The Logic of Unprovability

Mark Kamsma, University of Utrecht
Supervised by Jaap van Oosten

2016-06-20

Abstract

The first incompleteness theorem of Kurt Gödel states that a theory in which we can develop most of modern arithmetic is incomplete. We will take a look at such a theory: Peano Arithmetic (PA). We will develop the tools to formulate a sentence that essentially asserts in PA that it is not provable. Then we use this sentence to prove both Gödel's first and second incompleteness theorems, where the second states that PA cannot prove its own consistency. To prove Gödel's first incompleteness theorem also for consistent extensions of PA we will use Rosser sentences. These are sentences that are equivalent in PA to the assertion that a disproof of them occurs before any proof of them. After developing the necessary technical tools, we prove that Rosser sentences need not to be all provably equivalent but there are constructions where they are.

1 Introduction

In this paper we provide the theory that is needed to prove the famous incompleteness theorems of Kurt Gödel. Then we prove these theorems. After that we are ready to take a look at so called Rosser sentences.

In more detail: in section 2 we will take a look at recursive functions and some of their important properties. Then in section 3 we will talk about Peano Arithmetic, a powerful theory in which one can develop most of modern arithmetic. We will show that we can represent the recursive functions in Peano Arithmetic, which is why they are such an important tool. After that we are finally ready for Gödel's incompleteness theorems in section 4. The first of these theorems states that Peano Arithmetic is incomplete. The second theorem states that Peano Arithmetic cannot prove its own consistency. These theorems are based on the construction of a sentence that essentially says "I am not provable". We will then shift focus to Rosser sentences, these are sentences that basically say "a disproof of me occurs before any proof of me". In section 5 we will take a closer look at these Rosser sentences and how they can also be used to prove Gödel's first incompleteness theorem for Peano Arithmetic and also for consistent extensions of Peano Arithmetic. Then in section 6 we will develop some technical tools that allow us to reason about Rosser sentences. These tools are then used in section 7 where we prove that not all Rosser sentences need to be provably equivalent, but also that there are constructions where all Rosser sentences are provably equivalent.

This paper is heavily based on [2] and [1]. Sections 2, 3 and 4 are based on chapters 3, 4 and 5 in [1]. Unless noted otherwise, the definitions and results in those sections are from [1], possibly reformulated to match the style of this paper. Then section 5 functions as a bridge between the content of [1] and [2]: the main result in that section is from [1], but its definitions ready us for the next sections. Sections 6 and 7 are completely based on [2] and a bit of [3].

We will also need to agree on a convention for $\mathbb{N}$. In this paper we will use the convention that $0 \in \mathbb{N}$.

2 Recursive functions

2.1 Primitive recursive functions

Before diving into primitive recursive functions we will first agree on some notation. When we talk about the function $\frac{x}{y}$, we have not specified whether we mean $(x, y) \mapsto \frac{x}{y}$ or $(y, x) \mapsto \frac{x}{y}$ or even $(x, y, z) \mapsto \frac{x}{y}$. These are all different functions. That is what the $\lambda - notation$ is for. If $\vec{x}$ is a sequence of variables $x_1, \ldots, x_n$, then $\lambda\vec{x}.F$ denotes the function $(x_1, \ldots, x_n) \mapsto F(\vec{x})$.

Example 2.1. The functions $(x, y) \mapsto \frac{x}{y}$, $(y, x) \mapsto \frac{x}{y}$ and $(x, y, z) \mapsto \frac{x}{y}$ would be written as $\lambda xy.\frac{x}{y}$, $\lambda yx.\frac{x}{y}$ and $\lambda xyz.\frac{x}{y}$ respectively.

Now that is out of the way we can define primitive recursive functions. These functions will become very important in the next sections. In section 3 we will show that we can represent primitive recursive functions using logical formulas.

Definition 2.1. The class of *primitive recursive functions* is a subclass of all functions $\mathbb{N}^k \to \mathbb{N}$ (for any $k \in \mathbb{N}$). It contains the *zero function*, *successor function* and all *projections*, and is closed under *composition* and *primitive recursion*. To be more precise, it is generated by the following clauses:

1. the *zero function* $Z = \lambda x.0$ is primitive recursive;
2. the *successor function* $S = \lambda x.x + 1$ is primitive recursive;
3. the *projections* $\Pi_i^k = \lambda x_1, \ldots, x_k.x_i$ (for $1 \leq i \leq k$) are primitive recursive;
4. given primitive recursive functions $G_1, \ldots, G_l : \mathbb{N}^k \to \mathbb{N}$ and $H : \mathbb{N}^l \to \mathbb{N}$, then $\lambda\vec{x}.H(G_1(\vec{x}), \ldots, G_l(\vec{x}))$ is said to be defined by *composition* and is primitive recursive;
5. given primitive recursive functions $G : \mathbb{N}^k \to \mathbb{N}$ and $H : \mathbb{N}^{k+2} \to \mathbb{N}$, then we can define $F : \mathbb{N}^{k+1} \to \mathbb{N}$ by *primitive recursion* as follows:

$$F(0, \vec{x}) = G(\vec{x})$$
$$F(y + 1, \vec{x}) = H(y, F(y, \vec{x}), \vec{x})$$

and F is primitive recursive.

Note that in clause 5 it is possible that $k = 0$. In that case G is just a constant and F is defined by

$$F(0) = G$$
$$F(y + 1) = H(y, F(y))$$

which is also primitive recursive.

Definition 2.2. A *k-ary relation* is a subset of $\mathbb{N}^k$. One can define such a relation using its *characteristic function*. For a relation A, that is a function $\chi_A : \mathbb{N}^k \to \mathbb{N}$ such that

$$\chi_A(\vec{x}) = \begin{cases} 0 & \vec{x} \in A \\ 1 & \text{else} \end{cases}$$

We speak of a *primitive recursive relation* if the characteristic function of the relation is primitive recursive.

To get a grip on what kind of functions are primitive recursive we will consider a few examples. These are based on the examples in [1] on pages 34-35, but they are reformulated to fit the style of this paper.

Example 2.2. We will define the function $\lambda xy.x + y$ using primitive recursion to show that it is primitive recursive. We take G to be $\Pi^1_1(y)$, so we have $F(0, y) = G(y) = \Pi^1_1(y) = y = 0 + y$. We can define H by composition as $\lambda abc.S(\Pi^3_2(a, b, c))$. Then we have $F(x + 1, y) = H(x, F(x, y), y) = H(x, x + y, y) = S(\Pi^3_2(x, x + y, y)) = S(x + y) = (x + 1) + y$, and so F is defined by primitive recursion.

Example 2.3. Using the result from example 2.2 we can define $\lambda xy.xy$ by primitive recursion, and thus show that it is primitive recursive. Take G to be the zero function. Then $F(0, y) = G(y) = 0 = 0y$. We will compose H of projections and the function $\lambda xy.x+y$ from example 2.2 as follows: $H(a, b, c) = \Pi^3_2(a, b, c)+\Pi^3_3(a, b, c)$. Then we have $F(x+1, y) = H(x, F(x, y), y) = H(x, xy, y) = xy+y = (x + 1)y$, and so F is defined by primitive recursion.

We will end this section with a useful proposition which allows us to define primitive recursive functions by cases. Event hough this proposition is the same as proposition 3.3 in [1], its proof is heavily reliant on exercise 26 in [1].

Proposition 2.1. *If G_1, G_2 and H are primitive recursive functions $\mathbb{N}^k \to \mathbb{N}$, and F is defined by*

$$F(\vec{x}) = \begin{cases} G_1(\vec{x}) & H(\vec{x}) = 0 \\ G_2(\vec{x}) & \textit{else} \end{cases}$$

then F is also primitive recursive.

Proof. Define sg, the *sign function*, and its complement $\overline{sg}$ as follows

$$sg(x) = \begin{cases} 1 & x > 0 \\ 0 & \text{else} \end{cases} \qquad \overline{sg}(x) = \begin{cases} 0 & x > 0 \\ 1 & \text{else} \end{cases}$$

Note that the "else" case in both definitions is equivalent to $x = 0$ since we work in $\mathbb{N}$. So we can define sg using primitive recursion: set $sg(0) = 0$ and $sg(x + 1) = 1$. In the same way we can define $\overline{sg}$. Thus both sg and $\overline{sg}$ are primitive recursive.

We can now easily see that F is primitive recursive since

$$F(\vec{x}) = \overline{sg}(H(\vec{x}))G_1(\vec{x}) + sg(H(\vec{x}))G_2(\vec{x}).$$

Note that we used the fact that $\lambda xy.x + y$ and $\lambda xy.xy$ are primitive recursive, as we have seen in examples 2.2 and 2.3. ■

2.2 Total recursive functions

The primitive recursive functions are defined for every possible input. As will become clear, we will also want to be able to talk about functions that are not defined for every possible input. We denote such a *partial function* F that maps some of the elements of a set X to elements of a set Y as $F : X \rightharpoonup Y$.

Later on we will be interested in so called *computable functions*. These are the partial functions $F : \mathbb{N}^k \rightharpoonup \mathbb{N}$ for which there is an algorithm so that for all inputs for which F is defined this algorithm calculates the output of F.

We can extend the class of primitive recursive functions to a class of *partial recursive functions*. It is a well known result that the classes of partial recursive functions and the computable functions are the same.

Before defining this class of functions we will need to agree on one more piece of notation: the *Kleene equality* $\simeq$. For two partial functions $F : \mathbb{N}^k \rightharpoonup \mathbb{N}$ and $G : \mathbb{N}^k \rightharpoonup \mathbb{N}$ we let $F(x) \simeq G(x)$ mean that $F(x)$ is defined precisely when $G(x)$ is defined, and when they are defined we have $F(x) = G(x)$.

Definition 2.3. The class of *partial recursive functions* $\mathbb{N}^k \rightharpoonup \mathbb{N}$ (for any $k \in \mathbb{N}$) is generated by the following clauses.

1. All primitive recursive functions are partial recursive.
2. Given partial recursive functions $G_1, \ldots, G_l : \mathbb{N}^k \rightharpoonup \mathbb{N}$ and $H : \mathbb{N}^l \rightharpoonup \mathbb{N}$, then $F(\vec{x}) = \lambda\vec{x}.H(G_1(\vec{x}), \ldots, G_l(\vec{x}))$ defined by composition is partial recursive. Note that F is defined for all $\vec{x} \in \bigcap_{i=1}^{l} \mathrm{dom}(G_i)$ such that $(G_1(\vec{x}), \ldots, G_l(\vec{x})) \in \mathrm{dom}(H)$.
3. Given a partial recursive function $G : \mathbb{N}^{k+1} \rightharpoonup \mathbb{N}$, then $F : \mathbb{N}^k \rightharpoonup \mathbb{N}$ defined by $\mu y.G(\vec{x}, y) = 0$ is said to be defined by *minimization* and is partial recursive. This notation means that $F(\vec{x})$ is defined precisely when there exists a y such that $\forall i \leq y$ we have that $(\vec{x}, y) \in \mathrm{dom}(G)$ and $G(\vec{x}, y) = 0$, and $F(\vec{x})$ is then the smallest y that satisfies this property.

It is of course possible for partial recursive function to be defined for all its input values. In that case we also call it a *(total) recursive function*.

Definition 2.4. A partial recursive function $F : \mathbb{N}^k \rightharpoonup \mathbb{N}$ is called *(total) recursive* if $\mathrm{dom}(F) = \mathbb{N}^k$.

As we can speak about a primitive recursive relation, we can now also speak about a *recursive relation*. A k-ary recursive relation is a subset of $\mathbb{N}^k$ so that there is a recursive function that can determine for any $\vec{x} \in \mathbb{N}^k$ whether or not $\vec{x}$ is part of that relation.

There are also *recursively enumerable* (or *r.e.*) relations. A k-ary recursively enumerable relation is a subset of $\mathbb{N}^k$ so that there is a recursive function that enumerates all members of $\mathbb{N}^k$.

Before making this precise in definitions we point out that there are many bijections between $\mathbb{N}$ and $\mathbb{N}^k$ for any $k \in \mathbb{N}$. As we will see in subsection 4.1, some of these bijections are primitive recursive and thus recursive. So by interpreting the output of (partial) recursive functions using such a bijection we can act like they map to $\mathbb{N}^k$.

Definition 2.5. A subset $A \subseteq \mathbb{N}^k$ is called a *recursive relation* if its characteristic function χ_A is recursive.

Definition 2.6. A subset $A \subseteq \mathbb{N}^k$ is called *recursively enumerable* (or *r.e.*) if it is either empty or there is a recursive function $F : \mathbb{N} \to \mathbb{N}^k$ such that for all $\vec{a} \in A$ there is an $n \in \mathbb{N}$ with $F(n) = \vec{a}$. Also, if $\vec{a} \notin A$ then for all $n \in \mathbb{N}$ we have $F(n) \neq \vec{a}$.

The difference between a recursive relation and recursively enumerable one may not directly be clear. Every recursive relation is of course recursively enumerable. However, not every recursively enumerable relation is recursive. Suppose we are given a k-ary recursively enumerable relation A, and an arbitrary element $\vec{x} \in \mathbb{N}^k$. Then if $\vec{x}$ is part of A, we can recursively determine this. However, if $\vec{x}$ is not part of A then we can not necessarily recursively determine this.

We have slightly deviated here from the definition of recursively enumerable in [1] because it will suit us more later on. We will however prove in the following lemma that our definition and the definition from [1] are equivalent.

Lemma 2.1. *A set $A \subseteq \mathbb{N}^k$ is recursively enumerable iff there is a recursive set $B \subseteq \mathbb{N}^{k+1}$ such that $A = \{\vec{x} \in \mathbb{N}^k : \exists y((\vec{x}, y) \in B)\}$.*

Proof. The case where A or B is empty is clear. So we assume that A and B are not empty.

($\Rightarrow$) Suppose A is recursively enumerable, and let F be a recursive function that enumerates A. Now define $\chi_B : \mathbb{N}^{k+1} \to \mathbb{N}$ as follows:

$$\chi_B(\vec{x}, y) = \begin{cases} 0 & \text{if } F(y) = \vec{x} \\ 1 & \text{else} \end{cases}$$

Then clearly $B = \{(\vec{x}, y) \in \mathbb{N}^{k+1} : \chi_B(\vec{x}, y) = 0\}$ is satisfactory, and it is recursive because by definition χ_B is a recursive characteristic function for B.

($\Leftarrow$) Let χ_B be a recursive characteristic function for B. We now describe an algorithm that enumerates all elements of A. As we discussed earlier there is a partial recursive function that gives the same output as the algorithm. Because the algorithm outputs something on every input (input n gives the nth element of the enumeration), it is total recursive. We iterate through all elements $(\vec{x}, y)$ of $\mathbb{N}^{k+1}$ using a recursive bijection $\mathbb{N} \to \mathbb{N}^{k+1}$. At each step of the iteration we check if $\chi_B(\vec{x}, y)$ equals 0. If this is the case we output $\vec{x}$, otherwise we just continue.

For any $\vec{x} \in A$ we must have that $\chi(\vec{x}, y) = 0$ for some y. Since we iterate through all elements of $\mathbb{N}^{k+1}$ we must encounter $(\vec{x}, y)$ at some time, and thus $\vec{x}$ is output by the algorithm. For any $\vec{x}$ that is not an element of A there is no y such that $\chi(\vec{x}, y) = 0$, so $\vec{x}$ is never output by the algorithm. We conclude that this algorithm enumerates exactly the elements of A. ■

2.3 The recursion theorem

A useful property of the partial recursive functions is that there is a universal algorithm for them. That is, there is a partial recursive function Ψ such that given a partial recursive function $F : \mathbb{N}^k \rightharpoonup \mathbb{N}$ there is some integer e such that:

$$\Psi(k, e, \vec{x}) \backsimeq F(\vec{x}).$$

For Ψ to be truly a universal algorithm, we would have to replace $\vec{x}$ by a single integer coding $\vec{x}$. Since this can easily be done using primitive recursive functions, we will not concern us with it.

What is more important is the following notation. The integer e we described above is called the *index of* F. This allows us to write $\varphi_e^{(k)}$ for F. If $k = 1$ we may omit the superscript and just write φ_e for F.

In other words for every partial recursive function $F : \mathbb{N}^k \rightharpoonup \mathbb{N}$, there is some index e such that:

$$\varphi_e^{(k)}(\vec{x}) \backsimeq F(\vec{x}).$$

But as $\varphi_e^{(k)}(\vec{x})$ is just an abbreviation for $\Psi(k, e, \vec{x})$ we also have that $\varphi_e^{(k)}(\vec{x})$ is a partial recursive function for every e and k.

To conclude this section we will state the recursion theorem. We will do so without proof, as it is a well known result and proving it requires quite some tools that are beyond the scope of this paper. For the interested reader, a proof is provided in [1] (corollary 3.20). We will however state the primitive recursive version of the theorem (as mentioned in [1], page 52), as this makes it possible to formalize any functions we define using this theorem in the theory PA (we describe PA in the next section).

Theorem 2.1 (Kleene's recursion theorem)**.** *There is a primitive recursive function G_n such that for all $x_1, \ldots, x_n, e$:*

$$\varphi_{G_n(e)}^{(n)}(x_1, \ldots, x_n) \backsimeq \varphi_e^{(n+1)}(x_1, \ldots, x_n, G_n(e)).$$

The recursion theorem allows us to define a partial recursive function based on itself. More precise: we can define a partial recursive function based on another partial recursive function with some index e. We can then use the recursion theorem to find a value for e such that e is also the index of the partial recursive function we are defining, giving us a self-referential partial recursive function.

3 Peano arithmetic

Definition 3.1. We consider the language $\mathcal{L}_{PA}$ consisting of the constants 0 and 1, and the binary function symbols $\cdot$ and $+$. We can then define the theory PA of *Peano Arithmetic* in $\mathcal{L}_{PA}$, with the following axioms:

(A1) $\forall x \neg(x + 1 = 0)$

(A2) $\forall xy(x + 1 = y + 1 \rightarrow x = y)$

(A3) $\forall x(x + 0 = x)$

(A4) $\forall xy(x + (y + 1) = (x + y) + 1)$

(A5) $\forall x(x \cdot 0 = 0)$

(A6) $\forall xy(x \cdot (y + 1) = (x \cdot y) + x)$

(A7) $\forall \vec{x}[(\varphi(0, \vec{x}) \wedge \forall y(\varphi(y, \vec{x}) \rightarrow \varphi(y + 1, \vec{x}))) \rightarrow \forall y \varphi(y, \vec{x})]$ for every formula $\varphi(y, \vec{x})$

The first six axioms are pretty clear. The seventh item is actually a set of axioms, one for every formula $\varphi(y, \vec{x})$. Such a set of axioms is called an *axiom scheme*. In this case it states that one can use induction to prove that a formula holds. Which is why these axioms are called the *induction axioms* or the *induction scheme*.

Definition 3.2. It is clear that $\mathbb{N}$ with its usual addition and multiplication is a model of PA. We call this the *standard model* of PA and denote it by $\mathcal{N}$.

The theory of PA is actually very strong, and many properties of elementary number theory can be expressed and proved in it. The following proposition proves some basic properties of PA. More properties will be discussed in section 3.1. Properties (P1) and (P3) are proved in [1], proposition 4.1, the rest of the proof was left as an exercise (exercise 50).

Proposition 3.1. *The following basic properties hold in* PA*:*

(P1) $\text{PA} \vdash \forall xyz(x + (y + z) = (x + y) + z)$

(P2) $\text{PA} \vdash \forall xy(x + y = y + x)$

(P3) $\text{PA} \vdash \forall x(x = 0 \vee \exists y(x = y + 1))$

(P4) $\text{PA} \vdash \forall xy\exists z(x + z = y \vee x = y + z)$

(P5) $\text{PA} \vdash \forall xyz(x + z = y + z \to x = y)$

Proof. For each of these properties we will use the induction scheme to prove them. For all these proofs we reason in PA.

For (P1) we let $\varphi(z)$ be $\forall xy(x + (y + z) = (x + y) + z)$. Using (A3) we directly have $\text{PA} \vdash \varphi(0)$. Now suppose that $\varphi(z)$ holds, then using (A4) we see that $x+(y+(z+1)) = x+((y+z)+1) = (x+(y+z))+1 = ((x+y)+z)+1 = (x + y) + (z + 1)$. We conclude that $\text{PA} \vdash \forall z(\varphi(z) \to \varphi(z + 1))$ and thus $\text{PA} \vdash \forall z\varphi(z)$.

To prove (P2) we let $\varphi(x)$ be $\forall y(x + y = y + x)$. We will first prove $\varphi(0)$ and $\varphi(1)$, both using induction. So let $\psi_0(y)$ and $\psi_1(y)$ be $0 + y = y + 0$ and $1 + y = y + 1$ respectively. Clearly $\text{PA} \vdash \psi_0(0)$. Now suppose that $\psi_0(y)$ holds, then using (P1) and (A3) we find $0 + (y + 1) = (0 + y) + 1 = (y + 0) + 1 = y + 1 = (y + 1) + 0$. So $\text{PA} \vdash \forall y(\psi_0(y) \to \psi_0(y + 1))$ and therefore $\text{PA} \vdash \varphi(0)$. Since $\text{PA} \vdash \varphi(0)$ we see that $\text{PA} \vdash \psi_1(0)$ must also hold. Suppose that $\psi_1(y)$ holds, then using (P1) we get $1 + (y + 1) = (1 + y) + 1 = (y + 1) + 1$. Thus $\text{PA} \vdash \forall y(\psi_1(y) \to \psi_1(y+1))$ and we have now also proven that $\text{PA} \vdash \varphi(1)$. Now this is out of the way we only need to prove $\text{PA} \vdash \forall x(\varphi(x) \to \varphi(x+1))$. Suppose $\varphi(x)$ holds, then using (P1) multiple times and using the fact that $\text{PA} \vdash \varphi(1)$ we find $(x+1)+y = x+(1+y) = x+(y+1) = (x+y)+1 = (y+x)+1 = y+(x+1)$. This concludes our proof of (P2).

We can see (P3) very quickly, for let $\varphi(x)$ be $x = 0 \vee \exists y(x = y + 1)$. Then we see directly that $\text{PA} \vdash \varphi(0) \wedge \forall x\varphi(x + 1)$, so we conclude that $\text{PA} \vdash \forall x\varphi(x)$.

For (P4) let $\varphi(x)$ be $\forall y\exists z(x+z = y \vee x = y+z)$. Combining (P2) and (A3) we find that $\text{PA} \vdash \varphi(0)$ (namely, take $z = y$). Now suppose $\varphi(x)$ holds. Let y be any element, then there is some z such that $x + z = y$ or $x = y + z$. To prove $\varphi(x + 1)$ we reason by cases. First, suppose $z = 0$. Then we have $x = y$,

hence $x+1 = y+1$ so we find that $z = 1$ satisfies. If $z \neq 0$ there are two cases left: $x + z = y$ and $x = y + z$. In the first case we use (P3) to find a z' such that $z = z' + 1$. Then by (P1) and (P2) we have $y = x + z = x + (z' + 1) = x + (1 + z') = (x + 1) + z'$. In the last case we simply make use of (A4) once to see that $x+1 = (y+z)+1 = y+(z+1)$. This proves $\text{PA} \vdash \forall x(\varphi(x) \to \varphi(x+1))$ and so $\text{PA} \vdash \forall x(\varphi(x))$.

For (P5) we let $\varphi(z, x, y)$ be $x + z = y + z \to x = y$. Clearly $\text{PA} \vdash \varphi(0, x, y)$. Suppose that $x + z = y + z \to x = y$ holds, and suppose that $x + (z + 1) = y + (z + 1)$. Using (A4) we see that $x + (z + 1) = y + (z + 1)$ is equivalent to $(x + z) + 1 = (y + z) + 1$, thus by (A2) we have that $x + z = y + z$. Then using the induction hypothesis we conclude that $x = y$. So $\text{PA} \vdash \forall xy[\varphi(0, x, y) \wedge \forall z(\varphi(z, x, y) \to \varphi(z + 1, x, y))]$ and thus by induction $\text{PA} \vdash \forall xyz\varphi(z, x, y)$. ■

In what follows we will use proposition 3.1 without mentioning it, because the properties stated there are natural. The same holds for the axioms of PA. Although we might sometimes refer to them when things need to be a bit clearer.

It will be useful to add an ordering $<$ to PA. We can do this without adding axioms. The sentence $\exists z(x + (z + 1) = y)$ defines a discrete linear order with a least element which also satisfies the least number principle, as we will soon see. Because we will need this ordering a lot, we will introduce a new symbol for it.

Definition 3.3. The notation $x < y$ abbreviates $\exists z(x + (z + 1) = y)$. We will also use $\exists x < y\psi$ and $\forall x < y\psi$ as abbreviations for $\exists x(x < y \wedge \psi)$ and $\forall x(x < y \to \psi)$ respectively. Finally, like one would expect $x \leq y$ abbreviates $x < y \vee x = y$, and $x \neq y$ abbreviates $\neg(x = y)$.

Using this new notation we will now prove our claim about the ordering $<$ gives to PA. Eventhough this is also a proposition in [1] (proposition 4.2), its proof was left as an exercise (exercise 51).

Proposition 3.2. PA *proves that the ordering* $<$ *satisfies the following properties.*

(1) It is a discrete linear order. That is, for all x, y, z*, we have:*
$\neg(x < x)$,
$x < y \wedge y < z \to x < z$,
$x < y \vee x = y \vee y < x$,
$x < y \to (x + 1 \leq y)$.

(2) It has as least element 0. That is, for all x *we have* $x = 0 \vee 0 < x$.

(3) It satisfies the least number principle. That is, for all formulas ψ *we have* $\exists w\psi(w) \to \exists y(\psi(y) \wedge \forall x < y\neg\psi(x))$.

Proof. First we prove that $\text{PA} \vdash \forall x(x < x + 1)$. Recall that $x < x + 1$ is just an abbreviation for $\exists z(x + (z + 1) = x + 1)$. We can then let z be 0 and we are done.

Now for property (1), we will first prove that $\neg(x < x)$ for all x. Suppose that there would be an x such that $x < x$, then there would be some z such that $x + (z + 1) = x$ which is equivalent to $(z + 1) + x = 0 + x$. But this would mean that $z + 1 = 0$, which is impossible, hence $\neg(x < x)$. To prove

$x < y \wedge y < z \to x < z$, we let a and b be such that $x + (a + 1) = y$ and $y + (b + 1) = z$. Then $(x + (a + 1)) + (b + 1) = z$ thus $x + (((a + 1) + b) + 1) = z$, so $x < z$. The third item was $x < y \vee x = y \vee y < x$, we use (P4) here to see that there is some z such that either $x + z = y$ or $x = y + z$. If $z = 0$, then we have that $x = y$. Otherwise we can find a z' such that $z = z' + 1$, so we either have $x + (z' + 1) = y$ or $y + (z' + 1) = x$ thus $x < y$ or $y < x$. Now for the last item, suppose $x < y$ then there is some z such that $x + (z + 1) = y$. In case $z = 0$ we have $x + 1 = y$, otherwise there is some z' such that $z = z' + 1$. Filling this in we get $(x + 1) + (z' + 1) = y$, so $x + 1 < y$.

Property (2) is a direct consequence of (P3), for x is either 0 or we find some z such that $x = z + 1$, hence $0 + (z + 1) = x$ and so $0 < x$.

Only property (3) is now left to prove. We reason in PA and aim for a contradiction. So suppose that for a formula ψ we have a w such that $\psi(w)$, but that there is no smallest y such that $\psi(y)$. Let $\varphi(x)$ be the formula:

$$\forall y(\psi(y) \to x < y).$$

We will use induction to prove $\forall x \varphi(x)$. Since there is no smallest y such that $\psi(y)$ holds we cannot have $\psi(0)$ and thus we have $\varphi(0)$. Now suppose that $\varphi(x)$ holds, then we cannot have $\psi(x + 1)$. This can be seen as follows: since $x + 1$ cannot be the smallest element for which ψ holds, there must be a $z < x+1$ (and thus $z \leq x$) such that $\psi(z)$ holds. By the induction hypothesis on φ we must have $x < z$. So we conclude that $z \leq x$ and $x < z$ which is a contradiction. So no such z can exist and thus $\psi(x+1)$ cannot hold. By the induction hypothesis and the fact that $\psi(x + 1)$ does not hold we conclude that $\varphi(x + 1)$ holds.

Now we have proven $\forall x \varphi(x)$ by induction we see that $\varphi(w)$ must also hold. But then $\psi(w) \to w < w$, which is a contradiction with the assumption that $\psi(w)$. We conclude that there must be a smallest y such that $\psi(y)$ holds. ■

3.1 Coding sequences in PA

As mentioned earlier, PA is very strong and we can express and prove various properties of elementary number theory in it. It is exactly this strength that enables us to prove its incompleteness in section 4. But first we will use this strength to develop a tool for coding sequences in PA.

Our proofs in this section will require us to use standard arithmetical identities like the ones in proposition 3.1. So we will extend this proposition here with some more properties. We will however not prove these properties, since their proof is not very interesting. One can prove them by continuing the proof of proposition 3.1 in the same fashion.

Proposition 3.3 (proposition 3.1 continued)**.** *The following basic properties hold in* PA*:*

(P6) $\text{PA} \vdash \forall xyz((x \cdot y) \cdot z = x \cdot (y \cdot z))$

(P7) $\text{PA} \vdash \forall xy(x \cdot y = y \cdot x)$

(P8) $\text{PA} \vdash \forall xyz(x \cdot (y + z) = x \cdot y + x \cdot z)$

(P9) $\text{PA} \vdash \forall xyz(z \neq 0 \wedge x \cdot z = y \cdot z \to x = y)$

Now we have enough basic properties to prove that PA supports division with remainder. However, since the properties in proposition 3.3 are natural we will not mention it when we use them.

Theorem 3.1 (Division with remainder)**.**

$$\text{PA} \vdash \forall xy(y \neq 0 \to \exists ab(x = a \cdot y + b \wedge 0 \leq b < y))$$

Furthermore, PA *proves that* a *and* b *are unique.*

Proof. We will use induction on x here. Clearly $0 = 0 \cdot y + 0$. Assume that there are a and $0 \leq b < y$ such that $x = a \cdot y + b$. Then since $b < y$ we have that $b + 1 \leq y$ because $<$ is a discrete order. Suppose that $b + 1 < y$, then $x+1 = a \cdot y+(b+1)$. If on the other hand $b+1 = y$, then $x+1 = (a+1) \cdot y+0$. This completes our induction proof.

We now only need to prove uniqueness of a en b. Suppose there are x, y such that $x = a \cdot y + b = a' \cdot y + b'$ with $0 \leq b, b' < y$. Assume that $a < a'$. Then we have that $a + 1 \leq a'$, thus

$$x = a \cdot y + b < a \cdot y + y \leq a' \cdot y,$$

which is a contradiction. So we have $a' \leq a$, and by symmetry $a \leq a'$ so $a = a'$. It now also follows directly that $b = b'$. ■

We can use theorem 3.1 to define the *least common multiple* or *lcm* and *greatest common divisor* or *gcd*. For that we will introduce the shorthand notation $a|b$ for $\exists c(a \cdot c = b)$. Let $x, y \geq 1$, then clearly $x|x \cdot y \wedge y|x \cdot y$. By the least number principle we find a minimal $z > 0$ such that $x|z \wedge y|z$. We denote this z by $\text{lcm}(x, y)$. Clearly $\text{lcm}(x, y) \leq x \cdot y$.

By theorem 3.1 we can now find an a and $0 \leq b < \text{lcm}(x, y)$ such that $x \cdot y = a \cdot \text{lcm}(x, y) + b$. So we have $x|b \wedge y|b$. From this it follows that $b = 0$, because otherwise $0 < b < \text{lcm}(x, y)$, which would contradict the minimality of $\text{lcm}(x, y)$. This means that $x \cdot y = a \cdot \text{lcm}(x, y)$ for some a, and we will denote this a by $\text{gcd}(x, y)$. Since $\text{lcm}(x, y) = y \cdot w$ for some w we have $x \cdot y = \text{gcd}(x, y) \cdot y \cdot w$, thus $x = \text{gcd}(x, y) \cdot w$. Therefore $\text{gcd}(x, y)|x$, and $\text{gcd}(x, y)|y$ in the same way.

This is actually the greatest common divisor. For suppose there is some $g > \text{gcd}(x, y)$ with $g|x$ and $g|y$, then $x = g \cdot x'$ and $y = g \cdot y'$ for some x' and y'. Which means we would have $x \cdot y = g \cdot x' \cdot y$, so $x \cdot y' = x' \cdot y$ and thus $x|x' \cdot y$ and $y|x' \cdot y$. Let w be such that $x = \text{gcd}(x, y) \cdot w$, then clearly $x' < w$. By the definition of $\text{gcd}(x, y)$ we have $\text{gcd}(x, y) \cdot w \cdot y = x \cdot y = \text{gcd}(x, y) \cdot \text{lcm}(x, y)$ and thus $\text{lcm}(x, y) = w \cdot y$. But then $x' \cdot y < w \cdot y = \text{lcm}(x, y)$ which contradicts the minimality of $\text{lcm}(x, y)$. So there is no $g > \text{gcd}(x, y)$ with $g|x$ and $g|y$.

This gives us the necessary tools to formulate Bézout's theorem for PA, which is the final tool that is needed for coding sequences in PA. We will however omit the proof and only state the theorem here.

Theorem 3.2 (Bézout's theorem for PA)**.**

$$\text{PA} \vdash \forall xy \geq 1 \exists a \leq y, b \leq x(a \cdot x = b \cdot y + \text{gcd}(x, y))$$

Proof. See theorem 4.8 in [1]. ■

Now we can code a sequence of numbers. Suppose we are given a sequence $x_0, \ldots, x_{n-1}$. Note that the first index of our sequence is 0. This is convenient since we use the convention that $0 \in \mathbb{N}$.

Let $m = \max(x_0, \ldots, x_{n-1}, n)!$, then we claim that for all $0 \leq i < j < n$ we have that $m(i+1)+1$ and $m(j+1)+1$ are coprime. If they were not coprime, there would be a $d > 1$ such that $d|m(i+1)+1$ and $d|m(j+1)+1$. So there would be $d_1, d_2 \geq 1$ such that $d = d_1 \cdot d_2$, $d_1|m$ and $d_2|j-i$. Since $j - i < n$ we have $j - i|m$ by the definition of m and hence $d_2|m$. At least one of d_1 and d_2 has to be strictly greater than 1. Without loss of generality, assume $d_1 > 1$. Then $d_1|m$ and $d_1|m(i+1)+1$, which is a contradiction. So $m(i+1)+1$ and $m(j+1)+1$ are coprime.

Using this property we can apply the Chinese remainder theorem (which is based on Bézout's theorem) to find a number a such that $a \equiv x_i \bmod m(i+1)+1$ for all $0 \leq i < n$. So in a way, the pair (a, m) codes the sequence $x_0, \ldots, x_{n-1}$.

The next theorem states a few properties that will prove to be very useful. Namely that for every x there is a sequence starting with x, that every sequence can be extended by any x and that every element in the sequence is smaller than a. The last property may not seem directly useful, but it is a technical detail that we will need later on.

Before stating the theorem we will need to agree upon some notation once more. Let $\mathrm{rm}(x, y)$ denote the remainder of x divided by y, and $(a, m)_i$ will denote the ith element in the sequence coded by (a, m). That is $(a, m)_i = \mathrm{rm}(a, m \cdot (i+1)+1)$.

Theorem 3.3. *The following properties hold when coding sequences in* PA*:*

(1) $\mathrm{PA} \vdash \forall x \exists a m((a, m)_0 = x)$

(2) $\mathrm{PA} \vdash \forall l x a m \exists b n(\forall i < l((a, m)_i = (b, n)_i) \wedge (b, n)_l = x)$

(3) $\mathrm{PA} \vdash \forall a m i((a, m)_i \leq a)$

Proof. Again, we refer to [1]. This time to theorem 4.9. ■

3.2 Representing primtitve recursive functions in PA

The language $\mathcal{L}_{PA}$ does not have constants for all the natural numbers, but it will be useful to express the natural numbers in PA. To this end we will define an abbreviation for terms that represent the natural numbers in PA. We let the integer 0 be represented by $\bar{0}$, which is simply the term 0. Then for $n \in \mathbb{N}$ we define $\overline{n+1}$ to be $\overline{n}+1$. So for example, the number 3 would be represented by the term $\overline{3} = ((0+1)+1)+1$. Note that $\bar{n}$ is a term in PA, and the $+$ we use in its definition is the function symbol $+$ from $\mathcal{L}_{PA}$, not the $+$ from the natural numbers $\mathcal{N}$.

The difference between function symbol $+$ from $\mathcal{L}_{PA}$ and the addition operator $+$ from $\mathcal{N}$ is also very important in the following proposition. There they are linked together, but one should still keep in mind that they have different meanings. The same holds for the $\cdot$ and $<$ symbols.

The following proposition is taken from exercise 62 in [1].

Proposition 3.4. *Let n and m be elements of $\mathcal{N}$. Then the following properties hold:*

(1) $\text{PA} \vdash \overline{n} + \overline{m} = \overline{n+m}$;

(2) $\text{PA} \vdash \overline{n} \cdot \overline{m} = \overline{n \cdot m}$;

(3) $\text{PA} \vdash \overline{n} < \overline{m} \iff n < m$;

(4) $\text{PA} \vdash \forall x(x < \overline{n} \leftrightarrow x = \overline{0} \vee \ldots \vee x = \overline{n-1})$ *if* $n > 0$.

Proof. Property (1) follows directly from proposition 3.1 (P1), which describes the associativity of the $+$ operator in PA, and the fact that $\text{PA} \vdash 0 + 1 = 1$. Property (2) also follows from the basic algebraic properties of PA. When working out $\overline{n} \cdot \overline{m}$ we get a concatenation of $n \cdot m$ times the symbols 1 and $+$, which is $\overline{n \cdot m}$.

Property (3) requires a bit more work. Suppose that $\text{PA} \vdash \overline{n} < \overline{m}$, then we have that $\text{PA} \vdash \exists z(\overline{m} = \overline{n} + (z+1))$. That means that $\mathcal{N} \models \exists z(m = n + (z+1))$, from which it clearly follows that $n < m$. Suppose now that $n < m$, then $m - n \geq 1$ and thus $m = n + (z+1)$ with $z = m - n - 1$. So by (1) we have $\text{PA} \vdash \overline{m} = \overline{n} + (\overline{z} + 1)$ and thus $PA \vdash \overline{n} < \overline{m}$.

We will prove (4) using induction to n. Suppose $n = 1$, then we have to prove that $\text{PA} \vdash \forall x(x < 1 \leftrightarrow x = 0)$. If $x < 1$ we either have $x + 1 = 1$, in which case $x = 0$, or we have $x + 1 < 1$ but that is impossible since then $1 \leq x + 1 < 1$. So $x < 1 \to x = 0$, and the converse is trivial. Now suppose that (4) holds for n, we want to prove it for $n+1$. So let $x < \overline{n+1}$, then either $x + 1 = \overline{n+1} = \overline{n} + 1$, in which case $x = \overline{n}$ or $x + 1 < \overline{n+1} = \overline{n} + 1$ and thus $x < \overline{n}$, in which case we use the induction hypothesis to conclude that $x = \overline{0} \vee \ldots \vee x = \overline{n-1}$. Hence $x < \overline{n+1} \to x = \overline{0} \vee \ldots \vee x = \overline{n}$. The converse is again trivial, when using (3). ∎

We can classify formulas based on their quantifiers. We distinguish quantifiers that are *bounded* and quantifiers that are *unbounded*. The *bounded* quantifiers are of the form $\exists x < t\varphi(x)$ or $\forall x < t\varphi(x)$ where t is some term not containing x. The unbounded quantifiers are just all the other quantifiers.

Definition 3.4. An $\mathcal{L}_{PA}$-formula is a Δ_0-*formula* if all its quantifiers are bounded. A Σ_1-*formula* is one of the form $\exists x_1 \ldots x_k \varphi(x_1, \ldots, x_k)$ where $\varphi(x_1, \ldots, x_k)$ is a Δ_0-formula. Similarly, a Π_1-*formula* is of the form $\forall x_1 \ldots x_k \varphi(x_1, \ldots, x_k)$ with $\varphi(x_1, \ldots, x_k)$ a Δ_0-formula. Finally, a formula φ is called a Δ_1-*formula* if φ and $\neg\varphi$ are equivalent to a Σ_1-formula in PA.

When relating PA to $\mathcal{N}$ the Σ_1-formulas are very important because of Σ_1-*completeness*, which states that a Σ_1-sentence is true in $\mathcal{N}$ if and only if it is provable in PA. This is also stated in exercise 63 of [1]. We will present a proof here.

Theorem 3.4 (Σ_1-completeness)**.** *For every Σ_1-formula $\varphi(x_1, \ldots, x_k)$ we have that for all $n_1, \ldots, n_k \in \mathcal{N}$:*

$$\text{PA} \vdash \varphi(\overline{n_1}, \ldots, \overline{n_k}) \iff \mathcal{N} \models \varphi(n_1, \ldots, n_k).$$

In particular this means that a Σ_1-sentence is provable in PA *if and only if it is true in $\mathcal{N}$.*

Before proving theorem 3.4 we will introduce a useful lemma. This lemma actually says the same, but then for Δ_0-formulas. Before we state and prove that lemma, we will cover some theory on the form of formulas which we will need in our proof of the lemma. First, it is easy to see that the following equivalences hold for any formulas α and β (not only in PA, but in every theory):

$$\begin{aligned}
\neg\neg\alpha &\leftrightarrow \alpha,\\
\neg(\alpha \wedge \beta) &\leftrightarrow \neg\alpha \vee \neg\beta,\\
\neg(\alpha \vee \beta) &\leftrightarrow \neg\alpha \wedge \neg\beta,\\
\neg(\alpha \to \beta) &\leftrightarrow \alpha \wedge \neg\beta,\\
\neg\exists x\alpha(x) &\leftrightarrow \forall\neg\alpha(x),\\
\neg\forall x\alpha(x) &\leftrightarrow \exists\neg\alpha(x).
\end{aligned}$$

We can use these to bring a formula φ in a form φ' that is equivalent, but where each occurrence of the symbol $\neg$ is of the form $\neg\chi$ with χ an atomic formula. It will be useful to give this form a name.

Definition 3.5. For every formula, there is an equivalent formula such that each occurrence of the negation symbol only applies to atomic formulas. This form is called the *negation normal form.*

Lemma 3.1. *For every Δ_0-formula $\varphi(x_1, \ldots, x_k)$ we have that for all $n_1, \ldots, n_k \in \mathcal{N}$:*

$$\text{PA} \vdash \varphi(\overline{n_1}, \ldots, \overline{n_k}) \iff \mathcal{N} \models \varphi(n_1, \ldots, n_k).$$

Proof. The implication from the left to the right is trivial, since $\mathcal{N}$ is a model for PA. We will show the converse first for quantifier-free sentences. So let ψ be a quantifier-free sentence with $\mathcal{N} \models \psi$. Bring ψ into negation normal form to obtain ψ', we will show with induction to the number of binary logic symbols in ψ' that $\text{PA} \vdash \psi'$. When ψ' has 0 binary logic symbols, the possible forms are: $t_1 = t_2$, $t_1 < t_2$, $\neg(t_1 = t_2)$ and $\neg(t_1 < t_2)$ for terms t_1 and t_2. Since every term in $\mathcal{N}$ that does not contain variables evaluates to some integer, the interpretations $t_1^{\mathcal{N}}$ and $t_2^{\mathcal{N}}$ are integers. Which means that one of the cases $t_1^{\mathcal{N}} = t_2^{\mathcal{N}}$, $t_1^{\mathcal{N}} < t_2^{\mathcal{N}}$, $t_1^{\mathcal{N}} \neq t_2^{\mathcal{N}}$ or $\neg(t_1^{\mathcal{N}} < t_2^{\mathcal{N}})$ holds. Using proposition 3.4 we see that $\text{PA} \vdash \overline{t_1^{\mathcal{N}}} = \overline{t_2^{\mathcal{N}}}$, $\text{PA} \vdash \overline{t_1^{\mathcal{N}}} < \overline{t_2^{\mathcal{N}}}$, $\text{PA} \vdash \overline{t_1^{\mathcal{N}}} \neq \overline{t_2^{\mathcal{N}}}$ or $\text{PA} \vdash \neg(\overline{t_1^{\mathcal{N}}} < \overline{t_2^{\mathcal{N}}})$, depending on the case. Thus $\text{PA} \vdash \psi'$.

Now suppose that the induction hypothesis holds for quantifier-free sentences in negation normal form with m binary logic symbols, and that ψ' has $m+1$ binary logic symbols. Then ψ' is of one of the following forms: $\alpha \wedge \beta$, $\alpha \vee \beta$ or $\alpha \to \beta$. Since α and β both have at most m binary logic symbols, we can use the induction hypothesis to obtain

$$\text{PA} \vdash \alpha \iff \mathcal{N} \models \alpha,$$

and likewise for β. In the first case we have $\mathcal{N} \models \alpha$ and $\mathcal{N} \models \beta$. Thus $\text{PA} \vdash \alpha$ and $\text{PA} \vdash \beta$, from which it follows that $\text{PA} \vdash \alpha \wedge \beta$. In the second case we either have $\mathcal{N} \models \alpha$ or $\mathcal{N} \models \beta$, so $\text{PA} \vdash \alpha$ or $\text{PA} \vdash \beta$. From both it follows that $\text{PA} \vdash \alpha \vee \beta$. For the third case we either have $\mathcal{N} \models \beta$, thus $\text{PA} \vdash \beta$ and thus $\text{PA} \vdash \alpha \to \beta$, or we have $\mathcal{N} \models \neg\beta$, thus $\mathcal{N} \models \neg\alpha$. In which case we can apply

the induction hypothesis to the negation normal form of $\neg\alpha$ (note that bringing it in negation normal form does not change the amount of binary logic symbols) and conclude that PA $\vdash \neg\alpha$, so again PA $\vdash \alpha \to \beta$. In all three cases we see that PA $\vdash \psi'$.

Now that we have established the converse for quantifier-free formulas, we can finish the proof by showing that $\varphi(\overline{n_1}, \ldots, \overline{n_k})$ is equivalent to a quantifier-free sentence ψ in PA. Because then it is also equivalent to ψ in $\mathcal{N}$, and we find:

$$\mathcal{N} \models \varphi(n_1, \ldots, n_k) \implies \mathcal{N} \models \psi \implies \text{PA} \vdash \psi \implies \text{PA} \vdash \varphi(\overline{n_1}, \ldots, \overline{n_k}).$$

To transform $\varphi(\overline{n_1}, \ldots, \overline{n_k})$ into a quantifier-free sentence we will simply replace all of its subformulas containing quantifiers by something equivalent. The first occurrence of a quantifier has to be of the form $\forall y < t\chi(y)$ or $\exists y < t\chi(y)$. Here t is some term not containing any variables, because $\varphi(\overline{n_1}, \ldots, \overline{n_k})$ contains no free variables and we are looking at the first occurrence of a quantifier. So the interpretation $t^{\mathcal{N}}$ of t in $\mathcal{N}$ is just an integer. If the subformula containing the quantifier was of the form $\forall y < t\chi(y)$, it is equivalent in PA to $\chi(\overline{0}) \wedge \ldots \wedge \chi(\overline{t^{\mathcal{N}} - 1})$ (or $\neg\bot$ if $t^{\mathcal{N}} = 0$). If it was of the form $\exists y < t\chi(y)$, it is equivalent in PA to $\chi(\overline{0}) \vee \ldots \vee \chi(\overline{t^{\mathcal{N}} - 1})$ (or $\bot$ if $t^{\mathcal{N}} = 0$). We can now replace the subformula by another formula that is equivalent but where the quantifier has been eliminated. By repeating this progress of replacing subformulas we obtain the sentence ψ we were looking for. ■

Proof of theorem 3.4. We can now prove theorem 3.4. Again, the implication from the left to the right is trivial. So suppose that φ is a Σ_1-formula, and that for $n_1, \ldots, n_k \in \mathcal{N}$ we have that

$$\mathcal{N} \models \varphi(n_1, \ldots, n_k).$$

Note that $\varphi(n_1, \ldots, n_k)$ is actually of the form $\exists x_1 \ldots x_m \varphi'(n_1, \ldots, n_k, x_1, \ldots, x_m)$ for some Δ_0-formula φ'. Since $\varphi(n_1, \ldots, n_k)$ is true in $\mathcal{N}$ there must be $a_1, \ldots, a_m \in \mathcal{N}$ such that $\varphi'(n_1, \ldots, n_k, a_1, \ldots, a_m)$ is true in $\mathcal{N}$. Using lemma 3.1 we see that PA $\vdash \varphi'(\overline{n_1}, \ldots, \overline{n_k}, \overline{a_1}, \ldots, \overline{a_m})$, hence PA $\vdash \exists x_1 \ldots x_m \varphi'(\overline{n_1}, \ldots, \overline{n_k}, x_1, \ldots, x_m)$. So we conclude that PA $\vdash \varphi(\overline{n_1}, \ldots, \overline{n_k})$. ■

We will finish this subsection with a theorem that lets us represent primitive recursive functions as Δ_1-formulas. To do so we will introduce one more abreviation. We let $\exists! x \varphi(x)$ mean "there is exactly one x such that $\varphi(x)$", or more formally:

$$\exists x \forall y (\varphi(y) \leftrightarrow x = y).$$

The proof of this theorem is mostly taken from theorem 4.13 in [1], it is however adjusted a bit to make it work for Δ_1-formulas and not only Σ_1-formulas (like in [1]).

Theorem 3.5. *Let $F : \mathbb{N}^k \to \mathbb{N}$ be a primitive recursive function, then there is a Δ_1-formula $\varphi_F(\vec{x}, y)$ with $k+1$ free variables such that for all $n_1, \ldots, n_k \in \mathbb{N}$:*

$$\text{PA} \vdash \varphi(\overline{n_1}, \ldots, \overline{n_k}, \overline{F(n_1, \ldots, n_k)}), \tag{1}$$

$$\text{PA} \vdash \forall x_1, \ldots, x_k \exists! y \varphi(x_1, \ldots, x_k, y). \tag{2}$$

Proof. The primitive recursive functions are generated inductively. So we can use induction on their generation to prove theorem 3.5. For the zero function Z, successor function S and projections Π_i^k a formula is easy to find: $\varphi_Z(x,y) = (y=0)$, $\varphi_S(x,y) = (y = x+1)$ and $\varphi_{\Pi_i^k}(x_1,\ldots,x_k,y) = (y = x_i)$ respectively. These are all Δ_0-formulas and thus Δ_1-formulas. Furthermore, they clearly satisfy properties (1) and (2).

Now suppose that F is defined by composition of $H, G_1, \ldots, G_m$, that is

$$F(\vec{x}) = H(G_1(\vec{x}), \ldots, G_m(\vec{x})).$$

By the induction hypothesis we find Δ_1-formulas $\varphi_H, \varphi_{G_1}, \ldots, \varphi_{G_m}$ that satisfy properties (1) and (2). Now, let $\varphi_F(\vec{x}, y)$ be

$$\exists y_1 \ldots y_m(\varphi_{G_1}(\vec{x}, y_1) \wedge \ldots \wedge \varphi_{G_m}(\vec{x}, y_m) \wedge \varphi_H(y_1, \ldots, y_m, y)),$$

then φ_F is a Σ_1-formula. From the definition of φ_F is follows that it satisfies property (1). Property (2) follows from the fact that $\varphi_H, \varphi_{G_1}, \ldots, \varphi_{G_m}$ all satisfy property (2).

To prove that φ_F is a Δ_1-formula, we will show that $\neg\varphi_F(\vec{x}, y)$ is equivalent to

$$\exists y_1 \ldots y_m(\varphi_{G_1}(\vec{x}, y_1) \wedge \ldots \wedge \varphi_{G_m}(\vec{x}, y_m) \wedge \neg\varphi_H(y_1, \ldots, y_m, y)),$$

which we will call $\psi(\vec{x}, y)$. Then using the fact that φ_H is a Δ_1-formula (and thus $\neg\varphi_H$ is a Σ_1-formula) we see that this is indeed a Σ_1-formula. Note that $\neg\varphi_F(\vec{x}, y)$ is clearly equivalent to

$$\forall y_1 \ldots y_m(\neg(\varphi_{G_1}(\vec{x}, y_1) \wedge \ldots \wedge \varphi_{G_m}(\vec{x}, y_m)) \vee \neg\varphi_H(y_1, \ldots, y_m, y)).$$

Suppose that $\neg\varphi_F(\vec{x}, y)$ holds in PA. By property (2) of $\varphi_{G_1}, \ldots, \varphi_{G_m}$ there must be $y_1, \ldots, y_m$ such that $\varphi_{G_1}(\vec{x}, y_1) \wedge \ldots \wedge \varphi_{G_m}(\vec{x}, y_m)$, which means that $\neg\varphi_H(y_1, \ldots, y_m, y)$, and so $\psi(\vec{x}, y)$, must hold in PA. Now suppose that $\psi(\vec{x}, y)$ holds in PA, then we have $y_1, \ldots, y_m$ such that $\varphi_{G_1}(\vec{x}, y_1) \wedge \ldots \wedge \varphi_{G_m}(\vec{x}, y_m) \wedge \neg\varphi_H(y_1, \ldots, y_m, y)$. For any other sequence $y_1', \ldots, y_m'$ we have by property (2) of $\varphi_{G_1}, \ldots, \varphi_{G_m}$ that $\neg\varphi_{G_i}(\vec{x}, y_i')$ for at least one $1 \leq i \leq m$, hence $\neg(\varphi_{G_1}(\vec{x}, y_1) \wedge \ldots \wedge \varphi_{G_m}(\vec{x}, y_m))$. So we see that $\neg\varphi_F(\vec{x}, y)$ holds in PA, which concludes our proof of $\neg\varphi_F(\vec{x}, y)$ being equivalent to $\psi(\vec{x}, y)$.

The last possibility is that F is defined by primitive recursion. That is, there are functions G, H such that

$$\begin{aligned} F(0, \vec{x}) &= G(\vec{x}), \\ F(z+1, \vec{x}) &= H(z, F(z, \vec{x}), \vec{x}). \end{aligned}$$

Using the induction hypothesis we find Δ_1-formulas $\varphi_G(\vec{x}, y)$ and $\varphi_H(z, u, \vec{x}, y)$ that satisfy properties (1) and (2). We define $\varphi_F(z, \vec{x}, y)$ as follows:

$$\exists am(\varphi_G(\vec{x}, (a,m)_0) \wedge \forall i < z\varphi_H(i, (a,m)_i, \vec{x}, (a,m)_{i+1}) \wedge y = (a,m)_z).$$

The idea behind this is that the pair (a, m) encodes the sequence of outcomes of $F(0, \vec{x}), \ldots, F(z, \vec{x})$. We will show that properties (1) and (2) hold for φ_F by induction to z. We will use theorem 3.3 repeatedly here.

We first look at the case where $z = 0$. Then $\varphi_F(z, \vec{x}, y)$ is nothing more than $\exists am(\varphi_G(\vec{x}, (a,m)_0) \wedge y = (a,m)_0)$. By property (1) of φ_G we have that

$\varphi_G(\overline{x_1}, \ldots, \overline{x_k}, \overline{G(\vec{x})})$ holds. Then using theorem 3.3 we find a, m such that $(a, m)_0 = \overline{G(\vec{x})}$, and we can conclude that property (1) holds for φ_F. Since there is exactly one y' such that $\varphi_G(\vec{x}, y')$ holds for all $\vec{x}$ there also must be exactly one y such that $\varphi_F(0, \vec{x}, y)$ holds, which proves property (2) for φ_F.

Now suppose that properties (1) and (2) hold for $\varphi_F(z, \vec{x}, y)$, then we have a pair (a, m) that encodes the sequence $F(0, \vec{x}), \ldots, F(z, \vec{x})$. This is the only sequence that can satisfy

$$\varphi_G(\vec{x}, (a, m)_0) \wedge \forall i < z \varphi_H(i, (a, m)_i, \vec{x}, (a, m)_{i+1})$$

because of property (2) of φ_G and φ_H. Using theorem 3.3 we find an encoding (a', m') for a sequence that extends the sequence encoded by (a, m) by $F(z + 1, \vec{x})$. By property (1) of φ_H we find that this sequence satisfies

$$\varphi_G(\vec{x}, (a', m')_0) \wedge \forall i < (z + 1) \varphi_H(i, (a', m')_i, \vec{x}, (a', m')_{i+1}).$$

By definition of (a', m') we have that $(a', m')_{z+1} = \overline{F(z + 1, \vec{x})}$, thus property (1) is satisfied. By property (2) of φ_H and φ_G this is also the only possible sequence, so we conclude that property (2) is also satisfied for φ_F.

Recall that $(a, m)_i$ is just an abbreviation. For example $\varphi_G(\vec{x}, (a, m)_0)$ is just short for

$$\exists c, d < a (a = c \cdot (m + 1) + d \wedge 0 \leq d < m + 1 \wedge \varphi_G(\vec{x}, d)),$$

which is a Σ_1-formula, since φ_G is a Δ_1-formula. It is here that we used (3) from theorem 3.3. This means that φ_F is equivalent to a Σ_1-formula. Now we consider $\neg\varphi(z, \vec{x}, y)$, or equivalently

$$\forall a m (\neg(\varphi_G(\vec{x}, (a, m)_0) \wedge \forall i < z \varphi_H(i, (a, m)_i, \vec{x}, (a, m)_{i+1})) \vee y \neq (a, m)_z).$$

We define $\psi(z, \vec{x}, y)$ as

$$\exists a m (\varphi_G(\vec{x}, (a, m)_0) \wedge \forall i < z \varphi_H(i, (a, m)_i, \vec{x}, (a, m)_{i+1}) \wedge y \neq (a, m)_z),$$

which is equivalent to a Σ_1-formula. We claim that $\neg\varphi(z, \vec{x}, y)$ is equivalent to $\psi(z, \vec{x}, y)$. Suppose that $\neg\varphi(z, \vec{x}, y)$ holds in PA. Using property (2) of φ_G and φ_H we find a sequence $y_0, \ldots, y_z$ such that

$$\varphi_G(\vec{x}, y_0) \wedge \varphi_H(\overline{0}, y_0, \vec{x}, y_1) \wedge \ldots \wedge \varphi(\overline{z - 1}, y_{z-1}, \vec{x}, y_z)$$

holds in PA. By theorem 3.3 this means that we have a pair (a, m) encoding the sequence $y_0, \ldots, y_z$. Thus there are a, m such that

$$\varphi_G(\vec{x}, (a, m)_0) \wedge \forall i < z \varphi_H(i, (a, m)_i, \vec{x}, (a, m)_{i+1}),$$

which means that $y \neq (a, m)_z$ and so $\psi(z, \vec{x}, y)$ holds in PA. Now suppose that $\psi(z, \vec{x}, y)$ holds in PA, then we have a sequence encoded by (a, m) such that

$$\varphi_G(\vec{x}, (a, m)_0) \wedge \forall i < z \varphi_H(i, (a, m)_i, \vec{x}, (a, m)_{i+1}) \wedge y \neq (a, m)_z).$$

Furthermore, the sequence $(a, m)_0, \ldots, (a, m)_z$ is unique by property (2) of φ_G and φ_H. So for any (a', m') encoding a sequence we either have that

$$\varphi_G(\vec{x}, (a, m)_0) \wedge \forall i < z \varphi_H(i, (a, m)_i, \vec{x}, (a, m)_{i+1})$$

does not hold, or it holds in which case we must have that $(a', m')_z = (a, m)_z$ by uniqueness and so $y \neq (a', m')_z$. In both cases we conclude that $\neg\varphi_F(z, \vec{x}, y)$ holds in PA. So $\neg\varphi_F(z, \vec{x}, y)$ is indeed equivalent to ψ and we conclude that φ_F is a Δ_1-formula. ■

Definition 3.6. If, given a function $F : \mathbb{N}^k \to \mathbb{N}$, we can find a formula φ_F as in theorem 3.5 then it is called *provably recursive*. This formula φ_F is also said to *represent* F. However, that is in fact a weaker notion. Because property (2) is not required then, but instead we must have for all $n_1, \ldots, n_k$ that

$$\text{PA} \vdash \exists! y \varphi_F(\overline{n_1}, \ldots, \overline{n_k}, y).$$

In both cases φ_F does not even need to be a Δ_1-formula, a Σ_1-formula is good enough.

In the definition above we mention that for that definition we do not need φ_F to be a Δ_1-formula. However, we did prove theorem 3.5 for Δ_1-formulas because that will be needed later on when proving Gödel's incompleteness theorems.

3.3 Representing total recursive functions in PA

As we have seen in the previous subsection, primitive recursive functions are important because we can represent them in PA. Similar results hold for the total recursive functions. In this subsection we will take a look at these results.

The following theorem is not mentioned in [1], but is necessary later in sections 6 and 7.

Theorem 3.6. *Let $\varphi(x)$ be a Σ_1-formula, then the set $\{x \in \mathbb{N} : \mathcal{N} \models \varphi(x)\}$ is recursively enumerable.*

Proof. Since $\varphi(x)$ is a Σ_1-formula, it is of the form $\exists y_1, \ldots, y_k \psi(x, y_1, \ldots, y_k)$ for some Δ_0-formula $\psi(x, y_1, \ldots, y_k)$. As we argued in the proof of lemma 3.1, we can transform $\psi(x, y_1, \ldots, y_k)$ into a quantifier-free formula if $x, y_1, \ldots, y_k$ are integers. That means there is an algorithm that can simply check whether or not $\psi(x, y_1, \ldots, y_k)$ is true.

We will now describe an algorithm that enumerates all of $\{x \in \mathbb{N} : \mathcal{N} \models \varphi(x)\}$. We iterate through the set $\mathbb{N}^{k+1}$. For each element $(x, y_1, \ldots, y_k)$ we can check whether or not $\psi(x, y_1, \ldots, y_k)$, and thus $\varphi(x)$, is true. Whenever this is the case, we output x.

The described algorithm clearly enumerates exactly the elements of $\{x \in \mathbb{N} : \mathcal{N} \models \varphi(x)\}$, so this set is recursively enumerable. ■

Theorem 3.5 proves that we can represent primitive recursive functions in a slightly stronger manner than them just being Σ_1-represented, as we defined in definition 3.6. We can also represent total recursive functions, but they can only be Σ_1-represented.

Theorem 3.7. *Every total recursive function is Σ_1-represented in* PA.

The proof of theorem 3.7 requires some results that are beyond the scope of this paper. For the full proof we refer to [1] theorem 4.14. We will provide a sketch of the proof below.

Proof sketch of theorem 3.7. Suppose we are given a total recursive function $F : \mathbb{N}^k \to \mathbb{N}$. Using the results that are beyond the scope of this paper one is able to construct a primitive recursive set $A \subseteq \mathbb{N}^{k+2}$ with the following property. There is at least one y such that $(x_1, \ldots, x_k, y, m) \in A$, precisely when

$F(x_1, \ldots, x_k) = m$. By theorem 3.5 there is a Σ_1-formula $\varphi(x_1, \ldots, x_k, y, m)$ that represents A. That is, for all $x_1, \ldots, x_k, y, m \in \mathbb{N}$ we have:

$$(x_1, \ldots, x_k, y, m) \in A \Leftrightarrow \text{PA} \vdash \varphi(\overline{x_1}, \ldots, \overline{x_k}, \overline{y}, \overline{m}),$$
$$(x_1, \ldots, x_k, y, m) \notin A \Leftrightarrow \text{PA} \vdash \neg\varphi(\overline{x_1}, \ldots, \overline{x_k}, \overline{y}, \overline{m}).$$

This formula φ can be used to construct a Δ_0-formula $S(x_1, \ldots, x_k, z, m)$ such that:

$$\text{PA} \vdash \exists y m \varphi(x_1, \ldots, x_k, y, m) \leftrightarrow \exists! z \exists m S(x_1, \ldots, x_k, z, m).$$

Using the way $S(x_1, \ldots, x_k, z, m)$ is constructed one can check relatively easy that $\exists z S(x_1, \ldots, x_k, z, m)$ represents F. ■

4 Gödel's incompleteness theorems

The incompleteness theorems of Gödel rely on the power of PA. More precisely, they rely on the power of PA being able to make statements about its own theorems. To do so we first need to find some way of coding formulas and proofs. We will discuss the necessary tools to do so in subsection 4.1. After that we are finally ready to state and prove Gödel's incompleteness theorems.

4.1 Coding formulas and proofs

It is well known that there are bijections from $\mathbb{N}\times\mathbb{N}$ to $\mathbb{N}$. One of these bijections is

$$j(n, m) = \frac{1}{2}(n + m)(n + m + 1) + n,$$

which is a primitive recursive function because it is composed multplication and addition. In fact, the function j enumerates the elements in $\mathbb{N} \times \mathbb{N}$ as displayed in figure 1.

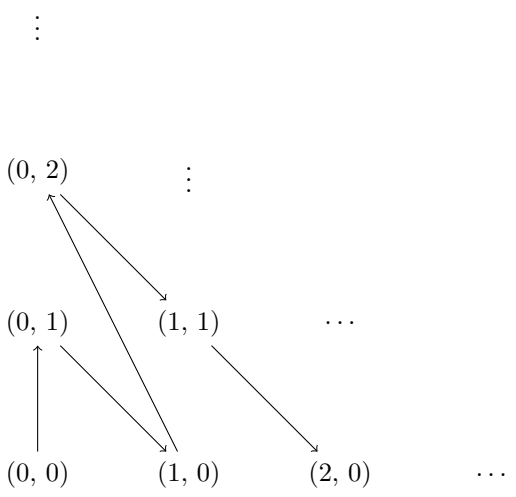


Figure 1: Bijection from $\mathbb{N} \times \mathbb{N}$ to $\mathbb{N}$.

Because j is bijective, we can find functions j_1 and j_2 such that

$$j(j_1(z), j_2(z)) = z.$$

In this specific case the functions j_1 and j_2 are primitive recursive. We can now define a bijective function from $\mathbb{N}^k$ to $\mathbb{N}$ for any k.

Definition 4.1. We define the bijections $j^k : \mathbb{N}^k \to \mathbb{N}$ as follows:

$$j^1(x) = x$$
$$j^{k+1}(x_1, \ldots, x_{k+1}) = j(j^k(x_1, \ldots, x_k), x_{k+1}).$$

The *projection functions*, $j_i^k : \mathbb{N} \to \mathbb{N}$ (with $1 \leq i \leq k$) are functions satisfying

$$j^k(j_1^k(z), \ldots, j_k^k(z)) = z$$

for all $z \in \mathbb{N}$, and are defined by

$$j_1^1(z) = z$$
$$j_i^{k+1} = \begin{cases} j_i^k(j_1(z)) & \text{if } 1 \leq i \leq k \\ j_2(z) & \text{if } i = k+1 \end{cases}$$

Using the fact that j, j_1 and j_2 are primitive recursive, we see that the functions j^k and j_i^k are also primitive recursive for all $k \in \mathbb{N}$ and $1 \leq i \leq k$.

These functions give us a way of coding sequences. A sequence $(x_1, \ldots, x_k)$ is nothing but an element of $\mathbb{N}^k$. Here we see $\mathbb{N}^0$ as the set that contains a unique element $(-)$, the empty sequence. We introduce the notation $\langle x_1, \ldots, x_k \rangle$ to denote the code for the sequence $(x_1, \ldots, x_k)$, and $\langle\rangle$ denotes the code for the empty sequence. So $\langle x_1, \ldots, x_k \rangle$ is just an integer, coding the sequence $(x_1, \ldots, x_k)$.

Definition 4.2. We code sequences as follows:

$$\langle\rangle = 0$$
$$\langle x_1, \ldots, x_k \rangle = j(k, j^k(x_1, \ldots, x_k)) + 1$$

Being able to code sequences gives us the necessary tools to code formulas and proofs. This means that to every formula φ we will be assigned some integer $\ulcorner \varphi \urcorner$ that uniquely determines the formula. This integer $\ulcorner \varphi \urcorner$ is called the *Gödel number* for formula φ.

We will code formulas as follows (in fact, we use the exact same way as in [1] section 5.1). We assume that the variables in $\mathcal{L}_{PA}$ are numbered $v_0, v_1, \ldots$, and we use the following "code book" where we from now on take $<$ to be a symbol of $\mathcal{L}_{PA}$.

0	1	v	$+$	$\cdot$	$=$	$<$	$\wedge$	$\vee$	$\to$	$\neg$	$\forall$	$\exists$
0	1	2	3	4	5	6	7	8	9	10	11	12

We code terms by recursion. So $\ulcorner 0 \urcorner = \langle 0 \rangle$, $\ulcorner 1 \urcorner = \langle 1 \rangle$, $\ulcorner v_i \urcorner = \langle 2, i \rangle$, $\ulcorner t + s \urcorner = \langle 3, \ulcorner t \urcorner, \ulcorner s \urcorner \rangle$ and $\ulcorner t \cdot s \urcorner = \langle 4, \ulcorner t \urcorner, \ulcorner s \urcorner \rangle$. We code formulas likewise by recursion, so for example $\ulcorner \varphi \wedge \psi \urcorner = \langle 7, \ulcorner \varphi \urcorner, \ulcorner \psi \urcorner \rangle$.

Because the coding of formulas is done recursively and the sequences are coded by primitive recursive functions, we can use primitive recursive functions to describe properties like "the main connective of this formula is $\wedge$".

Likewise we can specify a code book for proof trees and recursively code proof trees (this is worked out in [1], section 5.2). Then we can also use primitive recursive functions to express properties like "the conclusion of this tree is reached using $\rightarrow$ introduction". Even more important: we can create a primitive recursive predicate $\mathrm{Prf}(y, x)$ that says "y is the code of a correct proof in PA for the formula coded by x".

4.2 Gödel's first incompleteness theorem

The proof of Gödel's first incompleteness theorem heavily relies on the *diagonalization lemma*. Before proving this lemma we note that the properties "x codes a formula φ" and "y codes a term t" can be described using primitive recursive functions. This makes it possible to define a primitive recursive function $\mathrm{Sub}(x, y, i)$ as follows:

$$\mathrm{Sub}(x, y, i) = \begin{cases} \ulcorner \varphi[s/v_i] \urcorner & \text{if "}y\text{ codes a formula }\varphi\text{" and "}x\text{ codes a term }s\text{"} \\ 0 & \text{else} \end{cases}$$

Lemma 4.1 (Diagonalization lemma)**.** *Let φ be an $\mathcal{L}_{PA}$-formula with free variable v_0, then there is an $\mathcal{L}_{PA}$-formula ψ with the same free variables as φ except for v_0, such that:*

$$\mathrm{PA} \vdash \psi \leftrightarrow \varphi[\overline{\ulcorner \psi \urcorner}/v_0].$$

If φ is Π_1, then ψ can be chosen to be Π_1 as well.

Proof. The function $\mathrm{Sub}(x, y, i)$ is primitive recursive, so $\lambda xy.Sub(x, y, 0)$ is also primitive recursive. By theorem 3.5 there is a Δ_1-formula S that represents $\lambda xy.Sub(x, y, 0)$ in PA. We let T be the Δ_1-formula that represents the primitive recursive function $n \mapsto \ulcorner \overline{n} \urcorner$ in PA. Recall that we thus have the following properties for S and T and all $n, m \in \mathbb{N}$:

$$\mathrm{PA} \vdash S(\overline{n}, \overline{m}, \overline{\mathrm{Sub}(n, m, 0)}) \tag{1}$$

$$\mathrm{PA} \vdash \forall xy \exists! z S(x, y, z) \tag{2}$$

$$\mathrm{PA} \vdash T(\overline{n}, \overline{\ulcorner \overline{n} \urcorner}) \tag{3}$$

$$\mathrm{PA} \vdash \forall x \exists! z T(x, z) \tag{4}$$

Since φ has v_0 free, we can define a formula C to be equivalent to:

$$\forall xy(T(v_0, x) \wedge S(x, v_0, y) \rightarrow \varphi[y/v_0]),$$

and we let ψ be defined by:

$$C[\overline{\ulcorner C \urcorner}/v_0].$$

We claim that this is the ψ we are looking for. It is clear that if φ is a Π_1-formula, that C can be chosen to be a Π_1-formula and hence ψ is a Π_1-formula.

To prove the claim about ψ we reason in PA. Suppose that ψ and thus

$$\forall xy(T(\overline{\ulcorner C \urcorner}, x) \wedge S(x, \overline{\ulcorner C \urcorner}, y) \rightarrow \varphi[y/v_0])$$

holds. By (1) we have that

$$T(\overline{\ulcorner C \urcorner}, \overline{\ulcorner \overline{\ulcorner C \urcorner} \urcorner})$$

holds, and by (3) we have that

$$S(\overline{\ulcorner \overline{\ulcorner C \urcorner} \urcorner}, \overline{\ulcorner C \urcorner}, \overline{\ulcorner C[\overline{\ulcorner C \urcorner}/v_0] \urcorner})$$

holds. Therefore we can conclude that $\varphi[\overline{\ulcorner C[\overline{\ulcorner C \urcorner}/v_0] \urcorner}/v_0]$ holds, and thus by the definition of ψ we have $\varphi[\overline{\ulcorner \psi \urcorner}/v_0]$ so $\psi \to \varphi[\overline{\ulcorner \psi \urcorner}/v_0]$.

Now suppose that $\varphi[\overline{\ulcorner \psi \urcorner}/v_0]$ holds. For $x = \overline{\ulcorner \overline{\ulcorner C \urcorner} \urcorner}$ and $y = \overline{\ulcorner C[\overline{\ulcorner C \urcorner}/v_0] \urcorner}$ the formula

$$T(\overline{\ulcorner C \urcorner}, x) \wedge S(x, \overline{\ulcorner C \urcorner}, y) \tag{5}$$

holds by (1) and (3). By (2) and (4) these are the only values for x and y such that formula (5) holds. So the only values for x and y for which formula (5) holds are exactly those values such that

$$T(\overline{\ulcorner C \urcorner}, x) \wedge S(x, \overline{\ulcorner C \urcorner}, y) \to \varphi[y/v_0].$$

So we conclude that

$$\forall xy(T(\overline{\ulcorner C \urcorner}, x) \wedge S(x, \overline{\ulcorner C \urcorner}, y) \to \varphi[y/v_0])$$

holds, which is exactly ψ. Thus $\varphi[\overline{\ulcorner \psi \urcorner}/v_0] \to \psi$ and our proof is completed. ■

In section 4.1 we mentioned the existence of a primitive recursive predicate $\mathrm{Prf}(y, x)$ that says "y is the code of a correct proof in PA for the formula coded by x". Let $\overline{\mathrm{Prf}}(y, x)$ be a Σ_1-formula representing this predicate. We can define a *proof predicate* $\mathrm{Th}(x)$ as $\exists y \overline{\mathrm{Prf}}(y, x)$. In other words: $\mathrm{Th}(x)$ is true whenever the formula coded by x is provable in PA.

Proposition 4.1. *The proof predicate* $\mathrm{Th}(x)$ *described above satisfies the following properties, for any two* $\mathcal{L}_{PA}$ *formulas* φ *and* ψ*:*

(1) If $\mathrm{PA} \vdash \varphi$*, then* $\mathrm{PA} \vdash \mathrm{Th}(\overline{\ulcorner \varphi \urcorner})$.

(2) $\mathrm{PA} \vdash \mathrm{Th}(\overline{\ulcorner \varphi \urcorner}) \wedge \mathrm{Th}(\overline{\ulcorner \varphi \to \psi \urcorner}) \to \mathrm{Th}(\overline{\ulcorner \psi \urcorner})$.

(3) $\mathrm{PA} \vdash \mathrm{Th}(\overline{\ulcorner \varphi \urcorner}) \to \mathrm{Th}(\mathrm{Th}(\overline{\ulcorner \varphi \urcorner}))$.

Proof. For property (1) we suppose that $\mathrm{PA} \vdash \varphi$. That means there is a proof of φ, which can be coded by some $y \in \mathbb{N}$. Hence $\mathrm{PA} \vdash \overline{\mathrm{Prf}}(\overline{y}, \overline{\ulcorner \varphi \urcorner})$, thus $\mathrm{PA} \vdash \mathrm{Th}(\overline{\ulcorner \varphi \urcorner})$.

To prove property (2) we reason in PA. Suppose that $\mathrm{Th}(\overline{\ulcorner \varphi \urcorner})$ and $\mathrm{Th}(\overline{\ulcorner \varphi \to \psi \urcorner})$ hold, then there are x and y that code proofs for φ and $\varphi \to \psi$ respectively. We can combine these two proofs using $\to$ elimination to find a new proof coded by z for ψ. So $\overline{\mathrm{Prf}}(z, \overline{\ulcorner \psi \urcorner})$ holds and we have proven property (2).

Property (3) is actually a consequence of a more general result called *formalized* Σ_1*-completeness*. This theorem asserts that for any Σ_1-formula φ we have that $\mathrm{PA} \vdash \varphi \to \mathrm{Th}(\overline{\ulcorner \varphi \urcorner})$. Proving formalized Σ_1-completeness actually comes down to formalizing theorem 3.4 in PA. To see how this is exactly done we refer to theorem 5.7 of [1]. Because Th is a Σ_1-formula, we can apply this theorem to $\mathrm{Th}(\overline{\ulcorner \varphi \urcorner})$ and thereby obtain property (3). ■

The first incompleteness theorem of Gödel states that there is some sentence G, the *Gödel sentence*, that is *independent* of PA. That is, PA proves neither G nor $\neg G$.

Theorem 4.1 (Gödel's first incompleteness theorem)**.** *There is an $\mathcal{L}_{PA}$-sentence G, the Gödel sentence, that is independent of* PA.

Proof. Apply the diagonalization lemma, lemma 4.1, to the formula $\neg \operatorname{Th}(v_0)$ to obtain a formula G such that

$$\mathrm{PA} \vdash G \leftrightarrow \neg \operatorname{Th}(\overline{\ulcorner G \urcorner}).$$

To prove that $\mathrm{PA} \nvdash G$, we assume that $\mathrm{PA} \vdash G$ and aim for a contradiction. Using property (1) of proposition 4.1 we see that $\mathrm{PA} \vdash \operatorname{Th}(\overline{\ulcorner G \urcorner})$. But by the choice of G we also have that $\mathrm{PA} \vdash \neg \operatorname{Th}(\overline{\ulcorner G \urcorner})$. Thus PA would have to be inconsistent, which is not the case.

Now we suppose that $\mathrm{PA} \vdash \neg G$. Again we aim for a contradiction to prove that $\mathrm{PA} \nvdash \neg G$. By the choice of G we have that $\mathrm{PA} \vdash \operatorname{Th}(\overline{\ulcorner G \urcorner})$. Since $\mathcal{N}$ is a model of PA, we have that $\operatorname{Th}(\overline{\ulcorner G \urcorner})$ is tue in $\mathcal{N}$. So there is some x coding a proof of G in PA, thus $\mathrm{PA} \vdash G$. This would again mean that PA is inconsistent, which is not the case. ■

4.3 Gödel's second incompleteness theorem

The second incompleteness theorem of Gödel makes use of his first incompleteness theorem to show that PA does not prove its own consistency. To formalize this we need the following notation.

Definition 4.3. For a theory T we define the $\mathcal{L}_{PA}$-sentence $\operatorname{Con}(T)$ as

$$\neg \operatorname{Th}_T(\overline{\ulcorner \bot \urcorner}),$$

where $\operatorname{Th}_T(x)$ is a proof predicate in PA for the theory T. In particular, $\operatorname{Con}(\mathrm{PA})$ is $\neg \operatorname{Th}(\overline{\ulcorner \bot \urcorner})$.

We are now ready to formulate Gödels second incompleteness theorem. We will however first prove a result that is a bit more general, from which the second incompleteness theorem will easily follow. We first look at a lemma (taken from exercise 79 in [1]) that is necessary, then we will state and prove this more general result.

Lemma 4.2. *Given an $\mathcal{L}_{PA}$-formula φ that satisfies properties (1) and (2) in the same way as* Th *does in proposition 4.1, we have*

$$\mathrm{PA} \vdash \varphi(\overline{\ulcorner \alpha \wedge \beta \urcorner}) \leftrightarrow \varphi(\overline{\ulcorner \alpha \urcorner}) \wedge \varphi(\overline{\ulcorner \beta \urcorner}).$$

Proof. We first note that from properties (1) and (2) it follows that if $\mathrm{PA} \vdash \alpha \to \beta$, then $\mathrm{PA} \vdash \varphi(\overline{\ulcorner \alpha \urcorner}) \to \varphi(\overline{\ulcorner \beta \urcorner})$. We use property (1) to obtain $\mathrm{PA} \vdash \varphi(\overline{\ulcorner \alpha \to \beta \urcorner})$. Then by property (2) we have $\mathrm{PA} \vdash \varphi(\overline{\ulcorner \alpha \to \beta \urcorner}) \wedge \varphi(\overline{\ulcorner \alpha \urcorner}) \to \varphi(\overline{\ulcorner \beta \urcorner})$. So by combining the two we have $\mathrm{PA} \vdash \varphi(\overline{\ulcorner \alpha \urcorner}) \to \varphi(\overline{\ulcorner \beta \urcorner})$. We will use this property quite a few times in this proof.

Since $\mathrm{PA} \vdash \alpha \wedge \beta \to \alpha$ and $\mathrm{PA} \vdash \alpha \wedge \beta \to \beta$, we have:

$$\mathrm{PA} \vdash \varphi(\overline{\ulcorner \alpha \wedge \beta \urcorner}) \to \varphi(\overline{\ulcorner \alpha \urcorner})$$

and

$$\mathrm{PA} \vdash \varphi(\overline{\ulcorner \alpha \wedge \beta \urcorner}) \to \varphi(\overline{\ulcorner \beta \urcorner}).$$

So we conclude that

$$\mathrm{PA} \vdash \varphi(\overline{\ulcorner \alpha \wedge \beta \urcorner}) \to \varphi(\overline{\ulcorner \alpha \urcorner}) \wedge \varphi(\overline{\ulcorner \beta \urcorner}).$$

To prove the converse we note that $\mathrm{PA} \vdash \alpha \to (\beta \to \alpha \wedge \beta)$, thus:

$$\mathrm{PA} \vdash \varphi(\overline{\ulcorner \alpha \urcorner}) \to \varphi(\overline{\ulcorner \beta \to \alpha \wedge \beta \urcorner}).$$

Hence

$$\mathrm{PA} \vdash \varphi(\overline{\ulcorner \alpha \urcorner}) \wedge \varphi(\overline{\ulcorner \beta \urcorner}) \to \varphi(\overline{\ulcorner \beta \to \alpha \wedge \beta \urcorner}) \wedge \varphi(\overline{\ulcorner \beta \urcorner}) \to \varphi(\overline{\ulcorner \alpha \wedge \beta \urcorner}),$$

where we used property (2) again for the last step. ■

Theorem 4.2. *Given an $\mathcal{L}_{PA}$-formula φ that satisfies the same properties as* Th *in proposition 4.1, and any $\mathcal{L}_{PA}$-sentence ψ such that* $\mathrm{PA} \vdash \psi \leftrightarrow \neg\varphi(\overline{\ulcorner \psi \urcorner})$ *we have that*

$$\mathrm{PA} \vdash \psi \leftrightarrow \neg\varphi(\overline{\ulcorner \bot \urcorner}).$$

Proof. We have that $\mathrm{PA} \vdash \bot \to \psi$, so $\mathrm{PA} \vdash \varphi(\overline{\ulcorner \bot \urcorner}) \to \varphi(\overline{\ulcorner \psi \urcorner})$. By the assumption on ψ we then have $\mathrm{PA} \vdash \psi \to \neg\varphi(\overline{\ulcorner \psi \urcorner}) \to \neg\varphi(\overline{\ulcorner \bot \urcorner})$.

For the converse we use the assumption on ψ to see that:

$$\mathrm{PA} \vdash \varphi(\overline{\ulcorner \psi \urcorner}) \to \varphi(\overline{\ulcorner \neg\varphi(\overline{\ulcorner \psi \urcorner})\urcorner}).$$

Also by property (3) we have that $\mathrm{PA} \vdash \varphi(\overline{\ulcorner \psi \urcorner}) \to \varphi(\overline{\ulcorner \varphi(\overline{\ulcorner \psi \urcorner})\urcorner})$. Combining the two using lemma 4.2 we have:

$$\mathrm{PA} \vdash \varphi(\overline{\ulcorner \psi \urcorner}) \to \varphi(\overline{\ulcorner \neg\varphi(\overline{\ulcorner \psi \urcorner}) \wedge \varphi(\overline{\ulcorner \psi \urcorner})\urcorner}),$$

which is just $\mathrm{PA} \vdash \varphi(\overline{\ulcorner \psi \urcorner}) \to \varphi(\overline{\ulcorner \bot \urcorner})$. Taking the contraposition and using the assumption on ψ again we finally obtain

$$\mathrm{PA} \vdash \neg\varphi(\overline{\ulcorner \bot \urcorner}) \to \neg\varphi(\overline{\ulcorner \psi \urcorner}) \to \psi.$$

Which concludes our proof. ■

Theorem 4.3 (Gödel's second incompleteness theorem)**.**

$$\mathrm{PA} \nvdash \mathrm{Con(PA)}$$

Proof. Apply Gödel's first incompleteness theorem (theorem 4.1) to obtain a sentence G that is independent of PA and satisfies $\mathrm{PA} \vdash G \leftrightarrow \neg\,\mathrm{Th}(\overline{\ulcorner G \urcorner})$. We then apply theorem 4.2 with G in the role of ψ and Th in the role of φ to see that:

$$\mathrm{PA} \vdash G \leftrightarrow \mathrm{Con(PA)}.$$

Now, if PA would prove its own consistency it would also prove G. But G is independent of PA, so PA does not prove its own consistency. ■

5 Rosser sentences

In the proof of Gödel's first incompleteness theorem we reasoned that since $\mathrm{PA} \vdash \mathrm{Th}(\overline{\ulcorner G \urcorner})$ we must have $\mathcal{N} \models \mathrm{Th}(\overline{\ulcorner G \urcorner})$, which is valid because $\mathcal{N}$ is a model of PA. However, this proof cannot be used for consistent extensions of PA. For example, the theory $\mathrm{PA} \cup \{\neg G\}$ is a consistent extension of PA but $\mathcal{N}$ is not a model of it since $\mathcal{N} \models G$.

Using the so called Rosser trick it is possible to prove Gödel's first incompleteness theorem for any consistent extension of PA, using only its consistency and Σ_1-completeness (theorem 3.4). For this Rosser used a notion of one proof coming before another. To formalize this, we will introduce some new notation. The following definitions are taken from [2], and slightly adjusted to fit the context of this paper.

Definition 5.1. Let φ and ψ be $\mathcal{L}_{PA}$-sentences of the forms $\exists x \varphi'(x)$ and $\exists y \psi'(y)$ respectively. The we define $\varphi \preceq \psi$ to be an abbreviation of:

$$\exists x(\varphi'(x) \wedge \forall y < x \neg \psi'(y)),$$

and $\varphi \prec \psi$ will be an abbreviation of:

$$\exists x(\varphi'(x) \wedge \forall y \leq x \neg \psi'(y)).$$

Furthermore we let $\varphi \succeq \psi$ mean $\psi \preceq \varphi$ (likewise for $\succ$ and $\prec$), and $\varphi \equiv \psi$ means $\varphi \preceq \psi \wedge \psi \preceq \varphi$.

Recall that $\mathrm{Th}(\overline{\ulcorner \varphi \urcorner})$ is just short for $\exists x \overline{\mathrm{Prf}}(x, \overline{\ulcorner \varphi \urcorner})$. So we can apply this notation to the sentences $\mathrm{Th}(\overline{\ulcorner \varphi \urcorner})$ and $\mathrm{Th}(\overline{\ulcorner \neg \varphi \urcorner})$, getting $\mathrm{Th}(\overline{\ulcorner \varphi \urcorner}) \preceq \mathrm{Th}(\overline{\ulcorner \neg \varphi \urcorner})$. This gives a notion of a proof of φ 'coming before' a proof of $\neg\varphi$. This of course depends on how one codes proofs. We have established that here in the primitive recursive predicate Prf.

From here on it will no longer be necessary to fix a certain proof predicate. We will however need to formally define what we see as a proof predicate.

Definition 5.2. A Σ_1-sentence $\mathrm{Th}(x)$ is a *standard proof predicate* for an extension PA^+ of PA, if it satisfies the following properties for any $\mathcal{L}_{PA}$-sentences φ and ψ and any Σ_1-sentence σ:

(D1) If $\mathrm{PA}^+ \vdash \varphi$, then $\mathrm{PA}^+ \vdash \mathrm{Th}(\overline{\ulcorner \varphi \urcorner})$;

(D2) $\mathrm{PA}^+ \vdash \mathrm{Th}(\overline{\ulcorner \varphi \to \psi \urcorner}) \wedge \mathrm{Th}(\overline{\ulcorner \varphi \urcorner}) \to \mathrm{Th}(\overline{\ulcorner \psi \urcorner})$;

(D3) $\mathrm{PA}^+ \vdash \sigma \to \mathrm{Th}(\overline{\ulcorner \sigma \urcorner})$ (formalized Σ_1-completeness).

Note that we do not exclude the case that $\mathrm{PA}^+ = \mathrm{PA}$.

Clearly our usual proof predicate is a standard proof predicate, by proposition 4.1. Even though property (3) of 4.1 is not exactly formalized Σ_1-completeness, we have seen in its proof that our usual proof predicate possesses that property. So in what follows we can assume that at least one standard proof predicate exists. If we are talking about an extension PA^+ of PA, we can modify our proof predicate to accept the extra axioms of PA^+.

Let Th be any standard proof predicate for some extension PA^+ of PA (with possibly $\mathrm{PA}^+ = \mathrm{PA}$). Define the formula $\rho_{\mathrm{Th}}(x)$ to be

$$\mathrm{Th}(\langle \overline{10}, x\rangle) \preceq \mathrm{Th}(x).$$

So when φ is some $\mathcal{L}_{PA}$-sentence, $\rho_{\mathrm{Th}}(\overline{\ulcorner \varphi \urcorner})$ is

$$\mathrm{Th}(\overline{\ulcorner \neg\varphi \urcorner}) \preceq \mathrm{Th}(\overline{\ulcorner \varphi \urcorner}).$$

Definition 5.3. A *Rosser sentence* (for the standard proof predicate Th) is a sentence R such that

$$\mathrm{PA}^+ \vdash R \leftrightarrow \rho_{\mathrm{Th}}(\overline{\ulcorner R \urcorner}).$$

Where PA^+ is some extension of PA or PA itself, and Th is a standard proof predicate for PA^+.

Note that by the diagonalization lemma, lemma 4.1, every standard proof predicate has at least one Rosser sentence.

Essentially a Rosser sentencce asserts that a proof of its negation occurs before a proof of the sentence itself. It is exactly this property that allows us to prove Gödel's first incompleteness theorem using Rosser sentences.

Theorem 5.1 (Gödel's first incompleteness theorem using Rosser sentences)**.** *For any consistent extension* PA^+ *of* PA *we can find a Rosser sentence* R *that is independent of* PA^+*.*

Proof. Let Th^+ be a standard proof predicate for PA^+, we write ρ for ρ_{Th^+}. Note that because Th^+ is a Σ_1-sentence, it is of the form $\exists y \overline{Prf^+}(y, x)$ for some formula $\overline{Prf^+}(y, x)$. This means that $\overline{Prf^+}(y, x)$ actually says "y is the code of a proof in PA^+ for the formula coded by x". Now let R be a Rosser sentence for Th^+ , we claim that R is independent of PA^+.

Suppose that $\mathrm{PA}^+ \vdash R$, then we must have $\mathrm{PA}^+ \nvdash \neg R$ because PA^+ is consistent. This means that we have some natural number coding a proof for R in PA^+ and no natural number coding a proof of $\neg R$ in PA^+. Thus the sentence

$$\exists x \overline{Prf^+}(x, \overline{\ulcorner R \urcorner}) \wedge \forall y \leq x \neg \overline{\mathrm{Prf}^+}(y, \overline{\ulcorner \neg R \urcorner})$$

is true in $\mathcal{N}$. This is by definition exactly $\mathrm{Th}^+(\overline{\ulcorner R \urcorner}) \prec \mathrm{Th}^+(\overline{\ulcorner \neg R \urcorner})$. As this is a true Σ_1-sentence in $\mathcal{N}$, we can use Σ_1-completeness (theorem 3.4) to see that it holds in PA and hence in PA^+. Because

$$\begin{aligned}
\mathrm{PA}^+ \vdash \mathrm{Th}^+(\overline{\ulcorner R \urcorner}) \prec \mathrm{Th}^+(\overline{\ulcorner \neg R \urcorner}) \quad &\rightarrow \\
\neg(\mathrm{Th}^+(\overline{\ulcorner \neg R \urcorner}) \prec \mathrm{Th}^+(\overline{\ulcorner R \urcorner})) \quad &\rightarrow \\
\neg\rho(\overline{\ulcorner R \urcorner}) \quad &\rightarrow \\
\neg R, &
\end{aligned}$$

we have that $\mathrm{PA}^+ \vdash \neg R$ which gives a contradiction.

Now suppose that $\mathrm{PA}^+ \vdash \neg R$, then by consistency of PA^+ we have $\mathrm{PA}^+ \nvdash R$. Thus by the same reasoning as above we have a Σ_1-sentence that is true in $\mathcal{N}$:

$$\mathrm{Th}^+(\overline{\ulcorner \neg R \urcorner}) \prec \mathrm{Th}^+(\overline{\ulcorner R \urcorner}),$$

which is just $\rho(\overline{\ulcorner R \urcorner})$. Therefore $\mathrm{PA}^+ \vdash \rho(\overline{\ulcorner R \urcorner})$ and thus $\mathrm{PA}^+ \vdash R$ which again gives a contradiction. ■

6 Modal logic for Rosser sentences

As we have seen in section 5, Rosser sentences are quite an important tool when reasoning about PA and its extensions. This can lead to questions about these very Rosser sentences. One of those questions is whether or not all Rosser sentences are equivalent. This question is answered in section 7. To answer such questions we would like to have some sort of logic system in which we can reason about Rosser sentences. In this section we will show how one can develop such a system.

Definition 6.1. The language $\mathcal{L}_L$ consists of the standard boolean connectives $(\neg, \wedge, \vee, \rightarrow, \leftrightarrow)$, the one place connective $\Box$, the constants $\bot$ and $\top$ and countably many atoms $p_1, p_2, \ldots$.

The theory L is defined in the language $\mathcal{L}_L$ and has the following axiom schemes. Here A and B are $\mathcal{L}_L$ formulas.

(L1) All boolean tautologies (like $A \vee \neg A$).

(L2) $\Box(A \rightarrow B) \wedge \Box A \rightarrow \Box B$.

(L3) $\Box A \rightarrow \Box\Box A$.

(L4) $\Box(\Box A \rightarrow A) \rightarrow \Box A$.

It also has the following rule: $A/\Box A$. That is, whenever A is provable, $\Box A$ also is provable.

In modal logic $\Box A$ normally means "necessarily A", we will however read it a bit differently. For us it will mean "A is provable". This is made precise in the following definition.

Definition 6.2. An *interpretation* is a map $(\cdot)^*$ that assigns to each atom of $\mathcal{L}_L$ an $\mathcal{L}_{PA}$-formula, extended to assign an $\mathcal{L}_{PA}$-formula to each $\mathcal{L}_L$-formula while satisfying (for any two $\mathcal{L}_L$-formulas A, B):

(1) $\bot^*$ is $\bot$, and $\top^*$ is $\neg\bot$;

(2) $(A \wedge B)^*$ is $A^* \wedge B^*$ (and likewise for all other boolean connectives);

(3) $\Box A$ is $\mathrm{Th}(\overline{\ulcorner A^* \urcorner})$.

Where Th is some fixed standard proof predicate. So in fact, the interpretation $(\cdot)^*$ is based on Th.

The definitions above give us a way to reason about provability using modal logic. However, we want to reason about Rosser sentences. So we will need to introduce some notion of a proof occuring before another proof. To this end we will again use the symbols $\prec$ and $\preceq$.

Definition 6.3. We extend $\mathcal{L}_L$ to $\mathcal{L}_R$ by adding the symbols $\prec$ and $\preceq$. Also, the following formation rule is added in $\mathcal{L}_R$: $\Box A \prec \Box B$ and $\Box A \preceq \Box B$ are well formed formulas whenever A and B are. The Σ-*formulas* of $\mathcal{L}_R$ are those with principal connective $\Box$, $\prec$ or $\preceq$.

Our definition of an interpretation $(\cdot)^*$ naturally extends to $\mathcal{L}_R$ formulas by adding the requirement $(\Box A \preceq \Box B)^* = (\Box A)^* \preceq (\Box B)^*$ and likewise for $\prec$.

Lemma 6.1. *For any interpretation $(\cdot)^*$ we have that if A is a Σ-formula, then A^* is a Σ_1-formula.*

Proof. There are three possible principal connectives for A, namely $\Box$, $\prec$ or $\preceq$. This means that the interpretation A^* is of one of the following forms $\mathrm{Th}(\overline{\ulcorner\varphi\urcorner})$, $\mathrm{Th}(\overline{\ulcorner\varphi\urcorner}) \prec \mathrm{Th}(\overline{\ulcorner\psi\urcorner})$ or $\mathrm{Th}(\overline{\ulcorner\varphi\urcorner}) \preceq \mathrm{Th}(\overline{\ulcorner\psi\urcorner})$ for some formulas φ and ψ. All of these are Σ_1-formulas, so A^* is a Σ_1-formula. ■

One might ask if the following holds: if every interpretation of a formula A is a Σ_1-formula, then A is equivalent to some Σ-formula. This is however not the case. A counterexample to this would be $\Box A \prec \Box B \wedge \Box C \prec \Box D$ for distinct A, B, C, D. Clearly every interpretation of this formula is a Σ_1-formula, but it can never be equivalent to a Σ-formula.

6.1 The theories R^- and R

Definition 6.4. The theory R^- in the language $\mathcal{L}_R$ has the same rules as L and has the following axiom schemes.

(R1) All schemes from L, but then for all $\mathcal{L}_R$-formulas.

(R2) $A \to \Box A$, for every Σ-formula A.

(R3) The order axioms for $\preceq$ and $\prec$ (see below).

The theory R is R^- together with the rule $\Box A/A$.

To talk about the difference between R^- and R we first need one more definition.

Definition 6.5. A *derived rule* of some theory T is a rule which can be derived from the axioms and rules already present in that theory. So A/B is a derived rule of T if $T \vdash A \Rightarrow T \vdash B$.

We can actually show that R^- and R are different because $\Box A/A$ is not a derived rule of R^-. We will do this in subsection 6.3.

The *order axioms* mentioned in definition 6.4 are as follows. The first four schemes express that $\preceq$ pre-orders true formulas (keep in mind that all these formulas must have principal connective $\Box$):

$$A \to A \preceq A,$$
$$A \preceq B \to A,$$
$$A \preceq B \wedge B \preceq C \to A \preceq C,$$
$$A \vee B \to A \preceq B \vee B \prec A.$$

The next two schemes state that $\prec$ is the strict pre-order associated with $\preceq$:

$$A \prec B \to A \preceq B,$$
$$A \preceq B \to \neg(B \prec A).$$

The last scheme requires that all true formulas must occur before false ones:

$$A \wedge \neg B \to A \prec B.$$

We are now ready to state the main result of this section.

Theorem 6.1 (Arithmetical completeness of R)**.**

$$R \vdash A \iff \textit{for every interpretation } (\cdot)^*,\ \mathrm{PA} \vdash A^*.$$

Proof. One can prove

$$R \vdash A \Rightarrow \text{for every interpretation } (\cdot)^*,\ \mathrm{PA} \vdash A^*$$

right now. We will not fully write down the proof here, because it just comes down to checking that the set of theorems of PA is closed under the translated rules of R. This is done in [3] (lemma 2.3) for the theory L. Proving this for R just requires to additionally check for scheme (2), the rule $\Box A/A$ and the order axioms (refering to the numbering from definition 6.4).

Scheme (2) is just the translated version of formalized Σ_1-completeness. The rule $\Box A/A$ also holds for PA, because if $\mathrm{PA} \vdash (\Box A)^*$ we have $\mathcal{N} \models (\Box A)^*$ which is just $\mathcal{N} \models \mathrm{Th}(\overline{\ulcorner A^* \urcorner})$. This means that there is a proof for A^* in PA. The order axioms are easily checked with our definition of $\preceq$ and $\prec$ in PA.

Proving the converse requires a lot more work. We will focus on doing so in the rest of this section. ■

6.2 Kripke models

Kripke models are for modal logic what truth tables are for propositional logic. These models define which formulas are valid and which are not. In this subsection we will define Kripke models for R^-.

We will first define a *frame*. A frame is an irreflexive tree-like partial order $\lhd$ on some finite set. Because it is a tree-like order, we will call the elements of the finite set *nodes* (even though a common term for Kripke models would be worlds here). The root of the tree is the 'largest' element, that is: we have $b \lhd a$ whenever b is a descendant of a.

Definition 6.6. A *Kripke model* (for R^-) is a pair $(\lhd, \Vdash)$, where $\lhd$ is a frame and $\Vdash$ is a relation between the nodes of the frame and formulas of $\mathcal{L}_R$, satisfying the following.

(1) The relation $\Vdash$ respects boolean operations, for example: if a is a node, $a \Vdash A \wedge B$ iff $a \Vdash A$ and $a \Vdash B$. In particular, this also holds for negation: $a \nVdash A$ iff $a \Vdash \neg A$.

(2) We have that $\Box A$ holds at node a iff A holds at all of the descendants of a. That is, $a \Vdash \Box A$ iff for all $b \lhd a$, $b \Vdash A$.

(3) We have Σ-persistence, thus if A is a Σ-formula and $a \Vdash A$ for some node a, then for all $b \lhd a$ we have $b \Vdash A$.

(4) We require each of the order axioms (definition 6.4, item (3)) to hold at every node.

We read $a \Vdash A$ as "A holds at a", and we say that A is *valid in* $(\lhd, \Vdash)$ if A holds at every node. If A does not hold at some node of $(\lhd, \Vdash)$, we call $(\lhd, \Vdash)$ a *countermodel to* A.

To get a bit of an idea of how these models work, and because we will need it later, we will state a small lemma here. This lemma is not stated in [2], it is made up for this paper.

Lemma 6.2. *Every Kripke model is a countermodel to $\neg\Box A$ for every formula A.*

Proof. Let $(\lhd, \Vdash)$ be any Kripke model. Since it is finite, it must have a leaf node a. As there is no node b such that $b \lhd a$, we have $\forall b \lhd a(b \Vdash A)$ for every formula A. Hence we must have that $a \Vdash \Box A$, and so we have $a \nVdash \neg\Box A$. So $\neg\Box A$ does not hold at a and the Kripke model is thus a countermodel to $\neg\Box A$. ■

Kripke models have to respect boolean operations. So whether or not a certain formula holds at a certain node completely depends on whether or not its subformulas hold at that node. One would like to describe only a part of a Kripke model, namely the part that says something about a formula we are interested in, and then extend that to a complete Kripke model.

Definition 6.7. A set S is called *adequate* if it closed under subformulas and contains $\Box A \preceq \Box B$ and $\Box A \prec \Box B$ whenever it contains $\Box A$ and $\Box B$.

An *S-model* is a pair $(\lhd, \Vdash)$ such that requirements (1) - (4) from definition 6.6 hold for the formulas in S. An *A-model* is then an S-model where S is the smallest adequate set containing A.

We can now formalize that a part of a Kripke model can be extended to an actual Kripke model.

Lemma 6.3. *If S is adequate and $(\lhd, \Vdash)$ is an S-model, then there exists a relation $\Vdash'$ such that $(\lhd, \Vdash')$ is a Kripke model and $\Vdash$ and $\Vdash'$ agree on the formulas of S.*

Proof. We will not provide a complete proof here, but we will give an idea of why this would work. As we have stated before, what a Kripke model decides about a certain formula depends entirely on its subformulas. So to extend an S-model to a Kripke model, we only need to define what that Kripke model decides about the atoms that do not occur in S. Whether or not these atoms hold at a certain node does not influence whether or not the formulas in S hold at that node. An easy construction would be to let every atom that does not occur in S hold at every node. Of course, then some construction is necessary to make sure requirements (2) - (4) from definition 6.6 are met, but that is easy to do. ■

6.3 Completeness of R^-

In this subsection we will prove that the Kripke models we defined for R^- actually describe the valid formulas of R^-. This is formalized in the following theorem.

Theorem 6.2 (Completeness of R^-)**.** *For all $\mathcal{L}_R$-formulas A we have:*

$$R^- \vdash A \iff (1)$$
$$A \text{ is valid in all Kripke models} \iff (2)$$
$$A \text{ holds at the root in every Kripke model.} \quad (3)$$

In proving this theorem we will use a result from [3], namely corollary 1. This states that for any $\mathcal{L}_L$-formula A such that $L \nvdash A$, there is a Kripke model for L such that A does not hold at some node in that model. A Kripke model for L is a pair $(\lhd, \Vdash)$ such that (1) and (2) from definition 6.6 hold.

If $(\lhd, \Vdash)$ is a Kripke model (either for R^- or L), and a is some node in that model, then the restriction to a and all of its descendants is also a Kripke model. So when given a countermodel to some formula A, there is some node a where A does not hold. Restricting this countermodel to a and its descendants gives us a Kripke model where A does not hold at the root.

Proof of theorem 6.2. We first prove (1) $\Rightarrow$ (2). Let T be the set of the formulas that are valid in all models. We will show that this set contains all the instances of the axioms of R, and is closed under the rules of R. This is actually an extension of the proof of theorem 3.5 in [3], where a similar result was proved for the theory L. We will start with the axiom schemes from L.

(L1) Because every model has to respect boolean relations, boolean tautologies have to be valid in every model.

(L2) Suppose that $\Box(A \to B) \wedge \Box A$ holds at node a, then $A \to B \wedge A$ must hold at every descendant of a. So B must hold at every descendant of A, and therefore $\Box B$ holds at a.

(L3) Suppose $\Box A$ holds at node a, then A must hold at all of its descendants. So for each descendant b of a we have that A holds for all the descendants of b. So $\Box A$ holds at b, but that means that $\Box A$ holds at all the descendants of A. Thus $\Box\Box A$ holds at a.

(L4) Suppose that there is a model $(\lhd, \Vdash)$ where $\Box(\Box A \to A) \to \Box A$ does not hold at some node a (for some formula A). Then we must have that $a \Vdash \Box(\Box A \to A)$ and $a \nVdash \Box A$. Now let X be the set of all descendants of a where A does not hold. It must be non-empty because otherwise we would have $a \Vdash \Box A$. Let b be a $\lhd$-minimal element of X. For any $c \lhd b$ we have that $c \notin X$ thus $c \Vdash A$, so we have that $b \Vdash \Box A$. Because $a \Vdash \Box(\Box A \to A)$ and $b \lhd a$, we must have $b \Vdash \Box A \to A$. Since also $b \Vdash \Box A$ we must have $b \Vdash A$, which contradicts $b \in X$. So there can be no countermodel to $\Box(\Box A \to A) \to \Box A$.

That T is closed under the rule $A/\Box A$, we can see easily. For let $A \in T$, then it holds at every node in every model. So for any node a we have that A holds at all of its descendants, and thus $\Box A$ holds at a.

It now remains to show that T also contains all instances of (R2) and (R3). It is clear that T contains all instances of (R3) as these are the order axioms and are by definition required to hold at every node. For (R2) we let A be any Σ-formula, and suppose it holds at node a. By Σ-persistence we then must have that A holds at every descendant of a, and thus $\Box A$ holds at a.

The implication (2) $\Rightarrow$ (3) is direct. For the implication (3) $\Rightarrow$ (1) we will prove $\neg(1) \Rightarrow \neg(3)$ instead, as in theorem 3.1 of [2]. So assume that $R^- \nvdash A$. As we discussed before starting the proof of this theorem, a likewise result was already proved for L. We will aim to use that result by translating the $\mathcal{L}_R$-formula A to some $\mathcal{L}_L$-formula A'. For this, let S be an adequate set containing A and let $D_1, \ldots, D_n$ be the formulas of S with principal connective $\preceq$ or $\prec$. Let $p_1, \ldots, p_n$ be distinct atoms of $\mathcal{L}_L$, that do not occur in S. For any $\mathcal{L}_R$-formula B, the $\mathcal{L}_L$-formula B' will be the result of substituting p_i for D_i throughout B.

For any formula B we define $\boxdot B$ to be $B \wedge \Box B$. We let X be the set that contains the following $\mathcal{L}_L$-formulas:

- $\boxdot(p_i \rightarrow \Box p_i)$, for all $1 \leq i \leq n$;
- $\boxdot B'$, for each order axiom B only involving formulas of S.

The set X actually holds the translated version of the axioms of R^- that L is missing. Because the axioms of R^- are schemes it satisfies the rule $B(p)/B(C)$ for any two formulas B, C and any atom p. This means that since $R^- \nvdash A$, we must have $R^- \nvdash A'$, from which it follows that $L \nvdash \wedge X \rightarrow A'$ (here $\wedge X$ is the conjunction of all formulas in X).

We can now apply the result we discussed before we started this proof to find a Kripke model $(\lhd, \Vdash)$ such that $\wedge X \wedge \neg A'$ holds at its root. From this we will construct an S-model $(\lhd, \Vdash')$. We keep the same nodes and the same ordering $\lhd$ on them, and we define $\Vdash'$ as follows. For any $B \in S$ and any node a we let $\Vdash'$ be such that

$$a \Vdash' B \iff a \Vdash B'.$$

That this is indeed an S-model follows from the fact that $(\lhd, \Vdash)$ is a Kripke model for L. So it only remains to check (3) and (4) of definition 6.6. These easily follow from the fact that $\wedge X$ holds at the root of $(\lhd, \Vdash)$, which means $\wedge X$ is valid in $(\lhd, \Vdash)$ because for any formula B we have: if $\boxdot B$ holds at the root then B holds at every node.

To conclude we can now apply lemma 6.3 to extend $(\lhd, \Vdash')$ to a Kripke model for R^- where A does not hold at the root. ■

As was promised in subsection 6.1 we will show here that $\Box A/A$ is not a derived rule of R^-. We will not really need this result later on, but it is a nice example of how Kripke models and theorem 6.2 can be used.

Corollary 6.1. *The rule $\Box A/A$ is not a derived rule of R^-.*

Proof. Before really starting our proof we note that for any A we have

$$R^- \vdash \Box\bot \rightarrow \Box A.$$

This can be seen quickly by applying the rule $A/\Box A$ to obtain $R^- \vdash \Box(\bot \rightarrow A)$, after which we use axiom (L2) from definition 6.1.

We will now aim to show that $R^- \vdash \Box(\Box\top \prec \Box\bot)$ and then that $R^- \nvdash \Box\top \prec \Box\bot$. For that we reason in R^- by cases. Again using the rule $A/\Box A$ we have that $\Box\top$ holds. Now suppose that $\neg\Box\bot$ holds, then by the order axioms we have $\Box\top \prec \Box\bot$. This is a Σ-formula, so by axiom (R2) it follows that $\Box(\Box\top \prec \Box\bot)$ holds. If $\Box\bot$ would hold, then we can use the remark at the start

of this proof with $\Box\top \prec \Box\bot$ in the role of A to find $\Box(\Box\top \prec \Box\bot)$. So in either case $\Box(\Box\top \prec \Box\bot)$ holds.

To show that $R^- \nvdash \Box\top \prec \Box\bot$ we will use theorem 6.2 by giving a countermodel to $\Box\top \prec \Box\bot$. It will be enough to specify an S-model, where S is the smallest adequate set containing $\Box\top \prec \Box\bot$. Let $\lhd$ be an order only containing the point a. Then as a is a leaf node we have $a \Vdash \Box\top$ and $a \Vdash \Box\bot$, so we can set $a \Vdash \Box\bot \prec \Box\top$. By the order axioms we then must have $a \nVdash \Box\top \prec \Box\bot$. Extending this S-model to a Kripke model gives us a countermodel to $\Box\top \prec \Box\bot$ and thus by theorem 6.2 we have $R^- \nvdash \Box\top \prec \Box\bot$. ■

6.4 Completeness of R

As with R^- we would like to describe the valid formulas of R using Kripke models. In this subsection we will do so. To that end we will first need a few defintions.

Definition 6.8. For any $\mathcal{L}_R$-sentence A we let $\mathcal{S}A$ be

$$\wedge\{\Box B \to B : \Box B \text{ is a subformula of } A\}.$$

We call a Kripke model A-*sound* if $\mathcal{S}A$ holds at its root.

This enables us to define the *soundness rule*, which is the counterpart in $\mathcal{L}_R$ to the Σ_1-completeness of PA.

Definition 6.9 (Soundness rule)**.** For all Σ-formulas A:

$$(\mathcal{S}A \to A)/A.$$

For any interpretation $(\cdot)^*$ this is already a derived rule of PA. Because suppose that $(\mathcal{S}A \to A)^*$ is provable in PA, then it must be true in $\mathcal{N}$. Since $(\mathcal{S}A)^*$ is always true in $\mathcal{N}$, we have that A^* must be true in $\mathcal{N}$. Because A is a Σ-formula, we have that A^* is a Σ_1-formula and by Σ_1-completeness it is thus provable in PA.

The soundness rule happens to be a derived rule of R as well. However, seeing that requires a bit more work. It is actually a corollary to a more important theorem that expresses the completeness of R like theorem 6.2 did for R^-.

In [2] a new theory R^+ is introduced, which is R together with the soundness rule. We will however take a different approach (as is also suggested in [2]), which allows us to omit everything about R^+.

Theorem 6.3 (Completeness of R)**.** *For any $\mathcal{L}_R$-formula A we have:*

$$R \nvdash A \iff \textit{there is an } A\textit{-sound countermodel to } A.$$

Proof. For the implication from the left to the right we can use the proof from [2]. Suppose that $R \nvdash A$. For any integer n we mean by $\Box^n A$ that A is prefixed by n times a $\Box$. Because of the rule $\Box A/A$ we must have $R \nvdash \Box^n A$ for all n.

We define X to be the set

$$\{\Box B : \Box B \text{ is a subformula of } A\},$$

and N is the cardinality of this set. Because $R \nvdash \Box^{N+1}A$ and thus $R^- \nvdash \Box^{N+1}A$, we can use theorem 6.2 to find a countermodel $(\lhd, \Vdash)$ to $\Box^{N+1}A$. That means that there must be a sequence of nodes $a_{N+1} \lhd a_N \lhd \ldots \lhd a_1 \lhd a_0$ such that

$$a_0 \Vdash \neg\Box^{N+1}A, a_1 \Vdash \neg\Box^N A, \ldots, a_{N+1} \Vdash \neg A.$$

Now define X_i to be $\{\Box B \in X : a_i \Vdash \Box B\}$, then by Σ-persistence we have for $i \leq j$ that $X_i \subseteq X_j$. So we can apply the pigeonhole principle to find an i with $X_i = X_{i+1}$. This means that for every $\Box B \in X_{i+1}$ we also have that $a_i \Vdash \Box B$, thus $a_{i+1} \Vdash B$. So for all $\Box B \in X$ we must have $a_{i+1} \Vdash \Box B \to B$, which is the same as $a_{i+1} \Vdash \mathcal{S}A$. Restricting $(\lhd, \Vdash)$ to a_{i+1} and all of its descendants gives us an A-sound countermodel to A.

For the converse we will apply a lemma we state and prove later, lemma 6.9. It is here that we deviate from [2]. Given an A-sound countermodel to A, this lemma gives us an interpretation $(\cdot)^*$ such that $\mathrm{PA} \nvdash A^*$. Furthermore, we have already proved the implication

$$R \vdash A \Rightarrow \text{for every interpretation } (\cdot)^*,\ \mathrm{PA} \vdash A^*$$

from theorem 6.1. By assumption we have an A-sound countermodel to A, so we apply lemma 6.9 to obtain an interpretation $(\cdot)^*$ such that $\mathrm{PA} \nvdash A^*$. Using the contraposition of the implication we just mentioned, we find $R \nvdash A$. ■

We can now show that the soundness rule is actually a derived rule of R. The proof of this corollary gives us a nice example on how to use the completeness of R.

Corollary 6.2. *The soundness rule is a derived rule of R.*

Proof. Let A be a Σ-formula such that $R \vdash \mathcal{S}A \to A$, and suppose for a contradiction that $R \nvdash A$. Then by theorem 6.3 we find an A-sound countermodel $(\lhd, \Vdash)$ to A. As A is a Σ-formula, it cannot hold at the root because of Σ-persistence. Note that $\mathcal{S}(\mathcal{S}A \to A)$ is equivalent to $\mathcal{S}A$, so $(\lhd, \Vdash)$ is also $(\mathcal{S}A \to A)$-sound. Thus $\mathcal{S}A \to A$ must hold at every node, because otherwise we would have an $(\mathcal{S}A \to A)$-sound countermodel to $\mathcal{S}A \to A$. In particular it must hold at the root, so we have that both $\mathcal{S}A \to A$ and $\mathcal{S}A$ hold at the root which implies that A holds at the root. This gives us a contradiction, so we must have $R \vdash A$. ■

6.5 Arithmetical completeness of R

One way to describe a proof predicate is by using a total recursive function. We can do this as follows, for a total recursive function f we will think of $f(n)$ as a code for the set of theorems proved by proofs with code $\leq n$. For brevity's sake we will no longer speak about "a code for a set" or "a code for a proof", but we will just speak about a "set" or a "proof".

To actually define a proof predicate based on a total recursive function f we will need some Σ_1-formula $\mathrm{Th}_f(x)$ that essentially says $\exists y(x \in f(y))$. We could use theorem 3.7, which gives a Σ_1-formula $\varphi_f(y, x)$ representing f. This easily gives us a predicate $\mathrm{Th}_f(x)$. However, we would like to base this proof predicate

on the index of f so we can apply the recursion theorem. In [2] (section 5) it is mentioned that one can find a Σ_1-formula $\sigma(e,x)$ such that the following lemma holds.

Lemma 6.4. *We can find a Σ_1-formula $\sigma(e,x)$, such that for any index e,* PA *proves the following.*

(1) $\bigcup \text{range}(\varphi_e) = \{x : \sigma(e,x)\}$.

(2) For all $x, y \in \bigcup \text{range}(\varphi_e)$ we have that there is some n such that $x \in \varphi_e(n)$ and $y \notin \varphi_e(m)$ for all $m \leq n$, if and only if $\sigma(e,x) \prec \sigma(e,y)$.

Proof. It is not mentioned in [2] how one would construct such a formula $\sigma(e,x)$, so we will do so here. For this it will be useful to understand intuitively what the formula would mean. We want $\sigma(e,x)$ to hold whenever there is a proof of x according to φ_e. Since $\sigma(e,x)$ has to be a Σ_1-formula, it must be of the form $\exists y P(y,e,x)$ where we can interpret $P(y,e,x)$ as "according to φ_e there is proof $\leq y$ of x". For this we define the partial recursive function $p(y,e,x)$ as follows:

$$p(y,e,x) = \begin{cases} 0 & \text{if } x \in \varphi_e(y) \\ \text{undefined} & \text{else} \end{cases}$$

Note that with $x \in \varphi_e(y)$ we actually mean the partial recursive function that checks if x is an element of the finite set coded by $\varphi_e(y)$. Now $p(y,e,x)$ is defined (and is thus 0) exactly when there is proof $\leq y$ of x according to φ_e. If we let τ be the Σ_1-formula representing the Kleene T-predicate (as in [1]), we can now define $P(y,e,x)$ to be

$$\exists z \tau(\overline{3}, \overline{e_p}, j^3(y,e,x), z),$$

where e_p is the index of the function p we defined before. Now clearly $P(y,e,x)$ is a Σ_1-formula, so $\exists y P(y,e,x)$ is also a Σ_1-formula and we have found the desired definition for $\sigma(e,x)$. Verifying properties (1) and (2) that we mentioned above is easy now. ■

We can now rightfully speak about a proof predicate based on an index e of some total recursive function f. When we speak about the proof predicate Th_f based on some total recursive function f, we actually mean that it is based on the index e of f. Such a proof predicate is not necessarily a standard proof predicate in the sense of definition 5.2. We will however describe a way of constructing f so that we can prove in PA that Th_f is in fact a standard proof predicate, and so that it has some other useful properties we will state and prove later.

Constructing f and an interpretation $(\cdot)^*$ based on f. We will now construct a total recursive function f and an interpretation $(\cdot)^*$ based on f, given an A-sound countermodel to some $\mathcal{L}_R$-sentence A and a standard proof predicate Th. An interpretation $(\cdot)^*$ based on f is just an intertpretation such that $(\Box A)^*$ is mapped to $\text{Th}_f(\ulcorner A^* \urcorner)$.

So assume that $(\lhd, \Vdash)$ is an A-sound countermodel to A, and label its nodes $\{1, \ldots, n\}$ such that 1 is the root. For a finite adequate set containing A, we let S be that set together with the negations of the formulas in that set. We also

create a new node 0 that is not contained in the model and let $0 \Vdash B$ iff $1 \Vdash B$ for all $B \in S$.

Because Th is a standard proof predicate, it is a Σ_1-formula. Therefore the set $\{x : \mathrm{Th}(x)\}$ is recursively enumerable. So there must be a total recursive function g enumerating the set $\{x : \mathrm{Th}(x)\}$. So we can use $\exists y(x = g(y))$ to formalize "there is a proof of x".

Before continuing we will define a primitive recursive function $h : \mathbb{N} \to \{0, \ldots, n\}$ here and some term l in PA. Note that we will actually see h as a map from the natural numbers to the nodes of the Kripke model (together with 0).

Definition 6.10. To define a primitive recursive function $h : \mathbb{N} \to \{0, \ldots, n\}$ and some term l in PA we will use the recursion theorem, so in fact we define them based on some index e and let the recursion theorem provide a suitable index. Basically we will define h such that $h(0) = 0$ and $h(m+1) \trianglelefteq h(m)$, and l as the limit of h (i.e. $l = \lim_{m\to\infty} h(m)$) if that exists or $n+1$ otherwise.

So we set $h(0) = 0$. Now suppose that $h(m)$ is defined, we check if $g(m)$ is $\ulcorner l \neq \overline{j} \urcorner$ for some $j \triangleleft h(m)$ (or in the case $h(m) = 0$, for some $1 \leq j \leq n$). If this is the case, we set $h(m+1) = j$, otherwise we set $h(m+1) = h(m)$. This completes the definition of h and l.

We will now give the definition of f and the interpretation $(\cdot)^*$ based on f. More precisely, we will actually define a function $F(e, x)$ and the interpretation will be based on the function defined by the index e. Then we can use the recursion theorem to find the desired function f and intepretation $(\cdot)^*$.

First we define $(\cdot)^*$ for the atoms of $\mathcal{L}_R$. Let the atoms be $p_1, p_2, \ldots$, then if $p_k \in S$:

$$p_k^* = \vee\{l = \overline{i} : i \Vdash p_k \text{ and } 0 \leq i \leq n\} \wedge \overline{k} = \overline{k},$$

which is just $\bot \wedge \overline{k} = \overline{k}$ if p_k holds at no node. If $p_k \notin S$ we just set

$$p_k^* = l = 0 \wedge \overline{k} = \overline{k}.$$

Now no matter the index of f, we have that $(\cdot)^*$ is one-to-one, that $\{A^* : A \text{ is an } \mathcal{L}_R\text{-formula}\}$ is recursive and that we can reconstruct A from A^*.

We will define f in multiple stages. At each stage we define one or more outputs of f. We keep track of the amount of f that is defined in k_m: at stage m we have defined $f(n)$ for all $n < k_m$.

Stage m. Each stage will consist of two steps. Roughly speaking we want to output the formula that is coded by $g(m)$, unless it is of the form $\ulcorner B^* \urcorner$ for some $\Box B \in S$. In that case it depends on the value of $h(m)$ and what the node with number $h(m)$ decides about the formulas in S. We will now make this more precise.

Step 1. If $g(m)$ is of the form $\ulcorner B^* \urcorner$ for some $\Box B \in S$ we set $f(k_m) = f(k_m - 1)$, otherwise we set $f(k_m) = f(k_m - 1) \cup \{g(m)\}$. If k_m happens to be 0, then we interpret $f(k_m - 1)$ as the empty set.

Step 2. We only do something in this step when $m = 0$ or when $h(m) \neq h(m-1)$, otherwise we set $k_{m+1} = k_m + 1$ and go to stage $m+1$. We let Y be the set of B^* such that $\Box B \in S$ and $h(m) \Vdash \Box B$ and, if $m > 0$, $h(m-1) \nVdash \Box B$. We define the equivalence relation E on Y such that $B^* E C^*$ iff $h(m) \Vdash \Box B \preceq \Box C \wedge$

$\Box C \preceq \Box B$. Now let $E_1, \ldots, E_s$ be the equivalence classes of Y, in increasing order under $\prec$. For each $1 \leq i \leq s$ we set

$$f(k_m + i) = f(k_m) \cup E_1 \cup \ldots \cup E_i.$$

Now set $k_{m+1} = k_m + s + 1$ and go to stage $m + 1$.

This completes the definition of f.

Before we can actually do something useful with f, we first have to prove a few technical lemmas. In what follows we will use the same notation as in the definition of f.

Lemma 6.5. *For the term l from definition 6.10 we have the following properties:*

(1) $\mathrm{PA} \vdash \overline{0} \leq l \leq \overline{n}$,

(2) $\mathrm{PA} \vdash l = \overline{i} \to \mathrm{Th}(\ulcorner l \lhd \overline{i} \urcorner)$ *for all* $1 \leq i \leq n$,

(3) $\mathrm{PA} \vdash l = \overline{i} \to \neg\, \mathrm{Th}(\ulcorner l \neq \overline{j} \urcorner)$ *for all* $j \lhd i$ *(or* $1 \leq j \leq n$ *in case* $i = 0$*)*,

(4) $\mathcal{N} \models l = 0$.

Proof. We will only provide a sketch of the proof, for the full proof we refer to [3], lemma 4.1. Property (1) can easily be seen using the property that $h(m + 1) \trianglelefteq h(m)$ and the fact that $h(m)$ is defined for all m and $h(m) \leq n$ for all m.

For property (2) we note that PA proves that $h(m') \trianglelefteq h(m)$ for all m and $m' \geq m$. So $l = \overline{i}$ implies that there is some m such that $h(m) = \overline{i}$ so PA proves $l \trianglelefteq \overline{i}$ which is equivalent to $l = \overline{i} \vee l \lhd \overline{i}$. Since there is some m such that $h(m) = \overline{i}$ we have that PA proves $l \neq \overline{i}$, so PA must prove $l \lhd \overline{i}$.

For property (3) we reason in PA, assuming $l = \overline{i}$. If there would be some proof of $l \neq \overline{j}$ for $j \lhd i$ (or $1 \leq j \leq n$ in case $i = 0$), then we would have $l \trianglelefteq \overline{j}$ and thus $l \neq \overline{i}$. Which is a contradiction, so $l \neq \overline{j}$ is not provable.

Finally for (4), if $l = i > 0$ were to hold in $\mathcal{N}$ there must be some proof of $l \neq \overline{i}$ in PA. But then $\mathcal{N} \models l \neq i$, which gives a contradiction. So we must have $\mathcal{N} \models l = 0$. ■

Lemma 6.6. *For $B \in S$ and $0 \leq i \leq n$ we have:*

$$i \Vdash B \Rightarrow \mathrm{PA} \vdash l = \overline{i} \to B^*,$$
$$i \nVdash B \Rightarrow \mathrm{PA} \vdash l = \overline{i} \to \neg B^*.$$

Proof. We will prove this using induction on formulas. For atoms this lemma is obviously true, and the induction step for the boolean connectives is immediate.

If B is of the form $\Box C$ and $i \Vdash B$, then we reason in PA and assume $l = \overline{i}$. Since $l = \overline{i}$ there must be a smallest m such that $h(m) = i$. Then since $i \Vdash \Box C$ we have that f outputs C^* at stage m, thus $(\Box C)^*$ holds. Now if $i \nVdash \Box C$, then $\Box C$ holds at no parent of i. Since we assume $l = \overline{i}$ we must have $i \trianglelefteq h(m)$ for all m. So there is not stage at which C^* will be output by f, thus $(\neg \Box C)^*$ holds.

If B is of the form $\Box C \prec \Box D$, we again reason in PA. Assume that $i \Vdash \Box C \prec \Box D$ we must have $i \Vdash \Box C$. Thus C^* is ouput by f. Let m be the first stage where C^* is output by f. We must have $i \trianglelefteq h(m)$, so by Σ-persistence we cannot have $h(m) \Vdash \Box D \preceq \Box C$. This means that at stage m we either have that D^* is not output at all or that it is output after C^*. In both cases we have $(\Box C \prec \Box D)^*$. The case where B is of the form $\Box C \preceq \Box D$ is similar.

In the case that $i \nVdash \Box C \prec \Box D$ we note that $\neg(\Box C \prec \Box D)$ and $\neg\Box C \vee \Box D \preceq \Box C$ are equivalent. So simply applying the arguments above we see that the lemma also holds in this case. Finally, if we have $i \nVdash (\Box C \preceq \Box D)$ we can again use similar reasoning by noting that $\neg(\Box C \preceq \Box D)$ and $\neg\Box C \vee \Box D \prec \Box C$ are equivalent. ■

Corollary 6.3. *If $B \in S$ and $1 \Vdash B$, then $\mathcal{N} \models B^*$.*

Proof. Since $B \in S$ and $1 \Vdash B$ we must have $0 \Vdash B$. So using lemma 6.6 we find $\mathrm{PA} \vdash l = \overline{0} \to B^*$, so $\mathcal{N} \models l = 0 \to B^*$. By lemma 6.5 we have $\mathcal{N} \models l = 0$, so we conclude that $\mathcal{N} \models B^*$. ■

Lemma 6.7. *For all $\Box B \in S$ and all $0 \leq i \leq n$, we have:*

$$i \Vdash \Box B \Rightarrow \mathrm{PA} \vdash l = \overline{i} \to \mathrm{Th}(\ulcorner B^* \urcorner),$$
$$i \nVdash \Box B \Rightarrow \mathrm{PA} \vdash l = \overline{i} \to \neg\,\mathrm{Th}(\ulcorner B^* \urcorner).$$

Proof. Assume that $1 \leq i \leq n$ and suppose $i \Vdash \Box B$, then for all $j \lhd i$ we have $j \Vdash B$. As there are only finitely many such j, we can apply lemma 6.6 to find $\mathrm{PA} \vdash l \lhd \overline{i} \to B^*$ and therefore $\mathrm{PA} \vdash \mathrm{Th}(\ulcorner l \lhd \overline{i} \to B^* \urcorner)$. By lemma 6.5 we also have $\mathrm{PA} \vdash l = \overline{i} \to \mathrm{Th}(\ulcorner l \lhd \overline{i} \urcorner)$, so we have:

$$\mathrm{PA} \vdash l = \overline{i} \to (\mathrm{Th}(\ulcorner l \lhd \overline{i} \urcorner) \wedge \mathrm{Th}(\ulcorner l \lhd \overline{i} \to B^* \urcorner)),$$

and thus $\mathrm{PA} \vdash l = \overline{i} \to \mathrm{Th}(\ulcorner B^* \urcorner)$.

Now, still assuming $1 \leq i \leq n$, suppose that $i \nVdash \Box B$. Then there must be some $j \lhd i$ such that $j \Vdash \neg B$. Using lemma 6.6 we find $\mathrm{PA} \vdash l = \overline{j} \to \neg B^*$. Thus $\mathrm{PA} \vdash \mathrm{Th}(\ulcorner l = \overline{j} \to \neg B^* \urcorner)$, and thus $\mathrm{PA} \vdash \mathrm{Th}(\ulcorner B^* \to l \neq \overline{j} \urcorner)$. By lemma 6.5 we also have that $\mathrm{PA} \vdash l = \overline{i} \to \neg\,\mathrm{Th}(\ulcorner l \neq \overline{j} \urcorner)$. So, reasoning in PA under assumption that $l = \overline{i}$, we cannot have $\mathrm{Th}(\ulcorner B^* \urcorner)$ because then we would have both $\mathrm{Th}(\ulcorner l \neq \overline{j} \urcorner)$ and $\neg\,\mathrm{Th}(\ulcorner l \neq \overline{j} \urcorner)$. Thus we must have $\neg\,\mathrm{Th}(\ulcorner B^* \urcorner)$, which leads to the conclusion $\mathrm{PA} \vdash l = \overline{i} \to \neg\,\mathrm{Th}(\ulcorner B^* \urcorner)$.

Now assume that $i = 0$, and suppose that $0 \Vdash \Box B$. Because $(\lhd, \Vdash)$ is A-sound we have $i \Vdash \Box B \to B$, and thus $0 \Vdash B$. Furthermore, since 0 and 1 agree on all formulas in S we also have $1 \Vdash B$ and $1 \Vdash \Box B$. Hence $i \Vdash B$ for all $0 \leq i \leq n$, so by lemma 6.6 we have $\mathrm{PA} \vdash l = \overline{i} \to B^*$ for all $0 \leq i \leq n$. Since $\mathrm{PA} \vdash \overline{0} \leq i \leq \overline{n}$ by lemma 6.5, we must have $\mathrm{PA} \vdash B^*$ and thus $\mathrm{PA} \vdash \mathrm{Th}(\ulcorner B^* \urcorner)$. So we certainly have $\mathrm{PA} \vdash l = \overline{0} \to \mathrm{Th}(\ulcorner B^* \urcorner)$.

The last possibility is $i = 0$ and $0 \nVdash \Box B$. This means that there is some $1 \leq j \leq n$ such that $j \nVdash B$. Thus by lemma 6.6 we have $\mathrm{PA} \vdash \mathrm{Th}(\ulcorner l = \overline{j} \to \neg B^* \urcorner)$, and thus $\mathrm{PA} \vdash \mathrm{Th}(\ulcorner B^* \to l \neq \overline{j} \urcorner)$. Since also $\mathrm{PA} \vdash l = \overline{0} \to \neg\,\mathrm{Th}(\ulcorner l \neq \overline{j} \urcorner)$, we can again reason in PA under the assumption that $l = \overline{0}$ to find that $\neg\,\mathrm{Th}(\ulcorner B^* \urcorner)$ must hold. That is, $\mathrm{PA} \vdash l = \overline{0} \to \neg\,\mathrm{Th}(\ulcorner B^* \urcorner)$. ■

We now can finally prove that Th_f is indeed a standard proof predicate. We do so by showing that f outputs exactly the formulas that are theorems of PA according to the standard proof predicate Th.

Lemma 6.8.

$$\mathrm{PA} \vdash \forall x(\mathrm{Th}(x) \leftrightarrow \mathrm{Th}_f(x))$$

Proof. This amounts to showing that $\bigcup \mathrm{range}(f)$ and $\mathrm{range}(g)$ are the same. We reason in PA. By the construction of f the only formulas where these sets may not agree are those of the form B^* with $\Box B \in S$.

Since $\mathrm{PA} \vdash \overline{0} \leq l \leq \overline{n}$ we can reason in PA by cases. Using lemma 6.7 and the fact that S is finite, we see that PA proves for each B^* such that $\Box B \in S$ that $\mathrm{Th}(\ulcorner B^* \urcorner)$ exactly when $l \Vdash \Box B$ and thus when f outputs B^*. ■

Lemma 6.9.

$$\mathrm{PA} \nvdash A^*.$$

Proof. For all $0 \leq i \leq n$ we have that $\mathrm{PA} + l = \overline{i}$ is consistent. Since $(\lhd, \Vdash)$ is a countermodel to A, we must have $i \Vdash \neg A$ for some $1 \leq i \leq n$. So by lemma 6.6 we have $\mathrm{PA} \vdash l = \overline{i} \to \neg A^*$. But then if $\mathrm{PA} \vdash A^*$ we would have that $\mathrm{PA} + l = \overline{i}$ proves both A^* and $\neg A^*$, which contradicts its consistency. So we must have $\mathrm{PA} \nvdash A^*$. ■

We can now finally prove the arithmetical completeness of R.

Conclusion of the proof of theorem 6.1. The following was left to prove:

$$\text{for every interpretation } (\cdot)^*,\ \mathrm{PA} \vdash A^* \Longrightarrow R \vdash A.$$

We complete this proof by proving the contraposition of this statement. So suppose that $R \nvdash A$, then by theorem 6.3 we find an A-sound countermodel to A. We can use this countermodel to construct a total recursive function f as we described in this subsection. For the interpretation $(\cdot)^*$ based on f we have by lemma 6.9:

$$\mathrm{PA} \nvdash A^*.$$

This concludes our proof. ■

7 Equivalence of Rosser sentences

In this section we will present the final result we discuss in this paper. We will show that there are standard proof predicates where not all Rosser sentences are provably equivalent, but also that there are standard proof predicates where all Rosser sentences are provably equivalent. However, this still does not say anything about the usual proof predicate we defined and used in section 4.

We will base our proofs on existing standard proof predicates to construct new standard proof predicates. These new proof standard predicates should be equivalent in the sense that they prove the same formulas. This is made precise in the following definition.

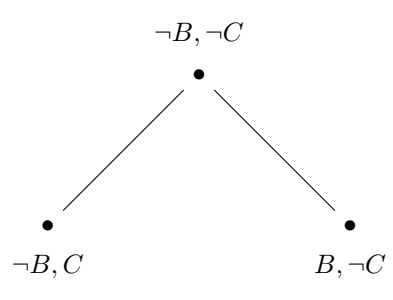
Definition 7.1. Two standard proof predicates Th and Th$'$ are *provably equivalent* if

$$\mathrm{PA} \vdash \forall x(\mathrm{Th}(x) \leftrightarrow \mathrm{Th}'(x)).$$

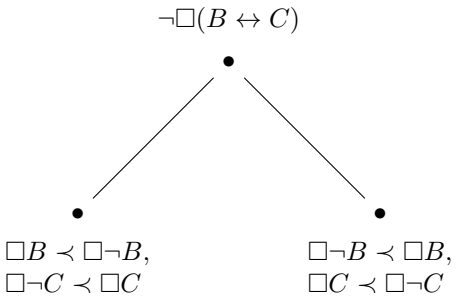
7.1 Inequivalent Rosser sentences

Theorem 7.1. *Given a standard proof predicate* Th*, we can find a provably equivalent standard proof predicate* Th$'$ *so that not all Rosser sentences for* Th$'$ *are provably equivalent.*

Proof. Let $B = \Box\bot \prec \Box(\bot \wedge \bot)$ and $C = \Box(\bot \wedge \bot) \prec \Box\bot$. We will describe a Kripke model in stages. First we make sure that the nodes satisfy the formulas as they do in figure 2a. We must then have $\neg\Box B$, $\neg\Box\neg B$, $\neg\Box C$ and $\neg\Box\neg C$ at the root. So we can extend the Kripke model so that the formulas in figure 2b hold at the corresponding nodes.



(a) Kripke model (1).



(b) Kripke model (2).

Figure 2: Kripke models for inequivalent rosser sentences.

Now define $A = \Box(B \leftrightarrow \Box\neg B \prec \Box B) \wedge \Box(C \leftrightarrow \Box\neg C \prec \Box C) \wedge \neg\Box(B \leftrightarrow C)$. By inspecting the Kripke model we specified we find that it is A-sound. Also by lemma 6.2 we have that it is a countermodel to $\neg\Box(B \leftrightarrow C)$ and thus to A. So we have an A-sound countermodel to A, therefore we can use corollary 6.3 to see that $\mathcal{N} \models A^*$ because A holds at the root. It is here that we use our standard proof predicate Th to construct a provably equivalent standard proof predicate Th$'$ and an interpretation $(\cdot)^*$ based on Th$'$ (Th and Th$'$ are provably equivalent by lemma 6.8).

This means that $\mathcal{N} \models (\Box(B \leftrightarrow \Box\neg B \prec \Box B))^*$, so

$$\mathcal{N} \models \mathrm{Th}'(\ulcorner B^* \leftrightarrow \mathrm{Th}'(\ulcorner\neg B^*\urcorner) \prec \mathrm{Th}'(\ulcorner B^*\urcorner)\urcorner)$$

and thus

$$\mathrm{PA} \vdash B^* \leftrightarrow \mathrm{Th}'(\ulcorner\neg B^*\urcorner) \prec \mathrm{Th}'(\ulcorner B^*\urcorner).$$

In other words, B^* is a Rosser sentence for Th$'$. In the same way C^* is a Rosser sentence for Th$'$. Now finally, because $\mathcal{N} \models \neg\,\mathrm{Th}'(B^* \leftrightarrow C^*)$ we have that $\mathrm{PA} \nvdash B^* \leftrightarrow C^*$ so B^* and C^* are not provably equivalent. ∎

7.2 Equivalent Rosser sentences

Our goal in this subsection will be to prove the following theorem. Before stating it however, we will need to extend our definition of standard proof predicate with two more properties. For a standard proof predicate Th, these properties are as follows:

(+1) PA proves that "$\{x : \mathrm{Th}(x)\}$ is closed under tautological consequence and it contains all Σ_1-sentences φ such that $\mathcal{N} \models \varphi$".

(+2) PA proves that for any formula $\varphi(x, y)$ we have "if $\mathrm{Th}(\ulcorner \forall xy \varphi(x, y) \urcorner)$, then for all constant terms t, s we have $\mathrm{Th}(\ulcorner \varphi(t, s) \urcorner)$.".

Note that the usual proof predicate possesses these properties. In [2] only the first property is required. However, as is pointed out in [4] this is not enough. So we have required one more property as is suggested in [4]. This will only make a difference in the proof of lemma 7.1 (lemma 6.3 in [2]).

Theorem 7.2. *Given a standard proof predicate* Th *also satisfying (+1) and (+2), we can find a provably equivalent standard proof predicate* Th' *so that all Rosser sentences for* Th' *are provably equivalent.*

Before proving theorem 7.2 we will once more describe a recursive function f using the recursion theorem, so that f defines a standard proof predicate Th_f (like we did in subsection 6.5). For this we again assume that g is a recursive function that, provably in PA, enumerates $\{x : \mathrm{Th}(x)\}$.

Definition of f based on Th**.** We will keep track of two things. One is the so called *Rosser list*, essentially this list keeps track of the sentences of which we have already seen that they are a Rosser sentence. The second thing is a bell that can ring. This bell will ring at most once, and it does so when we encounter the proof or disproof of a Rosser sentence. We will now give the precise definition of f in stages. This definition is in fact based on some index e for f that is provided by the recursion theorem.

Stage m. Each stage consists of two steps. The first step is always executed, the second step is only executed if the bell rings in the first step. As will become clear, once the bell has rung we will no longer go to the next stage. In other words: we only consider the stages in which the bell has not yet rung and the stage in which it rings (if it rings).

Step 1. If $g(m)$ is of the form $\ulcorner \varphi \urcorner$ or $\ulcorner \neg \varphi \urcorner$ for some φ on the Rosser list, we ring the bell and go directly to step 2. Otherwise we set $f(m) = g(m)$ and we take another look at $g(m)$. If $g(m)$ is $\ulcorner \varphi \leftrightarrow \mathrm{Th}_f(\ulcorner \neg \varphi \urcorner) \prec \mathrm{Th}_f(\ulcorner \varphi \urcorner) \urcorner$ for some φ then we add φ to the Rosser list, unless $\neg\varphi$ is already on the Rosser list or φ is $\neg\psi$ and ψ is already on the Rosser list. This ensures that at most one of φ and $\neg\varphi$ is on the Rosser list for any φ. Now we go to stage $m + 1$.

Step 2. If we have come here the bell has just rung. Suppose the Rosser list consists of $\{\varphi_1, \ldots, \varphi_n\}$. We distinguish two cases. In the first case $g(m) = \ulcorner \varphi_i \urcorner$ for some $1 \leq i \leq n$. Then we define $f(m), \ldots, f(m + 2n - 1)$ to be $\ulcorner \varphi_1 \urcorner, \ldots, \ulcorner \varphi_n \urcorner, \ulcorner \neg\varphi_1 \urcorner, \ldots, \ulcorner \neg\varphi_n \urcorner$ (in that order). In the second case we have $g(m) = \ulcorner \neg\varphi_i \urcorner$ for some $1 \leq i \leq n$. We then define $f(m), \ldots, f(m + 2n - 1)$ to be $\ulcorner \neg\varphi_1 \urcorner, \ldots, \ulcorner \neg\varphi_n \urcorner, \ulcorner \varphi_1 \urcorner, \ldots, \ulcorner \varphi_n \urcorner$ (in that order). In both cases we continue by enumerating all $\mathcal{L}_{PA}$-sentences.

This completes the definition of f, which is to be formalized in PA.

Before finally proving theorem 7.2 we will first prove three lemmas about f. In what follows we will use the terms and definitions from the definition of f.

Lemma 7.1. PA *proves that "if the bell rings, then* $\{x : \mathrm{Th}(x)\}$ *is inconsistent, that is* $\mathrm{Th}(\ulcorner\bot\urcorner)$*".*

Proof. We reason in PA. Suppose that the bell rings at stage m because $g(m) = \ulcorner\varphi\urcorner$ for some φ on the Rosser list. If $g(i) = \ulcorner\neg\varphi\urcorner$ for some $i < m$ then we have $\mathrm{Th}(\ulcorner\varphi\urcorner)$ and $\mathrm{Th}(\ulcorner\neg\varphi\urcorner)$, so clearly $\mathrm{Th}(\ulcorner\bot\urcorner)$ (by lemma 4.2). Otherwise we have $g(i) \neq \ulcorner\neg\varphi\urcorner$ for all $i < m$. So f outputs $\ulcorner\varphi\urcorner$ before $\ulcorner\neg\varphi\urcorner$, and so we have that

$$\mathrm{Th}_f(\ulcorner\varphi\urcorner) \prec \mathrm{Th}_f(\ulcorner\neg\varphi\urcorner).$$

Which is a Σ_1-sentence, thus $\mathrm{Th}(\ulcorner\mathrm{Th}_f(\ulcorner\varphi\urcorner) \prec \mathrm{Th}_f(\ulcorner\neg\varphi\urcorner)\urcorner)$.

Because φ is on the Rosser list, we must have that g outputs $\ulcorner\varphi \leftrightarrow \mathrm{Th}_f(\ulcorner\neg\varphi\urcorner) \prec \mathrm{Th}_f(\ulcorner\varphi\urcorner)\urcorner$. Thus g must output $\ulcorner\mathrm{Th}_f(\ulcorner\neg\varphi\urcorner) \prec \mathrm{Th}_f(\ulcorner\varphi\urcorner)\urcorner$ at some time.

It is here that we diverge from the proof in [2] and we continue as in [4]. Moving out of PA shortly, we can see that:

$$\mathrm{PA} \vdash \forall xy[\mathrm{Th}_f(x) \prec \mathrm{Th}_f(y) \rightarrow \neg(\mathrm{Th}_f(y) \prec \mathrm{Th}_f(x))].$$

So we must have:

$$\mathrm{PA} \vdash \mathrm{Th}(\ulcorner\forall xy[\mathrm{Th}_f(x) \prec \mathrm{Th}_f(y) \rightarrow \neg(\mathrm{Th}_f(y) \prec \mathrm{Th}_f(x))]\urcorner).$$

We now move back in PA, we can use property (+2) to see that we have $\mathrm{Th}(\ulcorner\mathrm{Th}_f(\ulcorner\varphi\urcorner) \prec \mathrm{Th}_f(\ulcorner\neg\varphi\urcorner) \rightarrow \neg(\mathrm{Th}_f(\ulcorner\neg\varphi\urcorner) \prec \mathrm{Th}_f(\ulcorner\varphi\urcorner))\urcorner)$. Therefore we have both $\mathrm{Th}(\ulcorner\neg(\mathrm{Th}_f(\ulcorner\neg\varphi\urcorner) \prec \mathrm{Th}_f(\ulcorner\varphi\urcorner))\urcorner)$ and $\mathrm{Th}(\ulcorner\mathrm{Th}_f(\ulcorner\neg\varphi\urcorner) \prec \mathrm{Th}_f(\ulcorner\varphi\urcorner)\urcorner)$. So using lemma 4.2 we can conclude that $\mathrm{Th}(\ulcorner\bot\urcorner)$ holds.

The other case, where $g(m) = \ulcorner\neg\varphi\urcorner$ for some φ on the Rosser list, does not use property (+2). We may again assume that $g(i) \neq \ulcorner\varphi\urcorner$ for all $i < m$, because otherwise $\mathrm{Th}(\ulcorner\bot\urcorner)$ would directly follow. This means that $\mathrm{Th}_f(\ulcorner\neg\varphi\urcorner) \prec \mathrm{Th}_f(\ulcorner\varphi\urcorner)$ is a true Σ_1-sentence, and thus we must have:

$$\mathrm{Th}(\ulcorner\mathrm{Th}_f(\ulcorner\neg\varphi\urcorner) \prec \mathrm{Th}_f(\ulcorner\varphi\urcorner)\urcorner).$$

Because φ is on the Rosser list, we must have:

$$\mathrm{Th}(\ulcorner\varphi \leftrightarrow \mathrm{Th}_f(\ulcorner\neg\varphi\urcorner) \prec \mathrm{Th}_f(\ulcorner\varphi\urcorner)\urcorner).$$

Since also $\mathrm{Th}(\ulcorner\neg\varphi\urcorner)$ we have:

$$\mathrm{Th}(\ulcorner\neg(\mathrm{Th}_f(\ulcorner\neg\varphi\urcorner) \prec \mathrm{Th}_f(\ulcorner\varphi\urcorner))\urcorner).$$

So we conclude that both $\mathrm{Th}(\ulcorner\mathrm{Th}_f(\ulcorner\neg\varphi\urcorner) \prec \mathrm{Th}_f(\ulcorner\varphi\urcorner)\urcorner)$ and $\mathrm{Th}(\ulcorner\neg(\mathrm{Th}_f(\ulcorner\neg\varphi\urcorner) \prec \mathrm{Th}_f(\ulcorner\varphi\urcorner))\urcorner)$ hold, so using lemma 4.2 we find $\mathrm{Th}(\ulcorner\bot\urcorner)$. ■

Lemma 7.2.

$$\mathrm{PA} \vdash \mathrm{range}(f) = \mathrm{range}(g).$$

Proof. We reason in PA. If the bell never rings then by construction f outputs the exact same sentences as g. If the bell rings f outputs all $\mathcal{L}_{PA}$-sentences, and so does g by lemma 7.1. ■

Note that lemma 7.2 actually proves that Th_f is a standard proof predicate and that it is provably equivalent to Th.

Lemma 7.3. *If φ is a Rosser sentence for Th_f, then φ is eventually put on the Rosser list.*

Proof. We do not reason in PA now. This means that the bell will never ring. Let φ be a Rosser sentence for Th_f, and assume that it is never put on the Rosser list. Then that must be because there is some Rosser sentence ψ for Th_f such that $\varphi = \neg\psi$ or $\neg\varphi = \psi$. One of these sentences must be true in $\mathcal{N}$, but no Rosser sentence can be true in $\mathcal{N}$. So φ must be on the Rosser list. ■

Conclusion of the proof of theorem 7.2. For any two Rosser sentences φ and ψ we must have, by lemma 7.3, that they are both on the Rosser list at some stage m. We can now reason in PA as follows: neither $\ulcorner\varphi\urcorner$ nor $\ulcorner\psi\urcorner$ has been output by g at a stage $\leq m$ and they are both on the Rosser list. By construction of f we have that f outputs $\ulcorner\neg\varphi\urcorner$ before $\ulcorner\varphi\urcorner$ iff it outputs $\ulcorner\neg\psi\urcorner$ before $\ulcorner\psi\urcorner$. In other words

$$\mathrm{Th}_f(\ulcorner\neg\varphi\urcorner) \prec \mathrm{Th}_f(\ulcorner\varphi\urcorner) \leftrightarrow \mathrm{Th}_f(\ulcorner\neg\psi\urcorner) \prec \mathrm{Th}_f(\ulcorner\psi\urcorner),$$

and thus $\varphi \leftrightarrow \psi$. ■

References

[1] J. van Oosten, *Gödel's Incompleteness Theorems*, Lecture notes, `http://www.staff.science.uu.nl/~ooste110/syllabi/godelmoeder.pdf` February 2015.

[2] D. Guaspari, R.M. Solovay, *Rosser sentences*, Annals of Mathematical Logic, Volume 16, Issue 1, pages 81-99, 6 June 1978.

[3] R.M. Solovay, *Provability interpretations of modal logic*, Isreal Journal of Mathematics, Volume 25, pages 287-304, 1976.

[4] C. von Bülow, *A remark on equivalent Rosser sentences*, Annals of Pure and Applied Logic, Volume 151, pages 62-67, 2008.