{Val}_s^{\mathfrak{M}}(t_n)).$$

Definição syn.5 (x-Variant). If s is a variável assignment for a estrutura $\mathfrak{M}$, then any variável assignment s' for $\mathfrak{M}$ which differs from s at most in what it assigns to x is called an *x-variant* of s. If s' is an x-variant of s we write $s' \sim_x s$. (Similarly for second-order variables X or u.)

Definição syn.6. If s is a variável assignment for a estrutura $\mathfrak{M}$ and $m \in |\mathfrak{M}|$, then the assignment $s[m/x]$ is the variável assignment defined by

$$s[m/y] = \begin{cases} m & \text{if } y \equiv x \\ s(y) & \text{otherwise,} \end{cases}$$

If X is an n-place relation variável and $M \subseteq |\mathfrak{M}|^n$, then $s[M/X]$ is the variável assignment defined by

$$s[M/y] = \begin{cases} M & \text{if } y \equiv X \\ s(y) & \text{otherwise.} \end{cases}$$

If u is an n-place function variável and $f\colon |\mathfrak{M}|^n \to |\mathfrak{M}|$, then $s[f/u]$ is the variável assignment defined by

$$s[f/y] = \begin{cases} f & \text{if } y \equiv u \\ s(y) & \text{otherwise.} \end{cases}$$

In each case, y may be any first- or second-order variável.

Definição syn.7 (Satisfaction). For second-order fórmulas φ, the definition of satisfaction is like **??** with the addition of:

1. $\varphi \equiv X^n(t_1, \ldots, t_n)$: $\mathfrak{M}, s \vDash \varphi$ iff $\langle \mathrm{Val}_s^{\mathfrak{M}}(t_1), \ldots, \mathrm{Val}_s^{\mathfrak{M}}(t_n)\rangle \in s(X^n)$.
2. $\varphi \equiv \forall X\, \psi$: $\mathfrak{M}, s \vDash \varphi$ iff for every $M \subseteq |\mathfrak{M}|^n$, $\mathfrak{M}, s[M/X] \vDash \psi$.
3. $\varphi \equiv \exists X\, \psi$: $\mathfrak{M}, s \vDash \varphi$ iff for at least one $M \subseteq |\mathfrak{M}|^n$ so that $\mathfrak{M}, s[M/X] \vDash \psi$.
4. $\varphi \equiv \forall u\, \psi$: $\mathfrak{M}, s \vDash \varphi$ iff for every $f\colon |\mathfrak{M}|^n \to |\mathfrak{M}|$, $\mathfrak{M}, s[f/u] \vDash \psi$.
5. $\varphi \equiv \exists u\, \psi$: $\mathfrak{M}, s \vDash \varphi$ iff for at least one $f\colon |\mathfrak{M}|^n \to |\mathfrak{M}|$ so that $\mathfrak{M}, s[f/u] \vDash \psi$.

Exemplo syn.8. Consider the fórmula $\forall z\, (X(z) \leftrightarrow \neg Y(z))$. It contains no second-order quantifiers, but does contain the second-order variáveis X and Y (here understood to be one-place). The corresponding first-order sentença $\forall z\, (P(z) \leftrightarrow \neg R(z))$ says that whatever falls under the interpretation of P does

not fall under the interpretation of R and vice versa. In a estrutura, the interpretation of a símbolo de predicado P is given by the interpretation $P^{\mathfrak{M}}$. But for second-order variáveis like X and Y, the interpretation is provided, not by the estrutura itself, but by a variável assignment. Since the second-order fórmula is not a sentença (it includes free variáveis X and Y), it is only satisfied relative to a estrutura $\mathfrak{M}$ together with a variável assignment s.

$\mathfrak{M}, s \vDash \forall z\,(X(z) \leftrightarrow \neg Y(z))$ whenever the membros of $s(X)$ are not membros of $s(Y)$, and vice versa, O.s., iff $s(Y) = |\mathfrak{M}| \setminus s(X)$. For instance, take $|\mathfrak{M}| = \{1, 2, 3\}$. Since no símbolos de predicado, símbolos de função, or símbolos de constante are involved, the domain of $\mathfrak{M}$ is all that is relevant. Now for $s_1(X) = \{1, 2\}$ and $s_1(Y) = \{3\}$, we have $\mathfrak{M}, s_1 \vDash \forall z\,(X(z) \leftrightarrow \neg Y(z))$.

By contrast, if we have $s_2(X) = \{1, 2\}$ and $s_2(Y) = \{2, 3\}$, $\mathfrak{M}, s_2 \nvDash \forall z\,(X(z) \leftrightarrow \neg Y(z))$. That's because $\mathfrak{M}, s_2[2/z] \vDash X(z)$ (since $2 \in s_2[2/z](X)$) but $\mathfrak{M}, s_2[2/z] \nvDash \neg Y(z)$ (since also $2 \in s_2[2/z](Y)$).

Exemplo syn.9. $\mathfrak{M}, s \vDash \exists Y\,(\exists y\, Y(y) \wedge \forall z\,(X(z) \leftrightarrow \neg Y(z)))$ if there is an $N \subseteq |\mathfrak{M}|$ such that $\mathfrak{M}, s[N/Y] \vDash (\exists y\, Y(y) \wedge \forall z\,(X(z) \leftrightarrow \neg Y(z)))$. And that is the case for any $N \neq \emptyset$ (so that $\mathfrak{M}, s[N/Y] \vDash \exists y\, Y(y)$) and, as in the previous example, $N = |\mathfrak{M}| \setminus s(X)$. In other words, $\mathfrak{M}, s \vDash \exists Y\,(\exists y\, Y(y) \wedge \forall z\,(X(z) \leftrightarrow \neg Y(z)))$ iff $|\mathfrak{M}| \setminus s(X)$ is non-empty, O.s., $s(X) \neq |\mathfrak{M}|$. So, the fórmula is satisfied, p.ex., if $|\mathfrak{M}| = \{1, 2, 3\}$ and $s(X) = \{1, 2\}$, but not if $s(X) = \{1, 2, 3\} = |\mathfrak{M}|$.

Since the fórmula is not satisfied whenever $s(X) = |\mathfrak{M}|$, the sentença

$$\forall X\, \exists Y\,(\exists y\, Y(y) \wedge \forall z\,(X(z) \leftrightarrow \neg Y(z)))$$

is never satisfied: For any estrutura $\mathfrak{M}$, the assignment $s(X) = |\mathfrak{M}|$ will make the sentença false. On the other hand, the sentence

$$\exists X\, \exists Y\,(\exists y\, Y(y) \wedge \forall z\,(X(z) \leftrightarrow \neg Y(z)))$$

is satisfied relative to any assignment s, since we can always find $M \subseteq |\mathfrak{M}|$ but $M \neq |\mathfrak{M}|$ (p.ex., $M = \emptyset$).

Exemplo syn.10. The second-order sentença $\forall X\, \forall y\, X(y)$ says that every 1-place relation, O.s., every property, holds of every object. That is clearly never true, since in every $\mathfrak{M}$, for a variable assignment s with $s(X) = \emptyset$, and $s(y) = a \in |\mathfrak{M}|$ we have $\mathfrak{M}, s \nvDash X(y)$. This means that $\varphi \to \forall X\, \forall y\, X(y)$ is equivalent in second-order logic to $\neg\varphi$, that is: $\mathfrak{M} \vDash \varphi \to \forall X\, \forall y\, X(y)$ iff $\mathfrak{M} \vDash \neg\varphi$. In other words, in second-order logic we can define $\neg$ using $\forall$ and $\to$.

Problema syn.1. Show that in second-order logic $\forall$ and $\to$ can define the other connectives:

1. Prove that in second-order logic $\varphi \wedge \psi$ is equivalent to $\forall X\,(\varphi \to (\psi \to \forall x\, X(x)) \to \forall x\, X(x))$.

2. Find a second-order formula using only $\forall$ and $\to$ equivalent to $\varphi \vee \psi$.

-NoValue-

syn.4 Semantic Notions

explanation

The central logical notions of *validity*, *entailment*, and *satisfiability* are defined the same way for second-order logic as they are for first-order logic, except that the underlying satisfaction relation is now that for second-order fórmulas. A second-order sentença, of course, is a fórmula in which all variables, including predicate and function variables, are bound.

sol:syn:sem:sec

Definição syn.11 (Validity). A sentence φ is *valid*, $\vDash \varphi$, iff $\mathfrak{M} \vDash \varphi$ for every estrutura $\mathfrak{M}$.

Definição syn.12 (Entailment). A set of sentences Γ *entails* a sentence φ, $\Gamma \vDash \varphi$, iff for every estrutura $\mathfrak{M}$ with $\mathfrak{M} \vDash \Gamma$, $\mathfrak{M} \vDash \varphi$.

Definição syn.13 (Satisfiability). A set of sentences Γ is *satisfiable* if $\mathfrak{M} \vDash \Gamma$ for some estrutura $\mathfrak{M}$. If Γ is not satisfiable it is called *unsatisfiable*.

-NoValue-

syn.5 Expressive Power

explanation

Quantification over second-order variables is responsible for an immense increase in the expressive power of the language over that of first-order logic. Second-order existential quantification lets us say that functions or relations with certain properties exist. In first-order logic, the only way to do that is to specify a non-logical symbol (O.s., a símbolo de função or símbolo de predicado) for this purpose. Second-order universal quantification lets us say that all subsets of, relations on, or functions from the domínio to the domínio have a property. In first-order logic, we can only say that the subsets, relations, or functions assigned to one of the non-logical symbols of the language have a property. And when we say that subsets, relations, functions exist that have a property, or that all of them have it, we can use second-order quantification in specifying this property as well. This lets us define relations not definable in first-order logic, and express properties of the domain not expressible in first-order logic.

sol:syn:exp:sec

Definição syn.14. If $\mathfrak{M}$ is a estrutura for a language $\mathcal{L}$, a relation $R \subseteq |\mathfrak{M}|^2$ is *definable* in $\mathcal{L}$ if there is some fórmula $\varphi_R(x, y)$ with only the variables x and y free, such that $R(a, b)$ holds (O.s., $\langle a, b\rangle \in R$) iff $\mathfrak{M}, s \vDash \varphi_R(x, y)$ for $s(x) = a$ and $s(y) = b$.

Exemplo syn.15. In first-order logic we can define the identity relation $\mathrm{Id}_{|\mathfrak{M}|}$ (O.s., $\{\langle a, a\rangle : a \in |\mathfrak{M}|\}$) by the formula $x = y$. In second-order logic, we can define this relation *without* $=$. For if a and b are the same membro of $|\mathfrak{M}|$, then they are membros of the same subsets of $|\mathfrak{M}|$ (since sets are determined by their membros). Conversely, if a and b are different, then they are not membros of the same subsets: p.ex., $a \in \{a\}$ but $b \notin \{a\}$ if $a \neq b$. So "being membros of

the same subsets of $|\mathfrak{M}|$" is a relation that holds of a and b iff $a = b$. It is a relation that can be expressed in second-order logic, since we can quantify over all subsets of $|\mathfrak{M}|$. Hence, the following fórmula defines $\mathrm{Id}_{|\mathfrak{M}|}$:

$$\forall X\,(X(x) \leftrightarrow X(y))$$

Problema syn.2. Show that $\forall X\,(X(x) \to X(y))$ (note: $\to$ not $\leftrightarrow$!) defines $\mathrm{Id}_{|\mathfrak{M}|}$.

Exemplo syn.16. If R is a two-place símbolo de predicado, $R^{\mathfrak{M}}$ is a two-place relation on $|\mathfrak{M}|$. Perhaps somewhat confusingly, we'll use R as the símbolo de predicado for R and for the relation $R^{\mathfrak{M}}$ itself. The *transitive closure* R^* of R is the relation that holds between a and b iff for some $c_1, \ldots, c_k$, $R(a, c_1)$, $R(c_1, c_2)$, $\ldots$, $R(c_k, b)$ holds. This includes the case if $k = 0$, O.s., if $R(a, b)$ holds, so does $R^*(a, b)$. This means that $R \subseteq R^*$. In fact, R^* is the smallest relation that includes R and that is transitive. We can say in second-order logic that X is a transitive relation that includes R:

$$\psi_R(X) \equiv \forall x\,\forall y\,(R(x, y) \to X(x, y)) \wedge \forall x\,\forall y\,\forall z\,((X(x, y) \wedge X(y, z)) \to X(x, z)).$$

The first conjunct says that $R \subseteq X$ and the second that X is transitive.

To say that X is the smallest such relation is to say that it is itself included in every relation that includes R and is transitive. So we can define the transitive closure of R by the fórmula

$$R^*(X) \equiv \psi_R(X) \wedge \forall Y\,(\psi_R(Y) \to \forall x\,\forall y\,(X(x, y) \to Y(x, y))).$$

We have $\mathfrak{M}, s \vDash R^*(X)$ iff $s(X) = R^*$. The transitive closure of R cannot be expressed in first-order logic.

-NoValue-

syn.6 Describing Infinite and Enumerável Domínios

sol:syn:siz:sec

A set M is (Dedekind) infinite iff there is uma injetiva function $f\colon M \to M$ which is not sobrejetiva, O.s., with $\mathrm{Im}(f) \neq M$. In first-order logic, we can consider a one-place símbolo de função f and say that the function $f^{\mathfrak{M}}$ assigned to it in a estrutura $\mathfrak{M}$ is injetiva and $\mathrm{Im}(f) \neq |\mathfrak{M}|$:

$$\forall x\,\forall y\,(f(x) = f(y) \to x = y) \wedge \exists y\,\forall x\,y \neq f(x).$$

If $\mathfrak{M}$ satisfies this sentença, $f^{\mathfrak{M}} : |\mathfrak{M}| \to |\mathfrak{M}|$ is injetiva, and so $|\mathfrak{M}|$ must be infinite. If $|\mathfrak{M}|$ is infinite, and hence such a function exists, we can let $f^{\mathfrak{M}}$ be that function and $\mathfrak{M}$ will satisfy the sentença. However, this requires that our language contains the non-logical symbol f which we use for this purpose.

In second-order logic, we can simply say that such a function *exists*. This no-longer requires f, and we obtain the sentença in pure second-order logic

$$\text{Inf} \equiv \exists u\, (\forall x\, \forall y\, (u(x) = u(y) \to x = y) \land \exists y\, \forall x\, y \neq u(x)).$$

$\mathfrak{M} \vDash \text{Inf}$ iff $|\mathfrak{M}|$ is infinite. We can then define $\text{Fin} \equiv \neg\text{Inf}$; $\mathfrak{M} \vDash \text{Fin}$ iff $|\mathfrak{M}|$ is finite. No single sentença of pure first-order logic can express that the domínio is infinite although an infinite set of them can. There is no set of sentenças of pure first-order logic that is satisfied in a estrutura iff its domain is finite.

Proposição syn.17. $\mathfrak{M} \vDash \text{Inf}$ *iff* $|\mathfrak{M}|$ *is infinite.*

Prova. $\mathfrak{M} \vDash \text{Inf}$ iff $\mathfrak{M}, s \vDash \forall x\, \forall y\, (u(x) = u(y) \to x = y) \land \exists y\, \forall x\, y \neq u(x)$ for some s. If it does, $s(u)$ is uma injetiva function, and some $y \in |\mathfrak{M}|$ is not in the range of $s(u)$. Conversely, if there is uma injetiva $f\colon |\mathfrak{M}| \to |\mathfrak{M}|$ with $\text{Im}(f) \neq |\mathfrak{M}|$, then $s(u) = f$ is such a variable assignment. □

A set M is enumerável if there is an enumeration

$$m_0, m_1, m_2, \ldots$$

of its membros (without repetitions but possibly finite). Such an enumeration exists iff there is um membro $z \in M$ and a function $f\colon M \to M$ such that z, $f(z)$, $f(f(z))$, ..., are all the membros of M. For if the enumeration exists, $z = m_0$ and $f(m_k) = m_{k+1}$ (or $f(m_k) = m_k$ if m_k is the last membro of the enumeration) are the requisite membro and function. On the other hand, if such a z and f exist, then z, $f(z)$, $f(f(z))$, ..., is an enumeration of M, and M is enumerável. We can express the existence of z and f in second-order logic to produce a sentença true in a estrutura iff the estrutura is enumerável:

$$\text{Count} \equiv \exists z\, \exists u\, \forall X\, ((X(z) \land \forall x\, (X(x) \to X(u(x)))) \to \forall x\, X(x))$$

Proposição syn.18. $\mathfrak{M} \vDash \text{Count}$ *iff* $|\mathfrak{M}|$ *is enumerável.*

Prova. Suppose $|\mathfrak{M}|$ is enumerável, and let m_0, m_1, ..., be an enumeration. By removing repetitions we can guarantee that no m_k appears twice. Define $f(m_k) = m_{k+1}$ and let $s(z) = m_0$ and $s(u) = f$. We show that

$$\mathfrak{M}, s \vDash \forall X\, ((X(z) \land \forall x\, (X(x) \to X(u(x)))) \to \forall x\, X(x))$$

Suppose $M \subseteq |\mathfrak{M}|$ is arbitrary. Suppose further that $\mathfrak{M}, s[M/X] \vDash (X(z) \land \forall x\, (X(x) \to X(u(x))))$. Then $s[M/X](z) \in M$ and whenever $x \in M$, also $(s[M/X](u))(x) \in M$. In other words, since $s[M/X] \sim_X s$, $m_0 \in M$ and if $x \in M$ then $f(x) \in M$, so $m_0 \in M$, $m_1 = f(m_0) \in M$, $m_2 = f(f(m_0)) \in M$, etc. Thus, $M = |\mathfrak{M}|$, and so $\mathfrak{M}, s[M/X] \vDash \forall x\, X(x)$. Since $M \subseteq |\mathfrak{M}|$ was arbitrary, we are done: $\mathfrak{M} \vDash \text{Count}$.

Now assume that $\mathfrak{M} \vDash \text{Count}$, O.s.,

$$\mathfrak{M}, s \vDash \forall X\, ((X(z) \land \forall x\, (X(x) \to X(u(x)))) \to \forall x\, X(x))$$

for some s. Let $m = s(z)$ and $f = s(u)$ and consider $M = \{m, f(m), f(f(m)), \dots\}$. M so defined is clearly enumerável. Then

$$\mathfrak{M}, s[M/X] \vDash (X(z) \wedge \forall x\,(X(x) \to X(u(x)))) \to \forall x\, X(x)$$

by assumption. Also, $\mathfrak{M}, s[M/X] \vDash X(z)$ since $M \ni m = s[M/X](z)$, and also $\mathfrak{M}, s[M/X] \vDash \forall x\,(X(x) \to X(u(x)))$ since whenever $x \in M$ also $f(x) \in M$. So, since both antecedent and conditional are satisfied, the consequent must also be: $\mathfrak{M}, s[M/X] \vDash \forall x\, X(x)$. But that means that $M = |\mathfrak{M}|$, and so $|\mathfrak{M}|$ is enumerável since M is, by definition. □

Problema syn.3. The sentença Inf ∧ Count is true in all and only infinito enumerável domains. Adjust the definition of Count so that it becomes a different sentença that directly expresses that the domain is infinito enumerável, and prove that it does.

Créditos das Imagens

Referências Bibliográficas