CARLETON UNIVERSITY

SCHOOL OF
MATHEMATICS AND STATISTICS

HONOURS PROJECT

TITLE: Gödel's Incompleteness Theorems

AUTHOR: Sharon Pragashkumar

SUPERVISOR: Brandon Fodden

DATE: August 25th, 2020

Acknowledgements

I would first like to express my sincerest gratitude to Brandon Fodden. His guidance and support throughout my project, has allowed me to explore my topic and deepen my understanding. He has been an exceptional supervisor and I am truly grateful for all the effort he has put in to helping me with my project. I would also like to thank my second reader, Kevin Cheung for his valuable suggestions. Finally, my friends and family (and dog, Rocky) have always been by my side encouraging and supporting me in all my endeavours. I am beyond thankful for each of you.

1 Introduction

Axiom systems, such as the Peano Axioms, attempt to characterize a mathematical structure. They allow us to describe mathematical objects and their interactions with each other. A first order theory in a language L consists of axioms and rules of inference of first order logic (see Appendix). We can also have non-logical axioms in a set we will call Γ. A formalist approach to mathematics is one that views mathematics as the study of axiom systems. Historically, the goal of the axiom system is to be able to derive any true sentence but mathematician, Kurt Gödel showed that this is not always possible, as we shall see.

In the years between 1900 and 1930 [1], mathematician David Hilbert began to highlight areas of research in logic. He believed axiom systems were the best way to study different areas in mathematics. Hilbert wanted to prove the existence of a common set of logical axioms for all mathematical reasoning. A few years later, logician and mathematician Kurt Gödel showed that a first order theory with nonlogical axioms Γ can prove a formula A if and only if A is true in every model of Γ. This is called *Gödel's Completeness Theorem* [3].

During this time, people were looking for an axiom system for which arithmetic on $\mathbb{N}$ is a model. Peano Arithmetic (PA) is such a system for arithmetic but one can show that it also has other nonstandard models [1]. By Gödel's Completeness Theorem, PA can prove formulas that are true in every model of PA. But we now have the question of whether PA can prove sentences that are true of arithmetic on $\mathbb{N}$, which is its intended model. If not, would we be able to add more axioms to Γ in order to create an extension of PA that can prove all sentences that are true of arithmetic on $\mathbb{N}$? In particular, is there a consistent extension of PA such that for any sentence A, we have that either A or $\neg A$ is a theorem? If this holds, then the extension is complete. However, Gödel showed that this was not the case:

Gödel's Incompleteness Theorem I [4]

Every consistent axiomatized extension Γ of PA is incomplete.

Gödel's first Incompleteness Theorem places limitations on what we can prove in an axiom system. This paper aims to prove the result and discuss the implications of it. Furthermore, we can ask if an extension of PA can prove its own consistency. Gödel's second Incompleteness Theorem will tell us that the answer to this question is no! We cannot have an axiomatic system which is both consistent and able to prove its own consistency. The theorem is stated here and will be seen in greater detail in Section 6.

Gödel's Incompleteness Theorem II [4]

No consistent, axiomatic extension of PA can prove its own consistency.

2 The formal system NN

We will introduce a formal system for arithmetic called NN where we follow the rules of first-order logic. The language of NN is L_{NN} and the axioms are listed in the Appendix. One can show that this list of axioms is not enough to prove all true statements from arithmetic. As an example, it can be shown that $\mathrm{NN} \vdash 0 \times S0 = 0$ but we can't show $\mathrm{NN} \vdash \forall x_1(0 \times x_1 = 0)$ since NN doesn't have any axioms which express induction, unlike PA. This report will prove Gödel's Incompleteness Theorem I for any extension of NN and this includes PA. We can list some basic properties of NN:

Theorem 2.1. *Let a,$b \in \mathbb{N}$. Then the following hold*

1. *If a equals b, then* $\mathrm{NN} \vdash (0^a = 0^b)$.
2. *If a is not equal to b, then* $\mathrm{NN} \vdash \neg(0^a = 0^b)$.
3. *If a is less than b, then* $\mathrm{NN} \vdash (0^a < 0^b)$.

4. *If a is not less than b, then* $\mathrm{NN} \vdash \neg(0^a < 0^b)$.

5. $\mathrm{NN} \vdash (0^a + 0^b) = 0^{a+b}$.

6. $\mathrm{NN} \vdash (0^a \times 0^b) = 0^{a\times b}$.

7. $\mathrm{NN} \vdash \forall x[(x < 0^a) \vee (x = 0^a) \vee (0^a < x)]$.

8. *If* $\mathrm{NN} \vdash A_x[0], \ldots, \mathrm{NN} \vdash A_x[0^{a-1}]$, *then* $\mathrm{NN} \vdash (x < 0^a) \to A$.

Properties (1)-(6) will be formally proved in the following section and are found in [1]. We will use these properties to help prove Gödel's Incompleteness Theorem. An advantage of NN is that it is finitely axiomatized, although it is a weaker system than PA. However, the axioms of NN were chosen in order to prove Gödel's first Incompleteness theorem. The theorem will hold for any extension of NN.

3 Expressibility

We will require a means of expressing relations on $\mathbb{N}$ in NN. This section will aim to do that.

Definition 3.1. *Let R be an n-ary relation on $\mathbb{N}$. If there is a formula A that has exactly n free variables $x_1, \ldots x_n$ such that for all $a_1, \ldots a_n$ in $\mathbb{N}$,*
***EX1**: If $R(a_1, \ldots a_n)$, then* $\mathrm{NN} \vdash A_{x_1,\ldots,x_n}[0^{a_1}, \ldots, 0^{a_n}]$.
***EX2**: If $\neg R(a_1, \ldots, a_n)$, then* $\mathrm{NN} \vdash \neg A_{x_1,\ldots,x_n}[0^{a_1}, \ldots, 0^{a_n}]$
then R is expressible. *We say that A expresses R.*

This definition explains how we can express a relation with a formula A by replacing the free variables $x_1, \ldots, x_n$ with $0^{a_1}, \ldots, 0^{a_n}$ for $a_1, \ldots, a_n$ in $\mathbb{N}$. In other words, an expressible relation on $\mathbb{N}$ is a relation that can be encoded in NN.

We can now express the relations known as "equality" and "less-than" on $\mathbb{N}$ in NN. Note that this is the proof of Theorem 2.1 for properties (1) through (4) which were:

1. If a equals b, then $\mathrm{NN} \vdash (0^a = 0^b)$.
2. If a is not equal to b, then $\mathrm{NN} \vdash \neg(0^a = 0^b)$.
3. If a is less than b, then $\mathrm{NN} \vdash (0^a < 0^b)$.
4. If a is not less than b, then $\mathrm{NN} \vdash \neg(0^a < 0^b)$.

Proof. The first property states that if a and b are equal natural numbers, then the representation of a and the representation of b in NN are the same. To prove 1, we have:

(1) $\forall x_1(x_1 = x_1)$	REF AX
(2) $0^a = 0^a$	$\forall$-ELIM
(3) $0^a = 0^b$	by assumption of 1, a equals b.

The second property states the opposite, saying that if a and b are not the same natural number, then the representations of a and b in NN are not equal. Since a and b are not the same natural number, let's take a to be greater than b and let $S(b)$ be the sentence "If the number a is greater than the number b, then $\mathrm{NN} \vdash \neg(0^a = 0^b)$". We will use induction on b to prove $S(b)$ holds for all $b \geq 0$:

Base Case: When $b = 0$, $S(0)$ says "If the natural number a is greater than 0, $\mathrm{NN} \vdash \neg(0^a = 0)$." Since we know $a > b$, we have that $a - 1$ is in $\mathbb{N}$ and

(1) $\forall x_1 \neg(Sx_1 = 0)$	NN1
(2) $\neg(S0^{a-1} = 0)$	$\forall - ELIM$
(3) $\neg(0^a = 0)$	notation

Inductive Step: Next, we suppose $b \geq 0$ and that $S(b)$ holds. We will now prove $S(b+1)$ which says "If a is a number greater than $b+1$, then $\mathrm{NN} \vdash \neg(0^a = 0^b)$". Since a is greater than $b+1$, we have that $a-1$ is greater than b which by our induction hypothesis, means NN $\vdash \neg(0^{a-1} = 0^b)$. Then we have the following:

(1) $\neg(0^{a-1} = 0^b)$	THM of NN
(2) $\forall x_1 \forall x_2[(Sx_1 = Sx_2) \rightarrow (x_1 = x_2)]$	NN2
(3) $\forall x_2((S0^{a-1} = Sx_2) \rightarrow (0^{a-1} = x_2))$	$\forall$-ELIM
(4) $(S0^{a-1} = S0^b) \rightarrow (0^{a-1} = 0^b)$	$\forall$-ELIM
(5) $\neg(S0^{a-1} = S0^b)$	MT on (1) and (4)
(6) $\neg(0^a = 0^{b+1})$	notation

Thus, $S(b)$ holds for all $b \geq 0$. But now suppose b is greater than but not equal to a. By the same argument as above (with a and b reversed), we have NN $\vdash \neg(0^b = 0^a)$. Then

(1) $\vdash \neg(0^b = 0^a)$	THM of NN
(2) $(0^a = 0^b) \rightarrow (0^b = 0^a)$	SYM
(3) $\neg(0^a = 0^b)$	MT

Thus, NN $\vdash \neg(0^a = 0^b)$ as needed.

The third property states that if the natural number a is less than the natural number b, then the representation of a in NN is less than the representation of b in NN. However, the proof of the third property uses the fourth, so we will prove the fourth first.

The fourth property states that if the natural number a is not less than the natural number b, then the representation of a is not less than the representation of b. We can let the sentence $S(b)$ state that "If a is not less than b, then NN $\vdash \neg(0^a < 0^b)$". We will use induction on b to show that $S(b)$ holds for all $b \geq 0$.

Base Case: When $b = 0$, $S(b)$ says "If a is not less than 0, NN $\vdash \neg(0^a < 0)$".

(1) $\forall x_1 \neg(x_1 < 0)$	NN7
(2) $\neg(0^a < 0)$	$\forall$-ELIM

Inductive Step: Next, we suppose S(b) holds. We will now prove $S(b+1)$ which says "If $a \geq b+1$, then NN $\vdash \neg(0^a < 0^{b+1})$". Assume $a \geq b+1$ so that we have $a > b$. By the induction hypothesis, this gives $\mathrm{NN} \vdash \neg(0^a < 0^b)$ and by property 2, we have $\mathrm{NN} \vdash \neg(0^a = 0^b)$. To get the desired result, we have:

(1) $\neg(0^a < 0^b)$	THM of NN
(2) $\neg(0^a = 0^b)$	THM of NN
(3) $\forall x_1 \forall x_2[(x_1 < Sx_2) \to ((x_1 < x_2) \vee (x_1 = x_2))]$	NN8
(4) $\forall x_2[(0^a < Sx_2) \to ((0^a < x_2) \vee (0^a = x_2))]$	$\forall$-ELIM
(5) $(0^a < S0^b) \to [(0^a < 0^b) \vee (0^a = 0^b)]$	$\forall$-ELIM
(6) $(0^a < 0^{b+1}) \to [(0^a < 0^b) \vee (0^a = 0^b)]$	notation
(7) $\neg(0^a < 0^{b+1}) \vee [(0^a < 0^b) \vee (0^a = 0^b)]$	DEF of $\to$
(8) $[\neg(0^a < 0^{b+1}) \vee (0^a < 0^b)] \vee (0^a = 0^b)$	ASSOC
(9) $\neg(0^a < 0^{b+1}) \vee (0^a < 0^b)$	DS on (2), (8)
(10) $\neg(0^a < 0^{b+1})$	DS on (1), (9)

Thus, $\mathrm{NN} \vdash \neg(0^a < 0^{b+1})$ as needed.

We can now prove property 3: suppose a is less than b. By property 2, we have $\mathrm{NN} \vdash \neg(0^a = 0^b)$ since a does not equal b and by property 4, we have $\mathrm{NN} \vdash \neg(0^b < 0^a)$ since b is not less than a. We have:

(1) $\neg(0^a = 0^b)$	THM of NN
(2) $\neg(0^b < 0^a)$	THM of NN
(3) $\forall x_1 \forall x_2[(x_1 < x_2) \vee (x_1 = x_2) \vee (x_2 < x_1)]$	NN9
(4) $\forall x_2[(0^b < x_2) \vee (0^b = x_2) \vee (x_2 < 0^b)]$	$\forall$-ELIM
(5) $(0^b < 0^a) \vee (0^b = 0^a) \vee (0^a < 0^b)$	$\forall$-ELIM
(6) $(0^b = 0^a) \vee (0^a < 0^b)$	DS on (2), (5)
(7) $0^a < 0^b$	DS on (1), (6), and SYM

Thus, we have $\mathrm{NN} \vdash 0^a < 0^b$ as required. □

We have now expressed the relations "equality" and "less-than" in NN. However, we can see from the following theorem that not all relations on $\mathbb{N}$ are expressible:

Theorem 3.2. *Assume NN is consistent. Then there exists a 1-ary relation on $\mathbb{N}$ that is not expressible in NN.*

A careful proof of this theorem will not be given here but we can give the general idea. The number of formulas of $\mathrm{L_{NN}}$ is countable and using an argument made by Cantor, it can be shown that the number of 1-ary relations on $\mathbb{N}$, which correspond to subsets of $\mathbb{N}$, is uncountable. Since the number of relations on $\mathbb{N}$ is larger than the number of formulas of $\mathrm{L_{NN}}$, there are not enough formulas to express all relations. This is worrisome but it turns out (seen in the next section) that all the relations which we need to express are indeed expressible!

We can also prove parts 5 and 6 of Theorem 2.1 which are:

5. $\mathrm{NN} \vdash 0^a + 0^b = 0^{a+b}$.
6. $\mathrm{NN} \vdash 0^a \times 0^b = 0^{a \times b}$.

Proof. For property 5, we first note that in $0^a + 0^b$, '$+$' represents the symbol of $\mathrm{L_{NN}}$ and that in 0^{a+b}, '$+$' represents the usual addition of natural numbers. We proceed by induction on b:

Base Case: Let $b = 0$, we should show $\mathrm{NN} \vdash 0^a + 0 = 0^a$:

(1) $\forall x_1(x_1 + 0 = x_1)$	NN3
(2) $0^a + 0 = 0^a$	$\forall$-ELIM

Inductive Step: We now assume $\mathrm{NN} \vdash 0^a + 0^b = 0^{a+b}$ and we need to show $\mathrm{NN} \vdash 0^a + 0^{b+1} = 0^{a+b+1}$:

(1) $\forall x_1 \forall x_2[x_1 + Sx_2 = S(x_1 + x_2)]$	NN4
(2) $\forall x_2[0^a + Sx_2 = S(0^a + x_2)]$	$\forall$-ELIM
(3) $0^a + S0^b = S(0^a + 0^b)$	$\forall$-ELIM
(4) $0^a + 0^{b+1} = S(0^a + 0^b)$	notation
(5) $0^a + 0^b = 0^{a+b}$	induction hypothesis
(6) $S(0^a + 0^b) = S0^{a+b}$	FUNC
(7) $S(0^a + 0^b) = 0^{a+b+1}$	notation
(8) $0^a + 0^{b+1} = 0^{a+b+1}$	TRAN

This completes the proof of property 5.

For property 6, we first note that in $0^a \times 0^b$, '$\times$' represents the symbol of L_{NN} and that in $0^{a\times b}$, '$\times$' represents the usual multiplication of natural numbers.

Base Case: Let $b = 0$. We must show $NN \vdash 0^a \times 0 = 0$.

(1) $\forall x_1(x_1 \times 0 = 0)$ NN5

(2) $0^a \times 0 = 0$ $\forall$-ELIM

Inductive Step: Now we assume $NN \vdash 0^a \times 0^b = 0^{a\times b}$ and we want to show $NN \vdash 0^a \times 0^{b+1} = 0^{a\times b+a}$:

(1) $\forall x_1 \forall x_2[x_1 \times Sx_2 = (x_1 \times x_2) + x_1]$	NN6
(2) $\forall x_2[0^a \times Sx_2 = (0^a \times x_2) + 0^a]$	$\forall$-ELIM
(3) $0^a \times S0^b = (0^a \times 0^b) + 0^a$	$\forall$-ELIM
(4) $0^a \times 0^{b+1} = (0^a \times 0^b) + 0^a$	notation
(5) $0^a \times 0^b = 0^{a\times b}$	induction hypothesis
(6) $(0^a \times 0^b) + 0^a = 0^{a\times b} + 0^a$	FUNC
(7) $0^a \times 0^{b+1} = 0^{a\times b} + 0^a$	TRAN
(8) $0^{a\times b} + 0^a = 0^{a\times b+a}$	by property 5
(9) $0^a \times 0^{b+1} = 0^{a\times b+a}$	TRAN

This completes the proof of Theorem 2.1 for properties (1) to (6). □

The following lemma will be used in a proof in the next section:

Lemma 3.3. *Let s and t be variable-free terms of L_{NN} and let x be a variable of L_{NN}. Then if $NN \vdash s = t$, we have that $NN \vdash (x = s) \leftrightarrow (x = t)$.*

Proof. Let y be a variable of L_{NN} that is different from x. We have the following:

(1) $s = t$	THM of NN
(2) $(y = t) \rightarrow [(x = y) \leftrightarrow (x = t)]$	EQ AX
(3) $(s = t) \rightarrow [(x = s) \leftrightarrow (x = t)]$	SUBST RULE
(4) $(x = s) \leftrightarrow (x = t)$	MP on (1) and (3)

□

4 Recursive and Expressible Relations

We have seen how to express a relation R on $\mathbb{N}$ satisfying both **EX1** and **EX2**. Though the method we used was able to express the equality and less than relations of $\mathbb{N}$ in NN, other more convoluted relations would be trickier and impractical to express with this approach. In addition, we've seen by Theorem 3.2, that there are some relations on $\mathbb{N}$ that are unexpressible. This section will aim to find another way to show the expressibility of relations on $\mathbb{N}$ by proving the following:

Theorem 4.1. *Let R be an n-ary relation on $\mathbb{N}$. If R is recursive, then R is expressible in NN.*

The proof of this theorem relies on background information regarding recursive functions which can be seen in the Appendix. We will first introduce the class of representable functions, then prove that every recursive function is representable, and also prove R is expressible if and only if K_R is representable.

Definition 4.2. *Let F be an n-ary function. F is* representable *if there exists a formula A with exactly $n+1$ free variables such that for all a_i in $\mathbb{N}$:*

$$\text{NN} \vdash A_{x_1,x_2,\ldots,x_n}[0^{a_1}, 0^{a_2}, \ldots, 0^{a_n}] \leftrightarrow (x_{n+1} = 0^{F(a_1,\ldots a_n)}).$$

The first step will be to show that all starting functions are representable:

Theorem 4.3. *Addition and multiplication are representable.*

Proof. First we will prove the formula $x_3 = x_1 + x_2$ represents addition by showing NN $\vdash (x_3 = 0^a + 0^b) \leftrightarrow (x_3 = 0^{a+b})$. Let a and b be numbers in $\mathbb{N}$. By property 5 from Theorem 2.1, we have NN $\vdash 0^a + 0^b = 0^{a+b}$. Then by Lemma 3.3, we can let $s = 0^a + 0^b$ and $t = 0^{a+b}$ and we get NN $\vdash (x_3 = 0^a + 0^b) \leftrightarrow (x_3 = 0^{a+b})$ as needed.

Next, we will prove the formula $x_3 = x_1 \times x_2$ represents multiplication by showing NN $\vdash (x_3 = 0^a \times 0^b) \leftrightarrow (x_3 = 0^{a\times b})$. Let a and b be numbers in $\mathbb{N}$.

By property 6 from Theorem 2.1, we get $\mathrm{NN} \vdash 0^a \times 0^b = 0^{a\times b}$. Then again by Lemma 3.3, we let $s = 0^a \times 0^b$ and $t = 0^{a\times b}$ to get $\mathrm{NN} \vdash (x_3 = 0^a \times 0^b) \leftrightarrow (x_3 = 0^{a\times b})$. □

Theorem 4.4. *The projection functions $U_{n,k}$ are representable.*

Proof. Fix $n, k \in \mathbb{N}$. We will prove the formula $(x_{n+1} = x_k) \wedge (x_1 = x_1) \wedge \cdots \wedge (x_n = x_n)$ represents the projection function $U_{n,k}$ by showing

$$\mathrm{NN} \vdash [(x_{n+1} = 0^{a_k}) \wedge (0^{a_1} = 0^{a_1}) \wedge \cdots \wedge (0^{a_n} = 0^{a_n})] \leftrightarrow (x_{n+1} = 0^{U_{n,k}(a_1,\ldots,a_n)}).$$

This follows easily since we have $U_{n,k}(a_1, \ldots, a_n) = a_k$ by definition, and for each i, REF AX and $\forall$-ELIM yields $0^{a_i} = 0^{a_i}$. □

We still need to show that $K_<$ is representable but first we will prove the following theorem since the result follows from it:

Theorem 4.5. *Let R be an n-ary relation. R is expressible if and only if K_R is representable.*

Proof. We will first prove the forward direction; assume R is expressible, and let A be a formula with free variables $x_1, x_2, \ldots, x_n$ that express R and also let $a_1, a_2, \ldots, a_n$ be natural numbers. We will show that the formula

$$[A \wedge (x_{n+1} = 0)] \vee [\neg A \wedge (x_{n+1} = S0)]$$

represents K_R. To help simplify notation, we denote $A_{x_1,\ldots,x_n}[0^{a_1}, \ldots, 0^{a_n}]$ by B. By the definition of representability (Definition 4.2), we need to prove:

$$\mathrm{NN} \vdash [(B\wedge(x_{n+1} = 0))\vee(\neg B\wedge(x_{n+1} = S0))] \leftrightarrow (x_{n+1} = 0^{K_R(a_1,a_2,\ldots,a_n)}). \qquad (*)$$

To show this, we consider the two cases: (1) $R(a_1, \ldots, a_n)$ and (2) $\neg R(a_1, \ldots, a_n)$:
Case (1): Suppose $R(a_1, \ldots, a_n)$. Then $K_R(a_1, \ldots, a_n) = 0$ so $(*)$ becomes:

$$\mathrm{NN} \vdash [(B \wedge (x_{n+1} = 0)) \vee (\neg B \wedge (x_{n+1} = S0))] \leftrightarrow (x_{n+1} = 0). \qquad (\star)$$

We also know by Definition 3.1, that $\mathrm{NN} \vdash B$ so the proof of $(\star)$ can be shown by the following argument in propositional logic. Let B be p, $(x_{n+1} = 0)$ be q, and let $(x_{n+1} = S0)$ be r. We can then rewrite $(\star)$ as $\mathrm{NN} \vdash [(p \wedge q) \vee (\neg p \wedge r)] \leftrightarrow q$. One can show that

$$p \to ([(p \wedge q) \vee (\neg p \wedge r)] \leftrightarrow q)$$

is a tautology of propositional logic. Since we have $\mathrm{NN} \vdash p$, it follows that $(\star)$ holds.

Case (2): The proof of (2) follows a similar argument except we now have $K_R(a_1, \dots, a_n) = 1$ since $\neg R(a_1, \dots, a_n)$ and from Definition 3.1, we see $\mathrm{NN} \vdash \neg B$. Thus, $(*)$ becomes:

$$\mathrm{NN} \vdash [(B \wedge (x_{n+1} = 0)) \vee (\neg B \wedge (x_{n+1} = S0))] \leftrightarrow (x_{n+1} = S0).$$

We can let p, q, r be the same from part (1) so that the tautology to use now becomes

$$\neg p \to ([(p \wedge q) \vee (\neg p \wedge r)] \leftrightarrow r),$$

which proves part (2) since we have $\mathrm{NN} \vdash \neg p$.

We can now prove the reverse direction. Assume K_R is representable. Then from Definition 4.2, we can let A be a formula with exactly $n + 1$ free variables with $a_1, \dots, a_n$ in $\mathbb{N}$ such that:

$$\mathrm{NN} \vdash A_{x_1, x_2, \dots, x_n}[0^{a_1}, 0^{a_2}, \dots, 0^{a_n}] \leftrightarrow (x_{n+1} = 0^{K_R(a_1, a_2, \dots, a_n)}). \qquad (\bullet)$$

We can show that the formula which expresses R is $A_{x_{n+1}}[0]$ by proving ***EX1*** and ***EX2*** from Definition 3.1.

Let $a_1, \dots, a_n \in \mathbb{N}$ and denote $A_{x_1, \dots, x_n, x_{n+1}}[0^{a_1}, \dots, 0^{a_n}, 0]$ by B.

Proof of *EX1* First assume $R(a_1, \dots, a_n)$ so that $K_R(a_1, \dots, a_n) = 0$. Then $(\bullet)$ becomes:

$$\mathrm{NN} \vdash A_{x_1, x_2, \dots, x_n}[0^{a_1}, 0^{a_2}, \dots, 0^{a_n}] \leftrightarrow (x_{n+1} = 0).$$

Using SUBST RULE to replace x_{n+1} with 0 gives $\mathrm{NN} \vdash B \leftrightarrow (0 = 0)$ which implies $\mathrm{NN} \vdash B$, since $\mathrm{NN} \vdash 0 = 0$.

Proof of *EX2* First assume $\neg R(a_1, \ldots, a_n)$ so that $K_R(a_1, \ldots, a_n) = 1$. Then $(\bullet)$ becomes:

$$\mathrm{NN} \vdash A_{x_1,x_2,\ldots,x_n}[0^{a_1}, 0^{a_2}, \ldots, 0^{a_n}] \leftrightarrow (x_{n+1} = S0).$$

Again, replacing x_{n+1} with 0 gives $\mathrm{NN} \vdash B \leftrightarrow (0 = S0)$. Since $\mathrm{NN} \vdash \neg(0 = S0)$, $\mathrm{NN} \vdash \neg B$. □

Theorem 4.6. *$K_<$ is representable.*

Proof. Since by Theorem 2.1 properties 2 and 4, the relation '$<$' is expressible in NN, $K_<$ is representable by Theorem 4.5. □

We have now seen that all starting functions are representable and can move on to show that the composition of representable functions is representable. In order to do that, we will need the following lemma of first order logic.

Lemma 4.7. *Let B be any formula, $y_1, \ldots, y_k$ be distinct, and $t_1, \ldots, t_k$ be variable free terms. Then,*

$$\vdash \exists y_1 \cdots \exists y_k[(y_1 = t_1) \wedge \cdots \wedge (y_k = t_k) \wedge B] \leftrightarrow B_{y_1,\ldots,y_k}[t_1, \ldots, t_k].$$

This lemma deals with the substitution and though the proof is not stated here, a full proof can be seen in [1] where it is Theorem 3 of Section 5.6.

Theorem 4.8. *The composition of representable functions is representable.*

Proof. Let G be a k-ary representable function and let H_i where $i = 1, \ldots, k$, be n-ary representable functions such that the function F is the composition of G and H_i:

$$F(a_1, \ldots, a_n) = G(H_1(a_1, \ldots, a_n), \ldots, H_k(a_1, \ldots, a_n)).$$

We will show that for free variables $x_1, \ldots, x_{n+1}$, formulas A^i that represent H_i and formula B that represents G, the formula

$$\exists y_1 \cdots \exists y_k (A^1 \wedge \cdots \wedge A^k \wedge B)$$

represents F. Let $x_1, \ldots, x_{n+1}, y_1, \ldots, y_k$ be distinct variables and since H_i is representable, let $x_1, \ldots, x_n, y_i$ be free variables such that for all $a_i \in \mathbb{N}$,

$$\mathrm{NN} \vdash A^i_{x_1,\ldots,x_n}[0^{a_1}, \ldots, 0^{a_n}] \leftrightarrow (y_i = 0^{H_i(a_1,\ldots,a_n)}).$$

Now let A_i denote $A^i_{x_1,\ldots,x_n}[0^{a_1}, \ldots, 0^{a_n}]$ and let $H_i(a_1, \ldots, a_n) = b_i$. Then the above expression turns into:

$$\mathrm{NN} \vdash A_i \leftrightarrow (y_i = 0^{b_i}). \qquad (*)$$

Similarly, since G is a representable function, there exists free variables, $y_1, \ldots, y_k, x_{n+1}$ be such that for all $b_i \in \mathbb{N}$,

$$\mathrm{NN} \vdash B_{y_1,\ldots,y_k}[0^{b_1}, \ldots, 0^{b_k}] \leftrightarrow (x_{n+1} = 0^{G(b_1,\ldots,b_k)}).$$

Letting $G(b_1, \ldots, b_k) = c$, gives

$$\mathrm{NN} \vdash B_{y_1,\ldots,y_k}[0^{b_1}, \ldots, 0^{b_k}] \leftrightarrow (x_{n+1} = 0^c). \qquad (**)$$

By $(*)$, we can replace $y_i = 0^{b_i}$ by A_i and by $(**)$, we can replace $B_{y_1,\ldots,y_k}[0^{b_1}, \ldots, 0^{b_k}]$ with $x_{n+1} = 0^c$. Now using Lemma 4.7, we let $t_i = 0^{b_i}$ to get

$$\mathrm{NN} \vdash \exists y_1 \cdots \exists y_k (A^1 \wedge \cdots \wedge A^k \wedge B) \leftrightarrow (x_{n+1} = 0^c),$$

as required. □

The next theorem is one of the last steps before proving the goal of this section. The proof will be left out due to space as it is tedious but it can be found in [1] as Theorem 6 of Section 7.3.

Theorem 4.9. *Let G be an $(n+1)$-ary function and let F be an n-ary function that is the minimalization of G. If G is representable, then F is representable.*

Theorem 4.10. *If F is a recursive function, then F is representable.*

Proof. Let F be a recursive function. By Theorems 4.3, 4.4, and 4.6, we know the starting functions of F are representable. Then by Theorems 4.8, and 4.9 we have that the composition and minimalization of functions is representable. Thus, F is representable. $\square$

Finally, we can prove our result:

Theorem 4.1. Every recursive relation is an expressible relation.

Proof. Let R be a recursive relation. By the definition of a recursive relation, K_R is a recursive function. By Theorem 4.10 K_R is representable, and so Theorem 4.5 implies R is expressible. $\square$

5 Gödel Coding

L_{NN} was designed to describe statements of arithmetic but Kurt Gödel found an ingenious way to express statements of logic in NN using a special coding system which is often called the *arithmetization of syntax*. It uses an injective function $g : \mathrm{FOR}_{\mathrm{NN}} \longrightarrow \mathbb{N}$ which takes a formula A of L_{NN} and fixes it to a natural number, a, known as the code or *Gödel Number* of A. We will also need $g(A)$ to satisfy two conditions Gö1 and Gö2.

Theorem 5.1. ***Coding of*** L_{NN} *There exists an injective function* $g : \mathrm{FOR}_{\mathrm{NN}} \longrightarrow \mathbb{N}$ *such that the following two conditions are satisfied:*

Gö1*: There exists an algorithm to calculate $g(A)$ for any given $A \in \mathrm{FOR}_{\mathrm{NN}}$.*

Gö2*: There exists an algorithm such that when given a Gödel Number a, the algorithm decides whether there is an $A \in \mathrm{FOR}_{\mathrm{NN}}$ such that $g(A) = a$. If there is such an A, the algorithm will find it.*

Proof. To prove this result, we first show that such an injective function g exists, then we can find the needed algorithms to make sure Gö1 and Gö2 are fulfilled.

In order to construct a function g, we can first build a "symbol number" function SN, which assigns an odd number to each of the symbols of L_{NN}. The values of this function are:

Symbol, s	**SN**(**s**)
(	1
)	3
$\neg$	5
$\vee$	7
$\forall$	9
$=$	11
0	13
S	15
$+$	17
$\times$	19
$<$	21
x_n	$21 + 2n$ with $n = 1, 2, \ldots$

Note since A $\in$ $\mathrm{FOR}_{\mathrm{NN}}$, A can be written as a string of symbols from 1 to n: $s_1 s_2 \cdots s_n$. Also, let $p_1 < \cdots < p_n$ be the first n primes. Then we can define the function g as

$$g(A) = p_1^{SN(s_1)} \times \cdots \times p_n^{SN(s_n)}.$$

To show that g is injective, suppose we have $g(A) = g(B)$ where $A, B \in \mathrm{FOR}_{\mathrm{NN}}$. Since A and B are formulas, we can rewrite A as $s_1 s_2 \cdots s_j$ and B as $b_1 b_2 \cdots b_k$. From above, we have:

$$g(A) = p_1^{SN(s_1)} \times \cdots \times p_j^{SN(s_j)} = p_1^{SN(b_1)} \times \cdots \times p_n^{SN(b_k)} = g(B).$$

By the unique factorization theorem for primes, we have that $j = k$, and $SN(s_m) = SN(b_m)$ for $1 \leq m \leq k$. It then follows that $s_m = b_m$. Thus, $A = B$ so g is injective.

We have completed the construction of our function g and can move on to find an algorithm for Gö1. Given formula A with symbols $s_1 s_2 \cdots s_n$,

1. Find the first n primes in increasing order.
2. Find $SN(s_i)$ using the symbol table, for each symbol in A.
3. Calculate $g(A) = p_1^{SN(s_1)} \times \cdots \times p_n^{SN(s_n)}$.

We now have an algorithm which calculates the value of $g(A)$ for any given formula A of L_{NN}. Lastly, we find an algorithm for Gö2. Note that by the way formulas of NN are constructed, there is an algorithm that will decide whether or not a given string of symbols is a formula. Given $a \in \mathbb{N}$,

1. If $a \leq 1$, go to 7.
2. Write a as a product of its primes in increasing order: $a = p_1^{e_1} \times \cdots \times p_n^{e_n}$.
3. If $p_1, \ldots p_n$ are NOT the first n primes, go to 7.
4. If any of $e_1, \ldots, e_n$ are even, go to 7.
5. Use the symbol table to find $s_1, \ldots, s_n$ such that $SN(s_i) = e_i$ where $1 \leq\ i \leq n$ and write the symbols out as $s_1 s_2 \cdots s_n$.
6. If $s_1 s_2 \cdots s_n$ is not a formula of L_{NN}, go to 7. If so, then A is $s_1 s_2 \cdots s_n$ and $g(A) = a$.
7. No, there is no such formula $A \in \mathrm{FOR}_{\mathrm{NN}}$ such that $g(A) = a$.

Thus, $g : \mathrm{FOR}_{\mathrm{NN}} \longrightarrow \mathbb{N}$ exists and satisfies **Gö1** and **Gö2**. $\square$

We can use g to assign a code to proofs. For the following definitions, let $\Gamma \subseteq \mathrm{FOR}_{\mathrm{NN}}$ and let A be a formula of L_{NN}.

Definition 5.2. *Suppose A has a proof $A_1, \ldots, A_n$ using Γ. Then $b \in \mathbb{N}$ is called the* code *of a proof of A using Γ if:*

$$b = p_1^{g(A_1)} \times \cdots \times p_n^{g(A_n)}$$

where $p_1 < \cdots < p_n$ are the first n primes.

Definition 5.3. *Define $PRF_\Gamma(a, b)$ as a 2-ary relation where a is the code of a formula A and b is the code of a proof of A using Γ.*

Definition 5.4. *Define $FOR(a)$ as a 1-ary relation where a is the code of a formula A of L_{NN}.*

Now that we have the definitions of FOR and PRF_Γ, one can show the following:

Theorem 5.5. *FOR is decidable.*

Proof. This follows from ***Gö2***. □

Theorem 5.6. *If Γ is axiomatized, then PRF_Γ is decidable.*

Proof. Let Γ be axiomatized. To show PRF_Γ is decidable, we construct an algorithm given a and b that decides if b is the code of a proof of A.

1. Use **Gö2** to check if a is the code of a formula A. If it is, find A such that $g(A) = a$.

2. Rewrite b as $b = p_1^{e_1} \times \cdots \times p_n^{e_n}$.

3. If primes p_i are the first n primes in increasing order, use **Gö2** to see if $e_1, \ldots, e_n$ are codes of formulas.

4. If yes, use **Gö2** to find formulas $A_1, \ldots, A_n$ such that $g(A_i) = e_i$.

5. Since Γ is axiomatized, we can recognize which A_i are in Γ. Using this, we can determine if each A_i is an axiom, an element of Γ, or is the result of applying a rule of inference on previous A_j, for $j < i$. Doing this, decide if $A_1, \ldots, A_n$ is a proof of A_n using Γ.

6. If yes, and if $A_n = A$, then b is the code of a proof of A using Γ.

Since we have given an algorithm, PRF_Γ is decidable. $\square$

Now that we have found a way to code sentences of $\mathrm{L_{NN}}$, we aim to find expressions for relations such as FOR and PRF_Γ. Recall that we are assuming the Church-Turing Thesis (see Appendix). Suppose we have a relation R that is decidable. By the Church-Turing Thesis, it is recursive and by Theorem 4.1, R is expressible. This means, there exists a formula A that expresses R. For example, since PRF_Γ and FOR are decidable, we know there are formulas Prf_Γ and For of $\mathrm{L_{NN}}$ which express each of these relations. Thus we have found a way to say "$A_1, \ldots, A_n$ is a proof of A" within $\mathrm{L_{NN}}$ itself!

6 Gödel's Incompleteness Theorems

In this section we will state and prove the goal of this paper: Gödel's Incompleteness Theorem I. First, it is important to note that Gödel assumed a stronger condition on consistency called ω-consistency. Later, Rosser found a way to remove this condition and prove the result with our regular definition of consistency. This section will use ω-consistency and the definition is as follows:

Definition 6.1. Γ *is* ω-consistent *if for every formula A of* $\mathrm{L_{NN}}$ *with exactly one free variable x, we have:*

If $\Gamma \vdash A_x[0^n]$ for each $n \in \mathbb{N}$, then $\neg\forall x A$ is not a theorem of Γ.

Even though Γ can prove A when the free variable x is replaced by the *nth* successor of 0, for all n in $\mathbb{N}$, we still cannot say that $\forall x A$ is a theorem of Γ. For example, there are nonstandard models of NN that contain elements that are not equal to any successor of 0 [2]. We can also show that ω-consistency is stronger than our regular definition of consistency.

Lemma 6.2. *If* Γ *is* ω*-consistent, then* Γ *is consistent.*

Proof. Let A be the formula $x_1 = x_1$. To show Γ is consistent, we will need to find a formula that cannot be proved by Γ. We have:

(1) $\forall x_1(x_1 = x_1)$ REF AX

(2) $0^n = 0^n$ $\forall$-ELIM

Thus $\Gamma \vdash A_{x_1}[0^n]$ for each n in $\mathbb{N}$. Then by the definition of ω-consistency, we have $\neg\forall x_1(x_1 = x_1)$ is not a theorem of Γ so Γ is consistent. $\square$

As further evidence that ω-consistency is not too extreme of a condition, we note that one can prove that if the standard interpretation $\mathcal{N}$ of L_{NN} is a model of Γ, then Γ is ω-consistent. See Lemma 2 in Section 7.4 of [1] for a proof.

We are now ready to state and prove **Gödel's Incompleteness Theorem I**.

Theorem 6.3 (Gödel's Incompleteness Theorem I)**.** *Every* ω*-consistent axiomatized extension* Γ *of NN is incomplete.*

Proof. To begin the proof of this theorem, our first goal will be to construct a sentence G and then show that neither G nor $\neg G$ are theorems of Γ. We can define a 2-ary relation R (dependent on Γ) on $\mathbb{N}$ as follows:

$R(a, b)$ if and only if both the subsequent conditions hold:

R1: a is the code of a formula A of Γ with one free variable x_1.

R2: b is the code of a proof of $A_{x_1}[0^a]$ using Γ.

Note that **R2** uses self reference by replacing the free variable x_1 in A with the a^{th} successor of 0, where a is the code of A. Next, by Theorem 5.6 we have that PRF_Γ is decidable since Γ is axiomatized. Now, we show that R is a decidable relation since we will then be able to deduce by the previous section, that R is recursive, and hence, expressible.

To show that R is decidable, we come up with an algorithm that decides whether R holds or not. We can use the following steps to do so. Given a and b,

1. Check by using **Gö2** if a is a code of a formula A of L_{NN}.
2. If it is, find the formula A and make sure it has exactly one free variable, x_1.
3. If it does, use **Gö2** to find $c = g(A_{x_1}[0^a])$. Note that this is the code of $A_{x_1}[0^a]$.
4. By Theorem 5.6, PRF_Γ is a decidable relation, so determine if $PRF_\Gamma(c, b)$ holds.

Since we have described an algorithm, R is a decidable relation. By the Church-Turing thesis, R is recursive. Hence, by Theorem 4.1, R is therefore expressible. This means that there exists a formula in NN that expresses R on $\mathbb{N}$ in terms of two free variables x_1, and x_2. We can call this formula B and then state the following from Definition 3.1 for all $a, b \in \mathbb{N}$:
EX1: If $R(a, b)$, then $\Gamma \vdash B_{x_1,x_2}[0^a, 0^b]$.
EX2: If $\neg R(a, b)$, then $\Gamma \vdash \neg B_{x_1,x_2}[0^a, 0^b]$.
Note that since $\mathrm{NN} \subseteq \Gamma$, Γ can replace NN in the conditions **EX1** and **EX2**.

Now let p be the code of the formula $\forall x_2 \neg B$ which has exactly one free variable x_1, meaning p satisfies **R1**. Then by **R2**, we have that

$$R(p, b) \text{ if and only if } b \text{ is the code of a proof of } \forall x_2 \neg B_{x_1}[0^p] \text{ using } \Gamma. \qquad (*)$$

We can let this formula be our sentence G: $\forall x_2 \neg B_{x_1}[0^p]$.

This has completed the first goal of the proof so we can begin the next portion which is to show that neither G nor $\neg G$ are theorems of Γ. We take an informal, closer look at the sentence G to better understand its meaning. Using DeMorgan's Law, we can rewrite G as $\neg \exists x_2 B_{x_1}[0^p]$. This says that there is no natural number b such that $B_{x_1,x_2}[0^p, 0^b]$. Since Γ cannot show such a b exists, we use the contrapositive of **EX1** to deduce that there is no $b \in \mathbb{N}$ such that $R(p, b)$. Then by $(*)$, there is no number b that is the code of a proof of $\forall x_2 \neg B_{x_1}[0^p]$, which implies that $\forall x_2 \neg B_{x_1}[0^p]$ (which is G) is not a theorem of Γ. Thus, G states *"I am not a theorem of* Γ*"*. So we have a sentence of $\mathrm{L_{NN}}$ that claims itself is not a theorem of an extension Γ of NN.

Finally, we will show neither G nor $\neg G$ is a theorem of Γ. First, suppose $\Gamma \vdash G$ and let b be the code of the proof of G using Γ. Then by $(*)$, $R(p, b)$, and so by **EX1**, it follows that $\Gamma \vdash B_{x_1,x_2}[0^p, 0^b]$. However, we have:

(1) $\forall x_2 \neg B_{x_1}[0^p]$	THM of Γ
(2) $\neg B_{x_1,x_2}[0^p, 0^b]$	$\forall$-ELIM

But since Γ is consistent, (2) gives a contradiction. Thus, G is not a theorem of Γ.

To prove $\neg G$ is not a theorem of Γ we need to utilize the definition of ω-consistency. We can see the formula $\neg B_{x_1}[0^p]$ has exactly one free variable, x_2 since x_1 is being replaced by the p^{th} successor of 0. Note that $\neg G$ is $\neg \forall x_2 \neg B_{x_1}[0^p]$ and to show that this is not a theorem of Γ by the definition of ω-consistency, we need to prove $\Gamma \vdash \neg B_{x_1,x_2}[0^p, 0^b]$ for all $b \in \mathbb{N}$. If we prove $\neg R(p, b)$, then we will get the desired result by **EX2**.

Let $b \in \mathbb{N}$ and suppose $R(p, b)$. By $(*)$, b is the code of a proof of G meaning $\Gamma \vdash G$. This contradicts what was proved above so we have $\neg R(p, b)$. Thus, **EX2** gives $\Gamma \vdash \neg B_{x_1,x_2}[0^p, 0^b]$ for all $b \in \mathbb{N}$ and so $\neg G$ is not a theorem of Γ. $\square$

We have finally completed the proof of Gödel's Incompleteness Theorem I!

In a model, a sentence is either true or false. Note that if G is false in the standard interpretation $\mathcal{N}$, then $\neg G$ is true which means there is always a true sentence (either G or ¬G) on $\mathbb{N}$ that is not a theorem of our extension of NN. We may add this true sentence as an axiom into Γ but then we can construct another sentence in a similar manner which is again, not a theorem of our extension. Thus, no matter how we axiomatize arithmetic it will either be inconsistent or incomplete.

However, we can find an extension Γ such that it is both consistent and complete, but not axiomatized, as we now describe. Let Γ be the set of all true sentences of L_{NN} in our standard model, $\mathcal{N}$. Since $\mathcal{N}$ is a model of Γ, by Gödel's Completeness Theorem I, Γ is a consistent extension of NN. Further, if a sentence A of L_{NN} is true in $\mathcal{N}$, then $A \in \Gamma$ since Γ consists of all true sentences in L_{NN}. Thus, such an A has a one line proof. If A is false in $\mathcal{N}$, then the negation of A is true and hence, has a proof in Γ by the same reasoning. Thus, Γ is complete!

This extension does not contradict the Incompleteness Theorem since it is not axiomatized. In fact, by the Incompleteness Theorem, there must not exist an algorithm to check if A is in Γ. This means there is no algorithm to tell what the true sentences in our standard interpretation are.

One might ask if Γ can prove its own consistency. To explore this idea, we will need the axiom scheme for induction which means we should now work with the stronger system of PA which can be seen as an extension of NN.

Recall that For and Prf_Γ are formulas of L_{NN} that express the relations FOR and PRF_Γ on $\mathbb{N}$. Let CON_{PA} be the sentence

$$\exists x_1[\mathit{For} \land \neg \exists x_2 \mathit{Prf}_{\mathrm{PA}}]$$

of L_{NN}. Informally, CON_{PA} says there is a formula of L_{NN} that does not have a proof in Γ. This is equivalent to stating that Γ is consistent. This brings us to Gödel's second result on incompleteness:

Theorem 6.4 (Gödel's Incompleteness Theorem II)**.** *If PA is consistent then CON_{PA} is not a theorem of PA.*

This theorem says that Peano Arithmetic cannot prove its own consistency! We know that PA is consistent since the standard interpretation is a model, but this model can only be constructed within certain systems such as ZFC set theory. PA alone cannot prove its own consistency.

The proof of this theorem is too difficult to get into here, but we can give the general idea. Since PA is an extension of NN, by the proof of Theorem 6.3 we have a sentence G_{PA} for which we showed "If PA is consistent, then G_{PA} is not a theorem of PA". However, one may formalize our proof of this entirely within PA! This is where the induction axiom of PA is required. Recall that, informally, G_{PA} says that G_{PA} is not a theorem of PA. Thus we have PA $\vdash CON_{\mathrm{PA}} \rightarrow G_{\mathrm{PA}}$. If we were to have PA $\vdash CON_{\mathrm{PA}}$, then by modus ponens (MP), we get PA $\vdash G_{\mathrm{PA}}$ which is a contradiction since G_{PA} is not a theorem of PA.

As a final thought, suppose someone were to show that PA is able to prove its own consistency, i.e. PA $\vdash CON_{\mathrm{PA}}$. If this was proved before Gödel gave his second Incompleteness Theorem, we would note that if PA was actually inconsistent then every sentence of L_{NN} would be a theorem of PA. Thus, CON_{PA} would also be a theorem and so even though we have the formal proof of PA $\vdash CON_{\mathrm{PA}}$, it still cannot tell us that PA is able to prove its own consistency.

Now that we have Gödel's second Incompleteness Theorem, suppose PA $\vdash CON_{PA}$ were established. By Theorem 6.4, we have that CON_{PA} is not a theorem of PA, which means PA is actually inconsistent!

In the year 1936, Rosser generalized Gödel's result by replacing the assumption of ω-consistency with consistency. Thus our result becomes:

Every consistent axiomatized extension of NN is incomplete.

In conclusion, we have stated and proved Gödel's Incompleteness Theorem and have explored the implications and limitations that have come forward because of it. The Incompleteness Theorems are some of the most significant results in mathematics and modern logic.

7 Appendix

This report takes Chapters 1 to 5 of [1] as background. A summary of this material is provided here.

7.1 First-Order Languages

We will use the first-order language L_{NN}:

Logical Symbols

(1) $\neg$, $\vee$, $\wedge$, $\rightarrow$, $\leftrightarrow$	logical connectives
(2) $\forall$, $\exists$	quantifiers
(3) (,)	punctuation
(4) $=$	symbol for equality
(5) $x_1, x_2, x_3, \ldots$	infinite list of variables

Note $\wedge, \rightarrow, \leftrightarrow$, and $\exists$ can be defined in terms of $\neg, \vee$, and $\forall$ in the usual way. For example, $A \rightarrow B$ is defined to be $\neg A \vee B$.

Non-Logical Symbols

(1) 0	constant symbol
(2) S	1-ary function symbol
(3) $+$, $\times$	2-ary function symbols
(4) $<$	2-ary relation symbol

We recall the notation 0^a, which denotes the a^{th} successor of 0 where a is a natural number. We define $0^0 = 0$ and $0^{a+1} = S0^a$ for $a \in \mathbb{N}$.

We can define terms, formulas, and sentences of L_{NN} as follows:

Terms: These are first defined by variables of NN and the constant 0. The application of a function symbol to terms is another term. An example is $S0$ since we use the 1-ary function S on our term 0.

Formulas: Let t_1 and t_2 be terms. Then $t_1 < t_2$ and $t_1 = t_2$ are formulas. If P and Q are formulas, then $\neg P$, $(P \vee Q)$, $(P \wedge Q)$, $(P \to Q)$, $(P \leftrightarrow Q)$, $\forall xP$ and $\exists xP$ are formulas.

Note that FOR_{NN} denotes the set of all formulas of NN and that $A_x[y]$ is obtained from the formula A by replacing all free occurrences of x by the term y.

Sentences: A sentence is a formula that does not have any free variables.

Definition 7.1. *An* interpretation *of* L_{NN} *consists of the following:*

1. *a nonempty set D*
2. *the constant symbol* 0 *is assigned to an element of D*
3. *the function symbol S is assigned to a 1-ary function on D*
4. *The function symbols $+$ and $\times$ are assigned to 2-ary functions on D*
5. *The relation symbol $<$ is assigned to a 2-ary relation on D*

In the standard interpretation $\mathcal{N}$ of L_{NN}, we take $D = \mathbb{N}$ and use the usual interpretations of $0, S, +, \times$, and $<$ on $\mathbb{N}$.

7.2 First-Order Logic

Here are the axioms, rules of inference, and some derived rules of inference: Let A, B, and C be formulas of L, and let s and t be terms.

Axioms of FO_L

(1) Propositional	$\neg A \vee A$	PROP AX
(2) Reflexive	$\forall x_1(x_1 = x_1)$	REF AX
(3) Substitution	$\forall x_n A \rightarrow A_{\mathrm{x_n}}[t]$	SUBST AX
(4) Equality	$(x_n = t) \rightarrow (A \leftrightarrow A_{\mathrm{x_n}}[t])$	EQ AX

Rules of Inference

(1) Associative Rule	$\frac{A\vee(B\vee C)}{(A\vee B)\vee C}$	ASSOC
(2) Contraction Rule	$\frac{A\vee A}{A}$	CTN
(3) Expansion Rule	$\frac{A}{B\vee A}$	EXP
(4) Cut Rule	$\frac{A\vee B, \neg A\vee C}{B\vee C}$	CUT
(5) Add $\forall$-Rule	$\frac{A\vee B}{A\vee\forall x_n B}$ with x_n not free in A	ADD $\forall$

Some ***Derived Rules of Inference*** are given here:

(1) Commutative Rule	$\frac{A\vee B}{B\vee A}$	CM
(2) Modus Ponens	$\frac{A, A\rightarrow B}{B}$	MP
(3) Disjunctive Syllogism	$\frac{A\vee B, \neg A}{B}$	DS
(4) Hypothetical Syllogism	$\frac{A\rightarrow B, B\rightarrow C}{A\rightarrow C}$	HS
(5) Modus Tollens	$\frac{A\rightarrow B, \neg B}{\neg A}$	MT
(6) $\forall$-Elimination Rule	$\frac{\forall x_n A}{A_{\mathrm{x_n}}[t]}$	$\forall$-ELIM
(7) Substitution Rule	$\frac{A}{A_{\mathrm{x_n}}[t]}$	SUBST RULE

Recall the universal closures of the following are the axioms of NN:

NN1 $\neg(Sx_1 = 0)$

NN2 $(Sx_1 = Sx_2) \to (x_1 = x_2)$

NN3 $x_1 + 0 = x_1$

NN4 $x_1 + Sx_2 = S(x_1 + x_2)$

NN5 $x_1 \times 0 = 0$

NN6 $x_1 \times Sx_2 = (x_1 \times x_2) + x_1$

NN7 $\neg(x_1 < 0)$

NN8 $(x_1 < Sx_2) \to [(x_1 < x_2) \vee (x_1 = x_2)]$

NN9 $(x_1 < x_2) \vee (x_1 = x_2) \vee (x_2 < x_1)$

The Peano axioms consists of the universal closures of **NN1** to **NN6**, along with:

$$[A_x[0] \wedge \forall x(A \to A_x[S_x])] \to \forall x A$$

which is the axiom of induction.

In addition, here are some derived rules which hold for NN:

(1) Tran Rule	$\frac{s=t, t=u}{s=u}$		TRAN
(2) Sym Rule	$\frac{s=t}{t=s}$		SYM
(3) Function Rule	$\frac{s=t}{F(s,u)=F(t,u)}$	(for F a function)	FUNC

7.3 Proof and Truth

Let Γ be a set of formulas of L_{NN}.

Definition 7.2. *Suppose we have a list of formulas $A_1, A_2, \ldots, A_n$ which are either in Γ, are axioms of NN, or are the result of applying rules of inference on previous A_i. Then $A_1, A_2, \ldots, A_n$ is called a* proof *of A_n and we write $\Gamma \vdash A_n$.*

Let I be an interpretation of L_{NN}. An assignment ϕ in I is a function that maps each x_i to an element of D. One can extend ϕ to terms, and use this to extend to a function, also called ϕ, that maps formulas to $\{T, F\}$ in the usual way.

Definition 7.3. *Let A be a formula. A is called* true in the interpretation I *if for every assignment ϕ in I, we always have $\phi(A) = T$. If $\phi(A) = F$ for all assignments then A is* false in I.

Definition 7.4. *An interpretation is a* model *of Γ if every formula in Γ is true in the interpretation.*

Definition 7.5. Γ *is* inconsistent *if there is a formula A such that $\Gamma \vdash A$ and $\Gamma \vdash \neg A$. Otherwise, Γ is called* consistent.

Note that one can prove that Γ is consistent if and only if there is a formula that is not a theorem of Γ.

Definition 7.6. Γ *is* complete *if for each sentence A, either $\Gamma \vdash A$ or $\Gamma \vdash \neg A$. If this is not true, then Γ is called* incomplete.

In 1929 [3], Gödel proved his Completeness Theorems, which we state here.

Theorem 7.7 (Gödel's Completeness Theorem I [3])**.** *Γ is consistent if and only if Γ has a model.*

Theorem 7.8 (Gödel's Completeness Theorem II [3])**.** *$\Gamma \vdash A$ if and only if A is true in every model of Γ.*

7.4 Recursive Functions and Relations

Recall the following definitions and theorems about recursive functions and relations on $\mathbb{N}^n$:

Definition 7.9. *K_R is called the* characteristic function *of an n-ary relation R. It is an n-ary function defined as:*

$$K_R(a_1, \ldots, a_n) = \begin{cases} 0 & \text{if } R(a_1, \ldots, a_n) \\ 1 & \text{if } \neg R(a_1, \ldots, a_n). \end{cases}$$

Definition 7.10 (Minimalization)**.** *Let G be an $(n+1)$-ary function such that for all $a_1, \ldots, a_n \in \mathbb{N}$, there exists $x \in \mathbb{N}$ with $G(a_1, \ldots, a_n, x) = 0$. The n-ary function F is found by* minimalization *if given such a function G we can define F for $a_i \in \mathbb{N}$ as*

$$F(a_1, \ldots, a_n) = \mu x[G(a_1, \ldots, a_n, x) = 0]$$

where $\mu x[G(a_1, \ldots, a_n, x) = 0]$ is the smallest natural number x such that $G(a_1, \ldots, a_n, x) = 0$ holds.

Definition 7.11. *An n-ary function F is* recursive *if there exists a finite sequence of functions $F_1, F_2, \ldots, F_m$ with $F_m = F$ and if one of the following conditions hold for $1 \leq k \leq m$:*

1. *F_k is a starting function (these are $+$, $\times$, $U_{n,k}$, and $K_<$ where $U_{n,k}$ is defined by $U_{n,k}(a_1, \ldots, a_n) = a_k$, and $K_<$ is the characteristic function of the $<$ relation).*
2. *$k > 1$ and F_k can be obtained using composition or minimalization on previous functions F_i where $1 \leq i \leq k-1$.*

Definition 7.12. *An n-ary relation R is* recursive *if K_R is a recursive function.*

Informally, we say a function F is computable if there is an algorithm to compute it. Similarly, we say a relation R is decidable if there exists an algorithm that will determine if terms are related to each other. One can show that the following hold:

Theorem 7.13. *Every recursive function is computable.*

Theorem 7.14. *Every recursive relation is decidable.*

If the reader is interested, the proof of the previous theorems can be found in Section 1.8 of [1]. The following is a hypothesis which cannot be proved due to the informal nature of our definition of a computable function but a brief explanation of why it should hold can also be found in the same section.

Church-Turing Thesis Every computable function is recursive.

The Church-Turing Thesis looks at the computability of recursive functions but can also be extrapolated to the decidability of recursive relations: *Every decidable relation is recursive.* We will make use of the Church-Turing Thesis, but it can be avoided by using deeper results on recursive relations (i.e. the relations that we show are decidable can directly be shown to be recursive) [1].

Definition 7.15. Γ *is* axiomatized *if there exists an algorithm that decides whether or not any given formula A is in Γ.*

References

[1] R. E. Hodel, *An Introduction to Mathematical Logic*, Dover Publications, Mineola, NY, 2013.

[2] M. R. Murty and B. Fodden, *Hilbert's Tenth Problem: An Introduction to Logic, Number Theory, and Computability*, Student Mathematical Library 88, American Mathematical Society, Providence, RI, 2019.

[3] K. Gödel, *Die Vollständigkeit der Axiome des logischen Funktionenkalküls*, Monatsh. Math. Phys. 37 (1930), no. 1, 349–360.

[4] K. Gödel, *Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme, I*, Monatsh. Math. Phys. v. 38 n. 1, (1931), 173-198.