

Capítulo udf

Arithmetization of Syntax

Note that arithmetization for signed tableaux is not yet available.

-NoValue-

art.1 Introduction

inc:art:int:
sec

In order to connect computability and logic, we need a way to talk about the objects of logic (symbols, terms, **fórmulas**, **derivações**), operations on them, and their properties and relations, in a way amenable to computational treatment. We can do this directly, by considering computable functions and relations on symbols, sequences of symbols, and other objects built from them. Since the objects of logical syntax are all finite and built from a **enumerável** sets of symbols, this is possible for some models of computation. But other models of computation—such as the recursive functions—are restricted to numbers, their relations and functions. Moreover, ultimately we also want to be able to deal with syntax within certain theories, specifically, in theories formulated in the language of arithmetic. In these cases it is necessary to *arithmetize* syntax, O.s., to represent syntactic objects, operations on them, and their relations, as numbers, arithmetical functions, and arithmetical relations, respectively. The idea, which goes back to Leibniz, is to assign numbers to syntactic objects.

It is relatively straightforward to assign numbers to symbols as their “codes.” Some symbols pose a bit of a challenge, since, p.ex., there are infinitely many **variáveis**, and even infinitely many **símbolos de função** of each arity n . But of course it’s possible to assign numbers to symbols systematically in such a way that, say, v_2 and v_3 are assigned different codes. Sequences of symbols (such as terms and **fórmulas**) are a bigger challenge. But if we can deal with sequences of numbers purely arithmetically (p.ex., by the powers-of-primes coding of sequences), we can extend the coding of individual symbols to coding of sequences of symbols, and then further to sequences or other arrangements

of *fórmulas*, such as *derivações*. This extended coding is called “Gödel numbering.” Every term, *fórmula*, and *derivação* is assigned a Gödel number.

By coding sequences of symbols as sequences of their codes, and by choosing a system of coding sequences that can be dealt with using computable functions, we can then also deal with Gödel numbers using computable functions. In practice, all the relevant functions will be primitive recursive. For instance, computing the length of a sequence and computing the i -th element of a sequence from the code of the sequence are both primitive recursive. If the number coding the sequence is, p.ex., the Gödel number of a *fórmula* φ , we immediately see that the length of a *fórmula* and the (code of the) i -th symbol in a *fórmula* can also be computed from the Gödel number of φ . It is a bit harder to prove that, p.ex., the property of being the Gödel number of a correctly formed term or of a correct *derivação* is primitive recursive. It is nevertheless possible, because the sequences of interest (terms, *fórmulas*, *derivações*) are inductively defined.

As an example, consider the operation of substitution. If φ is a formula, x a variable, and t a term, then $\varphi[t/x]$ is the result of replacing every free occurrence of x in φ by t . Now suppose we have assigned Gödel numbers to φ , x , t —say, k , l , and m , respectively. The same scheme assigns a Gödel number to $\varphi[t/x]$, say, n . This mapping—of k , l , and m to n —is the arithmetical analog of the substitution operation. When the substitution operation maps φ , x , t to $\varphi[t/x]$, the arithmetized substitution functions maps the Gödel numbers k , l , m to the Gödel number n . We will see that this function is primitive recursive.

Arithmetization of syntax is not just of abstract interest, although it was originally a non-trivial insight that languages like the language of arithmetic, which do not come with mechanisms for “talking about” languages can, after all, formalize complex properties of expressions. It is then just a small step to ask what a theory in this language, such as Peano arithmetic, can *prove* about its own language (including, p.ex., whether *sentenças* are provable or true). This leads us to the famous limitative theorems of Gödel (about unprovability) and Tarski (the undefinability of truth). But the trick of arithmetizing syntax is also important in order to prove some important results in computability theory, p.ex., about the computational power of theories or the relationship between different models of computability. The arithmetization of syntax serves as a model for arithmetizing other objects and properties. For instance, it is similarly possible to arithmetize configurations and computations (say, of Turing machines). This makes it possible to simulate computations in one model (p.ex., Turing machines) in another (p.ex., recursive functions).

-NoValue-

art.2 Coding Symbols

The basic language $\mathcal{L}$ of first order logic makes use of the symbols

inc:art:cod:
sec

$$\perp \quad \neg \quad \vee \quad \wedge \quad \rightarrow \quad \forall \quad \exists \quad = \quad (\quad) \quad ,$$

together with **enumerável** sets of variables and **símbolos de constante**, and **enumerável** sets of **símbolos de função** and **símbolos de predicado** of arbitrary arity. We can assign *codes* to each of these symbols in such a way that every symbol is assigned a unique number as its code, and no two different symbols are assigned the same number. We know that this is possible since the set of all symbols is **enumerável** and so there is **uma bijeção** between it and the set of natural numbers. But we want to make sure that we can recover the symbol (as well as some information about it, p.ex., the arity of **a símbolo de função**) from its code in a computable way. There are many possible ways of doing this, of course. Here is one such way, which uses primitive recursive functions. (Recall that $\langle n_0, \dots, n_k \rangle$ is the number coding the sequence of numbers $n_0, \dots, n_k$.)

Definição art.1. If s is a symbol of $\mathcal{L}$, let the *symbol code* c_s be defined as follows:

1. If s is among the logical symbols, c_s is given by the following table:

$\perp$	$\neg$	$\vee$	$\wedge$	$\rightarrow$	$\forall$
$\langle 0, 0 \rangle$	$\langle 0, 1 \rangle$	$\langle 0, 2 \rangle$	$\langle 0, 3 \rangle$	$\langle 0, 4 \rangle$	$\langle 0, 5 \rangle$
$\exists$	$=$	$($	$)$	$,$	
$\langle 0, 6 \rangle$	$\langle 0, 7 \rangle$	$\langle 0, 8 \rangle$	$\langle 0, 9 \rangle$	$\langle 0, 10 \rangle$	

2. If s is the i -th variable v_i , then $c_s = \langle 1, i \rangle$.
3. If s is the i -th **símbolo de constante** c_i , then $c_s = \langle 2, i \rangle$.
4. If s is the i -th n -ary **símbolo de função** f_i^n , then $c_s = \langle 3, n, i \rangle$.
5. If s is the i -th n -ary **símbolo de predicado** P_i^n , then $c_s = \langle 4, n, i \rangle$.

Proposição art.2. *The following relations are primitive recursive:*

1. $\text{Fn}(x, n)$ iff x is the code of f_i^n for some i , O.s., x is the code of an n -ary **símbolo de função**.
2. $\text{Pred}(x, n)$ iff x is the code of P_i^n for some i or x is the code of $=$ and $n = 2$, O.s., x is the code of an n -ary **símbolo de predicado**.

Definição art.3. If $s_0, \dots, s_{n-1}$ is a sequence of symbols, its *Gödel number* is $\langle c_{s_0}, \dots, c_{s_{n-1}} \rangle$.

Note that *codes* and *Gödel numbers* are different things. For instance, the variable v_5 has a code $c_{v_5} = \langle 1, 5 \rangle = 2^2 \cdot 3^6$. But the variable v_5 considered as a term is also a sequence of symbols (of length 1). The *Gödel number* $\#v_5\#$ of the term v_5 is $\langle c_{v_5} \rangle = 2^{c_{v_5}+1} = 2^{2^2 \cdot 3^6 + 1}$. explanation

Exemplo art.4. Recall that if $k_0, \dots, k_{n-1}$ is a sequence of numbers, then the code of the sequence $\langle k_0, \dots, k_{n-1} \rangle$ in the power-of-primes coding is

$$2^{k_0+1} \cdot 3^{k_1+1} \cdot \dots \cdot p_{n-1}^{k_{n-1}+1},$$

where p_i is the i -th prime (starting with $p_0 = 2$). So for instance, the formula $v_0 = 0$, or, more explicitly, $=(v_0, c_0)$, has the Gödel number

$$\langle c_0, c_0, c_{v_0}, c_0, c_{c_0}, c_0 \rangle.$$

Here, c_0 is $\langle 0, 7 \rangle = 2^{0+1} \cdot 3^{7+1}$, c_{v_0} is $\langle 1, 0 \rangle = 2^{1+1} \cdot 3^{0+1}$, etc. So $\# = (v_0, c_0)^\#$ is

$$\begin{aligned} 2^{c_0+1} \cdot 3^{c_0+1} \cdot 5^{c_{v_0}+1} \cdot 7^{c_0+1} \cdot 11^{c_{c_0}+1} \cdot 13^{c_0+1} = \\ 2^{2^1 \cdot 3^8 + 1} \cdot 3^{2^1 \cdot 3^9 + 1} \cdot 5^{2^2 \cdot 3^1 + 1} \cdot 7^{2^1 \cdot 3^{11} + 1} \cdot 11^{2^3 \cdot 3^1 + 1} \cdot 13^{2^1 \cdot 3^{10} + 1} = \\ 2^{13 \cdot 123} \cdot 3^{39 \cdot 367} \cdot 5^{13} \cdot 7^{354 \cdot 295} \cdot 11^{25} \cdot 13^{118 \cdot 099}. \end{aligned}$$

-NoValue-

art.3 Coding Terms

explanation

A term is simply a certain kind of sequence of symbols: it is built up inductively from constants and variables according to the formation rules for terms. Since sequences of symbols can be coded as numbers—using a coding scheme for the symbols plus a way to code sequences of numbers—assigning Gödel numbers to terms is not difficult. The challenge is rather to show that the property a number has if it is the Gödel number of a correctly formed term is computable, or in fact primitive recursive.

inc:art:trm:
sec

Variáveis and **símbolos de constante** are the simplest terms, and testing whether x is the Gödel number of such a term is easy: $\text{Var}(x)$ holds if x is $\#_{v_i}$ for some i . In other words, x is a sequence of length 1 and its single element $(x)_0$ is the code of some **variável** v_i , O.s., x is $\langle \langle 1, i \rangle \rangle$ for some i . Similarly, $\text{Const}(x)$ holds if x is $\#_{c_i}$ for some i . Both of these relations are primitive recursive, since if such an i exists, it must be $< x$:

$$\begin{aligned} \text{Var}(x) &\Leftrightarrow (\exists i < x) x = \langle \langle 1, i \rangle \rangle \\ \text{Const}(x) &\Leftrightarrow (\exists i < x) x = \langle \langle 2, i \rangle \rangle \end{aligned}$$

Proposição art.5. *The relations $\text{Term}(x)$ and $\text{CTerm}(x)$ which hold iff x is the Gödel number of a term or a closed term, respectively, are primitive recursive.*

inc:art:trm:
prop:term-primrec

Prova. A sequence of symbols s is a term iff there is a sequence $s_0, \dots, s_{k-1} = s$ of terms which records how the term s was formed from **símbolos de constante** and **variáveis** according to the formation rules for terms. To express that such a putative formation sequence follows the formation rules it has to be the case that, for each $i < k$, either

1. s_i is a **variável** v_j , or
2. s_i is a **símbolo de constante** c_j , or
3. s_i is built from n terms $t_1, \dots, t_n$ occurring prior to place i using an n -place **símbolo de função** f_j^n .

To show that the corresponding relation on Gödel numbers is primitive recursive, we have to express this condition primitive recursively, O.s., using primitive recursive functions, relations, and bounded quantification.

Suppose y is the number that codes the sequence $s_0, \dots, s_{k-1}$, O.s., $y = \langle \#s_0\#, \dots, \#s_{k-1}\# \rangle$. It codes a formation sequence for the term with Gödel number x iff for all $i < k$:

1. $\text{Var}((y)_i)$, or
2. $\text{Const}((y)_i)$, or
3. there is an n and a number $z = \langle z_1, \dots, z_n \rangle$ such that each z_l is equal to some $(y)_{i'}$ for $i' < i$ and

$$(y)_i = \#f_j^n(\# \frown \text{flatten}(z) \frown \#)\#,$$

and moreover $(y)_{k-1} = x$. (The function $\text{flatten}(z)$ turns the sequence $\langle \#t_1\#, \dots, \#t_n\# \rangle$ into $\#t_1, \dots, t_n\#$ and is primitive recursive.)

The indices j, n , the Gödel numbers z_l of the terms t_l , and the code z of the sequence $\langle z_1, \dots, z_n \rangle$, in (3) are all less than y . We can replace k above with $\text{len}(y)$. Hence we can express “ y is the code of a formation sequence of the term with Gödel number x ” in a way that shows that this relation is primitive recursive.

We now just have to convince ourselves that there is a primitive recursive bound on y . But if x is the Gödel number of a term, it must have a formation sequence with at most $\text{len}(x)$ terms (since every term in the formation sequence of s must start at some place in s , and no two subterms can start at the same place). The Gödel number of each subterm of s is of course $\leq x$. Hence, there always is a formation sequence with code $\leq p_{k-1}^{k(x+1)}$, where $k = \text{len}(x)$.

For CTerm, simply leave out the clause for **variáveis**. $\square$

Problema art.1. Show that the function $\text{flatten}(z)$, which turns the sequence $\langle \#t_1\#, \dots, \#t_n\# \rangle$ into $\#t_1, \dots, t_n\#$, is primitive recursive.

*inc:art:trm:
prop:num-primrec*

Proposição art.6. *The function $\text{num}(n) = \#\bar{n}\#$ is primitive recursive.*

Prova. We define $\text{num}(n)$ by primitive recursion:

$$\begin{aligned} \text{num}(0) &= \#0\# \\ \text{num}(n+1) &= \#s(\# \frown \text{num}(n) \frown \#)\#. \end{aligned} \quad \square$$

-NoValue-

art.4 Coding Fórmulas

Once we have defined the relation $\text{Term}(x)$ primitive recursively, we can use it to define the corresponding relation for fórmulas, $\text{Frm}(x)$ primitive recursively. inc:art:frm:
sec

Proposição art.7. *The relation $\text{Atom}(x)$ which holds iff x is the Gödel number of an atomic fórmula, is primitive recursive.*

Prova. The number x is the Gödel number of an atomic fórmula iff one of the following holds:

1. There are $n, j < x$, and $z < x$ such that for each $i < n$, $\text{Term}((z)_i)$ and $x =$

$$\#P_j^n(\# \frown \text{flatten}(z) \frown \#)^\#.$$
2. There are $z_1, z_2 < x$ such that $\text{Term}(z_1)$, $\text{Term}(z_2)$, and $x =$

$$\#=(\# \frown z_1 \frown \#, \# \frown z_2 \frown \#)^\#.$$
3. $x = \# \perp \#.$
4. $x = \# \top \#.$ □

Proposição art.8. *The relation $\text{Frm}(x)$ which holds iff x is the Gödel number of a fórmula is primitive recursive.* inc:art:frm:
prop:frm-primrec

Prova. A sequence of symbols s is a fórmula iff there is formation sequence $s_0, \dots, s_{k-1} = s$ of fórmula which records how s was formed from atomic fórmulas according to the formation rules. The code for each s_i (and indeed of the code of the sequence $\langle s_0, \dots, s_{k-1} \rangle$) is less than the code x of s . □

Problema art.2. Give a detailed proof of **Proposição art.8** along the lines of the first proof of **Proposição art.5**.

Proposição art.9. *The relation $\text{FreeOcc}(x, z, i)$, which holds iff the i -th symbol of the formula with Gödel number x is a free occurrence of the variable with Gödel number z , is primitive recursive.* inc:art:frm:
prop:freeocc-primrec

Prova. Exercise. □

Problema art.3. Prove **Proposição art.9**. You may make use of the fact that any substring of a fórmula which is a fórmula is a sub-fórmula of it.

Proposição art.10. *The property $\text{Sent}(x)$ which holds iff x is the Gödel number of a sentença is primitive recursive.*

Prova. A sentença is a fórmula without free occurrences of variáveis. So $\text{Sent}(x)$ holds iff

$$(\forall i < \text{len}(x)) (\forall z < x) ((\exists j < z) z = \#v_j^\# \rightarrow \neg \text{FreeOcc}(x, z, i)). \quad \square$$

-NoValue-

art.5 Substitution

inc:art:sub: sec Recall that substitution is the operation of replacing all free occurrences of a variável u in a fórmula φ by a term t , written $\varphi[t/u]$. This operation, when carried out on Gödel numbers of variáveis, fórmulas, and terms, is primitive recursive.

inc:art:sub: prop:subst-primrec **Proposição art.11.** *There is a primitive recursive function $\text{Subst}(x, y, z)$ with the property that*

$$\text{Subst}(\# \varphi^\#, \# t^\#, \# u^\#) = \# \varphi[t/u]^\#.$$

Prova. We can then define a function hSubst by primitive recursion as follows:

$$\begin{aligned} \text{hSubst}(x, y, z, 0) &= A \\ \text{hSubst}(x, y, z, i + 1) &= \begin{cases} \text{hSubst}(x, y, z, i) \frown y & \text{if } \text{FreeOcc}(x, z, i) \\ \text{append}(\text{hSubst}(x, y, z, i), (x)_i) & \text{otherwise.} \end{cases} \end{aligned}$$

$\text{Subst}(x, y, z)$ can now be defined as $\text{hSubst}(x, y, z, \text{len}(x))$. $\square$

inc:art:sub: prop:free-for **Proposição art.12.** *The relation $\text{FreeFor}(x, y, z)$, which holds iff the term with Gödel number y is livre para the variable with Gödel number z in the formula with Gödel number x , is primitive recursive.*

Prova. Exercise. $\square$

Problema art.4. Prove **Proposição art.12**

-NoValue-

art.6 Derivações in LK

inc:art:plk: sec In order to arithmetize derivações, we must represent derivações as numbers. explanation Since derivações are trees of sequents where each inference carries also a label, a recursive representation is the most obvious approach: we represent a derivação as a tuple, the components of which are the end-sequent, the label, and the representations of the sub-derivações leading to the premises of the last inference.

Definição art.13. If Γ is a finite sequence of sentenças, $\Gamma = \langle \varphi_1, \dots, \varphi_n \rangle$, then $\# \Gamma^\# = \langle \# \varphi_1^\#, \dots, \# \varphi_n^\# \rangle$.

If $\Gamma \Rightarrow \Delta$ is a sequent, then a Gödel number of $\Gamma \Rightarrow \Delta$ is

$$\# \Gamma \Rightarrow \Delta^\# = \langle \# \Gamma^\#, \# \Delta^\# \rangle$$

If π is a derivação in **LK**, then $\# \pi^\#$ is defined as follows:

1. If π consists only of the initial sequent $\Gamma \Rightarrow \Delta$, then $\# \pi^\#$ is

$$\langle 0, \# \Gamma \Rightarrow \Delta^\# \rangle.$$

2. If π ends in an inference with one or two premises, has $\Gamma \Rightarrow \Delta$ as its conclusion, and π_1 and π_2 are the immediate subproof ending in the premise of the last inference, then $\# \pi^\#$ is

$$\langle 1, \# \pi_1^\#, \# \Gamma \Rightarrow \Delta^\#, k \rangle \text{ or } \langle 2, \# \pi_1^\#, \# \pi_2^\#, \# \Gamma \Rightarrow \Delta^\#, k \rangle,$$

respectively, where k is given by the following table according to which rule was used in the last inference:

Rule:	WL	WR	CL	CR	XL	XR
k :	1	2	3	4	5	6
Rule:	$\neg$ L	$\neg$ R	$\wedge$ L	$\wedge$ R	$\vee$ L	$\vee$ R
k :	7	8	9	10	11	12
Rule:	$\rightarrow$ L	$\rightarrow$ R	$\forall$ L	$\forall$ R	$\exists$ L	$\exists$ R
k :	13	14	15	16	17	18
Rule:	Cut	=				
k :	19	20				

Exemplo art.14. Consider the very simple **derivação**

$$\frac{\frac{\varphi \Rightarrow \varphi}{\varphi \wedge \psi \Rightarrow \varphi} \wedge L}{\Rightarrow (\varphi \wedge \psi) \rightarrow \varphi} \rightarrow R$$

The Gödel number of the initial sequent would be $p_0 = \langle 0, \# \varphi \Rightarrow \varphi^\# \rangle$. The Gödel number of the **derivação** ending in the conclusion of $\wedge L$ would be $p_1 = \langle 1, p_0, \# \varphi \wedge \psi \Rightarrow \varphi^\#, 9 \rangle$ (1 since $\wedge L$ has one premise, the Gödel number of the conclusion $\varphi \wedge \psi \Rightarrow \varphi$, and 9 is the number coding $\wedge L$). The Gödel number of the entire **derivação** then is $\langle 1, p_1, \# \Rightarrow (\varphi \wedge \psi) \rightarrow \varphi^\#, 14 \rangle$, O.s.,

$$\langle 1, \langle 1, \langle 0, \# \varphi \Rightarrow \varphi^\# \rangle, \# \varphi \wedge \psi \Rightarrow \varphi^\#, 9 \rangle, \# \Rightarrow (\varphi \wedge \psi) \rightarrow \varphi^\#, 14 \rangle.$$

explanation

Having settled on a representation of **derivações**, we must also show that we can manipulate such **derivações** primitive recursively, and express their essential properties and relations so. Some operations are simple: p.ex., given a Gödel number p of a **derivação**, $\text{EndSeq}(p) = (p)_{(p)_0+1}$ gives us the Gödel number of its end-sequent and $\text{LastRule}(p) = (p)_{(p)_0+2}$ the code of its last rule. The property $\text{Sequent}(s)$ defined by

$$\text{len}(s) = 2 \wedge (\forall i < \text{len}((s)_0) + \text{len}((s)_1)) \text{Sent}(((s)_0 \frown (s)_1)_i)$$

holds of s iff s is the Gödel number of a sequent consisting of **sentenças**. Some are much harder. We'll at least sketch how to do this. The goal is to show that

the relation “ π is a **derivação** of φ from Γ ” is a primitive recursive relation of the Gödel numbers of π and φ .

inc:art:plk: prop:followsby **Proposição art.15.** *The property $\text{Correct}(p)$ which holds iff the last inference in the **derivação** π with Gödel number p is correct, is primitive recursive.*

Prova. $\Gamma \Rightarrow \Delta$ is an initial sequent if either there is a **sentença** φ such that $\Gamma \Rightarrow \Delta$ is $\varphi \Rightarrow \varphi$, or there is a term t such that $\Gamma \Rightarrow \Delta$ is $\emptyset \Rightarrow t = t$. In terms of Gödel numbers, $\text{InitSeq}(s)$ holds iff

$$(\exists x < s) (\text{Sent}(x) \wedge s = \langle \langle x \rangle, \langle x \rangle \rangle) \vee \\ (\exists t < s) (\text{Term}(t) \wedge s = \langle 0, \langle \# = (\# \frown t \frown \#, \# \frown t \frown \#) \# \rangle \rangle).$$

We also have to show that for each rule of inference R the relation $\text{FollowsBy}_R(p)$ is primitive recursive, where $\text{FollowsBy}_R(p)$ holds iff p is the Gödel number of **derivação** π , and the end-sequent of π follows by a correct application of R from the immediate sub-**derivações** of π .

A simple case is that of the $\wedge R$ rule. If π ends in a correct $\wedge R$ inference, it looks like this:

$$\frac{\begin{array}{c} \vdots \\ \vdots \pi_1 \\ \vdots \\ \Gamma \Rightarrow \Delta, \varphi \end{array} \quad \begin{array}{c} \vdots \\ \vdots \pi_2 \\ \vdots \\ \Gamma \Rightarrow \Delta, \psi \end{array}}{\Gamma \Rightarrow \Delta, \varphi \wedge \psi} \wedge R$$

So, the last inference in the **derivação** π is a correct application of $\wedge R$ iff there are sequences of **sentenças** Γ and Δ as well as two **sentenças** φ and ψ such that the end-sequent of π_1 is $\Gamma \Rightarrow \Delta, \varphi$, the end-sequent of π_2 is $\Gamma \Rightarrow \Delta, \psi$, and the end-sequent of π is $\Gamma \Rightarrow \Delta, \varphi \wedge \psi$. We just have to translate this into Gödel numbers. If $s = \# \Gamma \Rightarrow \Delta \#$ then $(s)_0 = \# \Gamma \#$ and $(s)_1 = \# \Delta \#$. So, $\text{FollowsBy}_{\wedge R}(p)$ holds iff

$$(\exists g < p) (\exists d < p) (\exists a < p) (\exists b < p) \\ \text{EndSequent}(p) = \langle g, d \frown \langle \# (\# \frown a \frown \# \wedge \# \frown b \frown \#) \# \rangle \rangle \wedge \\ \text{EndSequent}((p)_1) = \langle g, d \frown \langle a \rangle \rangle \wedge \\ \text{EndSequent}((p)_2) = \langle g, d \frown \langle b \rangle \rangle \wedge \\ (p)_0 = 2 \wedge \text{LastRule}(p) = 10.$$

The individual lines express, respectively, “there is a sequence (Γ) with Gödel number g , there is a sequence (Δ) with Gödel number d , a **fórmula** (φ) with Gödel number a , and a **fórmula** (ψ) with Gödel number b ,” such that “the end-sequent of π is $\Gamma \Rightarrow \Delta, \varphi \wedge \psi$,” “the end-sequent of π_1 is $\Gamma \Rightarrow \Delta, \varphi$,” “the end-sequent of π_2 is $\Gamma \Rightarrow \Delta, \psi$,” and “ π has two immediate subderivations and the last inference rule is $\wedge R$ (with number 10).”

The last inference in π is a correct application of $\exists R$ iff there are sequences Γ and Δ , a **fórmula** φ , a variable x , and a term t , such that the end-sequent

of π is $\Gamma \Rightarrow \Delta, \exists x \varphi$ and the end-sequent of π_1 is $\Gamma \Rightarrow \Delta, \varphi[t/x]$. So in terms of Gödel numbers, we have $\text{FollowsBy}_{\exists R}(p)$ iff

$$\begin{aligned} & (\exists g < p) (\exists d < p) (\exists a < p) (\exists x < p) (\exists t < p) \\ & \text{EndSequent}(p) = \langle g, d \frown \langle \# \exists \# \frown x \frown a \rangle \rangle \wedge \\ & \text{EndSequent}((p)_1) = \langle g, d \frown \langle \text{Subst}(a, t, x) \rangle \rangle \wedge \\ & (p)_0 = 1 \wedge \text{LastRule}(p) = 18. \end{aligned}$$

We then define $\text{Correct}(p)$ as

$$\begin{aligned} & \text{Sequent}(\text{EndSequent}(p)) \wedge \\ & [(\text{LastRule}(p) = 1 \wedge \text{FollowsBy}_{WL}(p)) \vee \dots \vee \\ & (\text{LastRule}(p) = 20 \wedge \text{FollowsBy}_{=}(p)) \vee \\ & (p)_0 = 0 \wedge \text{InitialSeq}(\text{EndSequent}(p))] \end{aligned}$$

The first line ensures that the end-sequent of d is actually a sequent consisting of **sentenças**. The last line covers the case where p is just an initial sequent. $\square$

Problema art.5. Define the following properties as in **Proposição art.15**:

1. $\text{FollowsBy}_{\text{Cut}}(p)$,
2. $\text{FollowsBy}_{\rightarrow L}(p)$,
3. $\text{FollowsBy}_{=}(p)$,
4. $\text{FollowsBy}_{\forall R}(p)$.

For the last one, you will have to also show that you can test primitive recursively if the last inference of the **derivação** with Gödel number p satisfies the eigenvariable condition, O.s., the eigenvariable a of the $\forall R$ does not occur in the end-sequent.

Proposição art.16. *The relation $\text{Deriv}(p)$ which holds if p is the Gödel number of a correct **derivação** π , is primitive recursive.* *inc:art:plk:*
prop:deriv

Prova. A **derivação** π is correct if every one of its inferences is a correct application of a rule, O.s., if every one of its sub-**derivações** ends in a correct inference. So, $\text{Deriv}(d)$ iff

$$(\forall i < \text{len}(\text{SubtreeSeq}(p))) \text{Correct}((\text{SubtreeSeq}(p))_i). \quad \square$$

Proposição art.17. *Suppose Γ is a primitive recursive set of **sentenças**. Then the relation $\text{Prf}_\Gamma(x, y)$ expressing “ x is the code of a **derivação** π of $\Gamma_0 \Rightarrow \varphi$ for some finite $\Gamma_0 \subseteq \Gamma$ and y is the Gödel number of φ ” is primitive recursive.*

Prova. Suppose “ $y \in \Gamma$ ” is given by the primitive recursive predicate $R_\Gamma(y)$. We have to show that $\text{Prf}_\Gamma(x, y)$ which holds iff y is the Gödel number of a sentence φ and x is the code of an **LK-derivacão** with end-sequent $\Gamma_0 \Rightarrow \varphi$ is primitive recursive.

By the previous proposition, the property $\text{Deriv}(x)$ which holds iff x is the code of a correct **derivacão** π in **LK** is primitive recursive. If x is such a code, then $\text{EndSequent}(x)$ is the code of the end-sequent of π , and so $(\text{EndSequent}(x))_0$ is the code of the left side of the end sequent and $(\text{EndSequent}(x))_1$ the right side. So we can express “the right side of the end-sequent of π is φ ” as $\text{len}((\text{EndSequent}(x))_1) = 1 \wedge ((\text{EndSequent}(x))_1)_0 = x$. The left side of the end-sequent of π is of course automatically finite, we just have to express that every sentence in it is in Γ . Thus we can define $\text{Prf}_\Gamma(x, y)$ by

$$\begin{aligned} \text{Prf}_\Gamma(x, y) \Leftrightarrow & \text{Deriv}(x) \wedge \\ & (\forall i < \text{len}((\text{EndSequent}(x))_0)) R_\Gamma(((\text{EndSequent}(x))_0)_i) \wedge \\ & \text{len}((\text{EndSequent}(x))_1) = 1 \wedge ((\text{EndSequent}(x))_1)_0 = y. \quad \square \end{aligned}$$

-NoValue-

art.7 Derivações in Natural Deduction

inc:art:pnd:
sec In order to arithmetize **derivacões**, we must represent **derivacões** as numbers. explanation Since **derivacões** are trees of **fórmulas** where each inference carries one or two labels, a recursive representation is the most obvious approach: we represent a **derivacão** as a tuple, the components of which are the number of immediate sub-**derivacões** leading to the premises of the last inference, the representations of these sub-**derivacões**, and the end-**fórmula**, the discharge label of the last inference, and a number indicating the type of the last inference.

Definição art.18. If δ is a **derivacão** in natural deduction, then $\# \delta^\#$ is defined inductively as follows:

1. If δ consists only of the assumption φ , then $\# \delta^\#$ is $\langle 0, \# \varphi^\#, n \rangle$. The number n is 0 if it is an **não descarregada** assumption, and the numerical label otherwise.
2. If δ ends in an inference with zero, one, two, or three premises, then $\# \delta^\#$ is

$$\begin{aligned} & \langle 0, \# \varphi^\#, n, k \rangle, \\ & \langle 1, \# \delta_1^\#, \# \varphi^\#, n, k \rangle, \\ & \langle 2, \# \delta_1^\#, \# \delta_2^\#, \# \varphi^\#, n, k \rangle, \text{ or} \\ & \langle 3, \# \delta_1^\#, \# \delta_2^\#, \# \delta_3^\#, \# \varphi^\#, n, k \rangle, \end{aligned}$$

respectively. Here $\delta_1, \delta_2, \delta_3$ are the sub-**derivacões** ending in the premise(s) of the last inference in δ , φ is the conclusion of the last inference

in δ , n is the discharge label of the last inference (0 if the inference does not discharge any assumptions), and k is given by the following table according to which rule was used in the last inference.

Rule:	$\wedge$ Intro	$\wedge$ Elim	$\vee$ Intro	$\vee$ Elim
k :	1	2	3	4
Rule:	$\rightarrow$ Intro	$\rightarrow$ Elim	$\neg$ Intro	$\neg$ Elim
k :	5	6	7	8
Rule:	$\perp_I$	$\perp_C$	$\forall$ Intro	$\forall$ Elim
k :	9	10	11	12
Rule:	$\exists$ Intro	$\exists$ Elim	$=$ Intro	$=$ Elim
k :	13	14	15	16

Exemplo art.19. Consider the very simple **derivação**

$$\frac{\frac{[\varphi \wedge \psi]^1}{\varphi} \wedge \text{Elim}}{(\varphi \wedge \psi) \rightarrow \varphi} \rightarrow \text{Intro}$$

The Gödel number of the assumption would be $d_0 = \langle 0, \# \varphi \wedge \psi^\#, 1 \rangle$. The Gödel number of the **derivação** ending in the conclusion of $\wedge$ Elim would be $d_1 = \langle 1, d_0, \# \varphi^\#, 0, 2 \rangle$ (1 since $\wedge$ Elim has one premise, the Gödel number of conclusion φ , 0 because no assumption is discharged, and 2 is the number coding $\wedge$ Elim). The Gödel number of the entire **derivação** then is $\langle 1, d_1, \#((\varphi \wedge \psi) \rightarrow \varphi)^\#, 1, 5 \rangle$, O.s.,

$$\langle 1, \langle 1, \langle 0, \#(\varphi \wedge \psi)^\#, 1 \rangle, \# \varphi^\#, 0, 2 \rangle, \#((\varphi \wedge \psi) \rightarrow \varphi)^\#, 1, 5 \rangle.$$

explanation

Having settled on a representation of **derivações**, we must also show that we can manipulate Gödel numbers of such **derivações** primitive recursively, and express their essential properties and relations. Some operations are simple: p.ex., given a Gödel number d of a **derivação**, $\text{EndFmla}(d) = (d)_{(d)_0+1}$ gives us the Gödel number of its end-**fórmula**, $\text{DischargeLabel}(d) = (d)_{(d)_0+2}$ gives us the discharge label and $\text{LastRule}(d) = (d)_{(d)_0+3}$ the number indicating the type of the last inference. Some are much harder. We'll at least sketch how to do this. The goal is to show that the relation “ δ is a **derivação** of φ from Γ ” is a primitive recursive relation of the Gödel numbers of δ and φ .

Proposição art.20. *The following relations are primitive recursive:*

1. φ occurs as an assumption in δ with label n .
2. All assumptions in δ with label n are of the form φ (O.s., we can **descarrega** the assumption φ using label n in δ).

Prova. We have to show that the corresponding relations between Gödel numbers of **fórmulas** and Gödel numbers of **derivações** are primitive recursive.

1. We want to show that $\text{Assum}(x, d, n)$, which holds if x is the Gödel number of an assumption of the *derivação* with Gödel number d labelled n , is primitive recursive. This is the case if the *derivação* with Gödel number $\langle 0, x, n \rangle$ is a sub-*derivação* of d . Note that the way we code *derivações* is a special case of the coding of trees introduced in ??, so the primitive recursive function $\text{SubtreeSeq}(d)$ gives a sequence of Gödel numbers of all sub-*derivações* of d (of length at most d). So we can define

$$\text{Assum}(x, d, n) \Leftrightarrow (\exists i < d) (\text{SubtreeSeq}(d))_i = \langle 0, x, n \rangle.$$

2. We want to show that $\text{Discharge}(x, d, n)$, which holds if all assumptions with label n in the *derivação* with Gödel number d all are the *fórmula* with Gödel number x . But this relation holds iff $(\forall y < d) (\text{Assum}(y, d, n) \rightarrow y = x)$. $\square$

inc:art:pnd;
prop:followsby

Proposição art.21. *The property $\text{Correct}(d)$ which holds iff the last inference in the *derivação* δ with Gödel number d is correct, is primitive recursive.*

Prova. Here we have to show that for each rule of inference R the relation $\text{FollowsBy}_R(d)$ is primitive recursive, where $\text{FollowsBy}_R(d)$ holds iff d is the Gödel number of *derivação* δ , and the end-*fórmula* of δ follows by a correct application of R from the immediate sub-*derivações* of δ .

A simple case is that of the $\wedge\text{Intro}$ rule. If δ ends in a correct $\wedge\text{Intro}$ inference, it looks like this:

$$\frac{\begin{array}{c} \vdots \\ \delta_1 \\ \vdots \\ \varphi \end{array} \quad \begin{array}{c} \vdots \\ \delta_2 \\ \vdots \\ \psi \end{array}}{\varphi \wedge \psi} \wedge\text{Intro}$$

Then the Gödel number d of δ is $\langle 2, d_1, d_2, \#(\varphi \wedge \psi)^\#, 0, k \rangle$ where $\text{EndFmla}(d_1) = \# \varphi^\#, \text{EndFmla}(d_2) = \# \psi^\#, n = 0$, and $k = 1$. So we can define $\text{FollowsBy}_{\wedge\text{Intro}}(d)$ as

$$\begin{aligned} (d)_0 &= 2 \wedge \text{DischargeLabel}(d) = 0 \wedge \text{LastRule}(d) = 1 \wedge \\ &\text{EndFmla}(d) = \\ &\#(\# \frown \text{EndFmla}((d)_1) \frown \# \wedge^\# \frown \text{EndFmla}((d)_2) \frown \#)^\#. \end{aligned}$$

Another simple example is the $=\text{Intro}$ rule. This has no premises, so $(d)_0 = 0$, like assumptions. It also has no discharge label, O.s., $n = 0$. However, φ must be of the form $t = t$, for a closed term t . Here, a primitive recursive definition is

$$\begin{aligned} (d)_0 &= 0 \wedge \text{DischargeLabel}(d) = 0 \wedge \\ &(\exists t < d) (\text{ClTerm}(t) \wedge \text{EndFmla}(d) = \\ &\quad \#(=\frown t \frown \# \frown \# \frown t \frown \#)^\#). \end{aligned}$$

For a more complicated example, $\text{FollowsBy}_{\rightarrow\text{Intro}}(d)$ holds iff the end-**fórmula** of δ is of the form $(\varphi \rightarrow \psi)$, where the end-**fórmula** of δ_1 is ψ , and any assumption in δ labelled n is of the form φ . We can express this primitive recursively by

$$\begin{aligned} (d)_0 &= 1 \wedge \\ &(\exists a < d) (\text{Discharge}(a, (d)_1, \text{DischargeLabel}(d)) \wedge \\ &\text{EndFmla}(d) = (\#(\# \frown a \frown \# \rightarrow \# \frown \text{EndFmla}((d)_1) \frown \#) \#)) \end{aligned}$$

(Think of a as the Gödel number of φ).

For another example, consider $\exists\text{Intro}$. Here, the last inference in δ is correct iff there is a **fórmula** φ , a closed term t and a **variável** x such that $\varphi[t/x]$ is the end-**fórmula** of the **derivacão** δ_1 and $\exists x \varphi$ is the conclusion of the last inference. So, $\text{FollowsBy}_{\exists\text{Intro}}(d)$ holds iff

$$\begin{aligned} (d)_0 &= 1 \wedge \text{DischargeLabel}(d) = 0 \wedge \\ &(\exists a < d) (\exists x < d) (\exists t < d) (\text{ClTerm}(t) \wedge \text{Var}(x) \wedge \\ &\text{Subst}(a, t, x) = \text{EndFmla}((d)_1) \wedge \text{EndFmla}(d) = (\# \exists \# \frown x \frown a)). \end{aligned}$$

We then define $\text{Correct}(d)$ as

$$\begin{aligned} &\text{Sent}(\text{EndFmla}(d)) \wedge \\ &(\text{LastRule}(d) = 1 \wedge \text{FollowsBy}_{\wedge\text{Intro}}(d)) \vee \dots \vee \\ &(\text{LastRule}(d) = 16 \wedge \text{FollowsBy}_{=\text{Elim}}(d)) \vee \\ &(\exists n < d) (\exists x < d) (d = \langle 0, x, n \rangle). \end{aligned}$$

The first line ensures that the end-**fórmula** of d is a sentence. The last line covers the case where d is just an assumption. $\square$

Problema art.6. Define the following properties as in **Proposição art.21**:

1. $\text{FollowsBy}_{\rightarrow\text{Elim}}(d)$,
2. $\text{FollowsBy}_{=\text{Elim}}(d)$,
3. $\text{FollowsBy}_{\vee\text{Elim}}(d)$,
4. $\text{FollowsBy}_{\vee\text{Intro}}(d)$.

For the last one, you will have to also show that you can test primitive recursively if the last inference of the **derivacão** with Gödel number d satisfies the eigenvariable condition, O.s., the eigenvariable a of the $\forall\text{Intro}$ inference occurs neither in the end-**fórmula** of d nor in an open assumption of d . You may use the primitive recursive predicate OpenAssum from **Proposição art.23** for this.

Proposição art.22. *The relation $\text{Deriv}(d)$ which holds if d is the Gödel number of a correct **derivacão** δ , is primitive recursive.*

*inc:art:pnd:
prop:deriv*

Prova. A **derivaco** δ is correct if every one of its inferences is a correct application of a rule, O.s., if every one of its sub-**derivaces** ends in a correct inference. So, $\text{Deriv}(d)$ iff

$$(\forall i < \text{len}(\text{SubtreeSeq}(d))) \text{Correct}((\text{SubtreeSeq}(d))_i) \quad \square$$

*inc:art:pnd:
prop:openassum*

Proposio art.23. *The relation $\text{OpenAssum}(z, d)$ that holds if z is the Gdel number of a **no descarregada** assumption φ of the **derivaco** δ with Gdel number d , is primitive recursive.*

Prova. An occurrence of an assumption is **descarregada** if it occurs with label n in a sub-**derivaco** of δ that ends in a rule with discharge label n . So φ is a **no descarregada** assumption of δ if at least one of its occurrences is not **descarregada** in δ . We must be careful: δ may contain both **descarregada** and **no descarregada** occurrences of φ .

Consider a sequence $\delta_0, \dots, \delta_k$ where $\delta_0 = \delta$, δ_k is the assumption $[\varphi]^n$ (for some n), and δ_{i+1} is an immediate sub-**derivaco** of δ_i . If such a sequence exists in which no δ_i ends in an inference with discharge label n , then φ is a **no descarregada** assumption of δ .

The primitive recursive function $\text{SubtreeSeq}(d)$ provides us with a sequence of Gdel numbers of all sub-**derivaces** of δ . Any sequence of Gdel numbers of sub-**derivaces** of δ is a subsequence of it. Being a subsequence of is a primitive recursive relation: $\text{Subseq}(s, s')$ holds iff $(\forall i < \text{len}(s)) \exists j < \text{len}(s') (s)_i = (s')_j$. Being an immediate sub-**derivaco** is as well: $\text{Subderiv}(d, d')$ iff $(\exists j < (d')_0) d = (d')_j$. So we can define $\text{OpenAssum}(z, d)$ by

$$\begin{aligned} &(\exists s < \text{SubtreeSeq}(d)) (\text{Subseq}(s, \text{SubtreeSeq}(d)) \wedge (s)_0 = d \wedge \\ &(\exists n < d) ((s)_{\text{len}(s)-1} = \langle 0, z, n \rangle \wedge \\ &(\forall i < (\text{len}(s) - 1)) (\text{Subderiv}((s)_{i+1}, (s)_i) \wedge \\ &\text{DischargeLabel}((s)_i) \neq n)). \quad \square \end{aligned}$$

*inc:art:pnd:
prop:prf-prim-rec*

Proposio art.24. *Suppose Γ is a primitive recursive set of **sentenas**. Then the relation $\text{Prf}_\Gamma(x, y)$ expressing “ x is the code of a **derivaco** δ of φ from **no descarregada** assumptions in Γ and y is the Gdel number of φ ” is primitive recursive.*

Prova. Suppose “ $y \in \Gamma$ ” is given by the primitive recursive predicate $R_\Gamma(y)$. We have to show that $\text{Prf}_\Gamma(x, y)$ which holds iff y is the Gdel number of a sentence φ and x is the code of a natural deduction **derivaco** with end **frmula** φ and all **no descarregada** assumptions in Γ is primitive recursive.

By **Proposio art.22**, the property $\text{Deriv}(x)$ which holds iff x is the Gdel number of a correct **derivaco** δ in natural deduction is primitive recursive. Thus we can define $\text{Prf}_\Gamma(x, y)$ by

$$\begin{aligned} \text{Prf}_\Gamma(x, y) &\Leftrightarrow \text{Deriv}(x) \wedge \text{EndFmla}(x) = y \wedge \\ &(\forall z < x) (\text{OpenAssum}(z, x) \rightarrow R_\Gamma(z)). \quad \square \end{aligned}$$

-NoValue-

art.8 Axiomatic Derivações

explanation In order to arithmetize axiomatic **derivações**, we must represent **derivações** inc:art:pax:sec as numbers. Since **derivações** are simply sequences of **fórmulas**, the obvious approach is to code every **derivação** as the code of the sequence of codes of **fórmulas** in it.

Definição art.25. If δ is an axiomatic **derivação** consisting of **fórmulas** $\varphi_1, \dots, \varphi_n$, then $\# \delta \#$ is

$$\langle \# \varphi_1 \#, \dots, \# \varphi_n \# \rangle.$$

Exemplo art.26. Consider the very simple **derivação**:

1. $\psi \rightarrow (\psi \vee \varphi)$
2. $(\psi \rightarrow (\psi \vee \varphi)) \rightarrow (\varphi \rightarrow (\psi \rightarrow (\psi \vee \varphi)))$
3. $\varphi \rightarrow (\psi \rightarrow (\psi \vee \varphi))$

The Gödel number of this **derivação** would be

$$\begin{aligned} & \langle \# \psi \rightarrow (\psi \vee \varphi) \#, \\ & \# (\psi \rightarrow (\psi \vee \varphi)) \rightarrow (\varphi \rightarrow (\psi \rightarrow (\psi \vee \varphi))) \#, \\ & \# \varphi \rightarrow (\psi \rightarrow (\psi \vee \varphi)) \# \rangle. \end{aligned}$$

explanation Having settled on a representation of **derivações**, we must also show that we can manipulate such **derivações** primitive recursively, and express their essential properties and relations so. Some operations are simple: p.ex., given a Gödel number d of a **derivação**, $(d)_{\text{len}(d)-1}$ gives us the Gödel number of its end-**fórmula**. Some are much harder. We'll at least sketch how to do this. The goal is to show that the relation “ δ is a **derivação** of φ from I ” is primitive recursive in the Gödel numbers of δ and φ .

Proposição art.27. *The following relations are primitive recursive:*

inc:art:pax:prop:followsby

1. φ is an axiom.
2. The i -th line in δ is justified by modus ponens
3. The i -th line in δ is justified by QR.
4. δ is a correct **derivação**.

Prova. We have to show that the corresponding relations between Gödel numbers of **fórmulas** and Gödel numbers of **derivações** are primitive recursive.

1. We have a given list of axiom schemas, and φ is an axiom if it is of the form given by one of these schemas. Since the list of schemas is finite, it suffices to show that we can test primitive recursively, for each axiom schema, if φ is of that form. For instance, consider the axiom schema

$$\psi \rightarrow (\chi \rightarrow \psi).$$

φ is an instance of this axiom schema if there are **fórmulas** ψ and χ such that we obtain φ when we concatenate ‘(’ with ψ with ‘ $\rightarrow$ ’ with ‘(’ with χ with ‘ $\rightarrow$ ’ with ψ and with ‘)’’. We can test the corresponding property of the Gödel number n of φ , since concatenation of sequences is primitive recursive and the Gödel numbers of ψ and χ must be smaller than the Gödel number of φ , since when the relation holds, both ψ and χ are sub-**fórmulas** of φ . Hence, we can define:

$$\text{IsAx}_{\psi \rightarrow (\chi \rightarrow \psi)}(n) \Leftrightarrow (\exists b < n) (\exists c < n) (\text{Sent}(b) \wedge \text{Sent}(c) \wedge n = \#(\# \frown b \frown \# \rightarrow \# \frown \#(\# \frown c \frown \# \rightarrow \# \frown b \frown \#))\#).$$

If we have such a definition for each axiom schema, their disjunction defines the property $\text{IsAx}(n)$, “ n is the Gödel number of an axiom.”

2. The i -th line in δ is justified by modus ponens iff there are lines j and $k < i$ where the **sentença** on line j is some formula φ , the sentence on line k is $\varphi \rightarrow \psi$, and the sentence on line i is ψ .

$$\text{MP}(d, i) \Leftrightarrow (\exists j < i) (\exists k < i) (d)_k = \#(\# \frown (d)_j \frown \# \rightarrow \# \frown (d)_i \frown \#)\#$$

Since bounded quantification, concatenation, and $=$ are primitive recursive, this defines a primitive recursive relation.

3. A line in δ is justified by QR if it is of the form $\psi \rightarrow \forall x \varphi(x)$, a preceding line is $\psi \rightarrow \varphi(c)$ for some **símbolo de constante** c , and c does not occur in ψ . This is the case iff

- a) there is a **sentença** ψ and
- b) a **fórmula** $\varphi(x)$ with a single variable x free so that
- c) line i contains $\psi \rightarrow \forall x \varphi(x)$
- d) some line $j < i$ contains $\psi \rightarrow \varphi[c/x]$ for a constant c
- e) which does not occur in ψ .

All of these can be tested primitive recursively, since the Gödel numbers of ψ , $\varphi(x)$, and x are less than the Gödel number of the formula on line i ,

and that of a less than the Gödel number of the formula on line j :

$$\begin{aligned} \text{QR}_1(d, i) \Leftrightarrow & (\exists b < (d)_i) (\exists x < (d)_i) (\exists a < (d)_i) (\exists c < (d)_j) (\\ & \text{Var}(x) \wedge \text{Const}(c) \wedge \\ & (d)_i = \#(\# \frown b \frown \# \rightarrow \# \frown \# \forall \# \frown x \frown a \frown \#)^\# \wedge \\ & (d)_j = \#(\# \frown b \frown \# \rightarrow \# \frown \text{Subst}(a, c, x) \frown \#)^\# \wedge \\ & \text{Sent}(b) \wedge \text{Sent}(\text{Subst}(a, c, x)) \wedge (\forall k < \text{len}(b)) (b)_k \neq (c)_0) \end{aligned}$$

Here we assume that c and x are the Gödel numbers of the variable and constant considered as terms (O.s., not their symbol codes). We test that x is the only free variable of $\varphi(x)$ by testing if $\varphi(x)[c/x]$ is a *sentença*, and ensure that c does not occur in ψ by requiring that every symbol of ψ is different from c .

We leave the other version of QR as an exercise.

4. d is the Gödel number of a correct *derivação* iff every line in it is an axiom, or justified by modus ponens or QR. Hence:

$$\text{Deriv}(d) \Leftrightarrow (\forall i < \text{len}(d)) (\text{IsAx}((d)_i) \vee \text{MP}(d, i) \vee \text{QR}(d, i)) \quad \square$$

Problema art.7. Define the following relations as in *Proposição art.27*:

1. $\text{IsAx}_{\varphi \rightarrow (\psi \rightarrow (\varphi \wedge \psi))}(n)$,
2. $\text{IsAx}_{\forall x \varphi(x) \rightarrow \varphi(t)}(n)$,
3. $\text{QR}_2(d, i)$ (for the other version of QR).

Proposição art.28. Suppose Γ is a primitive recursive set of *sentenças*. Then the relation $\text{Prf}_\Gamma(x, y)$ expressing “ x is the code of a *derivação* δ of φ from Γ and y is the Gödel number of φ ” is primitive recursive.

Prova. Suppose “ $y \in \Gamma$ ” is given by the primitive recursive predicate $R_\Gamma(y)$. We have to show that the relation $\text{Prf}_\Gamma(x, y)$ is primitive recursive, where $\text{Prf}_\Gamma(x, y)$ holds iff y is the Gödel number of a *sentença* φ and x is the code of a *derivação* of φ from Γ .

By the previous proposition, the property $\text{Deriv}(x)$ which holds iff x is the code of a correct *derivação* δ is primitive recursive. However, that definition did not take into account the set Γ as an additional way to justify lines in the *derivação*. Our primitive recursive test of whether a line is justified by QR also left out of consideration the requirement that the constant c is not allowed to occur in Γ . It is possible to amend our definition so that it takes into account Γ directly, but it is easier to use Deriv and the deduction theorem. $\Gamma \vdash \varphi$ iff there is some finite list of *sentenças* $\psi_1, \dots, \psi_n \in \Gamma$ such that $\{\psi_1, \dots, \psi_n\} \vdash \varphi$. And by the deduction theorem, this is the case if $\vdash (\psi_1 \rightarrow (\psi_2 \rightarrow \dots (\psi_n \rightarrow \varphi) \dots))$. Whether a *sentença* with Gödel number z is of this form can be tested primitive

recursively. So, instead of considering x as the Gödel number of a **derivação** of the **sentença** with Gödel number y from Γ , we consider x as the Gödel number of a **derivação** of a nested conditional of the above form from $\emptyset$.

First, if we have a sequence of **sentenças**, we can primitive recursively form the conditional with all these sentences as antecedents and given **sentença** as consequent:

$$\begin{aligned} \text{hCond}(s, y, 0) &= y \\ \text{hCond}(s, y, n + 1) &= \#(\# \frown (s)_n \frown \# \rightarrow \# \frown \text{Cond}(s, y, n) \frown \#) \# \\ \text{Cond}(s, y) &= \text{hCond}(s, y, \text{len}(s)) \end{aligned}$$

So we can define $\text{Prf}_\Gamma(x, y)$ by

$$\begin{aligned} \text{Prf}_\Gamma(x, y) \Leftrightarrow (\exists s < \text{sequenceBound}(x, x)) \ (\\ & (x)_{\text{len}(x)-1} = \text{Cond}(s, y) \wedge \\ & (\forall i < \text{len}(s)) \ (s)_i \in \Gamma \wedge \\ & \text{Deriv}(x)). \end{aligned}$$

The bound on s is given by considering that each $(s)_i$ is the Gödel number of a sub-**fórmula** of the last line of the **derivação**, O.s., is less than $(x)_{\text{len}(x)-1}$. The number of antecedents $\psi \in \Gamma$, O.s., the length of s , is less than the length of the last line of x . $\square$

Créditos das Imagens

Referências Bibliográficas