Friendly Logic Reading Notes
- Reading for my daughters

Patrick X. Li [1] [2]

September 1, 2021

[1]This note is based on reading the book "A Friendly Introduction to Mathematical Logic", 2nd, by Christopher C. Leary and Lars Kristiansen. Other references go to lecture notes from Joshua Wiscons at `http://webpages.csus.edu/wiscons/teaching/math161_f18.html`, and the lecture notes from Benjamin Rossman at `http://www.math.toronto.edu/rossman/MAT309.html`.

[2]To my daughters: I hope you not to be afraid of the world not understandable, at least math is partially deductible.

Contents

Chapter 1

Structure and Languages

1.1 Languages

Definition 1.1.1. A **first-order language** $\mathcal{L}$ is an infinite collection of distinct *symbols*, of one of the following types:

1. *Parentheses*: $($, $)$.
2. *Connectives* "or", "not": $\vee$, $\neg$.
3. *Quantifier* "for all": $\forall$.
4. *Variables*, one for each positive integer n: v_1, v_2, $\ldots$, v_n, $\ldots$. The set of variable symbols is denoted by $Vars$.
5. *Equality symbol* "equal to": $=$.
6. *Constant symbols*: Some set of zero or more symbols.
7. *Function symbols*: For each positive integer n, some set of zero or more n-ary function symbols.
8. *Relation symbols*: For each positive integer n, some set of zero or more n-ary relation symbols.

Remark 1.1.1. Item 1-5 in Definition 1.1.1 are always included implicitly or explicitly.

Example 1.1.1. Definition 1.1.1 does not have $\exists$ and $\wedge$ explicitly. But we can write by the following:

- $(\exists v)\alpha$ represents $\Big(\neg(\forall v)(\neg\alpha)\Big)$.

- $\alpha \wedge \beta$ represents $\Big(\neg\big(\neg\alpha \vee \neg\beta\big)\Big)$.
- $\alpha \to \beta$ represents $\Big((\neg\alpha) \vee \beta\Big)$. Thus, we have $\neg\beta \to \neg\alpha$ represented by $\Big((\neg\neg\beta) \vee (\neg\alpha)\Big) \iff \Big((\beta) \vee (\neg\alpha)\Big) \iff \alpha \to \beta$.

  The equivalence of $\alpha \to \beta \iff \neg\beta \to \neg\alpha$ is what we called **contrapositive**.

Example 1.1.2. We can have some degree of freedom when specifying a language. Think about the concept of groups in abstract algebra. If we want to write a language of groups, we can have either one in the following:

1. $$\mathcal{L}_G = \{0, +\},$$

   where 0 is the constant symbol for identity element, + is the binary function symbol for operation which we implicitly refer to the additive operation.

2. $$\mathcal{L}_G = \{1, ^{-1}, \cdot\},$$

   where 1 is the constant symbol for identity element, $^{-1}$ is the unary function symbol to represent the inverse of an element of the group, $\cdot$ is the binary function symbol for operation which we implicitly refer to multiplicative operation.

1.2 Terms and Formulas

Pick a language $\mathcal{L}$. We want to decide which "strings" of symbols have meaning. So we try to define the **nouns** of our language. They are built up from variables and constants by using functions, but not relations. This gives us the definition of **terms** at Definition 1.2.1.

Definition 1.2.1. A **term** of a language $\mathcal{L}$ is a **nonempty finite string** t **of symbols** from $\mathcal{L}$ such that either:

1. t is a variable, or
2. t is a *constant symbol*, or

3. $t :\equiv ft_1t_2\dots t_n$ where f is an n-ary *function symbol* of $\mathcal{L}$ and each e t_i is a term. (Note: Here the symbol $:\equiv$ is a meta-linguistic symbol that means equivalence on each side, it is not as part of the first-order language $\mathcal{L}$.)

Remark 1.2.1. 3 in Definition 1.2.1 is recursive definition.

Example 1.2.1. Pick the language $\mathcal{L}_{NT}$. Examples of terms are:

1. $t :\equiv v_1$
2. $t :\equiv 0$
3. $t :\equiv S0$
4. $t : +S0S0$ (Do you know what this means? $1+1$)

Terms are like to **nouns**.
Formulas are like to **assertions**.

Definition 1.2.2. Let $\mathcal{L}$ be a first-order language. A **formula of** $\mathcal{L}$ is a **nonempty finite string** ϕ **of symbols** from $\mathcal{L}$ such that either:

1. $\phi :\equiv = t_1t_2$ where t_1 and t_2 are terms of $\mathcal{L}$, or
2. $\phi :\equiv Rt_1t_2\dots t_n$, where R is an n-ary relation symbol of $\mathcal{L}$ and t_1, t_2, $\dots$, t_n are all terms of $\mathcal{L}$, or
3. $\phi :\equiv (\neg\alpha)$, where α is a formula of $\mathcal{L}$, or
4. $\phi :\equiv (\alpha \vee \beta)$, where α and β are formulas of $\mathcal{L}$, or
5. $\phi :\equiv (\forall v)(\alpha)$ where v is a variable and α is a formula of $\mathcal{L}$.

Definition 1.2.3. The **atomic formulas of** $\mathcal{L}$ are those formulas that satisfy (1) and (2) in Definition (1.2.2).[1]

$3-5$ in Definition 1.2.2 are recursive definition (again).

A formula is like an assertion, the result is either **True** or **False**. If a string of symbols does not contain either $=$ or any other relation symbol from the language, this string **cannot** be a formula.

Remark 1.2.2. As implied by Example 1.1.1, Definition 1.2.2 contains implicitly $\exists$ and $\wedge$.

[1]We can consider equality as one relation, therefore, (2) in Definition 1.2.2 contains (1).

Example 1.2.2. let $\mathcal{L}$ be any first-order language. What is the intended meaning of

$$(\exists v_1)(\exists v_2)(\exists v_3)\Big(\neg\big((v_1 = v_2) \vee (v_1 = v_3) \vee (v_2 = v_3)\big)\Big)?$$

$$\begin{aligned} & (\exists v_1)(\exists v_2)(\exists v_3)\Big(\neg\big((v_1 = v_2) \vee (v_1 = v_3) \vee (v_2 = v_3)\big)\Big) \\ \Rightarrow\quad & (\exists v_1)(\exists v_2)(\exists v_3)\Big((v_1 \neq v_2) \wedge (v_1 \neq v_3) \wedge (v_2 \neq v_3)\Big) \end{aligned}$$

1.3 Sentences

Formulas can be True, False, or indeterminate. We want to narrow down the scope to exclude "indeterminate" formulas. This is done by introducing the concept of sentences: Sentences of $\mathcal{L}$ are the formulas that can be either True or False in a given mathematical model.

Definition 1.3.1. The **language of number theory**

$$\mathcal{L}_{NT} \text{ is: } \{0, S, +, \cdot, E, <\}$$

where:

- 0 is a constant symbol,
- S is a unary function symbol (stand for the successor function such that $S(x) = x + 1$ for x is some number),
- $+$, $\cdot$, E are binary function symbols, with the meaning of addition ($+32$ refers to $3+2$), multiplication ($\cdot 32$ refers to $3 \cdot 2$ or $3 * 2$), exponentiation ($E32$ refers to be 3^2), respectively,
- $<$ is a binary relation symbol, with the meaning of $x < y$ for x is less than y with x and y some numbers.

Example 1.3.1. Find the following formulas in $\mathcal{L}_{NT}$ is True, False, or indeterminate:

1. $(\forall x)(\forall y)[(x < y) \vee (x = y) \vee (y < x)]$
2. $(\exists x)(x < 0)$
3. $\neg(\forall x)[(y < x) \vee (y = x)]$

Formula (1) "says" either $x < y$, or $x = y$, or $y < x$ for any arbitrary numbers x and y. If we interpret $<$ in the usual way, this formula is a true statement.

Formula (2) "says" there exists $x < 0$, which is false in $\mathcal{L}_{NT}$ since 0 is the smallest in $\mathcal{L}_{NT}$.

Formula (3) "says" not every x is greater than or equal to y. The truth of this statement depends on what y is. This formula does not say anything about what y is, thus it is indeterminate, and we say y is a "free" variable.

Definition 1.3.2. Let v be a variable and ϕ be a formula. v **is free in** ϕ if it is one of the following:

1. ϕ is atomic according to Definition 1.2.3 and v occurs in ϕ (is a symbol in ϕ). Example: $v = 0$, $v < 0$.
2. $\phi :\equiv (\neg\alpha)$ and v is free in α. Example: $\neg(v = 0)$, $\neg(v < 0)$.
3. $\phi :\equiv (\alpha \vee \beta)$ and v is free in α or β. Example: $\big(\neg(v < 0)\big) \vee (0 = 0)$.
4. $\phi :\equiv (\forall u)(\alpha)$ where v is not u and v is free in α. Example: $(\forall u)(v = 0)$, $(\forall u)(v < u)$, but NOT $(\forall v)(v = 0)$.

If v is free in ϕ, we shall denote $\phi(v)$ instead of ϕ.

Definition 1.3.3. A **sentence** in a language $\mathcal{L}$ is a formula of $\mathcal{L}$ with **no** free variable according to Definition 1.3.2.

Example 1.3.2. Consider $\mathcal{L} = \{\sin^2, \cos^2, 1\}$, where $\sin^2$ and $\cos^2$ are unary functions, 1 is constant.

- $\sin^2 x + \cos^2 x = 1$ is **not** a sentence, since the variable x is free.
- $(\forall x)\big(\sin^2 x + \cos^2 x = 1\big)$ is a sentence, since the variable x is not free.

1.4 Structures

Defining the structure/model is the beginning of **semantics**: the structure/model would assign some meaning to strings of symbols in terms, formulas, sentences, so we can determine whether terms, formulas, sentences are True or False.

Definition 1.4.1. Let $\mathcal{L}$ be a language. An **$\mathcal{L}$-structure** $\mathfrak{A}$ is a nonempty set A called the **universe of** $\mathfrak{A}$ together with:

1. an element $c^{\mathfrak{A}}$ of A for each constant symbol c of $\mathcal{L}$,
2. an n-ary function $f^{\mathfrak{A}} : A^n \to A$ for each n-ary *function symbol* f of $\mathcal{L}$,
3. an n-ary relation $R^{\mathfrak{A}} \subseteq A^n$ for each n-ary *relation symbol* R of $\mathcal{L}$.

Sometimes, we call it **$\mathcal{L}$-model**.

Example 1.4.1. Let a language $\mathcal{L} = \{0, f, S\}$ with 0 as constant, f as binary function, S as unary relation.

Define $\mathcal{L}$-structure as $\mathfrak{M}$ by:

- $M = \{a, b, \Delta\}$.
- $0^{\mathfrak{M}} = a \in M$.
- $f^{\mathfrak{M}}$ with the following table:

$f^{\mathfrak{M}}$	a	b	Δ
a	Δ	b	b
b	a	Δ	b
Δ	a	a	Δ

  Then $\mathfrak{M} = (M, 0^{\mathfrak{M}}, f^{\mathfrak{M}}, S^{\mathfrak{M}})$ is an $\mathcal{L}$-structure.

- $S^{\mathfrak{M}} = \{a, b\} \subseteq M$, which means $S^{\mathfrak{M}}a$ and $S^{\mathfrak{M}}b$.

Example 1.4.2. For the language of number theory $\mathcal{L}_{NT} : \{0, S, +, \cdot, E, <\}$ in Definition 1.3.1, the **standard structure** is:

$$\mathfrak{N} = (\mathbb{N}, 0^{\mathfrak{N}}, S^{\mathfrak{N}}, +^{\mathfrak{N}}, \cdot^{\mathfrak{N}}, E^{\mathfrak{N}}, <^{\mathfrak{N}}),$$

or in short, without superscript

$$\mathfrak{N} = (\mathbb{N}, 0, S, +, \cdot, E, <)$$

We have:

- $0^{\mathfrak{N}} = 0 \in \mathbb{N}$
- $S^{\mathfrak{N}}(x) = x + 1$ with $x, x + 1 \in \mathbb{N}$
- $+^{\mathfrak{N}}(x, y) = x + y$
- $\cdot^{\mathfrak{N}}(x, y) = x \cdot y$
- $E^{\mathfrak{N}}(x, y) = x^y$
- $<^{\mathfrak{N}} \subseteq \mathbb{N}^2$ with $(x, y) \in <^{\mathfrak{N}} \Leftrightarrow x < y$,
  where $<^{\mathfrak{N}} = \{(0, 1), (0, 2), \ldots, (1, 2), \ldots\}$

1.5 Truth in a Structure

Assignment functions tie symbols together with the structures.

Example 1.5.1. Let $\mathcal{L}_R = \{0, 1, -, +, \cdot\}$ where 0 and 1 are constants, $-$ is unary function, $+$ and $\cdot$ are binary function. Consider:

- $\mathfrak{Q} = (\mathbb{Q}, 0^{\mathbb{Q}}, 1^{\mathbb{Q}}, -^{\mathbb{Q}}, +^{\mathbb{Q}}, \cdot^{\mathbb{Q}})$, or in short $\mathfrak{Q} = (\mathbb{Q}, 0, 1, -, +, \cdot)$, with the standard interpretation of operations in rational numbers.

- $\mathfrak{Z} = (\mathbb{Z}, 0, 1, -, +, \cdot)$ with the standard interpretation of operations in integer numbers.

$\mathfrak{Q}$ and $\mathfrak{Z}$ are $\mathcal{L}_R$-structures.
Take a look at the formulas:

1. $\phi :\equiv (\forall x)\Big[(x \neq 0) \to \Big[(\exists y)(x \cdot y = 1)\Big]\Big]$

2. $\psi(x) :\equiv (x \neq 0) \to \Big[(\exists y)(x \cdot y = 1)\Big]$

3. $\rho(x) :\equiv (\exists y)(x \cdot y = 1)$

We have the following:

- $\mathfrak{Q} \models \phi$, but $\mathfrak{Z} \not\models \phi$. Think about ϕ with $x = 2$ as an example.

- $\psi(x)$ has a free variable x. $\mathfrak{Q} \models \psi(x)$ because $\mathfrak{Q} \models \phi$. $\mathfrak{Z} \not\models \psi(x)$ because $\mathfrak{Z} \not\models (\forall x)\psi(x)$, one example is to just pick $x = 2$.

- $\rho(x)$ has a free variable x. $\mathfrak{Q} \not\models \rho(x)$ because $\mathfrak{Q} \not\models (\forall x)\rho(x)$, think about $x = 0$. Also, with the same reasoning, $\mathfrak{Z} \not\models \rho(x)$ because $\mathfrak{Z} \not\models (\forall x)\rho(x)$, think about once again $x = 0$.

- But if we choose $x = -1$, then $\mathfrak{Q} \models \rho(-1)$ and $\mathfrak{Z} \models \rho(-1)$, where $\rho(-1)$ is a sentence by substituting x in $\rho(x)$ with -1.

In this example, we implicitly use the assignment by allowing $x = 2$, or $x = -1$. The formal definition of assignment comes below.

Definition 1.5.1. Let $\mathfrak{A}$ be an $\mathcal{L}$-structure. a **variable assignment function** (into $\mathfrak{A}$) is a function $s : \ Vars \to A$ that assigns to each variable an element of the universe A (of the $\mathcal{L}$-structure $\mathfrak{A}$).

Example 1.5.2. In $\mathcal{L}_{NT}$ standard structure $\mathfrak{N}$, the following are variable assignment functions:

$$s_1 : \ Vars \to \mathbb{N}, \quad s_1(v_i) = 2i + 1$$

e.g. $s_1(v_3) = 7$, $s_1(v_5) = 11$.

$$s_2 : \ Vars \to \mathbb{N}, \quad s_2(v_i) = 3,$$

which means $s_2(v_1) = 3$, $s_2(v_3) = 3$, and $s_2(v_5) = 3$.

Variable assignment functions need not be injective or bijective. Also, we can extend assignment from variables to $\mathcal{L}$-terms.

Definition 1.5.2. Let $\mathfrak{A}$ be an $\mathcal{L}$-structure and a variable assignment function $s : Vars \to A$. The function

$$\overline{s} : \{\mathcal{L}\text{-term}\} \to A$$

is called the **term assignment function** (generated by s), where the term is by Definition 1.2.1, defined as:

1. If t is a variable, $\overline{s}(t) = s(t)$.
2. If t is a constant symbol c, then $\overline{s}(t) = \overline{s}(c) = c^{\mathfrak{A}}$.
3. If $t :\equiv ft_1t_2 \cdots t_n$, then $\overline{s}(t) = f^{\mathfrak{A}}(\overline{s}(t_1), \overline{s}(t_2), \ldots, \overline{s}(t_n))$.

Example 1.5.3. In $\mathcal{L}_{NT}$ standard structure $\mathfrak{N}$ shown in Example 1.4.2, consider the formula $t :\equiv SS0$. Then for $s_1(v_i) = 2i + 1$,

$$\overline{s_1}(t) = S^{\mathfrak{N}}(S^{\mathfrak{N}}(\overline{s_1}(0))) = S^{\mathfrak{N}}(S^{\mathfrak{N}}(0^{\mathfrak{N}})) = S^{\mathfrak{N}}(S^{\mathfrak{N}}(0)) = S^{\mathfrak{N}}(1) = 2.$$

You may notice the form of $s_1(v_1)$ does not impact the result of $\overline{s_1}(t)$ for $t :\equiv SS0$.

Example 1.5.4. In $\mathcal{L}_{NT}$ standard structure $\mathfrak{N}$, consider the formula:

$$t :\equiv E(v_3, SS0) + v_1 \cdot v_3 + v_5$$

- For $s_1(v_i) = 2i + 1$, $\overline{s_1}(t) = E(7, 2) + 3 \cdot 7 + 11 = 81$.
- For $s_2(v_i) = 3$, $\overline{s_2}(t) = E(3, 2) + 3 \cdot 3 + 3 = 21$.

Definition 1.5.3. Suppose a variable assignment function $s : Vars \to A$ and x is a variable and $a \in A$, then $s[x|a]$ is called an **x-modification of s** (into $\mathfrak{A}$) defined as:

$$s[x|a](v) = \begin{cases} a & \text{if } v = x \\ s(v) & \text{otherwise} \end{cases}$$

An x-modification of s is just like s, except that we assign the variable x with some $a \in A$.

Example 1.5.5. Let $t :\equiv E(v_3, SS0) + (v_1 \cdot v_3) + v_5$. The variable assignment function is $s_1(v_i) = 2i + 1$.

Then $\overline{s_1}(t) = 81$ from previous Example 1.5.4.

$\overline{s_1}[v_3|2](t) = E(2, SS0) + (v_1 \cdot 2) + v_5 = 2^2 + 3 \cdot 2 + 11 = 21$ where we "replace" v_3 by 2, and then evaluate the rest of the formula by $s_1(v_i) = 2i+1$.

Definition 1.5.4. Let $\mathfrak{A}$ be an $\mathcal{L}$-structure, ϕ is an $\mathcal{L}$-formula by Definition 1.2.2, and $s : Vars \to A$ is a variable assignment function by Definition 1.5.1. Then **$\mathfrak{A}$ satisfies ϕ with assignment** s, or **$\mathfrak{A}$ models ϕ with assignment** s, write $\mathfrak{A} \models \phi[s]$, in the following circumstances:

1. If $\phi :\equiv = t_1 t_2 :\equiv t_1 = t_2$, then $\overline{s}(t_1) = \overline{s}(t_2)$,
2. If $\phi :\equiv R t_1 t_2 \ldots t_n$, then $\big(\overline{s}(t_1), \overline{s}(t_2), \ldots, \overline{s}(t_n)\big) \in R^{\mathfrak{A}}$,
3. If $\phi :\equiv (\neg\alpha)$, then $\mathfrak{A} \not\models \alpha[s]$ where $\not\models$ means "does not satisfy" or "does not model",
4. If $\phi :\equiv (\alpha \vee \beta)$, then $\mathfrak{A} \models \alpha[s]$ or $\mathfrak{A} \models \beta[s]$,
5. If $\phi :\equiv (\forall x)(\alpha)$, then $\forall b \in A$, $\mathfrak{A} \models \alpha[s[x|b]]$.

Example 1.5.6. In $\mathcal{L}_{NT}$, consider the formula

$$\phi :\equiv (\forall v_1)\Big[(v_1 = 0) \vee (\forall v_2)(v_2 < v_1 + v_2)\Big].$$

We show for any variable assignment function s, $\mathfrak{N} \models \phi[s]$ by the following:

$$\begin{aligned}
& \mathfrak{N} \models \phi[s] \\
\iff\quad & \forall m \in \mathbb{N},\, \mathfrak{N} \models \Big[(v_1 = 0) \vee (\forall v_2)(v_2 < v_1 + v_2)\Big]\big[s[v_1|m]\big] \\
\iff\quad & \forall m \in \mathbb{N},\, \Big(\mathfrak{N} \models (v_1 = 0)\big[s[v_1|m]\big], \\
& \quad \text{or } \mathfrak{N} \models (\forall v_2)(v_2 < v_1 + v_2)\big[s[v_1|m]\big]\Big) \\
\iff\quad & \forall m \in \mathbb{N},\, \Big((m = 0^{\mathfrak{N}} = 0 \in \mathbb{N}), \\
& \quad \text{or } \forall r \in \mathbb{N},\, \mathfrak{N} \models (v_2 < v_1 + v_2)\big[s[v_1|m]\big]\big[v_2|r\big]\Big) \\
\iff\quad & \forall m \in \mathbb{N},\, \Big(m = 0 \text{ or } (\forall r \in \mathbb{N})\big(r < m + r\big)\Big) \\
\iff\quad & \mathit{True}
\end{aligned}$$

Let Γ be a set of $\mathcal{L}$-formula. Then $\mathfrak{A}$ satisfies Γ with assignment s, denoted by $\mathfrak{A} \models \Gamma[s]$ if for each $\gamma \in \Gamma$, $\mathfrak{A} \models \gamma[s]$.

Note the difference among the following statements in Lemma 1.5.1, Proposition 1.5.1, and Corollary 1.5.1:

Lemma 1.5.1. Let s_1 and s_2 be variable assignment functions into a structure $\mathfrak{A}$ such that $s_1(v) = s_2(v)$ for every variable v that occurs in the **term** t. Then $\overline{s_1}(t) = \overline{s_2}(t)$.

Proof. Use induction on the complexity of the term t:

- Base case:
  - $t :\equiv v$, with v a variable. Then $\overline{s_1}(t) = s_1(v) = s_2(v) = \overline{s_2}(t)$.
  - $t :\equiv c$, with c a constant. Then $\overline{s_1}(t) = c^{\mathfrak{A}} = \overline{s_2}(t)$.
- Induction case: $t = ft_1 \dots t_n$, and $\overline{s_1}(t_i) = \overline{s_2}(t_i)$. Then $\overline{s_1}(t) = f^{\mathfrak{A}}\big(\overline{s_1}(t_1), \dots, \overline{s_1}(t_n)\big) = f^{\mathfrak{A}}\big(\overline{s_2}(t_1), \dots, \overline{s_2}(t_n)\big) = \overline{s_2}(t)$.

□

Proposition 1.5.1. Let s_1 and s_2 be variable assignment functions into a structure $\mathfrak{A}$. Let ϕ be a **formula**. If $s_1(v) = s_2(v)$ for every *free variable* v in ϕ, then $\mathfrak{A} \models \phi[s_1]$ if and only if $\mathfrak{A} \models \phi[s_2]$.

Corollary 1.5.1. Let ϕ be a **sentence** in the language $\mathcal{L}$ and $\mathfrak{A}$ is an $\mathcal{L}$-structure. Then $\mathfrak{A} \models \phi$ if and only if $\mathfrak{A} \models \phi[s]$ for some assignment function s.

Another way to interpret Corollary 1.5.1 is: either $\mathfrak{A} \models \phi[s]$ for all assignment functions s, or $\mathfrak{A} \models \phi[s]$ for **no** assignment function s.

Definition 1.5.5. Let ϕ be a formula in the language $\mathcal{L}$ and $\mathfrak{A}$ is an $\mathcal{L}$-structure. Then $\mathfrak{A}$ is a **model** of ϕ, denoted by $\mathfrak{A} \models \phi$, if and only if $\mathfrak{A} \models \phi[s]$ for every assignment function s.

If Φ is a set of $\mathcal{L}$-formulas, then $\mathfrak{A}$ **models** Φ, denoted by $\mathfrak{A} \models \Phi$, if and only if $\mathfrak{A} \models \phi$ for all $\phi \in \Phi$.

Example 1.5.7. In $\mathcal{L}_{NT}$ standard structure $\mathfrak{N}$, consider the variable assignment function that assigns $s(v_i) = 2i$. For example, $s(v_1) = 2 \cdot 1 = 2$.

Consider the following:

1. the formula $\phi(v_1) :\equiv v_1 + v_1 = SSSS0$.

   To show $\mathfrak{A} \models \phi[s]$, LHS has:

$$\begin{aligned} \overline{s}(v_1 + v_2) &\iff \overline{s}(v_1) \ +^{\mathfrak{A}} \ \overline{s}(v_2) \\ &\iff 2 \ +^{\mathfrak{A}} \ 2 \\ &\iff 4. \end{aligned}$$

   RHS has:

$$\begin{aligned} \overline{s}(SSSS0) &\iff S^{\mathfrak{A}}(S^{\mathfrak{A}}(S^{\mathfrak{A}}(S^{\mathfrak{A}}(0^{\mathfrak{A}})))) \\ &\iff 4. \end{aligned}$$

   So LHS and RHS are equivalent, and thus $\mathfrak{A} \models \phi[s]$.

2. the sentence $\sigma :\equiv (\forall v_1)\neg(\forall v_2)\neg(v_1 = v_2 + v_2)$.

   First, note that $\neg(\forall v_2)\neg$ is equivalent to $(\exists v_2)$. So the sentence σ means $(\forall v_1)(\exists v_2)(v_1 = v_2 + v_2)$. We know the sentence σ is false in the standard structure. To argue this formally, let s be an arbitrary variable assignment function. Then:

$$\begin{aligned} & \mathfrak{N} \models \sigma[s] \\ \iff & \text{For every } a \in \mathbb{N}, \mathfrak{N} \models \neg(\forall v_2)\neg(v_1 = v_2 + v_2)s[v_1|a] \\ \iff & \text{For every } a \in \mathbb{N}, \text{ exists } b \in \mathbb{N}, \\ & \mathfrak{N} \models \big(v_1 = v_2 + v_2\big)s[v_1|a][v_2|b]. \end{aligned}$$

   Consider if $v_1 = a = 3$, it requires $v_2 = b = 1.5$, there is no such $b \in \mathbb{N}$, therefore $\mathfrak{N} \not\models \sigma[s]$.

1.6 Substitutions and Substitutability

Suppose $\mathfrak{A} \models \forall x \exists y \neg(x = y)$. This sentence is true in any structure $\mathfrak{A}$ such that A has at least two elements.

If we replace x by y, then the sentence becomes $\exists y \neg(y = y)$, which is false in any structure. We can not substitute x by arbitrary. In other words, we need to specify rules of substitutability to avoid this kind of problem: the problem of attempting to substitute a term inside a quantifier that binds a variable involved in the term.

Definition 1.6.1. Let u be a **term**, x be a variable, and t be a term. The **term** u_t^x ("u by t replacing x", or write $u(x = t)$) is defined as:

1. If u is a variable not equal to x, then u_t^x is u.
2. If u is x, then u_t^x is t.
3. If u is a constant symbol, then u_t^x is u.
4. If $u :\equiv f u_1 u_2 \dots u_n$, where f is an n-ary function symbol and u_i are terms, then
$$u_t^x \text{ is } f(u_1)_t^x (u_2)_t^x \dots (u_n)_t^x.$$

u_t^x can be also written as $u|_{x=t}$ where x in the term u is replaced by the term t.

Example 1.6.1. Let t be $g(c)$ and let u be $f(x, y) + h(z, x, g(x))$. Then
$$u_t^x \text{ is } f(g(c), y) + h(z, g(c), g(g(c))).$$

Definition 1.6.2. Let ϕ be an **$\mathcal{L}$-formula**, x be a variable, t be a term. The **formula** ϕ_t^x is defined as:

1. If $\phi :\equiv = u_1 u_2$, then ϕ_t^x is $= (u_1)_t^x (u_2)_t^x$.
2. If $\phi :\equiv R u_1 u_2 \dots u_n$, then ϕ_t^x is $R(u_1)_t^x (u_2)_t^x \dots (u_n)_t^x$.
3. If $\phi :\equiv \neg(\alpha)$, then ϕ_t^x is $\neg(\alpha_t^x)$.
4. If $\phi :\equiv (\alpha \vee \beta)$, then ϕ_t^x is $(\alpha_t^x \vee \beta_t^x)$.
5. If $\phi :\equiv (\forall y)(\alpha)$, then
$$\phi_t^x = \begin{cases} \phi & \text{if } x \text{ is } y \\ (\forall y)(\alpha_t^x) & \text{otherwise.} \end{cases}$$

Example 1.6.2. Let

$$\phi :\equiv \Big(P(x,y) \to [(\forall x)(Q(g(x),z)) \vee (\forall y)(R(x,h(x))]\Big).$$

Let t be the term $g(c)$, then

$$\phi_t^x :\equiv \Big(P(g(c),y) \to \big[(\forall x)(Q(g(x),z)) \vee (\forall y)(R(g(c),h(g(c)))\big]\Big).$$

The idea of a term is substitutable for a variable in a formula is that: we will not substitute a variable contained in the term that is bounded by a quantifier.

Example 1.6.3. In $\mathcal{L}_{NT}$, consider the formula

$$\phi :\equiv (\forall y)(x + y = z) \vee (\forall x)(x \cdot x = x).$$

If $t :\equiv y + w$,

$$\phi_t^x :\equiv (\forall y)(y + w + y = z) \vee (\forall x)(x \cdot x = x).$$

Definition 1.6.3. Let ϕ be an $\mathcal{L}$-formula, x be a variable, t be a term. Then t **is substitutable for** x **in** ϕ if

1. ϕ is atomic by Definition 1.2.3, or
2. $\phi :\equiv \neg(\alpha)$ and t is substitutable for x in α, or
3. $\phi :\equiv (\alpha \vee \beta)$ and t is substitutable for x in both α and β, or
4. $\phi :\equiv (\forall y)(\alpha)$ and either
   (a) x is not free in ϕ by Definition 1.3.2, or
   (b) (x is free) y does not occur in t and t is substitutable for x in α.

Example 1.6.4. Let $t :\equiv yz + z$, the formula ϕ is substitutable for x in $\mathcal{L}_{NT}$?

1. $\phi :\equiv (\forall y)(Sx = y)$: *No*, x is free, but y occurs in t.
2. $\phi :\equiv (\forall y)\big(y = 0 \vee (\forall x)(x = y)\big)$: *Yes*, x is not free in ϕ.
3. $\phi :\equiv (y = x) \vee (\forall \omega)(E(\omega, x) > \omega)$: *Yes*, $y = x$ is atomic; and x is free in $(\forall \omega)(E(\omega, x) > \omega)$, ω does not occur in t, and t is substitutable for x in $(E(\omega, x) > \omega)$ that is atomic.

ϕ_t^x is defined regardless of t is substitutable or not for x in ϕ.

If we are concerned with preserving the truth of formulas after performing substitutions, we need to restrict operations under "if t is substitutable for x for ϕ".

1.7 Logical Implication

A large number of mathematic study is on: If this statement is true, then the other statement is true? That is, by assuming a structure/model satisfies some formulas/sentences, then some other sentences are true.

To formalize this question, we consider the concept of "logically imply".

Definition 1.7.1. Let Δ and Γ be sets of $\mathcal{L}$-formulas. Then Δ **logically implies** Γ, denoted by $\Delta \models \Gamma$ if for every $\mathcal{L}$-structure $\mathfrak{A}$, $\mathfrak{A} \models \Delta \to \mathfrak{A} \models \Gamma$.

Definition 1.7.1 says that if Δ is true in $\mathcal{L}$-structure $\mathfrak{A}$, then Γ is true in $\mathcal{L}$-structure $\mathfrak{A}$. For Δ to be true in $\mathcal{L}$-structure $\mathfrak{A}$, it has to be the case that $\mathfrak{A} \models \Delta[s]$ for *every* assignment function s.

Example 1.7.1. Consider a language $\mathcal{L} = \{e, \cdot\}$, with e a constant symbol, and $\cdot$ is a binary function symbol. Let

$$\begin{aligned}
\phi_1 &:\equiv \Big((\forall x \forall y \forall z)\big[(xy)z = x(yz)\big]\Big) \\
\phi_2 &:\equiv \Big((\forall x)\big(xe = ex = x\big)\Big) \\
\phi_3 &:\equiv \Big((\forall x)(\exists y)\big(xy = yx = e\big)\Big) \\
\phi_4 &:\equiv \Big((\forall x)\big(x \cdot x = e\big)\Big) \\
\psi &:\equiv \Big(xy = yx\Big)
\end{aligned}$$

Let $\Delta = \{\phi_1, \phi_2, \phi_3, \phi_4\}$ and $\Psi = \{\psi\}$. We want to show $\Delta \models \Psi$.

Let $\mathfrak{M}$ be any $\mathcal{L}$-structure. We must show

$$\mathfrak{M} \models \Delta \to \mathfrak{M} \models \Psi.$$

That is to show $\mathfrak{M} \models \{\phi_1, \phi_2, \phi_3, \phi_4\} \to \mathfrak{M} \models \Big(xy = yx\Big) \iff \mathfrak{M} \models (\forall x \forall y)\Big(xy = yx\Big)$.

Assume $\mathfrak{M} \models \{\phi_1, \phi_2, \phi_3, \phi_4\}$. Let $a, b \in M$. Then

- There is a $a' \in M$ such that $a'a = e$.
- There is a $b' \in M$ such that $bb' = e$.

$$\begin{aligned}
& a\cdot b\in M \\
\Rightarrow\ & (a\cdot b)(a\cdot b)=e^{\mathfrak{M}} \quad \text{by } \phi_4 \\
\Rightarrow\ & a\cdot(b\cdot(a\cdot b))=e^{\mathfrak{M}} \quad \text{by } \phi_1 \\
\Rightarrow\ & a'\big(a\cdot(b\cdot(a\cdot b))\big)=a'\cdot e^{\mathfrak{M}} \\
\Rightarrow\ & (a'\cdot a)\cdot\big(b\cdot(a\cdot b)\big)=a' \quad \text{by } \phi_1 \text{ and } \phi_2 \\
\Rightarrow\ & e^{\mathfrak{M}}\cdot\big(b\cdot(a\cdot b)\big)=a' \quad \text{by } \phi_3 \\
\Rightarrow\ & b\cdot(a\cdot b)=a' \quad \text{by } \phi_2 \\
\Rightarrow\ & (b\cdot a)\cdot b\cdot e^{\mathfrak{M}}=a' \quad \text{by } \phi_1 \text{ and } \phi_2 \\
\Rightarrow\ & (b\cdot a)\cdot b\cdot e^{\mathfrak{M}}\cdot b'=a'\cdot b' \\
\Rightarrow\ & (b\cdot a)\cdot b\cdot b'=a'\cdot b' \quad \text{by } \phi_2 \\
\Rightarrow\ & (b\cdot a)\cdot e^{\mathfrak{M}}=a'\cdot b' \quad \text{by } \phi_3 \\
\Rightarrow\ & b\cdot a=a'\cdot b' \quad \text{by } \phi_2 \\
\Rightarrow\ & b\cdot a=e^{\mathfrak{M}}\cdot a'\cdot b'\cdot e^{\mathfrak{M}} \quad \text{by } \phi_2 \\
\Rightarrow\ & b\cdot a=a\cdot a\cdot a'\cdot b'\cdot b\cdot b \quad \text{by } \phi_4 \\
\Rightarrow\ & b\cdot a=a\cdot(a\cdot a')\cdot(b'\cdot b)\cdot b=a\cdot e^{\mathfrak{M}}\cdot e^{\mathfrak{M}}\cdot b \quad \text{by } \phi_3 \\
\Rightarrow\ & b\cdot a=a\cdot b \quad \text{by } \phi_2
\end{aligned}$$

Thus, $\Delta \models \Psi$.

Definition 1.7.2. Let ϕ be an $\mathcal{L}$-formula. Then ϕ is **valid** if $\emptyset \models \phi$, or in other words, if ϕ is true in every $\mathcal{L}$-structure with every assignment function s, denoted by $\models \phi$.

Example 1.7.2. In previous Example 1.7.1, we have $\Delta \models \Psi$, which is $\{\phi_1,\phi_2,\phi_3,\phi_4\} \models \{\psi\}$.

If we use $\phi :\equiv \Big((\phi_1 \wedge \phi_2 \wedge \phi_3 \wedge \phi_4) \to \psi\Big)$, then ϕ is logically valid, denoted as $\models \phi$.

$\models$ has multiple meanings. If a structure is on the left, such as $\mathfrak{A} \models \sigma$, it works on the truth of a formula in a single structure. If a set of sentences is on the left, such as $\Gamma \models \sigma$, it works on logical implication.

Example 1.7.3. $\mathcal{L}$ has a unary relation P. Let

$$\phi :\equiv \Big((\forall x)P(x) \to (\exists x)P(x)\Big).$$

Then ϕ is logically valid. Denote $\mathfrak{M}$ as an arbitrary structure and s is an arbitrary variable assignment function. We want to show $\mathfrak{M} \models \phi[s]$.

If $\mathfrak{M} \not\models \Big((\forall x)P(x)\Big)[s]$, then $\mathfrak{M} \models \phi[s]$ holds by the fact that if the premise is False, the logical implication is True, which is $False \to Anything$ is always *True*.

Suppose $\mathfrak{M} \models \Big((\forall x)P(x)\Big)[s]$. So for all $m \in \mathbb{M}$, $\mathfrak{M} \models \Big(P(x)\Big)[s[x|m]]$. Thus, there exists some $m \in \mathbb{M}$, $\mathfrak{M} \models \Big(P(x)\Big)[s[x|m]]$. This means $\mathfrak{M} \models \Big((\exists x)P(x)\Big)[s]$, and thus $\mathfrak{M} \models \phi[s]$.

Chapter 2

Deductions

2.1 Motivation

A proof in mathematics is a sequence of statements, each statement can be justified by referring to previous statements.

We need three parts:

1. Start by specifying a set Λ of $\mathcal{L}$-formulas, called the **logical axioms**.
2. Then specify a set of **nonlogical axioms**, Σ.
3. Finally develop some **rules of inference (RofI in short)** , by ordered pairs (Γ, ϕ) where Γ is a **finite** set of formulas and ϕ is a formula.

A set of logical axioms is fixed, the collection of RofI is fixed as well. But the set of nonlogical axioms may vary in each deduction.

To build a mathematical proof, we need to satisfy:

1. Given a formula θ, we need an algorithm to decide whether or not θ is a logical axiom.
2. Given a **finite** set of formulas Γ and a formula θ, we need an algorithm to decide whether or not (Γ, θ) is a RofI.
3. Γ has to be a **finite** set of formulas (for each RofI (Γ, θ)).
4. Each logical axiom has to be **valid** by Definition 1.7.2.
5. RofI has to preserve truth: for each RofI (Γ, θ), $\Gamma \models \theta$.

2.2 Deductions

Assume a language $\mathcal{L}$, a set of logical axioms Λ as a fixed set of $\mathcal{L}$-formula, and a set of ordered pairs (Γ, ϕ) as RofI.

Definition 2.2.1. Let Σ be a set of nonlogical axioms as a collection of $\mathcal{L}$-formulas. Let D be a **finite** sequence $(\phi_1, \phi_2, \dots, \phi_n)$ of $\mathcal{L}$-formulas. Then D is a **deduction from** Σ if for each i, $1 \leq i \leq n$:

1. $\phi_i \in \Lambda$, that is ϕ_i is a logical axiom, or
2. $\phi_i \in \Sigma$, that is ϕ_i is a nonlogical axiom, or
3. There exists a RofI (Γ, ϕ_i) such that $\Gamma \subseteq \{\phi_1, \phi_2, \dots, \phi_{i-1}\}$.

Definition 2.2.2. If there is a deduction from Σ, the last line of which is the formula ϕ, then it is called a **deduction from** Σ **of** ϕ, denoted by $\Sigma \vdash \phi$.

Example 2.2.1. Consider a language $\mathcal{L} = \{P\}$, where P is a binary relation symbol. A set of axioms Σ is:

$$\begin{aligned} \Sigma = \{ \quad & \forall x P(x,x), \\ & P(u,v), \\ & P(u,v) \to P(v,u) \\ & P(v,u) \to P(u,u) \quad \}. \end{aligned}$$

Let $\Lambda = \emptyset$.
Consider a set of rules of inference to be the rule modus ponens:

$$\{(\{\alpha, \alpha \to \beta\}, \beta) \mid \alpha \text{ and } \beta \text{ are formulas of } \mathcal{L}\}.$$

We can write a deduction from Σ of the formula $P(u,u)$ as follows:

$$\begin{aligned} & P(u,v) \\ & P(u,v) \to P(v,u) \\ & P(v,u) \\ & P(v,u) \to P(u,u) \\ & P(u,u). \end{aligned}$$

Definition 2.2.1 is a "bottom-up" definition. Another way to take a "top-down" approach to specify the collection of deduction from Σ in the following proposition.

Definition 2.2.3. Assume Λ, Σ, the set of (Γ, θ) are fixed. Define

$$theorem_\Sigma = \{\phi \mid \Sigma \vdash \phi\}$$

as the set of all formulas generated from Σ by deductions.

Example 2.2.2. In Example 2.2.1,

$$theorem_\Sigma = \Sigma \cup \{P(v, u),\ P(u, u)\}.$$

Proposition 2.2.1. Given a set of $\mathcal{L}$-formulas Λ (for logical axioms) and Σ (for nonlogical axioms), and a collection of RofI (Γ, θ), the set $theorem_\Sigma = \{\phi \mid \Sigma \vdash \phi\}$ is the smallest set C such that:

1. $\Lambda \subseteq C$.
2. $\Sigma \subseteq C$.
3. If (Γ, θ) is a RofI and $\Gamma \subseteq C$, then $\theta \in C$.

Proof. First part to prove: $theorem_\Sigma$ satisfies 1-3.

- If $\alpha \in \Sigma \cup \Lambda$, then $D = (\alpha)$ is a deduction of α, so $\alpha \in theorem_\Sigma$. Thus, $\Sigma \cup \Lambda \subseteq theorem_\Sigma$.
- Let (Γ, θ) be a RofI with $\Gamma \subseteq theorem_\Sigma$. Γ is finite, so we can write $\Gamma = \{\alpha_1, \alpha_2, \ldots, \alpha_m\} \subseteq theorem_\Sigma$. Thus, there is a deduction D_i for α_i from Σ. We can construct $D_1 \cup D_2 \cup \ldots \cup D_m \cup \{\phi\}$ that is a valid deduction of ϕ from Σ by the rule of inference (Γ, θ).

Second part to prove: If C satisfies 1-3, then $theorem_\Sigma \subseteq C$. Let $\phi \in theorem_\Sigma$. Proceed by induction on the deduction (the set) of ϕ from Σ. If $\phi \in \Sigma \cup \Lambda \subseteq C$, then $\phi \in C$. Otherwise, assume there is a RofI (Γ, ϕ) such that $\Gamma \subseteq C$ by induction, thus, by 3: If (Γ, θ) is a RofI and $\Gamma \subseteq C$, then $\theta \in C$; we have $\phi \in C$. □

2.3 The Logical Axioms

We need to construct a collection Λ of logical axioms for a first-order language $\mathcal{L}$: given a formula ϕ, we can tell whether $\phi \in \Lambda$ or $\phi \notin \Lambda$.

Definition 2.3.1 (Equality axioms)**.** Assume the equality symbol $=$ is a part of the language $\mathcal{L}$. There are three groups of axioms for this symbol:

1. For each variable x in $\mathcal{L}$,
$$x = x \quad \text{is in } \Lambda. \tag{E1}$$

2. For all variables x_1, x_2, …, x_n; y_1, y_2, …, y_n, and all n-ary function symbols f:
$$\begin{aligned}&[(x_1 = y_1) \wedge (x_2 = y_2) \wedge \ldots \wedge (x_n = y_n)] \\ &\to (f(x_1, x_2, \ldots, x_n) = f(y_1, y_2, \ldots, y_n)) \text{ is in } \Lambda.\end{aligned} \tag{E2}$$

3. Let R be an n-ary relation symbol.
$$\begin{aligned}&[(x_1 = y_1) \wedge (x_2 = y_2) \wedge \ldots \wedge (x_n = y_n)] \\ &\to (R(x_1, x_2, \ldots, x_n) \to R(y_1, y_2, \ldots, y_n)) \text{ is in } \Lambda.\end{aligned} \tag{E3}$$

Equality Axioms (E2) and (E3) are axioms to allow substitution of equals for equals.

Definition 2.3.2 (Quantifier Axioms)**.** For every variable x in $\mathcal{L}$, every $\mathcal{L}$-term t, and every $\mathcal{L}$-formula ϕ, assume the term t is substitutable for the variable x in ϕ, then:
$$(\forall x)(\phi) \to \phi^x_t \text{ is in } \Lambda, \tag{Q1}$$

$$\phi^x_t \to (\exists x)(\phi) \text{ is in } \Lambda, \tag{Q2}$$

Roughly speaking, Quantifier Axiom (Q1) can be regarded as universal instantiation. It says if for all x, ϕ holds, then $\phi(x = t)$ holds.

Quantifier Axiom (Q2) can be regarded as existence generalization. It says if $\phi(x = t)$ holds, then there exists x such that ϕ holds.

Definition 2.3.3 (Logical Axioms)**.** The set Λ of logical axioms is the collection of all formulas as such:

$$x = x \quad \text{for each variable } x. \tag{E1}$$

$$\begin{aligned}&[(x_1 = y_1) \wedge (x_2 = y_2) \wedge \ldots \wedge (x_n = y_n)] \\ \to \quad &(f(x_1, x_2, \ldots, x_n) = f(y_1, y_2, \ldots, y_n)).\end{aligned} \tag{E2}$$

$$\begin{aligned}&[(x_1 = y_1) \wedge (x_2 = y_2) \wedge \ldots \wedge (x_n = y_n)] \\ \to \quad &(R(x_1, x_2, \ldots, x_n) \to R(y_1, y_2, \ldots, y_n)).\end{aligned} \tag{E3}$$

$$(\forall x \phi) \to \phi^x_t, \text{ if } t \text{ is substitutable for } x \text{ in } \phi \tag{Q1}$$

$$\phi^x_t \to (\exists x \phi), \text{ if } t \text{ is substitutable for } x \text{ in } \phi \tag{Q2}$$

2.4 Rules of Inference

Two types of inference rules:

1. For propositional consequence.
2. For quantifiers.

2.4.1 Propositional Consequence

Consider a restricted language $\mathcal{P}$ of **propositional logic** with:

1. a set of propositional variables: A, B, C, ...
2. the connectives: $\vee$ and $\neg$.

- Notice there is no quantifiers, no relation symbols, no function symbols, no constants.
- Formulas are the collection of all ϕ where
  - ϕ is a propositional variable, or
  - ϕ is $(\neg\alpha)$, or is $(\alpha \vee \beta)$, with α and β being formulas of propositional logic.
- Its variable assignment function is $v : \{\textit{propositional variable}\} \to \{T, F\}$, where assigns each propositional variable to either T for True, or F for False.
- Similar to the term assignment function at Definition 1.5.2 and the definition of modeling the formula with assignment at Definition 1.5.4, we can extend v to a function $\overline{v}$ that assigns True or False to any propositional formula as:

$$\overline{v}(\phi) = \begin{cases} v(\phi) & \text{if } \phi \text{ is a propositional variable} \\ F & \text{if } \phi :\equiv (\neg\alpha) \text{ and } \overline{v}(\alpha) = T \\ F & \text{if } \phi :\equiv (\alpha \vee \beta) \text{ and } \overline{v}(\alpha) = \overline{v}(\beta) = F \\ T & \text{otherwise.} \end{cases} \tag{2.1}$$

Definition 2.4.1. A propositional formula ϕ is a **tautology** if and only if $\overline{v}(\phi) = T$ for **any assignment function** v. Or, a propositional formula ϕ is a tautology if it is always **True**, e.g. $A \vee (\neg A)$.

Lemma 2.4.1. If an $\mathcal{L}$-formula θ is not valid, then θ_P is not a tautology. Or equivalently, if θ_P is a tautology, then θ is valid.

The definition of valid is by Definition 1.7.2:

"ϕ is **valid** if $\emptyset \models \phi$, or in other words, if ϕ is true in every $\mathcal{L}$-structure with every assignment function s, denoted by $\models \phi$."

One important point to remember: the converse of Lemma 2.4.1 is **NOT** true, which is, we can **not** say if θ is valid, then θ_P is a tautology.

Proof. Let $\mathcal{L}$ be a first-order language, θ be an $\mathcal{L}$-formula. Let $A_1, \ldots, A_n$ be the propositional variables in the formula θ_P. Each variable A_i corresponds to a subformula ψ_i of θ, for $i = 1, \ldots, n$.

For any $\mathcal{L}$-structure $\mathfrak{A}$, define the variable assignment function $v_{\mathfrak{A}}$ by

$$v_{\mathfrak{A}} = \begin{cases} T & \text{if } \mathfrak{A} \models \psi_i \\ F & \text{otherwise.} \end{cases}$$

We want to prove the claim:

$$\mathfrak{A} \not\models \theta \iff \overline{v}_{\mathfrak{A}}(\theta_P) = F.$$

This is equivalent to prove

$$\overline{v}_{\mathfrak{A}}(\theta_P) = T \iff \mathfrak{A} \models \theta.$$

We prove this claim by induction over the complexity of θ.

1. θ is an atomic formula, then $\psi_1 = \theta$. The claim follows by the definition of $v_{\mathfrak{A}}$.

2. θ is of the form $(\forall x)(\alpha)$, then $\psi_1 = \theta$. The claim follows by the definition of $v_{\mathfrak{A}}$.

3. θ is of the form $(\neg\alpha)$. We have

$$\mathfrak{A} \not\models (\neg\alpha) \iff \mathfrak{A} \models \alpha \iff \overline{v}_{\mathfrak{A}}(\alpha_P) = T \iff \overline{v}_{\mathfrak{A}}((\neg\alpha)_P) = F$$

   where the second equivalence is by induction hypothesis, while the last equivalent is by propositional formula assignment function defined at Equation 2.1.

4. θ is of the form $\alpha \vee \beta$.

$$\begin{aligned} & \mathfrak{A} \not\models (\alpha \vee \beta) \\ \Longleftrightarrow \quad & \mathfrak{A} \not\models \alpha \text{ and } \mathfrak{A} \not\models \beta \\ \Longleftrightarrow \quad & \overline{v}_{\mathfrak{A}}(\alpha_P) = F \text{ and } \overline{v}_{\mathfrak{A}}(\beta_P) = F \quad \text{by induction hypothesis} \\ \Longleftrightarrow \quad & \overline{v}_{\mathfrak{A}}((\alpha \vee \beta)_P) = F \quad \text{by Equation 2.1.} \end{aligned}$$

This completes the proof of the first statement: If an $\mathcal{L}$-formula θ is not valid, then θ_P is not a tautology.

To prove the second statement: If θ_P is a tautology, then θ is valid.

Assume θ is *not* valid. Then by Definition 1.7.2 of validity, there exists an $\mathcal{L}$-structure $\mathfrak{A}$ such that $\mathfrak{A} \not\models \theta$. By the first statement that we just proved, there exists a variable assignment function v such that $\overline{v}_{\mathfrak{A}}(\theta_P) = F$. Thus, by the definition of tautology at Definition 2.4.1, θ_P is not a tautology. Therefore, if θ_P is a tautology, then θ is valid. □

Example 2.4.1. One way to check whether a given propositional formula ϕ is a tautology is by constructing a truth table. Suppose ϕ is $A \to (B \to A)$, translated to $\neg A \vee (\neg B \vee A)$. The truth table is:

A	B	$\neg A$	$\vee$	$(\neg B$	$\vee$	$A)$
T	T	F	T	F	T	T
T	F	F	T	T	T	T
F	T	T	T	F	F	F
F	F	T	T	T	T	F

Thus, the truth table tells us $A \to (B \to A)$ is tautology.

Procedure 2.4.1. To transfer a $\mathcal{L}$-formula β to a corresponding formula β_P of propositional logic (so that we can testify the tautology of β_P):

1. Find all subformula of β of the form $\forall x(\alpha)$ that are not in the scope of another quantifier. Replace them with propositional variables systemically (rename them with some propositional variables).

2. Find all atomic formulas that remain, and replace them systemically with new propositional variables.

3. After these replacements, β is transferred to a propositional formula β_P.

- Notice: If β_P is a tautology, then β is valid in the sense of Definition 1.7.2. However, the converse is **not true**. For example, let β is

$$[(\forall x)(\theta) \wedge (\forall x)(\theta \to \rho)] \to (\forall x)(\rho).$$

  β is valid. But β_P is $[A \wedge B] \to C$ is not a tautology.

Example 2.4.2. Suppose

$$\beta :\equiv \Big(\big((\forall x)P(x)\big) \wedge Q(c,z)\Big) \to \Big(Q(c,z) \vee \big((\forall)xP(x)\big)\Big).$$

1. Replace the quantified subformula $\forall xP(x)$ with a propositional variable B: $B = \forall xP(x)$. Translate β to:

$$(B \wedge Q(c,z)) \to (Q(c,z) \vee B).$$

2. Replace the atomic formula $Q(c,z)$ with a propositional variable A: $A = Q(c,z)$. Translate β further to:

$$(B \wedge A) \to (A \vee B).$$

3. β_P is $(B \wedge A) \to (A \vee B)$. Translate further into $\neg(B \wedge A) \vee (A \vee B)$, thus is $(\neg B \vee \neg A) \vee (A \vee B)$. It is easy to verify β_P is a tautology, by noticing $\neg B \vee B$ and $\neg A \vee A$ are $True$.

Another way to verify β_P is a tautology is to look at the truth table:

A	B	$(\neg B$	$\vee$	$\neg A)$	$\vee$	$(A$	$\vee$	$B)$
T	T	F	F	F	T	T	T	T
T	F	T	T	F	T	T	T	F
F	T	F	T	T	T	F	T	T
F	F	T	T	T	T	F	F	F

Since β_P is tautology, we have β is valid by Lemma 2.4.1.

Definition 2.4.2. Let Γ_P be a set of propositional formulas and ϕ_P is a propositional formula. Then ϕ_P is a **propositional consequence of** Γ_P if every truth assignment that makes each propositional formula in Γ_P true also makes ϕ_P true.

By this definition of propositional consequence at Definition 2.4.2, ϕ_P is a tautology if and only if ϕ_P is a propositional consequence of $\emptyset$.

Lemma 2.4.2. Let $\Gamma_P = \{\gamma_{1P}, \gamma_{2P}, \dots, \gamma_{nP}\}$ be a nonempty finite set of propositional formulas and ϕ_P be a propositional formula, then ϕ_P is a propositional consequence of Γ_P if and only if

$$[\gamma_{1P} \wedge \gamma_{2P} \wedge \cdots \gamma_{nP}] \to \phi_P$$

is a tautology.

Definition 2.4.3. Let Γ be a nonempty finite set of $\mathcal{L}$-formulas and ϕ be an $\mathcal{L}$-formula. Then ϕ is a **propositional consequence of** Γ if ϕ_P is a propositional consequence of Γ_P, where ϕ_P and Γ_P are the results of applying the transferring procedure at Procedure 2.4.1 uniformly to ϕ and all formulas in Γ.

Combining Definition 2.4.2 and Lemma 2.4.2, we have another definition of propositional consequence of Γ:

Definition 2.4.4. Let $\Gamma = \{\gamma_1, \gamma_2, \dots, \gamma_n\}$ be a nonempty finite set of $\mathcal{L}$-formulas and ϕ be an $\mathcal{L}$-formula.

Let $\Gamma_P = \{\gamma_{1P}, \gamma_{2P}, \cdots, \gamma_{nP}\}$ and ϕ_P be the results of applying the transferring procedure at Procedure 2.4.1 uniformly to ϕ and all formulas in Γ. Then ϕ is a **propositional consequence of** Γ if

$$[\gamma_{1P} \wedge \gamma_{2P} \wedge \cdots \gamma_{nP}] \to \phi_P$$

is a tautology.

Definition 2.4.5. Let Γ be a finite set of $\mathcal{L}$-formulas, ϕ be an $\mathcal{L}$-formula, and ϕ is a *propositional consequence of* Γ. Then, (Γ, ϕ) is a **rule of inference of type (PC)**.

What rule (PC) at Definition 2.4.5 says is that if we can prove γ_1 and γ_2, and $[\gamma_1 \wedge \gamma_2 \to \phi]_P$ is a tautology, then we can prove ϕ.

Also if ϕ is a formula and we can prove ϕ_P is a tautology, then rule (PC) at Definition 2.4.5 asserts ϕ in any deduction, using $\Gamma = \emptyset$.

2.4.2 Quantifier Rules

Motivation: Suppose without making any particular assumptions about x (x is not free),

- if we can prove "x is an apple", then it seems reasonable to claim that we have proved "$(\forall x)$ x is an apple".

- And if we can prove "there exists orange" from the assumption that "x is an apple", then from the assumption "$(\exists x)$ x is an apple", we should be able to prove "there exists orange".

Definition 2.4.6. Assume the variable x is not free in the formula ψ. Then both of the following are **rules of inference of type (QR)**:

$$(\{\psi \to \phi\},\ \psi \to (\forall x\phi))$$

$$(\{\phi \to \psi\},\ (\exists x\phi) \to \psi)$$

Here "not making any particular assumptions about x" is presented formally by "x is not free in ψ".

(QR) rules in Definition 2.4.6 explain as: If x is not free in ψ, then

1. From the formula $\psi \to \phi$, we may deduce $\psi \to (\forall x\phi)$.
2. From the formula $\phi \to \psi$, we may deduce $(\exists x\phi) \to \psi$.

Definition 2.4.7. A set of RofI is **decidable**, if given a finite set of $\mathcal{L}$-formulas Γ and an $\mathcal{L}$-formula ϕ, there exists an algorithm that can decide in a finite amount of time whether (Γ, ϕ) is a RofI or not.

Definition 2.4.8. A set of logical axioms Λ is **decidable**, if given an $\mathcal{L}$-formula θ, there exists an algorithm that can decide in a finite amount of time whether $\theta \in \Lambda$ or not.

Definition 2.4.9. A set of nonlogical axioms Σ is **decidable**, if given an $\mathcal{L}$-formula θ, there exists an algorithm that can decide in a finite amount of time whether $\theta \in \Sigma$ or not.

2.5 Soundness Theorem

Our objective is to prove the Soundness Theorem: If $\Sigma \vdash \phi$, then $\Sigma \models \phi$.

The "Soundness" refers to our deduction system (logical axioms Λ and RofI (Γ, θ)), "Soundness" says:

- If we can prove *something*, *something* is *True.*

Recap our deductive system (with logical axioms Λ and RofI (Γ, θ) at Section 2.1), we require:

1. The logical axiom Λ is decidable: an algorithm to decide whether or not $\theta \in \Lambda$:.

2. The set of logical axioms Λ is **valid** by Definition 1.7.2 : $\models \Lambda$

3. Γ is a **finite** set of formulas for each RofI (Γ, θ).

4. The RofI (Γ, θ) is decidable: An algorithm to decide whether or not (Γ, θ) is a RofI, given a **finite** set of formulas Γ and a formula θ.

5. RofI preserve truth: If (Γ, θ) is RofI, the $\Gamma \models \theta$.

Requirement 1, 3, 4 are satisfied by construction of logical axioms and inference rules presented at Section 2.3 and at Section 2.4 respectively. Here in the following we validate Requirement 2 and Requirement 5.

Theorem 2.5.1. The logical axioms are valid, $\models \Lambda$.

Proof. To prove Theorem 2.5.1, consider $\alpha \in \Lambda$ and let s be any arbitrary variable assignment function with any arbitrary $\mathcal{L}$-structure $\mathfrak{A}$, we need to show $\mathfrak{A} \models \alpha[s]$.

Specifically, we need to go over Logical Axiom (E1), (E2), (E3), (Q1), (Q2) to verify each axiom is valid.

We sketch how to verify (E2), (E3), and (Q1) in the following:

1. Consider (E2)):

$$\begin{gathered}[(x_1 = y_1) \wedge (x_2 = y_2) \wedge \ldots \wedge (x_n = y_n)] \\ \rightarrow (f(x_1, x_2, \ldots, x_n) = f(y_1, y_2, \ldots, y_n)).\end{gathered}$$

   Fix a structure $\mathfrak{A}$ and a variable assignment function $s : Vars \rightarrow A$. We need to show:

$$\mathfrak{A} \models \Big([(x_1 = y_1) \wedge (x_2 = y_2) \wedge \ldots \wedge (x_n = y_n)] \rightarrow \\ (f(x_1, x_2, \ldots, x_n) = f(y_1, y_2, \ldots, y_n)) \Big)[s].$$

   This means: Assume $(\mathfrak{A}, s)$ has $x_1 = y_1$, $x_2 = y_2$, …, $x_n = y_n$, then we have $s(x_1) = s(y_1)$, $s(x_2) = s(y_2)$, …, and $s(x_n) = s(y_n)$, and we need to prove $\mathfrak{A} \models (f(x_1, x_2, \ldots, x_n) = f(y_1, y_2, \ldots, y_n))[s]$.

   From the definition of model with assignment at Definition 1.5.4, that is to show:

$$\overline{s}(f(x_1, x_2, \ldots, x_n)) = \overline{s}(f(y_1, y_2, \ldots, y_n)).$$

By the definition of term assignment function at Definition 1.5.2, that is to show:

$$f^{\mathfrak{A}}(\overline{s}(x_1), \overline{s}(x_2), \ldots, \overline{s}(x_n)) = f^{\mathfrak{A}}(\overline{s}(y_1), \overline{s}(y_2), \ldots, \overline{s}(y_n)).$$

This is true, since $\overline{s}(x_i) = s(x_i) = s(y_i) = \overline{s}(y_i)$, and since $f^{\mathfrak{A}}$ is a function. Thus we show (E2) is valid.

2. Consider (E3), where $\alpha \in \Lambda$ has the form:

$$[(x_1 = y_1) \wedge (x_2 = y_2) \wedge \ldots \wedge (x_n = y_n)] \\ \to (R(x_1, x_2, \ldots, x_n) \to R(y_1, y_2, \ldots, y_n)).$$

We want to prove $\mathfrak{A} \models \alpha[s]$.

Assume $\mathfrak{A} \models \big((x_1 = y_1) \wedge (x_2 = y_2) \wedge \ldots \wedge (x_n = y_n)\big)[s]$, and $\mathfrak{A} \models \big(R(x_1, x_2, \ldots, x_n)\big)[s]$. From the definition of model with assignment at Definition 1.5.4, that is:

$$\mathfrak{A} \models \Big(s(x_1) = s(y_1) \wedge s(x_2) = s(y_2) \wedge \ldots \wedge s(x_n) = s(y_n)\Big)$$

and

$$\mathfrak{A} \models \Big(\big(s(x_1), s(x_2), \ldots, s(x_n)\big) \in R^{\mathfrak{A}}\Big).$$

Thus, we have:

$$\mathfrak{A} \models \Big(\big(s(y_1), s(y_2), \ldots, s(y_n)\big) \in R^{\mathfrak{A}}\Big).$$

That is: $\mathfrak{A} \models \alpha[s]$.

3. Consider (Q1): if t is substitutable for x in ϕ, then $(\forall x \phi) \to \phi^x_t$. We have $\alpha :\equiv (\forall x \phi) \to \phi^x_t$, we want to prove $\mathfrak{A} \models \alpha[s]$.

Assume $\mathfrak{A} \models (\forall x \phi)[s]$, we need to prove $\mathfrak{A} \models \phi^x_t[s]$.

By the definition of satisfaction with assignment at Definition 1.5.4, assumption $\mathfrak{A} \models (\forall x \phi)[s]$ is $\mathfrak{A} \models \phi\big[s[x|a]\big]$ for all element $a \in A$. Let $a = \overline{s}(t)$, we have $\mathfrak{A} \models \phi\big[s[x|\overline{s}(t)]\big]$.

What we need to prove is: $\mathfrak{A} \models \phi\big[s[x|\overline{s}(t)]\big] \iff \mathfrak{A} \models \phi^x_t\big[s\big]$, and it is shown by Theorem 2.6.1.[1]

Thus, we just show $\mathfrak{A} \models \phi^x_t\big[s\big]$, so (Q1) is valid.

[1]Informally speaking, Theorem 2.6.1 says that suppose ϕ is true in $\mathfrak{A}$ with assignment function s, and $x = \overline{s}(t)$. Then under the assumption that t is substitutable for x in ϕ, if we change the formula ϕ by replacing x by t, then ϕ^x_t would be true in $\mathfrak{A}$ with assignment function s. The substitution of x by t happening at assignment function s or happening at the formula ϕ have the same impact on the 'final valuation of the formula by assignment'.

□

Theorem 2.5.2. If (Γ, θ) is a rule of inference, then $\Gamma \models \theta$.

Proof. To prove Theorem 2.5.2, we validate each inference rule (PC) at Definition (2.4.5) and (QR) at Definition (2.4.6).

1. Take inference rule (PC): Assume (Γ, θ) is a rule of type (PC). Then Γ is finite. Denote it by $\Gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_n\}$. By Lemma 2.4.2, we have

   $$\big[\gamma_{1P} \wedge \gamma_{2P} \wedge \ldots \wedge \gamma_{nP}\big] \to \theta_P$$

   is a tautology, where $\Gamma_P = \{\gamma_{1P}, \gamma_{2P}, \ldots, \gamma_{nP}\}$ is the set of propositional formula corresponding to Γ and θ_P the propositional formula corresponding to θ.

   Then by Lemma 2.4.1: If ϕ_P is a tautology, then ϕ is valid. We have

   $$\big[\gamma_1 \wedge \gamma_2 \wedge \ldots \wedge \gamma_n\big] \to \theta$$

   is valid. Thus, we just show $\Gamma \models \theta$.

2. Take one of the quantifier rules (QR): $(\{\psi \to \phi\},\ \psi \to (\forall x \phi))$.

   Suppose x is not free in ψ. We need to show that $(\psi \to \phi) \models [\psi \to (\forall x \phi)]$.

   So assume $\mathfrak{A} \models (\psi \to \phi)[s]$ for every assignment function s. Assume an arbitrary assignment function $t : \ Vars \to A$. We need to show $\mathfrak{A} \models (\psi \to \forall x \phi)[t]$.

   Assume $\mathfrak{A} \models \psi[t]$. We need to show $\mathfrak{A} \models \forall x \phi[t]$, which is for any $a \in A$, we need to show that $\mathfrak{A} \models \phi\big[t[x|a]\big]$.

   By assumption that $\mathfrak{A} \models (\psi \to \phi)$ for any assignment function, we have $\mathfrak{A} \models (\psi \to \phi)\big[t[x|a]\big]$. Since $\mathfrak{A} \models \psi[t]$, and t and $t[x|a]$ agree on all the free variables of ψ (where x is not free in ψ by assumption), by proposition 1.5.1 :

   $\Big[$If $s_1(v) = s_2(v)$ for every *free variable* v in ϕ, then $\mathfrak{A} \models \phi[s_1]$ if and only if $\mathfrak{A} \models \phi[s_2]\Big]$,

   we have $\mathfrak{A} \models \psi[t[x|a]]$. Together with $\mathfrak{A} \models (\psi \to \phi)\big[t[x|a]\big]$, we know $\mathfrak{A} \models \phi[t[x|a]]$, and we finish.

□

Theorem 2.5.3 (Soundness Theorem)**.** If $\Sigma \vdash \phi$, then $\Sigma \models \phi$.

Remark 2.5.1.

- In Theorem 2.5.3, Σ is a collection of $\mathcal{L}$-formulas.
- ϕ is a $\mathcal{L}$-formula.
- $\Sigma \vdash \phi$ means that ϕ is deduction from Σ by Definition 2.2.1. That is: Σ proves ϕ.
- $\Sigma \models \phi$ means that Σ logically implies ϕ by Definition 1.7.1. That is: ϕ is True in Σ.

Proof. To prove Theorem 2.5.3, let $theorem_\Sigma = \{\phi | \Sigma \vdash \phi\}$, as what Proposition 2.2.1 does. And let $X = \{\phi | \Sigma \models \phi\}$. We need to show $theorem_\Sigma \subseteq X$. Basically we need to show the following:

1. $\Sigma \subseteq X$. For any $\sigma \in \Sigma$, then obviously $\Sigma \models \sigma$, so $\Sigma \subseteq X$.
2. The set of logical axioms $\Lambda \subseteq X$. Logical axioms are valid by Theorem 2.5.1, they are *True* in any structure. This means $\Sigma \models \lambda$ for any $\lambda \in \Lambda$. So $\Lambda \subseteq X$.
3. If (Γ, θ) is a RofI with $\Gamma \subseteq X$. Assume a structure $\mathfrak{A}$ such that $\mathfrak{A} \models \Gamma$, we need to show $\mathfrak{A} \models \theta$:

   Since now $\Gamma \subseteq X$, so $\mathfrak{A} \models \Sigma \Rightarrow \mathfrak{A} \models \Gamma$. By Theorem 2.5.2: (Γ, θ) is RofI, thus $\Gamma \models \theta$. Therefore, we have $\mathfrak{A} \models \Gamma \models \theta$, which is $\mathfrak{A} \models \theta$.

□

The Soundness Theorem 2.5.3 ties the notions of deducibility ($\vdash$) with logical implication ($\models$): If there is $\Sigma \vdash \theta$ (a deduction from Σ to θ), then $\Sigma \models \theta$ (Σ logically implies θ).

In plain words: what we deduce (under our deduction system) is true.

2.6 Two Technical Lemmas

Example 2.6.1. Consider in $\mathcal{L}_{NT}$ the term $u = x \cdot v$ and $t = y + z$. Then $u_t^x = u|_{x=(y+z)} = (y+z) \cdot v$. The vaf (variable assignment function) s is:

$Vars$	s
x	12
y	3
z	7
v	4
$\vdots$	$\vdots$

where $s(x) = 12$, $s(y) = 3$, and so on.

Suppose another vaf s' that copies from s, except s' assigns x to $\overline{s}(t)$, which is $\overline{s}(y + z) = 3 + 7 = 10$:

$Vars$	s	s'
x	12	10
y	3	3
z	7	7
v	4	4
$\vdots$	$\vdots$	$\vdots$

If we compare $\overline{s}(u_t^x)$ and $\overline{s'}(u)$:

$$\overline{s}(u_t^x) = \overline{s}((y+z) \cdot v) = (3+7) \cdot 4 = 40$$

$$\overline{s'}(u) = \overline{s'}(x \cdot v) = 10 \cdot 4 = 40$$

So, $\overline{s}(u_t^x) = \overline{s'}(u)$. The value of the formula that s assigns to u_t^x is the same as that s' assigns to u.

Lemma 2.6.1. Suppose u is a term, x is a variable, t is a term. Consider a variable assignment function $s : Vars \to A$ and that $s' = s\big[x|\overline{s}(t)\big]$. Then $\overline{s}(u_t^x) = \overline{s'}(u)$.

Notice that vaf s and vaf s' are different only at the variable x where s' sets $x = \overline{s}(t)$. Thus, $\overline{s'}(x) = \overline{s}(t)$ and $s'(y) = s(y)$ for $y \neq x$.

Proof. To prove Lemma 2.6.1, notice u is a term. So by induction on the complexity of the term u, there are the following cases:

- u is a variable that is exactly x.

$$\overline{s}(u_t^x) = \overline{s}(x_t^x) = \overline{s}(t) = \overline{s'}(x) = \overline{s'}(u).$$

- u is a variable y different from x.

$$\overline{s}(u_t^x) = \overline{s}(y_t^x) = \overline{s}(y) = s(y) = s'(y) = \overline{s'}(u).$$

- u is a constant symbol c.

$$\overline{s}(u_t^x) = \overline{s}(c_t^x) = \overline{s}(c) = c^{\mathfrak{A}} = \overline{s'}(u).$$

- u is an n-ary function $f(r_1, r_2, \ldots, r_n)$, with each r_i a term.

$$\begin{aligned}
& \overline{s}(u_t^x) \\
= \; & \overline{s}\big(\big[f(r_1, r_2, \ldots, r_n)\big]_t^x\big) \\
= \; & \overline{s}\big(f\big((r_1)_t^x, (r_2)_t^x, \ldots, (r_n)_t^x\big)\big) \\
= \; & f^{\mathfrak{A}}\big(\overline{s}\big[(r_1)_t^x\big], \overline{s}\big[(r_2)_t^x\big], \ldots, \overline{s}\big[(r_n)_t^x\big]\big) \quad \text{by definition of } \overline{s} \\
= \; & f^{\mathfrak{A}}\big(\overline{s'}(r_1), \overline{s'}(r_2), \ldots, \overline{s'}(r_n)\big) \quad \text{by inductive hypothesis} \\
= \; & \overline{s'}\big(f(r_1, r_2, \ldots, r_n)\big) \quad \text{by definition of } \overline{s'} \\
= \; & \overline{s'}(u)
\end{aligned}$$

□

Remark 2.6.1. As long as t is substitutable for x in ϕ, two different ways of evaluating the truth of "ϕ, where we interpret x as t" mean the same:

1. The first way is to form the formula ϕ_t^x and checking if $\mathfrak{A} \models \phi_t^x[s]$.

2. The second way is to change the assignment function s to s' by interpreting x as $\overline{s}(t)$, and checking whether ϕ is true with this new assignment function s'.

Theorem 2.6.1. Suppose ϕ is an $\mathcal{L}$-formula, x is a variable, t is a term, and t is substitutable for x in ϕ. Suppose $s : Vars \to A$ is a variable assignment function and $s' = s[x|\overline{s}(t)]$. Then $\mathfrak{A} \models \phi_t^x[s]$ if and only if $\mathfrak{A} \models \phi[s']$.

Proof. To prove Theorem 2.6.1, we use induction on the complexity of ϕ:

1. $\phi :\equiv u_1 = u_2$ where u_1 and u_2 are terms.

$$\begin{aligned} & \mathfrak{A} \models \phi^x_t[s] \\ \Rightarrow\ & \mathfrak{A} \models \Big((u_1)^x_t = (u_2)^x_t\Big)[s] \\ \Rightarrow\ & \overline{s}\big((u_1)^x_t\big) = \overline{s}\big((u_2)^x_t\big) \\ & \quad \text{by model with assignment at Definition 1.5.4} \\ \Rightarrow\ & \overline{s'}(u_1) = \overline{s'}(u_2) \quad \text{by Lemma 2.6.1} \\ \Rightarrow\ & \mathfrak{A} \models \phi[s'] \end{aligned}$$

2. $\phi :\equiv R(u_1, u_2, \ldots, u_n)$.

$$\begin{aligned} & \mathfrak{A} \models \phi^x_t[s] \\ \Rightarrow\ & \mathfrak{A} \models \Big(R\big((u_1)^x_t, (u_2)^x_t, \ldots, (u_n)^x_t\big)\Big)[s] \\ \Rightarrow\ & \Big(\overline{s}\big((u_1)^x_t\big), \overline{s}\big((u_2)^x_t\big), \ldots, \overline{s}\big((u_n)^x_t\big)\Big) \in R^{\mathfrak{A}} \\ & \quad \text{by Definition of satisfaction 1.5.4} \\ \Rightarrow\ & \Big(\overline{s'}(u_1), \overline{s'}(u_2), \ldots, \overline{s'}(u_n)\Big) \in R^{\mathfrak{A}} \quad \text{by Lemma 2.6.1} \\ \Rightarrow\ & \mathfrak{A} \models \phi[s'] \end{aligned}$$

3. ϕ involves the connectives $\vee$ and $\neg$. It follows immediately from the inductive hypothesis.

4. $\phi :\equiv \forall y \psi$. It has two sub cases: x is y; x is not y.

   (a) x is y: ϕ^x_t is ϕ by Definition 1.6.2. So $\mathfrak{A} \models \phi^x_t[s] \iff \mathfrak{A} \models \phi[s]$. s and s' agree on all free variables in ϕ (x is y and y is bounded by $\forall$, thus x is not free). By proposition 1.5.1, $\mathfrak{A} \models \phi[s] \iff \mathfrak{A} \models \phi[s']$. Thus we have $\mathfrak{A} \models \phi^x_t[s] \iff \mathfrak{A} \models \phi[s']$.

   (b) x is not y:

      i. $\phi :\equiv \forall y \psi$, x is not y, x is *not* free in ψ. Since x is not free in ψ, ψ^x_t is ψ. Thus, ϕ^x_t is ϕ. Since s and s' agree on all free variables in ϕ, again by proposition1.5.1, we have:

      $$\mathfrak{A} \models \phi^x_t[s] \iff \mathfrak{A} \models \phi[s] \iff \mathfrak{A} \models \phi[s']$$

      ii. $\phi :\equiv \forall y \psi$, x is not y, x is free in ψ. As the given condition in this theorem: t is substitutable for x in ϕ, by Definition

1.6.3, we have y does not occur in t and t is substitutable for x in ψ.
We have:

$$\begin{aligned} & \mathfrak{A} \models \phi_t^x[s] \\ \iff & \mathfrak{A} \models (\forall y)(\psi_t^x)[s] \\ \iff & \mathfrak{A} \models (\psi_t^x)\big[s[y|a]\big] \quad \text{for every } a \in A. \end{aligned}$$

and

$$\begin{aligned} & \mathfrak{A} \models \phi[s'] \\ \iff & \mathfrak{A} \models (\forall y)(\psi)[s'] \\ \iff & \mathfrak{A} \models \psi\big[s'[y|a]\big] \quad \text{for every } a \in A. \end{aligned}$$

Since x is not y, for any $a \in A$, $s'[y|a] = s[y|a][x|\overline{s}(t)]$. By inductive hypothesis with t is substitutable for x in ψ, we have:

$$\mathfrak{A} \models (\psi_t^x)\big[s[y|a]\big] \iff \mathfrak{A} \models \psi\big[s'[y|a]\big].$$

Therefore, we have

$$\mathfrak{A} \models \phi_t^x[s] \iff \mathfrak{A} \models \phi[s']$$

□

2.7 properties of Our Deductive System

We can show in our deductive system, the equality is an equivalence relation.

Theorem 2.7.1.

1. $\vdash x = x$.

2. $\vdash x = y \to y = x$.

3. $\vdash (x = y \land y = z) \to x = z$.

Proof.

1. $\vdash x = x$: this is logical axiom of type (E1).

2. $\vdash x = y \to y = x$: Assume we have $x = y$, and $x = x$ from (E1).

$$\begin{aligned}&\big[x = y \wedge x = x\big] \to \big[x = x \to y = x\big] \quad \text{(E3)}\\ &x = x \quad \text{(E1)}\\ &x = y \to y = x \quad \text{(PC)}\end{aligned}$$

3. $\vdash (x = y \wedge y = z) \to x = z$: Assume $y = z$, and $x = x$ from (E1).

$$\begin{aligned}&\big[x = x \wedge y = z\big] \to \big[x = y \to x = z\big] \quad \text{(E3)}\\ &x = x \quad \text{(E1)}\\ &(x = y \wedge y = z) \to x = z \quad \text{(PC)}\end{aligned}$$

□

Lemma 2.7.1. $\Sigma \vdash \theta$ if and only if $\Sigma \vdash \forall x\theta$.

Proof.

1. To prove $\Sigma \vdash \theta \to \Sigma \vdash \forall x\theta$: [2]

$$\begin{array}{ll} \theta & \\ y = y & \text{By logical axiom } \Lambda \\ y = y \to \theta & \text{(PC): } \theta \text{ is True, then (anything} \to \theta) \\ y = y \to \forall x\theta & \text{(QR): } (\{\psi \to \phi\}, \psi \to (\forall x\phi)),\ x \text{ is not free in } y = y \\ \forall x\theta & \text{(PC): } \big(y = y \wedge (y = y \to \forall x\theta)\big) \to \forall x\theta \end{array}$$

2. To prove $\Sigma \vdash \forall x\theta \to \Sigma \vdash \theta$, notice that $\theta^x_x = \theta$ and x is substitutable of x at θ. Assume $\Sigma \vdash \forall x\theta$.

$$\begin{array}{ll} \forall x\theta & \\ \forall x\theta \to \theta^x_x & \text{(Q1): } (\forall x\phi) \to \phi^x_t \\ \theta^x_x & \text{(PC): } \big(\forall x\theta \wedge (\forall x\theta \to \theta^x_x)\big) \to \theta^x_x \\ \theta & \end{array}$$

□

[2] Another way is to use

$$[(\forall y(y = y)) \vee \neg(\forall y(y = y))] \to \theta.$$

Then by (QR) with x is not free at $[(\forall y(y = y)) \vee \neg(\forall y(y = y))]$ to obtain:

$$[(\forall y(y = y)) \vee \neg(\forall y(y = y))] \to (\forall x\theta).$$

Then by (PC) with $[(\forall y(y = y)) \vee \neg(\forall y(y = y))]$ is a tautology, thus $\forall x\theta$ is a propositional consequence of the implication.

Remark 2.7.1. Lemma 2.7.1 seems strange: *Assume* Σ *is* $\{x = \overline{5}\}$.

Then certainly we have $\Sigma \vdash x = \overline{5}$, so by this lemma, we conclude $\Sigma \vdash (\forall x)(x = \overline{5})$. Does it mean that we have proved everything is equal to five?

If $x = \overline{5}$ is true in a model $\mathfrak{A}$, this means that $\mathfrak{A} \models x = \overline{5}$ for every assignment function s. So for every $a \in A$, there exists an assignment function s such that $s(x) = a$, it must be every element of A is equal to 5.

Our deduction of $(\forall x)(x = \overline{5})$ preserves the truth, but our *assumption that* Σ *is* $\{x = \overline{5}\}$ implicitly restricts the universe A to contain every element equal to 5.

Lemma 2.7.2 (add delete for-all quantifier equivalence in deduction)**.** Suppose that $\Sigma \vdash \theta$. If Σ' is formed by taking any $\sigma \in \Sigma$ and adding or deleting a universal quantifier whose scope is the entire formula, then $\Sigma' \vdash \theta$.

Proof. By Lemma 2.7.1, $\Sigma' \vdash \Sigma$, so as long as $\Sigma \vdash \theta$, we have $\Sigma' \vdash \theta$. □

Remark 2.7.2. Lemma 2.7.2 implies: if we know $\Sigma \vdash \theta$, we can assume every element of Σ is a sentence by quoting Lemma 2.7.2 several times to replace each $\sigma \in \Sigma$ with its universal closure (to make the variable not free).

The following theorem says that there is a deduction of ϕ from the assumption θ if and only if there is a deduction of the implication $\theta \to \phi$.

Theorem 2.7.2 (The Deduction Theorem)**.** Suppose θ is a sentence and Σ is a set of formulas. Then

$$\Sigma \cup \{\theta\} \vdash \phi \iff \Sigma \vdash (\theta \to \phi).$$

Proof. To prove Theorem 2.7.2:

1. To show $\Sigma \vdash (\theta \to \phi) \Rightarrow \Sigma \cup \{\theta\} \vdash \phi$:

   Assume $\Sigma \vdash (\theta \to \phi)$, then obviously $\Sigma \cup \{\theta\} \vdash (\theta \to \phi)$. Together with $\Sigma \cup \{\theta\} \vdash \theta$, we have $\Sigma \cup \{\theta\} \vdash \phi$ by (PC): ϕ is a propositional consequence of θ and $(\theta \to \phi)$.

2. To show $\Sigma \cup \{\theta\} \vdash \phi \Rightarrow \Sigma \vdash (\theta \to \phi)$:

   Assume $C = \{\phi | \Sigma \vdash (\theta \to \phi)\}$. If we can show that C contains $\Sigma \cup \{\theta\}$, C contains all the nonlogical axioms Λ, and C is closed under the rules of inference, then by proposition 2.2.1 we know that the set $theorem_{\Sigma \cup \{\theta\}} = \{\phi | \Sigma \cup \{\theta\} \vdash \phi\} \subseteq C$. In other words, we know that if $\Sigma \cup \{\theta\} \vdash \phi$, then $\Sigma \vdash (\theta \to \phi)$ by $\phi \in C$.

(a) To show $\Sigma \subseteq C$: Let $\sigma \in \Sigma$, then $\Sigma \vdash \sigma$. By (PC): $\theta \to$ True is True (for either θ is False or True), we have $\Sigma \vdash (\theta \to \sigma)$.

(b) To show $\theta \in C$: $\Sigma \vdash \theta \to \theta$ as $\theta \to \theta$ is a tautology.

(c) To show $\Lambda \subseteq C$: the nonlogical axiom $\lambda \in \Lambda$ is True in Σ. Thus, $\Sigma \vdash (\theta \to \lambda)$. Thus, $\Lambda \subseteq C$.

(d) Let RofI (Γ, θ) with $\Gamma \subseteq C$. Let $\Gamma = \{\gamma_1, \dots, \gamma_n\}$, we have $\Sigma \vdash (\theta \to \gamma_i)$.

- For (PC) at Definition 2.4.5: ϕ is a propositional consequence of $\Gamma = \{\gamma_1, \dots, \gamma_n\}$. We want to show $\phi \in C$.
  Then, $(\theta \to \phi)$ is a propositional consequence of
  $$\Gamma' = \{(\theta \to \gamma_1), \dots, (\theta \to \gamma_n)\}.$$
  $\Sigma \vdash (\theta \to \gamma_i)$ implies $\Sigma \vdash (\theta \to \phi)$. Therefore, we have $\phi \in C$.
- For (QR) at Definition 2.4.6: here we check universal rule: $\Big(\{\psi \to \phi\},\ \psi \to (\forall x \phi)\Big)$ with x is not free at ψ. Denote $\Gamma = \{\psi \to \phi\}$ and $\alpha :\equiv \psi \to (\forall x \phi)$. We want to show $\alpha \in C$.
  Assume $\Gamma \subseteq C$.
  $$\begin{array}{ll} \Gamma \subseteq C & \\ \Rightarrow \Sigma \vdash \theta \to (\psi \to \phi) & \\ \Rightarrow \Sigma \vdash (\theta \wedge \psi) \to \phi & \text{(PC)} \\ \Rightarrow \Sigma \vdash (\theta \wedge \psi) \to \forall x \phi & \text{(QR): } \theta \text{ is a sentence,} \\ & \text{so } x \text{ is not free in } \theta, \\ & \text{and } x \text{ is not free at } \psi \\ \Rightarrow \Sigma \vdash \theta \to \big(\psi \to \forall x \phi\big) & \text{(PC)} \\ \Rightarrow \alpha \in C & \text{By } \alpha :\equiv \psi \to (\forall x \phi) \end{array}$$
  (QR) existence rule works out analogously.

Thus, Proposition 2.2.1 applies.

□

2.8 Nonlogical Axioms

Here we show some examples of sets of nonlogical axioms.

Example 2.8.1. Nonlogical Axioms of linear algebra: Suppose the language $\mathcal{L}$ consisting of one binary function symbol $\oplus$ and infinitely many unary function symbols $c\cdot$, one for each real number c. Also $\mathcal{L}$ has one constant symbol 0. Then one way to list the nonlogical axioms of a vector space is as follows:

1. $(\forall x)(\forall y) x \oplus y = y \oplus x$.
2. $(\forall x)(\forall y)(\forall z) x \oplus (y \oplus z) = (x \oplus y) \oplus z$.
3. $(\forall x) x \oplus 0 = x$.
4. $(\forall x)(\exists y) x \oplus y = 0$.
5. $(\forall x) 1 \cdot x = x$.
6. $(\forall x)(c_1 c_2) \cdot x = c_1 \cdot (c_2 \cdot x)$.
7. $(\forall x)(\forall y) c \cdot (x \oplus y) = c \cdot x \oplus c \cdot y$.
8. $(\forall x)(c_1 + c_2) \cdot x = c_1 \cdot x \oplus c_2 \cdot x$.

Example 2.8.2. Nonlogical Axioms for dense linear order without endpoints: Suppose the language $\mathcal{L}$ is $\{<\}$. The nonlogical axioms are:

1. $(\forall x)(\forall y)(x < y \vee x = y \vee y < x)$.
2. $(\forall x)(\forall y)[x = y \to \neg x < y]$.
3. $(\forall x)(\forall y)(\forall z)[(x < y \wedge y < z) \to x < z]$.
4. $(\forall x)(\forall y)[x < y \to ((\exists z)(x < z \wedge z < y))]$.
5. $(\forall x)(\exists y)(\exists z)(y < x \wedge x < z)$.

Example 2.8.3. Nonlogical Axioms for number theory. Suppose the language $\mathcal{L}_{NT} = \{0, S, +, \cdot, E, <\}$. The set of nonlogical axioms, called N **(Robinson Arithmetic)**, is a minimal set of assumptions to describe a bare-bones version of the usual operations on the set of natural numbers:

1. $(\forall x) \neg Sx = 0$.
2. $(\forall x)(\forall y)[Sx = Sy \to x = y]$.
3. $(\forall x) x + 0 = x$.
4. $(\forall x)(\forall y) x + Sy = S(x + y)$.

5. $(\forall x)x \cdot 0 = 0$.
6. $(\forall x)(\forall y)x \cdot Sy = (x \cdot y) + x$.
7. $(\forall x)xE0 = S0$.
8. $(\forall x)(\forall y)xE(Sy) = (xEy) \cdot x$.
9. $(\forall x)\neg x < 0$.
10. $(\forall x)(\forall y)[x < Sy \leftrightarrow (x < y \vee x = y)]$.
11. $(\forall x)(\forall y)[(x < y) \vee (x = y) \vee (y < x)]$.

- The standard structure for $\mathcal{L}_{NT}$ is:

  $$\mathfrak{N} = (\mathbb{N}, 0^{\mathfrak{N}}, S^{\mathfrak{N}}, +^{\mathfrak{N}}, \cdot^{\mathfrak{N}}, E^{\mathfrak{N}}, <^{\mathfrak{N}}), \text{ or in short } = (\mathbb{N}, 0, S, +, \cdot, E, <)$$

- If a is a natural number that $a \in \mathbb{N}$, then $\overline{a}$ is the $\mathcal{L}_{NT}$-term $\underbrace{SSS \ldots S}_{a\,S's}0$.

  In other words, $\overline{a}$ is the canonical term (symbol) in the language that refers to the natural number a in the universe $\mathbb{N}$ of $\mathcal{L}_{NT}$.

Lemma 2.8.1. For natural numbers $a, b \in \mathbb{N}$.

1. If $a = b$ (holds in $\mathbb{N}$), then $N \vdash \overline{a} = \overline{b}$.
2. If $a \neq b$, then $N \vdash \overline{a} \neq \overline{b}$.
3. If $a < b$, then $N \vdash \overline{a} < \overline{b}$.
4. If $a \not< b$, then $N \vdash \overline{a} \not< \overline{b}$.
5. $N \vdash \overline{a} + \overline{b} = \overline{a + b}$.
6. $N \vdash \overline{a} \cdot \overline{b} = \overline{a \cdot b}$.
7. $N \vdash \overline{a}E\overline{b} = \overline{a^b}$.

N is strong enough to prove these statements in Lemma 2.8.1, but it is not strong enough to prove the commutative property:

Example 2.8.4. N does not prove addition is commutative.

Proof. Consider $\mathcal{L}_{NT}$ as $\{0, S, +\}$. We narrow our scope to prove the first 4 axioms of N at 2.8.3 do not prove that addition is commutative:

1. $N_1 :\equiv (\forall x)\neg Sx = 0$.
2. $N_2 :\equiv (\forall x)(\forall y)[Sx = Sy \to x = y]$.
3. $N_3 :\equiv (\forall x)x + 0 = x$.
4. $N_4 :\equiv (\forall x)(\forall y)x + Sy = S(x + y)$.

In the following we construct a model $\mathfrak{A}$ such that

$$\mathfrak{A} \models \{N_1, N_2, N_3, N_4\} \text{ and } \mathfrak{A} \not\models (\forall x)(\forall y)x + y = y + x :$$

- The universe of $\mathfrak{A}$, denoted by A, is the natural numbers extended by two non-standard numbers: $A = \mathbb{N} \cup \{\alpha, \beta\}$.
- The constant $0^{\mathfrak{A}} = 0$.
- Let the unary function S in $\mathfrak{A}$ as
$$S^{\mathfrak{A}}(x) = \begin{cases} x + 1 & \text{if } x \in \mathbb{N} \\ x & \text{if } x \in \{\alpha, \beta\}. \end{cases}$$
- Let the binary function $+$ in $\mathfrak{A}$ as
$$x +^{\mathfrak{A}} y = \begin{cases} x + y & \text{if } x, y \in \mathbb{N} \\ x & \text{if } x \in \{\alpha, \beta\} \text{ and } y \in \mathbb{N} \\ y & \text{otherwise.} \end{cases}$$

By our construction, we have:

$$\alpha +^{\mathfrak{A}} \beta = \beta \neq \alpha = \beta +^{\mathfrak{A}} \alpha$$

and thus $\mathfrak{A} \not\models (\forall x)(\forall y)x + y = y + x$.

On the other hand, it is obvious that $\mathfrak{A} \models \{N_1, N_2, N_3\}$. We need to verify $\mathfrak{A} \models N_4$ by arguing for all x, $y \in \mathbb{N} \cup \{\alpha, \beta\}$:

$$x +^{\mathfrak{A}} S^{\mathfrak{A}}(y) = S^{\mathfrak{A}}(x +^{\mathfrak{A}} y) \qquad (*)$$

- It is obvious that (*) holds for $x, y \in \mathbb{N}$.
- For $x \in \{\alpha, \beta\}$ and $y \in \mathbb{N}$: LHS of (*) is $x +^{\mathfrak{A}} S^{\mathfrak{A}}(y) = x$, RHS of (*) is $S^{\mathfrak{A}}(x +^{\mathfrak{A}} y) = S^{\mathfrak{A}}(x) = x$, thus (*) holds.
- For $x \in \mathbb{N} \cup \{\alpha, \beta\}$ and $y \in \{\alpha, \beta\}$: LHS of (*) is $x +^{\mathfrak{A}} S^{\mathfrak{A}}(y) = x +^{\mathfrak{A}} y = y$. RHS of (*) is $S^{\mathfrak{A}}(x +^{\mathfrak{A}} y) = S^{\mathfrak{A}}(y) = y$. Thus, (*) holds.

□

2.9 Recap

This chapter shows:

- The Soundness Theorem shows that deductions preserve truth: what we deduce ($\vdash$) (under our deduction system) is true ($\models$).
- Deduction Theorem indicates the deduction behaves like proofs behave.

Chapter 3

Completeness and Compactness

3.1 Naively

We have established a particular deductive system, with logical axioms Λ and rules of inference (Γ, θ).

Consider Σ a set of $\mathcal{L}$-formulas denoted as nonlogical axioms, and ϕ is an $\mathcal{L}$-formula. The Soundness Theorem 2.5.3:

$$\text{If } \Sigma \vdash \phi, \text{ then } \Sigma \models \phi,$$

This Theorem shows that the deductive system preserves truth: if ϕ is deduced from Σ, then ϕ is *True* in any model of Σ.

The Completeness Theorem is this chapter will give us the reverse of the Soundness Theorem:

$$\text{If } \Sigma \models \phi, \text{ then } \Sigma \vdash \phi.$$

However, this is **NOT** saying that we are able to prove any statement of first-order logic that is a true statement (about the natural numbers in the language $\mathcal{L}_{NT}$). What Completeness Theorem says is the deductive system is complete by the following Definition 3.1.1.

Definition 3.1.1. A deductive system with a collection of logical axioms Λ and a collection of rules of inference is **complete** if for every set of nonlogical axioms Σ and every $\mathcal{L}$-formula ϕ,

$$\text{If } \Sigma \models \phi, \text{ then } \Sigma \vdash \phi.$$

- What Definition 3.1.1 says is that if ϕ is an $\mathcal{L}$-formula that is *True in every model of* Σ, then there will be a deduction from Σ of ϕ.

3.2 Completeness Theorem

3.2.1 The Theorem

- Fix the collection of nonlogical axioms Σ and an $\mathcal{L}$-formula ϕ.
- Assume $\mathcal{L}$ is countable: so $\mathcal{L}$-formulas can be enumerated as an infinite list: $\alpha_1, \alpha_2, \ldots$.

What we want to prove is the following Theorem 3.2.1:

Theorem 3.2.1 (Completeness Theorem)**.** Σ is a set of $\mathcal{L}$-formula and ϕ is an $\mathcal{L}$-formula. If $\Sigma \models \phi$, then $\Sigma \vdash \phi$.

Definition 3.2.1. Let Σ be a set of $\mathcal{L}$-formulas. Then Σ is **inconsistent** if there is a deduction from Σ of $[(\forall x)x = x] \wedge \neg[(\forall x)x = x]$.

Definition 3.2.2. Σ is **consistent** if it is *not inconsistent* in Definition 3.2.1.

- Definition 3.2.1 says Σ is inconsistent if Σ proves a contradiction.
- If Σ is inconsistent, then there is a deduction from Σ of every $\mathcal{L}$-formula: If Σ is inconsistent, and ϕ is a $\mathcal{L}$-formula, then $\Sigma \vdash \phi$.

Rephrasing the problem:

1. Assume ϕ is a sentence. **Why?**

   Lemma 2.7.1 states:
   $$\Sigma \vdash \theta \iff \Sigma \forall x \theta.$$

   Repeating this for all free variables in ϕ, say $x_1, \ldots, x_n$, then
   $$\Sigma \vdash \phi \iff \Sigma \vdash \forall x_1, \ldots, x_n \, \phi$$

   where $\forall x_1, \ldots, x_n \, \phi$ is a sentence.

2. Assume Σ contains ONLY sentences. **Why?**

   By Lemma 2.7.2: Suppose that $\Sigma \vdash \theta$. If Σ' is formed by taking any $\sigma \in \Sigma$ and adding or deleting a universal quantifier whose scope is the entire formula, then $\Sigma' \vdash \theta$.

3. Denote the symbol $\perp$ to represent the contradictory sentence $[(\forall x)x = x] \wedge \neg[(\forall x)x = x]$. $\perp$ is a sentence that is in every language and is true in **no** structure.

   Then we say Σ is consistent if $\Sigma \nvdash \perp$.

4. Assume ϕ is $\perp$. **Why?**

   Given $\Sigma \models \perp \rightarrow \Sigma \vdash \perp$, we can prove $\Sigma \models \phi \rightarrow \Sigma \vdash \phi$:

$$\begin{aligned}
\Sigma \models \phi \quad &\Rightarrow \quad \Sigma \cup (\neg\phi) \models \perp \text{ because no models of } \Sigma \cup (\neg\phi) \\
&\Rightarrow \quad \Sigma \cup (\neg\phi) \vdash \perp \\
&\Rightarrow \quad \Sigma \vdash (\neg\phi \rightarrow \perp) \text{ by Deduction Theorem 2.7.2} \\
&\Rightarrow \quad \Sigma \vdash \phi \text{ by (PC) with tautology } (\neg P \rightarrow False) \rightarrow P
\end{aligned}$$

   The last derivation uses the statement that for a sentence η,

$$\Sigma \vdash \eta \iff \Sigma \cup \{\neg\eta\} \vdash \perp .$$

   We will prove this statement by the following Lemma 3.2.1.

5. Therefore, to prove the Completeness Theorem, we need to prove:

$$\Sigma \models \perp \rightarrow \Sigma \vdash \perp$$

   for Σ any set of sentences.

6. By contrapositive

$$\Sigma \nvdash \perp \rightarrow \Sigma \not\models \perp,$$

   that is to prove the **Model Existence Theorem**:

   If Σ is a consistent set of sentences, then Σ has a model.

7. In other words, Σ is a consistent set of $\mathcal{L}$-sentences, our goal is to construct a model. We roughly do it in 3 steps.

STEP 1 We expand both $\mathcal{L}$ and Σ to $\mathcal{L}'$ and Σ', such as $\mathcal{L}' = \mathcal{L} \cup \{\text{lots of constants}\}$

STEP 2 We build a model $\mathfrak{M}$ of Σ' in $\mathcal{L}'$. The universe is essentially the set of variable-free terms of $\mathcal{L}'$.

STEP 3 We restrict $\mathfrak{M}$ to $\mathcal{L}$.

3.2.2 The Proof

Lemma 3.2.1. For a sentence η,

$$\Sigma \vdash \eta \iff \Sigma \cup \{\neg\eta\} \vdash \perp .$$

Proof. Assume $\Sigma \vdash \eta$. Then $\Sigma \cup \{\neg\eta\} \vdash \eta$. Obviously, we have $\Sigma \cup \{\neg\eta\} \vdash \neg\eta$. So by (PC), we have $\Sigma \cup \{\neg\eta\} \vdash \{\eta$ and $\neg\eta\} \vdash \perp$.

Assume $\Sigma \cup \{\neg\eta\} \vdash \perp$. By the Deduction Theorem 2.7.2, we have $\Sigma \vdash \neg\eta \rightarrow \perp$. $\{\neg\eta \rightarrow \perp\} \rightarrow \eta$ is a tautology. Thus, by (PC), we have $\Sigma \vdash \{\neg\eta \vdash \perp\} \rightarrow \eta$. □

A quick historical recap:

- The Completeness Theorem was originated by Kurt Gödel.
- The proof presented here is by Leon Henkin. It is involved with using the so-called *Henkin Structure*. The following Example 3.2.1 gives some detail on how this structure looks like.

Example 3.2.1 (Henkin Structure)**.** Let $\mathcal{L}$ be $\{0, f, g, R\}$, where 0 is a constant, f is a unary function symbol, g is a binary function symbol, and R is a 3-ary relation symbol.

Define a $\mathcal{L}$-structure $\mathfrak{B}$ as follows:

- The universe B is the set of all variable-free $\mathcal{L}$-terms.
- The constant $0^{\mathfrak{B}}$ is the term 0.
- The functions $f^{\mathfrak{B}}$ and $g^{\mathfrak{B}}$ are defined as adding f and g in front of the terms such that $f^{\mathfrak{B}}(t)$ is ft and $g^{\mathfrak{B}}(t, s)$ is gts with t and s are variable-free terms.
- For the 3-ary relation symbol, define a relation $R^{\mathfrak{B}}$ that is a subset of B^3. We can arbitrarily define as:

$$R^{\mathfrak{B}} = \{(r, s, t) \in B^3 | \text{ the total number of function symbols in } r,\ s,\ t \text{ is even } \}.$$

STEP 1

1. Let $\mathcal{L}_0 = \mathcal{L}$. Define :

$$\mathcal{L}_1 = \mathcal{L}_0 \cup \{c_1, c_2, \dots\}$$

where each c_i is a constant symbol that is *not* in $\mathcal{L}_0$.

- Remark: The constants c_i added to form $\mathcal{L}_1$ are called **Henkin constants**. These constants allow us to ensure whenever Σ contains $\exists x\phi(x)$, in our constructed model $\mathfrak{A}$, there will be an element in constant symbol c such that $\mathfrak{A} \models \phi(c)$.[1]

Lemma 3.2.2. If Σ is a consistent set of $\mathcal{L}_0$-sentences and $\mathcal{L}_1$ is an extension by constants of $\mathcal{L}_0$, then Σ is consistent as $\mathcal{L}_1$-sentences.

Proof. Prove by contradiction. Suppose Σ is not consistent as $\mathcal{L}_1$-sentences. Then there exists $D_1 = (\alpha_1, \dots, \alpha_m = \perp)$ as a deduction of $\perp$, with α_i an $\mathcal{L}_1$-formula.

- Then since the deduction is by definition finite, there exists k such that the new *Henkin constants* occurring in D_1 are c_1, c_2, …, c_k.
- c_1, c_2, …, c_k are Henkin constants, so we can find corresponding variables v_1, v_2, …, v_k which c_1, c_2, …, c_k replaced for. This implies that these variables v_1, v_2, …, v_k are not in D_1.
- Now define

$$D_0 = \left(\left(\alpha_1\right)^{c_1,\dots,c_k}_{v_1,\dots,v_k}, \dots, \left(\alpha_m\right)^{c_1,\dots,c_k}_{v_1,\dots,v_k} \right)$$

  Remember $\alpha_m = \perp$, so $\left(\alpha_m\right)^{c_1,\dots,c_k}_{v_1,\dots,v_k} = \perp$. Here we replace (backward) the Henkin constants c_1, c_2, …, c_k by the variables v_1, v_2, …, v_k only exist in $\mathcal{L}_0$ but not in $\mathcal{L}_1$.

[1]Here $\phi(c)$ is the result of replacing the free occurrence x in $\exists x\phi(x)$ by c, so $\phi(c)$ is actually ϕ^x_c, or write as $\phi|_{x=c}$.

- We can show $D_0 \vdash \perp$ and D_0 is in $\mathcal{L}_0$: Denote the elements of $D_0 = \{\beta_1, \ldots, \beta_m\}$ where $\beta_i = \big(\alpha_i\big)^{c_1,\ldots,c_k}_{v_1,\ldots,v_k}$ and in particular $\beta_m = \big(\alpha_m\big)^{c_1,\ldots,c_k}_{v_1,\ldots,v_k} = \perp$. Use induction on the elements of D_1: α_i can be element of Σ, or logical equality axioms (E1), (E2), (E3); α_i can be logical quantifier axioms (Q1) or (Q2); α_i can be the result of RofI (Γ, α_i).
  - α_i is the element of Σ, or logical equality axioms (E1), (E2), (E3): Substituting c_1, …, c_k by v_1, …, v_k on α_i keeps β_i as the element of Σ or the equality axioms. Thus, β_i is an element of the deduction in D_0 by the same reason as α_i in D_1.
  - α_i can be logical quantifier axioms (Q1) or (Q2): for example, if α_i is $(\forall x)\theta' \to {\theta'}^x_{t'}$, then β_i as $(\forall x)\theta \to \theta^x_t$ is also a quantifier axiom, given θ is ${\theta'}^{c_1,\ldots,c_k}_{v_1,\ldots,v_k}$, t' is substitutable for x and t as ${t'}^{c_1,\ldots,c_k}_{v_1,\ldots,v_k}$ is substitutable for x. Thus, β_i is an element of the deduction in D_0 by the same reason as α_i in D_1.
  - α_i is the result of RofI (Γ, α_i) in $\mathcal{L}_1$-sentences, then ($\Gamma^{c_1,\ldots,c_k}_{v_1,\ldots,v_k}$, β_i) is the RofI that derives β_i in $\mathcal{L}_0$-sentences.
- Therefore, we proof this lemma by contradiction.

□

2. Expand Σ.

- List all $\mathcal{L}_0$-sentences of the form $\exists x\theta$:

$$\exists x_1\theta_1, \exists x_2\theta_2, \ldots$$

  e.g. we have $\mathcal{L}_0$-sentences $\{\exists v_1(v_1 \le v_1),\ \exists v_2(\forall v_3(v_2 \ne v_3)),\ \exists v_4(v_4 > 0)\}$, then we take x_1 to be v_1, x_2 to be v_2, x_3 to be v_4, and $\theta_1 :\equiv (v_1 \le v_1)$, $\theta_2 :\equiv (\forall v_3(v_2 \ne v_3))$, $\theta_3 :\equiv (v_4 > 0)$.
- Define

$$\begin{aligned} \phi_i &:\equiv (\exists x_i\theta_i) \to (\theta_i)^{x_i}_{c_i}, \quad i = 1, 2, \ldots \\ H_1 &= \{\phi_i | i \ge 1\} \end{aligned}$$

  where ϕ_i is **Henkin axiom** that says "c_i is a witness for $\exists x_i\theta_i$". Notice c_i is Henkin constant of $\mathcal{L}_1$.
- Let $\Sigma_0 = \Sigma$, and define $\Sigma_1 = \Sigma_0 \cup H_1$. By doing this, we add to Σ countably many Henkin axioms.

Lemma 3.2.3. If Σ_0 is a consistent set of sentences and Σ_1 is created by adding Henkin Axioms to Σ_0, then Σ_1 is consistent as $\mathcal{L}_1$-sentences.

Proof. Prove by contradiction, that is Σ_1 is not consistent as $\mathcal{L}_1$-sentences. Then let m be smallest integer such that

$$\Sigma \cup \{\phi_1, \dots, \phi_m, \phi_{m+1}\}$$

is inconsistent, where ϕ_i is Henkin axiom for $i = 1, \dots, m+1$. m exists because the deduction is finite.

So we have:

$$\begin{aligned}
& \Sigma \cup \{\phi_1, \dots, \phi_{m+1}\} \vdash \perp \\
\Rightarrow\quad & \Sigma \cup \{\phi_1, \dots, \phi_m\} \vdash (\phi_{m+1} \to \perp) \quad \text{by Deduction Theorem 2.7.2} \\
\Rightarrow\quad & \Sigma \cup \{\phi_1, \dots, \phi_m\} \vdash (\neg\phi_{m+1} \vee \perp) \quad \text{by } a \to b \iff \neg a \vee b \\
\Rightarrow\quad & \Sigma \cup A \vdash (\neg\phi_{m+1} \vee \perp) \quad \text{by denoting } A = \{\phi_1, \dots, \phi_m\} \\
\Rightarrow\quad & \Sigma \cup A \vdash \neg\phi_{m+1} \\
& \quad \phi_{m+1} :\equiv \exists x\theta \to \theta^x_c \text{ so } \phi_{m+1} :\equiv \neg\exists x\theta \vee \theta^x_c \\
\Rightarrow\quad & \Sigma \cup A \vdash \neg(\neg\exists x\theta \vee \theta^x_c) \\
\Rightarrow\quad & \Sigma \cup A \vdash (\exists x\theta \wedge \neg\theta^x_c) \\
& \quad \text{Example 1.1.1 gives } \exists x\theta \iff \neg\forall x\neg\theta \\
\Rightarrow\quad & \Sigma \cup A \vdash \neg\forall x\neg\theta \text{ and } \Sigma \cup A \vdash \neg\theta^x_c
\end{aligned}$$

Keep $\Sigma \cup A \vdash \neg\forall x\neg\theta$.

Focus on $\Sigma \cup A \vdash \neg\theta^x_c$. Analogous to the proof in Lemma 3.2.2, we can take a deduction of $\neg\theta^x_c$ and replace each Henkin constant c by a new variable z, the result is still a deduction. Thus,

$\Sigma \cup A \vdash \neg\theta^x_c$

$\Rightarrow$ $\Sigma \cup A \vdash \neg\theta^x_z$

$\Rightarrow$ $\Sigma \cup A \vdash (\forall z)(\neg\theta^x_z)$ by Lemma 2.7.1: $\Sigma \vdash \theta \iff \Sigma \vdash \forall x\theta$

Axiom Q1: $(\forall x)(\phi) \to \phi^x_t$ if t is substitutable for x in ϕ
leads to $(\forall z)(\neg\theta^x_z) \to \neg(\theta^x_z)^z_x$,
since x is substitutable for z (remember z is a new variable)

$\Rightarrow$ $\Sigma \cup A \vdash \neg(\theta^x_z)^z_x$

$\Rightarrow$ $\Sigma \cup A \vdash \neg\theta$

$\Rightarrow$ $\Sigma \cup A \vdash \forall x\neg\theta$ by Lemma 2.7.1: $\Sigma \vdash \theta \iff \Sigma \vdash \forall x\theta$

$\Rightarrow$ $\Sigma \cup A \vdash \perp$ together by $\Sigma \cup A \vdash \neg\forall x\neg\theta$

$\Rightarrow$ $\Sigma \cup \{\phi_1, \ldots, \phi_m\}$ is inconsistent

But m is the smallest integer, contradiction observes. Thus, Σ_1 is consistent. □

3. Repeat

- $\mathcal{L}_0 \subseteq \mathcal{L}_1 \subseteq \mathcal{L}_2 \subseteq \ldots$: extend with countably many new Henkin constants each time.
- $\Sigma_0 \subseteq \Sigma_1 \subseteq \Sigma_2 \subseteq \ldots$: for Σ_i to Σ_{i+1}, extend with Henkin Axioms H_{i+1} for the $\mathcal{L}_i$-sentences.
- Define
$$\mathcal{L}' = \bigcup_i \mathcal{L}_i \text{ and } \hat{\Sigma} = \bigcup_i \Sigma_i.$$

Lemma 3.2.4. $\hat{\Sigma}$ is consistent as $\mathcal{L}'$-sentences.

Proof. Prove by contradiction. Assume $\hat{\Sigma} = \bigcup_i \Sigma_i$ is not consistent. This means $\bigcup_i \Sigma_i \vdash \perp$.

Then there exists finite set $\Gamma \subseteq \bigcup_i \Sigma_i$ and $\Gamma \vdash \perp$.

Choose k large enough such that $\Gamma \subseteq \Sigma_k$, then

$$\Gamma \vdash \perp \text{ implies } \Sigma_k \vdash \perp .$$

Contradiction observes as Σ_k is consistent.

□

4. Further extend $\hat{\Sigma}$ to Σ' to ensure either $\sigma \in \Sigma'$ or $\neg\sigma \in \Sigma'$ for all $\mathcal{L}'$-sentences σ:

   - Enumerate all $\mathcal{L}'$-sentences:
   $$\sigma_1, \sigma_2, \ldots$$
   - Let $\Sigma^0 = \hat{\Sigma}$.
   - $\Sigma^{k+1} = \begin{cases} \Sigma^k \cup \{\sigma_k\} & \text{if } \Sigma^k \cup \{\sigma_k\} \text{ is consistent} \\ \Sigma^k \cup \{\neg\sigma_k\} & \text{otherwise} \end{cases}$
   - $\Sigma' = \bigcup_k \Sigma^k$.

   Lemma 3.2.5. Σ^{k+1} is consistent.

   Proof. This is to prove either $\Sigma^k \cup \{\sigma_k\}$ is consistent or $\Sigma^k \cup \{\neg\sigma_k\}$ is consistent.

   Prove by contradiction. Assume both $\Sigma^k \cup \{\sigma_k\}$ and $\Sigma^k \cup \{\neg\sigma_k\}$ are inconsistent. This means
   $$\Sigma^k\{\sigma_k\} \vdash\perp \text{ and } \Sigma^k\{\neg\sigma_k\} \vdash\perp .$$
   By Deduction Theorem 2.7.2, we have
   $$\Sigma^k \vdash \sigma_k \rightarrow\perp \text{ and } \Sigma^k \vdash \neg\sigma_k \rightarrow\perp .$$
   This means
   $$\Sigma^k \vdash \{\sigma_k \rightarrow\perp, \neg\sigma_k \rightarrow\perp\} \vdash\perp .$$
   This contradicts Σ^k is consistent. Therefore, either $\Sigma^k \cup \{\sigma_k\}$ is consistent or $\Sigma^k \cup \{\neg\sigma_k\}$ is consistent. If $\Sigma^k \cup \{\sigma_k\}$ is inconsistent, then $\Sigma^k \cup \{\neg\sigma_k\}$ is consistent.

   [Why not both $\Sigma^k \cup \{\sigma_k\}$ and $\Sigma^k \cup \{\neg\sigma_k\}$ are consistent? If they are, then $\Sigma^k \cup \{\sigma_k, \neg\sigma_k\}$ is consistent, but $\{\sigma_k, \neg\sigma_k\} \vdash\perp$.] □

5. Since every Σ^{k+1} is consistent, we have Σ' is consistent. And for every formula σ, either $\sigma \in \Sigma'$ or $\neg\sigma \in \Sigma'$. Therefore, we conclude STEP 1: we have obtained $\mathcal{L}'$ and Σ'. Σ' is consistent as $\mathcal{L}'$-sentences. For every σ as a $\mathcal{L}'$-sentence, either $\sigma \in \Sigma'$ or $\neg\sigma \in \Sigma'$.

   Lemma 3.2.6. If σ is a $\mathcal{L}'$-sentence, then $\sigma \in \Sigma' \iff \Sigma' \vdash \sigma$.

   Proof. $\sigma \in \Sigma' \Rightarrow \Sigma' \vdash \sigma$ is obvious.

   So assume $\Sigma' \vdash \sigma$. If $\sigma \notin \Sigma'$, then $\neg\sigma \in \Sigma'$. Thus $\Sigma' \vdash \sigma$ and $\Sigma' \vdash \neg\sigma$, so $\Sigma' \vdash\perp$. But Σ' is consistent, contradiction observes.

   □

STEP 2: Constructing a model $\mathfrak{M}$ of Σ' in $\mathcal{L}'$-sentences.

1. Remember from Definition 1.4.1, an **$\mathcal{L}'$-model or $\mathcal{L}'$-structure** shall contain the universe, constants, functions, relations.

2. Idea on how to construct the universe M for the model $\mathfrak{M}$ (remember the universe is a set):

   - We start from the minimum: consider constant symbols only as elements of the universe: If we have function "+", we want $c_1 + c_2$ to be in our universe as well, but $c_1 + c_2$ is a term, not a constant. Thus, we can **NOT** use constant symbols only as elements of the universe.
   - What about using variable-free terms only as elements of universe, where a term by Definition 1.2.1? If we have the variable-free terms $c_1 + c_2$ and c_3, somewhere in Σ' there is a sentence $c_1 + c_2 = c_3$. Then Σ' demands in the universe $c_1 + c_2$ to be the same as c_3, but this is not possible because they are two different terms.
   - Solution: Use *equivalence class* of variable-free terms.

   Definition 3.2.3. Let T be the set of all variable-free $\mathcal{L}'$-terms. Define a relation $\sim$:
   $$t_1 \sim t_2 \iff (t_1 = t_2) \in \Sigma'$$

   Lemma 3.2.7. $\sim$ is an equivalence relation on T.

   Proof. It mainly uses Theorem 2.7.1: $\vdash x = x$, $\vdash x = y \to y = x$, $\vdash (x = y \land y = z) \to x = z$.

   We need to prove $\sim$ is symmetric, reflexive, transitive.

   To show symmetric, assume $t_1 \sim t_2$, we need to prove $t_2 \sim t_1$ that is to prove $(t_2 = t_1) \in \Sigma'$. From $t_1 \sim t_2$, we know $(t_1 = t_2) \in \Sigma'$ and thus $\Sigma' \vdash (t_1 = t_2)$ by Lemma 3.2.6. By Theorem 2.7.1, we know $\Sigma' \vdash (t_2 = t_1)$. Then by applying again Lemma 3.2.6, $\Sigma' \vdash (t_2 = t_1) \iff (t_2 = t_1) \in \Sigma'$.

   Reflexive means $t_1 \sim t_1 \iff (t_1 = t_1) \in \Sigma'$. By Theorem 2.7.1, we know $\Sigma' \vdash (t_1 = t_1)$ for any t_1 as $\mathcal{L}'$-sentence. By Lemma 3.2.6, $\Sigma' \vdash (t_1 = t_1) \iff (t_1 = t_1) \in \Sigma'$. Thus, we have $t_1 \sim t_1$.

Transitive means $t_1 \sim t_2$ and $t_2 \sim t_3$ implies $t_1 \sim t_3$. That is to prove $(t_1 = t_2) \in \Sigma'$ and $(t_2 = t_3) \in \Sigma'$ implies $(t_1 = t_3) \in \Sigma'$.

By Lemma 3.2.6, $(t_1 = t_2) \in \Sigma'$ and $(t_2 = t_3) \in \Sigma'$ imply that $\Sigma' \vdash (t_1 = t_2)$ and $\Sigma' \vdash (t_1 = t_3)$. By Theorem 2.7.1, $\Sigma' \vdash (t_1 = t_2 \wedge t_1 = t_3)$ implies $\Sigma' \vdash (t_1 = t_3)$. By Lemma 3.2.6, $\Sigma' \vdash (t_1 = t_3) \iff (t_1 = t_3) \in \Sigma'$, and thus $t_1 \sim t_3$.

□

3. **Universe** at the model $\mathfrak{M}$: $M = \{[t]\}$, where $[t]$ is the set of all variable-free terms s of the language $\mathcal{L}'$ such that $s \sim t$ according to Σ'.

   For example, if $\mathcal{L}'$ includes non-Henkin constant k inherited from $\mathcal{L}$, Henkin constants $c_1, c_2, \ldots$, a binary function symbol f, then the universe includes $[c_1]$ and $[f(k, c_2)]$.

4. **Constants** at the model $\mathfrak{M}$: $c^{\mathfrak{M}} = [c]$.

5. **Functions** at the model $\mathfrak{M}$:

$$f^{\mathfrak{M}}\big([t_1], [t_2], \ldots, [t_n]\big) = \big[f(t_1, t_2, \ldots, t_n)\big].$$

   For example, $[c_1] +^{\mathfrak{M}} [c_2] = [c_1 + c_2]$.

   Lemma 3.2.8. Function $f^{\mathfrak{M}}$ is well-defined.

   Proof. Assume $[t_1] = [t_2]$, we need to prove $f^{\mathfrak{M}}([t_1]) = f^{\mathfrak{M}}([t_2])$, which is $[f(t_1)] = [f(t_2)]$. By equivalence relation $\sim$ at Definition 3.2.3, we need to show if $(t_1 = t_2) \in \Sigma'$, then $\big(f(t_1) = f(t_2)\big) \in \Sigma'$.

$$\begin{aligned}
& x = y \to f(x) = f(y) \quad \text{by (E2)} \\
\Rightarrow\quad & \Sigma' \vdash \Big(x = y \to f(x) = f(y)\Big) \\
\Rightarrow\quad & \Sigma' \vdash \Big(\forall x \forall y (x = y \to f(x) = f(y))\Big) \quad \text{by Lemma 2.7.1} \\
\Rightarrow\quad & \Sigma' \vdash \Big(t_1 = t_2 \to f(t_1) = f(t_2)\Big) \\
& \quad \text{by (Q1) twice for } x \text{ by } t_1 \text{ and } y \text{ by } t_2 \\
& \text{with } (t_1 = t_2) \in \Sigma' \iff \Sigma' \vdash (t_1 = t_2) \quad \text{by Lemma 3.2.6} \\
\Rightarrow\quad & \Sigma' \vdash f(t_1) = f(t_2) \quad \text{by rule (PC) at Definition 2.4.5} \\
\Rightarrow\quad & \big(f(t_1) = f(t_2)\big) \in \Sigma' \quad \text{by Lemma 3.2.6}
\end{aligned}$$

□

6. **Relations**: $R^{\mathfrak{M}}\big([t_1], \dots, [t_n]\big) \iff Rt_1 \dots t_n \in \Sigma'$.

   Lemma 3.2.9. Relation $R^{\mathfrak{M}}\big([t_1], \dots, [t_n]\big) \iff Rt_1 \dots t_n \in \Sigma'$ is well-defined.

   Proof. Assume the relation is unary: $R^{\mathfrak{M}}([x]) \iff Rx \in \Sigma'$. We need to prove if $[t_1] = [t_2]$, then

   $$R^{\mathfrak{M}}([t_1]) \text{ is True } \iff R^{\mathfrak{M}}([t_2]) \text{ is True .}$$

   By equivalence relation $\sim$ at Definition 3.2.3, we need to show if $(t_1 = t_2) \in \Sigma'$, then $(Rt_1 = Rt_2) \in \Sigma'$.

   $$\begin{aligned}
   & x = y \to Rx = Ry \quad \text{by (E3)} \\
   \Rightarrow \quad & \Sigma' \vdash \Big(x = y \to Rx = Ry\Big) \\
   \Rightarrow \quad & \Sigma' \vdash \Big(\forall x \forall y (x = y \to Rx = Ry)\Big) \quad \text{by Lemma 2.7.1} \\
   \Rightarrow \quad & \Sigma' \vdash \Big(t_1 = t_2 \to Rt_1 = Rt_2\Big) \\
   & \quad \text{by (Q1) twice for } x \text{ by } t_1 \text{ and } y \text{ by } t_2 \\
   & \text{with } (t_1 = t_2) \in \Sigma' \iff \Sigma' \vdash (t_1 = t_2) \quad \text{by Lemma 3.2.6} \\
   \Rightarrow \quad & \Sigma' \vdash Rt_1 = Rt_2 \quad \text{by rule (PC) at Definition 2.4.5} \\
   \Rightarrow \quad & \big(Rt_1 = Rt_2\big) \in \Sigma' \quad \text{by Lemma 3.2.6}
   \end{aligned}$$

   □

   Proposition 3.2.1. $\mathfrak{M} \models \Sigma'$

   Proof. We prove $\sigma \in \Sigma' \iff \mathfrak{M} \models \sigma$ for all sentences σ $(\in \Sigma')$. We proceed by induction on the complexity of σ. σ is a sentence that is either $\sigma :\equiv t_1 = t_2$, $\sigma :\equiv R(t_1, t_2, \dots, t_n)$, $\sigma :\equiv (\neg\alpha)$, $\sigma :\equiv (\alpha \vee \beta)$, or $\sigma :\equiv (\forall x)(\phi)$ by Definition 1.2.2.

   A useful fact: If s is a variable assignment function (**vaf**) by Definition 1.5.1 into $\mathfrak{M}$ and t is a variable-free term, then the term assignment function by Definition 1.5.2 is $\overline{s}(t) = [t]$. [2]

[2] Term assignment function is for a variable t: $\overline{s}(t) = s(t) = [t]$, for a constant symbol c: $\overline{s}(c) = c^{\mathfrak{M}} = [c]$, for a function $ft_1 \dots t_n$: $\overline{s}(ft_1 \dots t_n) = f^{\mathfrak{M}}\big(\overline{s}(t_1), \dots, \overline{s}(t_n)\big) = f^{\mathfrak{M}}\big([t_1], \dots, [t_n]\big) = [f\big(t_1, \dots, t_n\big)]$ by construction of the model $\mathfrak{M}$.

(a) $\sigma :\equiv t_1 = t_2$. Since σ is a sentence, t_1 and t_2 are variable-free terms.
Assume $\sigma :\equiv t_1 = t_2 \in \Sigma'$. By definition 3.2.3,

$$\begin{aligned} & \sigma :\equiv t_1 = t_2 \in \Sigma' \\ \iff\ & t_1 \sim t_2 \\ \iff\ & [t_1] = [t_2] \\ \iff\ & \overline{s}(t_1) = \overline{s}(t_2) \text{ for all vaf } s \text{ into } \mathfrak{M} \\ \iff\ & \mathfrak{M} \models t_1 = t_2 \text{ by Definition 1.5.4} \end{aligned}$$

(b) $\sigma :\equiv R(t_1, t_2, \ldots, t_n)$, where t_1, t_2, …, t_n are variable free terms. Assume $\sigma :\equiv R(t_1, \ldots, t_n) \in \Sigma'$. By construction of the model $\mathfrak{M}$,

$$\begin{aligned} & \sigma :\equiv R(t_1, \ldots, t_n) \in \Sigma' \\ \iff\ & R^{\mathfrak{M}}\big([t_1], \ldots, [t_n]\big),\ \text{or}\ \big([t_1], \ldots, [t_n]\big) \in R^{\mathfrak{M}} \\ \iff\ & \big(\overline{s}(t_1), \ldots, \overline{s}(t_n)\big) \in R^{\mathfrak{M}} \text{ for all vaf } s \text{ into } \mathfrak{M} \\ \iff\ & \mathfrak{M} \models R(t_1, \ldots, t_n) \text{ by Definition 1.5.4} \end{aligned}$$

(c) $\sigma :\equiv \neg\alpha$, where by induction hyphothesis $\alpha \in \Sigma' \iff \mathfrak{M} \models \alpha$. Σ' is maximal and consistent, this means $\sigma \in \Sigma' \iff \alpha \notin \Sigma'$. Then,

$$\begin{aligned} & \sigma \in \Sigma' \\ \iff\ & \alpha \notin \Sigma' \\ \iff\ & \mathfrak{M} \not\models \alpha \\ \iff\ & \mathfrak{M} \models \neg\alpha \text{ by Definition 1.5.4} \\ \iff\ & \mathfrak{M} \models \sigma \end{aligned}$$

(d) $\sigma :\equiv \alpha \vee \beta$, where by induction hyphothesis $\alpha \in \Sigma' \iff \mathfrak{M} \models \alpha$ and $\beta \in \Sigma' \iff \mathfrak{M} \models \beta$. Then,

$$\begin{aligned} & \sigma \in \Sigma' \\ \iff\ & \alpha \in \Sigma' \text{ or } \beta \in \Sigma' \\ \iff\ & \mathfrak{M} \models \alpha \text{ or } \mathfrak{M} \models \beta \\ \iff\ & \mathfrak{M} \models (\alpha \vee \beta) \text{ by Definition 1.5.4} \\ \iff\ & \mathfrak{M} \models \sigma \end{aligned}$$

(e) $\sigma :\equiv (\forall x)(\phi)$, where by induction hyphothesis $\phi \in \Sigma' \iff \mathfrak{M} \models \phi$. We need to show $(\forall x)(\phi) \in \Sigma' \iff \mathfrak{M} \models (\forall x)(\phi)$. We work out $\iff$ by two ways $\Longrightarrow$ and $\Longleftarrow$:

$\Longrightarrow$: Assume $(\forall x)(\phi) \in \Sigma'$. We need to prove $\mathfrak{M} \models (\forall x)(\phi)$.

$$\begin{aligned}
& \mathfrak{M} \models \sigma \\
\iff\; & \mathfrak{M} \models (\forall x)(\phi) \\
\iff\; & \mathfrak{M} \models (\forall x)\phi[s] \text{ for any vaf } s \text{ into } \mathfrak{M} \\
\iff\; & \mathfrak{M} \models \phi\Big[s\big[x|m\big]\Big] \text{ for any } m \in M \\
& \quad \text{by Definition 1.5.4} \\
\iff\; & \mathfrak{M} \models \phi\Big[s\big[x|[t]\big]\Big] \text{ for any variable-free term } t \\
& \quad \text{(and thus } [t] \in M) \\
\iff\; & \mathfrak{M} \models \phi\Big[s\big[x|\overline{s}(t)\big]\Big] \text{ by the useful fact} \\
& t \text{ is variable-free} \\
& \quad \text{and thus is substitutable for } x \text{ in } \phi \\
\iff\; & \mathfrak{M} \models \phi^x_t[s] \text{ by Theorem 2.6.1} \\
\iff\; & \mathfrak{M} \models \phi^x_t \text{ since } \phi^x_t \text{ is variable-free} \\
\iff\; & \phi^x_t \in \Sigma' \text{ by induction hypothesis:} \\
& \quad \phi^x_t \text{ has fewer quantifiers than } \sigma \\
\iff\; & \Sigma' \vdash \phi^x_t \text{ by Lemma 3.2.6}
\end{aligned}$$

To conclude, we know $\sigma \in \Sigma'$, so $\Sigma' \vdash (\forall x)(\phi)$ by Lemma 3.2.6. Also, by (Q1), we have $\Sigma' \vdash (\forall x)(\phi) \to \phi^x_t$ since t is variable-free and thus is substitutable for x in ϕ. Thus, we have $\Sigma' \vdash \phi^x_t$, which means $\mathfrak{M} \models \sigma$.

$\Longleftarrow$: Assume $\sigma \notin \Sigma'$, we need to prove $\mathfrak{M} \not\models \sigma$.

$$\begin{aligned}
& \sigma \notin \Sigma' \\
\Rightarrow\ & \neg\sigma \in \Sigma' \text{ by maximality of } \Sigma' \\
\Rightarrow\ & \Sigma' \vdash \neg\sigma \text{ by Lemma 3.2.6} \\
\Rightarrow\ & \Sigma' \vdash \neg(\forall x)(\phi) \\
\Rightarrow\ & \Sigma' \vdash (\exists x)(\neg\phi) \text{ by Example 1.1.1: } (\exists x)\alpha = \neg(\forall x)(\neg\alpha) \\
\Rightarrow\ & (\exists x)(\neg\phi) \in \Sigma' \text{ by Lemma 3.2.6} \\
& \quad \text{Henkin axiom in } \Sigma' \text{ is as } (\exists x\theta) \to (\theta)^x_c \\
& \quad\quad \text{for some Henkin constant } c \\
\Rightarrow\ & \Big((\exists x)(\neg\phi) \to (\neg\phi)^x_c\Big) \in \Sigma' \\
\Rightarrow\ & \Sigma' \vdash \Big((\exists x)(\neg\phi) \wedge \big((\exists x)(\neg\phi) \to (\neg\phi)^x_c\big)\Big) \\
& \quad\quad \text{by Lemma 3.2.6} \\
\Rightarrow\ & \Sigma' \vdash (\neg\phi)^x_c \\
\Rightarrow\ & (\neg\phi)^x_c \in \Sigma' \text{ by Lemma 3.2.6} \\
\Rightarrow\ & \mathfrak{M} \models (\neg\phi)^x_c \text{ by induction hypothesis:} \\
& \quad\quad (\neg\phi)^x_c \text{ has fewer quantifiers} \\
\Rightarrow\ & \mathfrak{M} \models (\exists x)(\neg\phi) \text{ by (Q2)} \\
\Rightarrow\ & \mathfrak{M} \not\models \neg(\exists x)(\neg\phi) \\
\Rightarrow\ & \mathfrak{M} \not\models (\forall x)(\phi) \text{ by } (\exists x)(\neg\phi) \iff \neg(\forall x)(\phi)
\end{aligned}$$

□

STEP 3: Restrict the model $\mathfrak{M}$ (of Σ' in $\mathcal{L}'$-sentences) to $\mathcal{L}$.

1. By Proposition 3.2.1, we know $\mathfrak{M} \models \Sigma'$.
2. We know $\Sigma \subseteq \Sigma'$.
3. We know $\mathfrak{M}$ is an $\mathcal{L}'$-structure. We want to restrict $\mathfrak{M}$ back to the original $\mathcal{L}$-language, denoted by $\mathfrak{M}|_{\mathcal{L}}$.

   Notice the difference between $\mathcal{L}$ and $\mathcal{L}'$ is on those Henkin constant symbols that we added to $\mathcal{L}$ in our construction. So what we are going to do is to ignore these extra constant symbols when we do the restriction, but these elements are still in $\mathfrak{M}|_{\mathcal{L}}$.

4. Define $\mathfrak{M}|_{\mathcal{L}}$ for the model $\mathfrak{M}$ restricted to $\mathcal{L}$:[3]

   - The universe of $\mathfrak{M}|_{\mathcal{L}}$ is the same as $\mathfrak{M}$. Notice that by doing this, the universe of $\mathfrak{M}|_{\mathcal{L}}$ contains elements not necessarily in $\mathcal{L}$-language, for example, it could contain $[f(k, c_2)]$ where k is constant symbols from $\mathcal{L}$-language and c_2 is Henkin constant added by our construction to $\mathcal{L}$-language.
   - Any constant symbol, function symbol, relation symbol of the original $\mathcal{L}$-language are the same as in $\mathfrak{M}$.
   - Ignore those constant symbols, function symbols, relation symbols added to $\mathcal{L}'$ language by our construction.

   Then we want to show $\mathfrak{M}|_{\mathcal{L}} \models \Sigma$.

Lemma 3.2.10. If σ is an $\mathcal{L}$-sentence, then $\mathfrak{M} \models \sigma \iff \mathfrak{M}|_{\mathcal{L}} \models \sigma$.

Proof. Use induction on the complexity of σ as an $\mathcal{L}$-sentence, prove $\sigma \in \Sigma' \iff \mathfrak{M}|_{\mathcal{L}} \models \sigma$. That is $\sigma :\equiv t_1 = t_2$, $\sigma :\equiv R(t_1, t_2, \ldots, t_n)$, $\sigma :\equiv (\neg\alpha)$, $\sigma :\equiv (\alpha \vee \beta)$, or $\sigma :\equiv (\forall x)(\phi)$ by Definition 1.2.2.

The useful fact remains:

- s is a variable assignment function (**vaf**) by Definition 1.5.1 into $\mathfrak{M}|_{\mathcal{L}}$. $s : Vars \to M|_{\mathcal{L}}$ has no change as in Proposition 3.2.1 since the domain $Vars$ is the same. (The difference is on the range that is restricted to those elements in the universe that are related to variable-free terms of the language $\mathcal{L}$ only.)
- Suppose t is a variable-free term in $\mathcal{L}$, then the term assignment function by Definition 1.5.2 is $\overline{s}(t)|_{\mathcal{L}} = [t]$, with

$$\overline{s}|_{\mathcal{L}} : \{\mathcal{L} - \text{term}\} \to M|_{\mathcal{L}}.$$

  $\overline{s}|_{\mathcal{L}}$ is the function $\overline{s}$ in Proposition 3.2.1 restricted at the domain of $\mathcal{L}$ − terms.

Since the useful fact remains, the proof of the following cases remains the same as in Proposition 3.2.1:

[3] $\mathfrak{M}|_{\mathcal{L}}$ is called the **reduct** of $\mathfrak{M}$, which is analogous to a field can be regarded as a group in algebra.

(a) $\sigma :\equiv t_1 = t_2$, where t_1 and t_2 are variable-free terms in the language $\mathcal{L}$.

Assume $\sigma :\equiv t_1 = t_2 \in \Sigma'$. By definition 3.2.3,

$$\begin{aligned} & \sigma :\equiv t_1 = t_2 \in \Sigma' \\ \iff & t_1 \sim t_2 \\ \iff & [t_1] = [t_2] \\ \iff & \overline{s}|_{\mathcal{L}}(t_1) = \overline{s}|_{\mathcal{L}}(t_2) \text{ for all vaf } s \text{ into } \mathfrak{M}|_{\mathcal{L}} \\ \iff & \mathfrak{M}|_{\mathcal{L}} \models t_1 = t_2 \text{ by Definition 1.5.4} \end{aligned}$$

(b) $\sigma :\equiv R(t_1, t_2, \ldots, t_n)$, where t_1, t_2, …, t_n are variable-free terms in the language $\mathcal{L}$.

Assume $\sigma :\equiv R(t_1, \ldots, t_n) \in \Sigma'$. By construction of the model $\mathfrak{M}|_{\mathcal{L}}$,

$$\begin{aligned} & \sigma :\equiv R(t_1, \ldots, t_n) \in \Sigma' \\ \iff & R^{\mathfrak{M}|_{\mathcal{L}}}\big([t_1], \ldots, [t_n]\big), \text{ or } \big([t_1], \ldots, [t_n]\big) \in R^{\mathfrak{M}|_{\mathcal{L}}} \\ \iff & \big(\overline{s}|_{\mathcal{L}}(t_1), \ldots, \overline{s}|_{\mathcal{L}}(t_n)\big) \in R^{\mathfrak{M}|_{\mathcal{L}}} \text{ for all vaf } s \text{ into } \mathfrak{M}|_{\mathcal{L}} \\ \iff & \mathfrak{M}|_{\mathcal{L}} \models R(t_1, \ldots, t_n) \text{ by Definition 1.5.4} \end{aligned}$$

(c) $\sigma :\equiv \neg\alpha$, where by induction hyphothesis $\alpha \in \Sigma' \iff \mathfrak{M}|_{\mathcal{L}} \models \alpha$. Σ' is maximal and consistent, this means $\sigma \in \Sigma' \iff \alpha \notin \Sigma'$. Then,

$$\begin{aligned} & \sigma \in \Sigma' \\ \iff & \alpha \notin \Sigma' \\ \iff & \mathfrak{M}|_{\mathcal{L}} \not\models \alpha \\ \iff & \mathfrak{M}|_{\mathcal{L}} \models \neg\alpha \text{ by Definition 1.5.4} \\ \iff & \mathfrak{M}|_{\mathcal{L}} \models \sigma \end{aligned}$$

(d) $\sigma :\equiv \alpha \vee \beta$, where by induction hyphothesis $\alpha \in \Sigma' \iff \mathfrak{M}|_{\mathcal{L}} \models$

α and $\beta \in \Sigma' \iff \mathfrak{M}|_{\mathcal{L}} \models \beta$. Then,

$$\begin{aligned} & \sigma \in \Sigma' \\ \iff\ & \alpha \in \Sigma' \text{ or } \beta \in \Sigma' \\ \iff\ & \mathfrak{M}|_{\mathcal{L}} \models \alpha \text{ or } \mathfrak{M}|_{\mathcal{L}} \models \beta \\ \iff\ & \mathfrak{M}|_{\mathcal{L}} \models (\alpha \vee \beta) \text{ by Definition 1.5.4} \\ \iff\ & \mathfrak{M}|_{\mathcal{L}} \models \sigma \end{aligned}$$

The following case requires modification.

(e) $\sigma :\equiv (\forall x)(\phi)$, where by induction hyphothesis $\phi \in \Sigma' \iff \mathfrak{M}|_{\mathcal{L}} \models \phi$. We need to show $(\forall x)(\phi) \in \Sigma' \iff \mathfrak{M}|_{\mathcal{L}} \models (\forall x)(\phi)$. Analogously, we work out $\iff$ by two ways $\implies$ and $\impliedby$:

$\implies$: Assume $(\forall x)(\phi) \in \Sigma'$. We need to prove $\mathfrak{M}|_{\mathcal{L}} \models \sigma$.

$$\begin{aligned} & \mathfrak{M}|_{\mathcal{L}} \models \sigma \\ \iff\ & \mathfrak{M}|_{\mathcal{L}} \models (\forall x)(\phi) \\ \iff\ & \mathfrak{M}|_{\mathcal{L}} \models (\forall x)\phi[s] \text{ for any vaf } s \text{ into } \mathfrak{M}|_{\mathcal{L}} \\ \iff\ & \mathfrak{M}|_{\mathcal{L}} \models \phi\Big[s\big[x|m\big]\Big] \text{ for any } m \in M|_{\mathcal{L}} \text{ by Definition 1.5.4} \\ \iff\ & \mathfrak{M}|_{\mathcal{L}} \models \phi\Big[s\big[x\big|[t]\big]\Big] \text{ for any variable-free term } t \\ & \quad \text{in } \mathcal{L} \text{ (and thus } [t] \in M|_{\mathcal{L}}) \\ \iff\ & \mathfrak{M}|_{\mathcal{L}} \models \phi\Big[s\big[x\big|\overline{s}|_{\mathcal{L}}(t)\big]\Big] \text{ by the useful fact} \\ & \quad t \text{ is variable-free and thus is substitutable for } x \text{ in } \phi \\ \iff\ & \mathfrak{M}|_{\mathcal{L}} \models \phi^x_t[s] \text{ by Theorem 2.6.1} \\ \iff\ & \mathfrak{M}|_{\mathcal{L}} \models \phi^x_t \text{ since } \phi^x_t \text{ is variable-free} \\ \iff\ & \phi^x_t \in \Sigma' \text{ by induction hypothesis:} \\ & \quad \phi^x_t \text{ has fewer quantifiers than } \sigma \\ \iff\ & \Sigma' \vdash \phi^x_t \text{ by Lemma 3.2.6} \end{aligned}$$

To conclude, we know $\sigma \in \Sigma'$, so $\Sigma' \vdash (\forall x)(\phi)$ by Lemma 3.2.6. Also, by (Q1), we have $\Sigma' \vdash (\forall x)(\phi) \to \phi^x_t$ since t is variable-free and thus is substitutable for x in ϕ. Thus, we have $\Sigma' \vdash \phi^x_t$, which means $\mathfrak{M}|_{\mathcal{L}} \models \sigma$.

$\Longleftarrow$: Assume $\sigma \notin \Sigma'$, we need to prove $\mathfrak{M}|_{\mathcal{L}} \not\models \sigma$.

$\sigma \notin \Sigma'$

$\Rightarrow$ $\neg\sigma \in \Sigma'$ by maximality of Σ'

$\Rightarrow$ $\Sigma' \vdash \neg\sigma$ by Lemma 3.2.6

$\Rightarrow$ $\Sigma' \vdash \neg(\forall x)(\phi)$

$\Rightarrow$ $\Sigma' \vdash (\exists x)(\neg\phi)$ by Example 1.1.1: $(\exists x)\alpha = \neg(\forall x)(\neg\alpha)$

$\Rightarrow$ $(\exists x)(\neg\phi) \in \Sigma'$ by Lemma 3.2.6

Henkin axiom in Σ' is as $(\exists x\theta) \to (\theta)^x_c$ for some Henkin constant c

$\Rightarrow$ $\Big((\exists x)(\neg\phi) \to (\neg\phi)^x_c\Big) \in \Sigma'$

$\Rightarrow$ $\Sigma' \vdash \Big((\exists x)(\neg\phi) \wedge \big((\exists x)(\neg\phi) \to (\neg\phi)^x_c\big)\Big)$ by Lemma 3.2.6

$\Rightarrow$ $\Sigma' \vdash (\neg\phi)^x_c$

ϕ^x_c is not $\mathcal{L}$-sentence since there exists Henkin constant c.
Analogous to Lemma 3.2.2, we can replace backward c
by a variable z in $\mathcal{L}$ but not in $\mathcal{L}'$, likewise for
any other Henkin constants existing in the deduction toward $\neg\phi^x_c$

$\Rightarrow$ $\Sigma' \vdash (\neg\phi)^x_z$

$\Rightarrow$ $(\neg\phi)^x_z \in \Sigma'$ by Lemma 3.2.6

$\Rightarrow$ $\mathfrak{M}|_{\mathcal{L}} \models (\neg\phi)^x_z$ by induction hypothesis:
$(\neg\phi)^x_z$ has fewer quantifiers

$\Rightarrow$ $\mathfrak{M}|_{\mathcal{L}} \models (\exists x)(\neg\phi)$ by (Q2)

$\Rightarrow$ $\mathfrak{M}|_{\mathcal{L}} \not\models \neg(\exists x)(\neg\phi)$

$\Rightarrow$ $\mathfrak{M}|_{\mathcal{L}} \not\models (\forall x)(\phi)$ by $(\exists x)(\neg\phi) \iff \neg(\forall x)(\phi)$

$\square$

****

3.3 Compactness

Theorem 3.3.1 (Compactness)**.** Let Σ be any set of formulas. There is a model of Σ if and only if every finite subset Σ_0 of Σ has a model.

Proof.

- $\Longrightarrow$: If $\mathfrak{M} \models \Sigma$, then obviously $\mathfrak{M} \models \Sigma_0$ for every $\Sigma_0 \subseteq \Sigma$.

- $\Longleftarrow$: Assume every finite subset of Σ has a model. Prove by contradiction: assume Σ has no model $\Sigma \models \perp$.

$$\begin{aligned} & \Sigma \models \perp \\ \Rightarrow\ & \Sigma \vdash \perp \text{ by Completeness Theorem 3.2.1} \\ \Rightarrow\ & \Sigma_o \vdash \perp \text{ for some finite } \Sigma_0 \subseteq \Sigma \\ & \quad \text{since deductions are finite by Definition 2.2.1} \\ \Rightarrow\ & \Sigma_0 \models \perp \text{ by Soundness Theorem 2.5.3} \\ \Rightarrow\ & \Sigma_0 \text{ has no model} \\ \Rightarrow\ & \text{Observe contradiction} \end{aligned}$$

□

Definition 3.3.1. Σ is **satisfiable** if there is a model of Σ. Σ is **finitely satisfiable** if every finite subset of Σ has a model.

What Compactness Theorem 3.3.1 says: Σ is satisfiable if and only if Σ is finitely satisfiable.

Corollary 3.3.1. $\Sigma \models \theta$ if and only if there is a finite subset $\Sigma_0 \subseteq \Sigma$ such that $\Sigma_0 \models \theta$.

Proof.

$$\begin{aligned} & \Sigma \models \theta \\ \Longleftrightarrow\ & \Sigma \vdash \theta \text{ by Completeness Theorem 3.2.1} \\ \Longleftrightarrow\ & \Sigma_0 \vdash \theta \text{ for some finite subset } \Sigma_0 \subseteq \Sigma \text{ since deduction is finite} \\ \Longleftrightarrow\ & \Sigma_0 \models \theta \text{ by Soundness Theorem 2.5.3} \end{aligned}$$

□

3.4 Applications

3.4.1 Finite or Infinite

The property of "being finite" is **not** a first-order property.

Example 3.4.1. Let $\mathcal{L}_G = \{\cdot, ^{-1}, 1\}$ where $\cdot$ is binary as $x \cdot y$, $^{-1}$ is unary as x^{-1}, 1 is constant. The axioms for a **group** are:

$$\begin{aligned} \gamma_1 &:\equiv (x \cdot y) \cdot z = x \cdot (y \cdot z) \\ \gamma_2 &:\equiv x \cdot 1 = x \wedge 1 \cdot x = x \\ \gamma_3 &:\equiv x \cdot x^{-1} = 1 \wedge x^{-1} \cdot x = 1 \end{aligned}$$

Question 3.4.1. Is there a set of formulas Σ such that $G \models \Sigma$ if and only if G is a group?

Answer 3.4.1. Yes. Choose $\Sigma = \{\gamma_1, \gamma_2, \gamma_3\}$.

Question 3.4.2. Is there a set of formulas Σ such that $G \models \Sigma$ if and only if G is a group with at most 4 elements?

Answer 3.4.2. Yes. Choose $\Sigma = \{\gamma_1, \gamma_2, \gamma_3, \sigma\}$ where

$$\sigma :\equiv \exists x_1, x_2, x_3, x_4 \forall y \left(y = x_1 \vee y = x_2 \vee y = x_3 \vee y = x_4\right).$$

Question 3.4.3. Is there a set of formulas Σ such that $G \models \Sigma$ if and only if G is an **infinite** group?

Answer 3.4.3. Yes. Choose

$$\Sigma = \Big\{\gamma_1, \gamma_2, \gamma_3\Big\} \cup \Big\{\alpha_k \Big| \, k \geq 2\Big\}$$

where α_k means there exists k distinct elements:

$$\alpha_k :\equiv \left(\exists x_1, \ldots, x_k \Big(\bigwedge_{1 \leq i < j \leq k} (x_i \neq x_j)\Big)\right). \tag{3.1}$$

For instance,

$$\begin{aligned} \alpha_2 \ &:\equiv \ \Big(\exists x_1, x_2 \Big(x_1 \neq x_2\Big)\Big) \\ \alpha_3 \ &:\equiv \ \Big(\exists x_1, x_2, x_3 \Big((x_1 \neq x_2) \wedge (x_1 \neq x_3) \wedge (x_2 \neq x_3)\Big)\Big) \\ &\ \vdots \end{aligned}$$

Question 3.4.4. Is there a set of formulas Σ such that $G \models \Sigma$ if and only if G is a finite group?

Answer 3.4.4. Assume such a Σ does exist. Denote

$$\begin{aligned} \alpha_2 \ &:\equiv \ \Big(\exists x_1, x_2 \Big(x_1 \neq x_2\Big)\Big) \\ \alpha_3 \ &:\equiv \ \Big(\exists x_1, x_2, x_3 \Big((x_1 \neq x_2) \wedge (x_1 \neq x_3) \wedge (x_2 \neq x_3)\Big)\Big) \\ &\ \vdots \end{aligned}$$

Notice that $G \models \alpha_k$ if and only if G has at least k elements.
Define $\hat{\Sigma} = \Sigma \cup \{\alpha_k \big| k \geq 2\}$. Apply Compactness Theorem 3.3.1:

- Pick a finite subset $A \subseteq \hat{\Sigma}$.
- Let m be the largest integer such that $\alpha_m \in A$.
- Let C_m be the cyclic group with m elements. Then

$$\begin{aligned} &C_m \models \alpha_k \text{ for all } k \leq m \\ &C_m \models \Sigma \text{ by assumption} \end{aligned}$$

- $A \subseteq \Sigma \cup \{\alpha_2, \ldots, \alpha_m\}$ and $C_m \models \Sigma \cup \{\alpha_2, \ldots, \alpha_m\}$. Thus, $C_m \models A$.
- Since we can pick the subset A arbitrarily, this implies that every finite subset of $\hat{\Sigma}$ has a model. By Compactness Theorem 3.3.1, $\hat{\Sigma}$ has a model $\hat{G}$: $\hat{G} \models \hat{\Sigma}$.
- $\hat{G} \models \alpha_k$ for all $k \geq 2$, thus, $\hat{G}$ is infinite.
- Since $\Sigma \subseteq \hat{\Sigma}$, we have $\hat{G} \models \Sigma$, this means $\hat{G} \models \Sigma$ and $\hat{G}$ is an infinite group, **not** a finite group. contradiction is observed.

Therefore, the "finiteness" can **not** be axiomized in first-order.

Example 3.4.2. Consider $\mathcal{L}_0 = \{<\}$. The axioms for a linear order are:

$$\begin{aligned} L_1 &:\equiv \forall x \forall y \big((x < y) \vee (x = y) \vee (x > y)\big) \\ L_2 &:\equiv \forall x \big(\neg(x < x)\big) \\ L_3 &:\equiv \forall x \forall y \forall z \big[(x < y \wedge y < z) \rightarrow x < z\big] \end{aligned}$$

Then, using similar arguements as in Example 3.4.1 with α_k from (3.1) for the sentence "there exists k distinct elements", there is **no** set of axioms Σ such that $\mathfrak{M} \models \Sigma$ if and only if $\mathfrak{M}$ s a finite linear order (model/structure).

Theorem 3.4.1. Σ is a set of sentences. If Σ has arbitrarily large finite models, then Σ has an infinite model.

Proof. Suppose Σ has arbitrarily large finite models. Then let α_k be the sentence that "there exists at least k distinct elements" from (3.1).

Consider the set

$$\Phi = \Sigma \cup \{\alpha_k | k \geq 2\}.$$

Pick an arbitrary finite subset $A \subseteq \Phi$. Let m be the largest integer such that $\alpha_m \in A$. Then $A \subseteq \Sigma \cup \{\alpha_2, \ldots, \alpha_m\}$. Since Σ has arbitrarily large finite models, we can find a model for $\Sigma \cup \{\alpha_2, \ldots, \alpha_m\}$, which is also a model of A. Since we pick A arbitrarily, by Compactness Theorem 3.3.1, Φ has a model. Notice that any model of Φ is infinite, and it is also a model of Σ. Thus, Σ has an infinite model.

$\square$

3.4.2 Model of Natural Numbers

Example 3.4.3. Suppose the $\mathcal{L}_{NT}$-structure $\mathfrak{N}$ whose universe is the set of natural number $\mathbb{N}$.

Definition 3.4.1. Suppose $\mathfrak{M}$ and $\mathfrak{N}$ are two $\mathcal{L}$-structure. Then we call $\mathfrak{M}$ and $\mathfrak{N}$ are **isomorphic** and write $\mathfrak{M} \cong \mathfrak{N}$, if there is a bijection $i : M \to N$ such that for each constant symbol c of $\mathcal{L}$, $i(c^{\mathfrak{M}}) = c^{\mathfrak{N}}$, for each k-ary function symbol f and for each $m_1, \ldots, m_k \in M$,

$$i(f^{\mathfrak{M}}(m_1, \ldots, m_k)) = f^{\mathfrak{N}}(i(m_1), \ldots, i(m_k)),$$

and for each k-ary relation symbol R in $\mathcal{L}$,

$$(m_1, \ldots, m_k) \in R^{\mathfrak{M}} \text{ if and only if } (i(m_1), \ldots, i(m_k)) \in R^{\mathfrak{N}}.$$

The function i is called an **isomorphism**.

Definition 3.4.2. Suppose $\mathfrak{M}$ be an $\mathcal{L}$-structure. The **theory of** $\mathfrak{M}$ is

$$Th(\mathfrak{M}) = \{\phi | \mathfrak{M} \models \phi \text{ for } \phi \text{ as an } \mathcal{L}\text{-formula}\}$$

Question 3.4.5. Is there a set of formulas Σ such that $\mathfrak{M} \models \Sigma$ if and only if $\mathfrak{M} \cong \mathfrak{N}$?

Answer 3.4.5. Suppose such a Σ exists. So by assumption, we have $\mathfrak{M} \models \Sigma$ if and only if $\mathfrak{M} \cong \mathfrak{N}$.

Expand $\mathcal{L}_{NT}$ to $\mathcal{L} = \mathcal{L}_{NT} \cup \{c\}$ where c is a new constant symbol.

Let Γ be the following set of formulas:

$$\begin{aligned} \gamma_0 &:\equiv 0 < c \\ \gamma_1 &:\equiv S0 < c \\ \gamma_2 &:\equiv SS0 < c \\ &\vdots \end{aligned}$$

Corollary 3.4.1. Every finite subset of $\Sigma \cup \Gamma$ has a model.

Proof. Denote

$$\Theta = \Sigma \cup \Gamma.$$

Pick arbitrarily a finite subset $A \subseteq \Theta$. Let k be the largest integer such that $\gamma_k \in A$. Thus, A is a subset of $\Theta_k = \Sigma \cup \{\gamma_1, \ldots, \gamma_k\}$. Θ_k has a model $\mathfrak{N}_k$ whose universe is $\mathbb{N}$, the functions and relations are the same as $\mathfrak{N}$ with the extra constant symbol goes to $c^{\mathfrak{N}_k} = k+1$. This implies that $A \subseteq \Theta_k$ has a model $\mathfrak{N}_k$ as well. Since we pick arbitrarily the finite subset $A \subseteq \Theta$, by Compactness Theorem 3.3.1, Θ has a model.

□

Denote the model of Θ as $\mathfrak{M}'$. Now we shrink back from $\mathcal{L}$ to $\mathcal{L}_{NT}$ by "forgetting" the constant symbol c, denote $\mathfrak{M} = \mathfrak{M}'|_{\mathcal{L}_{NT}}$. By our construction, $\mathfrak{M} \models \Sigma$. But $\mathfrak{M}$ is not isomorphic to $\mathfrak{N}$, since $\mathfrak{M}$ has the new constant symbol $c^{\mathfrak{M}} = k+1$ and $\overline{k} < c^{\mathfrak{M}}$ holds for all $k \in \mathbb{N}$. So $\mathfrak{M} \ncong \mathfrak{N}$. Therefore, there is **no** set of formulas Σ such that $\mathfrak{M} \models \Sigma \iff \mathfrak{M} \cong \mathfrak{N}$.

Definition 3.4.3. If $\mathfrak{M}$ and $\mathfrak{N}$ are $\mathcal{L}$-structure, then $\mathfrak{M}$ and $\mathfrak{N}$ are **elementarily equivalent** if $Th(\mathfrak{M}) = Th(\mathfrak{N})$, denoted by

$$\mathfrak{M} \equiv \mathfrak{N}.$$

Example 3.4.3 just shows $\mathfrak{M} \equiv \mathfrak{N} \not\Rightarrow \mathfrak{M} \cong \mathfrak{N}$. In fact, we can have a generalization.

Theorem 3.4.2. If $\mathfrak{N}$ is infinite, then there exists structures $\mathfrak{M}$ of arbitrarily large cardinality such that $\mathfrak{M} \equiv \mathfrak{N}$ but $\mathfrak{M} \ncong \mathfrak{N}$.

Roughly speaking, the cardinality of a set is the number of elements in the set.

3.4.3 Superstructure with Special Element

We present an example of how to construct hyperreals from the language of ordered ring.

Example 3.4.4. Let $\mathcal{L}_{OR} = \{+, \cdot, 0, 1, <\}$ be the language of ordered ring.

We want to create an $\mathcal{L}_{OR}$-structure $\mathfrak{R}^*$ such that

1. $\mathfrak{R} \subseteq \mathfrak{R}^*$
2. $\mathfrak{R} \equiv \mathfrak{R}^*$

3. $\mathfrak{R}^*$ contains infinitesimal element such that there exists an element $\epsilon \in \mathfrak{R}^*$ with
$$0 < \epsilon < r \text{ for every } r \in \mathbb{R}.$$

Answer 3.4.6.

- Expand the langauge of ordered ring to the language of real numbers: from $\mathcal{L}_{OR}$ to $\mathcal{L}_R = \mathcal{L}_{OR} \cup \{c_r | r \in \mathbb{R}\}$
  - Define $\mathfrak{R}$ as $\mathcal{L}_R$-structure, with $c_r^{\mathfrak{R}} = r \in \mathbb{R}$
  - Let $\Sigma_1 = Th(\mathfrak{R})$ in $\mathcal{L}_R$, which is the collection of first-order $\mathcal{L}_R$-formulas that are true statements about the real numbers.
- Add one more constant, which will point to an infinitesimal element
$$\mathcal{L} = \mathcal{L}_R \cup \{a\}.$$
- Denote the following set of sentences:
$$\Sigma_2 = \{\alpha_r | r \in \mathbb{R}, r > 0\},$$
where
$$\alpha_r :\equiv (0 < a) \wedge (a < c_r), \text{ or } \alpha_r :\equiv 0 < a < c_r.$$

Corollary 3.4.2. Every finite subset of $\Sigma_1 \cup \Sigma_2$ has a model.

Proof. Pick arbitrarily a finite subset $A \subseteq \Sigma_1 \cup \Sigma_2$.

- There is a smallest element $r_0 \in \mathbb{R}$ such that $\alpha_{r_0} \in A$. Thus $A \subseteq \Sigma_1 \cup \{0 < a < r \,|\, r \geq r_0\}$
- Make $\tilde{\mathfrak{R}}$ an $\mathcal{L}$-structure by extending $\mathfrak{R}$ with
$$a^{\tilde{\mathfrak{R}}} = \frac{r_0}{2}.$$
  * Thus $0^{\tilde{\mathfrak{R}}} < a^{\tilde{\mathfrak{R}}} < r_0 \leq r^{\tilde{\mathfrak{R}}}$ for all $r \geq r_0$.
  * Thus, $\tilde{\mathfrak{R}} \models \alpha_r$ for all $r \geq r_0$.
  * Also, $\tilde{\mathfrak{R}} \models \Sigma_1 = Th(\mathfrak{R})$.
- Thus $\tilde{\mathfrak{R}} \models A$.

- Since we arbitrarily pick the finite subset A of $\Sigma_1 \cup \Sigma_2$, by Compactness Theorem 3.3.1, $\Sigma_1 \cup \Sigma_2$ has a model.

□

- Denote $\mathfrak{R}^*$ as the model of $\Sigma_1 \cup \Sigma_2$. Note that
  1. $\mathfrak{R} \subseteq \mathfrak{R}^*$, since we can identify with $r \longleftrightarrow c_r^{\mathfrak{R}^*}$
  2. $\mathfrak{R}^* \equiv \mathfrak{R}$, since $\mathfrak{R}^* \models Th(\mathfrak{R})$.
  3. $\mathfrak{R}^*$ contains the infinitesimal element $a^{\mathfrak{R}^*}$, since

$$\mathfrak{R}^* \models \{0 < a < c_r \,|\, r \in \mathbb{R}, r > 0\}$$

Definition 3.4.4. Any element $a \in \mathfrak{R}^*$ satisfying $0 < a < r$, $\forall r \in \mathbb{R}$ is called an **infinitesimal** element.

Example 3.4.5. Let $s, t \in \mathbb{R}$, where $\mathbb{R}$ is the universe of the model $\mathfrak{R}$. Then $r = s \iff |s - t| = a$ for some infinitesimal element $a \in \mathbb{R}^*$ (where $\mathbb{R}^*$ is the universe of the model $\mathfrak{R}^*$).

Answer 3.4.7.

$$\begin{aligned}
& |s-t| \text{ is infinitesimal} \\
\iff & |s-t| < r \text{ for all } r \in \mathbb{R} \text{ and } r > 0 \\
\iff & |s-t| = 0 \text{ since the absolute value } |s-t| \geq 0 \\
\iff & s = t
\end{aligned}$$

So two real numbers are equal if and only if they are infinitesimally close.

Definition 3.4.5. Any element $a \in \mathbb{R}^*$ with $a > r$ for all $r \in \mathbb{R}$ is called **infinite**.

Example 3.4.6. $a \in \mathbb{R}$ is infinitesimal if and only if a^{-1} is infinite.

Answer 3.4.8.

1. $\mathfrak{R}$ models $\forall x(x \neq 0 \to \exists y(xy = 1))$. Since $\mathfrak{R}^* \equiv \mathfrak{R}$, $\mathfrak{R}^*$ models this too! This ensures the existence of the inverse for non-zero elements in $\mathbb{R}$.

2. $\mathfrak{R}$ models $\forall x \forall y(0 < x < y \to 0 < y^{-1} < x^{-1})$. Since $\mathfrak{R}^* \equiv \mathfrak{R}$, $\mathfrak{R}^*$ models this too!

3. Thus,

$$
\begin{aligned}
& 0 < a < r \text{ for all } r \in \mathbb{R},\ r > 0 \\
\iff\ & 0 < r^{-1} < a^{-1} \text{ for all } r \in \mathbb{R}, r > 0 \\
\iff\ & 0 < s < a^{-1} \text{ for all } s \in \mathbb{R}, s > 0 \\
\iff\ & a^{-1} \text{ is infinite}
\end{aligned}
$$

Chapter 4

Incompleteness from Two Points of View

4.1 Introduction

- We have already proved that **the deduction system** is sound and complete by the Soundness Theorem 2.5.3 and the Completeness Theorem 3.2.1: Roughly speaking, the soundness and completeness is: We can prove a formula if and only if the formula is true in every structures/models.

4.1.1 Language of Natural Numbers

- Focus on the language of natural numbers $\mathcal{L}_{NT}$, with some structure $\mathfrak{N}$.
- Denote the set of sentences of $\mathcal{L}_{NT}$ that are true in the structure $\mathfrak{N}$ as **Theory of** $\mathfrak{N}$, or $Th(\mathfrak{N})$.

Definition 4.1.1. A set of nonlogical axioms Σ in a language $\mathcal{L}$ is called **complete** if for every $\mathcal{L}$-sentence σ, either $\Sigma \vdash \sigma$ or $\Sigma \vdash \neg\sigma$.

- A deduction system is *complete* means that for a given model of Σ, the deduction system proves/deduces ($\Sigma \vdash \theta$) those true formulas ($\Sigma \models \theta$).

- A set of axioms is *complete* means that the axioms can prove or refute any sentence.
- The set of axioms is *complete* is **stronger** condition (more restricted) than saying a deduction system is *complete*.

Definition 4.1.2. A set of axioms Σ is an **axiomatization of** $Th(\mathfrak{N})$ if for every sentence $\sigma \in Th(\mathfrak{N})$, $\Sigma \vdash \sigma$.

4.1.2 Incompleteness of nonlogical axioms of $\mathfrak{N}$

- It is trivial that $Th(\mathfrak{N})$ is the axiomatization of $Th(\mathfrak{N})$ itself.
- If we look for not so trivial answer: can we find a set of nonlogical axioms Σ that can decide whether or not a formula is an axiom (Σ is decidable in the sense of Definition 2.4.9).
- In short, we look for: a complete, consistent, decidable set of nonlogical axioms for $\mathfrak{N}$.
- The answer: **No,** by Gödel First Incompleteness Theorem.
- The idea is: Given any complete, consistent, decidable set of axioms for $\mathfrak{N}$, we can find a sentence σ that is true about natural numbers ($\sigma \in Th(\mathfrak{N})$), but σ is not provable from the the collection of axioms.

4.2 Complexity of Formulas

4.2.1 Language and Standard Structure of Natural Numbers

- The language of number theory:

$$\mathcal{L}_{NT} = \{0, S, +, \cdot, E, <\}.$$

- Standard Structure/Model of the natural number:

$$\mathfrak{N} = \{\mathbb{N}, 0, S, +, \cdot, E, <\}.$$

- A set of nonlogical axioms N, called **Robinson Arithmetic**, defined at Example 2.8.3:

1. $(\forall x)\neg Sx = 0$.
2. $(\forall x)(\forall y)[Sx = Sy \to x = y]$.
3. $(\forall x)x + 0 = x$.
4. $(\forall x)(\forall y)x + Sy = S(x + y)$.
5. $(\forall x)x \cdot 0 = 0$.
6. $(\forall x)(\forall y)x \cdot Sy = (x \cdot y) + x$.
7. $(\forall x)xE0 = S0$.
8. $(\forall x)(\forall y)xE(Sy) = (xEy) \cdot x$.
9. $(\forall x)\neg\, x < 0$.
10. $(\forall x)(\forall y)[x < Sy \longleftrightarrow (x < y \vee x = y)]$.
11. $(\forall x)(\forall y)[(x < y) \vee (x = y) \vee (y < x)]$.

Definition 4.2.1. Let x be a variable that does not occur in the term t, we have the following abbreviations, called **bounded quantifiers**:

1. $(\forall x < t)\phi \Leftrightarrow \forall x(x < t \to \phi)$
2. $(\forall x \leq t)\phi \Leftrightarrow \forall x((x < t \vee x = t) \to \phi)$
3. $(\exists x < t)\phi \Leftrightarrow \exists x(x < t \wedge \phi)$
4. $(\exists x \leq t)\phi \Leftrightarrow \exists x((x < t \vee x = t) \wedge \phi)$

The nonlogical axioms N in Example 2.8.3 is robust enough to prove every true statement and to refute every false statement in $\mathfrak{N}$ that contains **only bounded quantifiers**.

Any potential candidate for an axiomatization must be at least as strong as N, this implies that when we look for a formula that is true in $\mathfrak{N}$ and not provable must contain at least **some unbounded quantifiers**.

Definition 4.2.2. The collection of **Σ-formulas** is the smallest set of $\mathcal{L}_{NT}$ formulas such that:

1. Every atomic formula is a Σ-formula.
2. Every negation of an atomic formula is a Σ-formula.
3. If α and β are Σ-formula, then $\alpha \wedge \beta$ and $\alpha \vee \beta$ are Σ-formula.
4. If α is a Σ-formula, x is a variable that does not occur in the term t, then the following are Σ-formula: $(\forall x < t)\alpha$, $(\forall x \leq t)\alpha$, $(\exists x < t)\alpha$, $(\exists x \leq t)\alpha$.

5. If α is a Σ-formula, x is a variable, then $(\exists x)\alpha$ is a Σ-formula.

The Robinson Arithmetic N in Example 2.8.3 is robust enough to prove every true Σ-sentence.

Definition 4.2.3. The collection of Π-**formulas** is the smallest set of $\mathcal{L}_{NT}$-formula such that:

1. Every atomic formula is a Π-formula.
2. Every negative of an atomic formula is a Π-formula.
3. If α and β are Π-formulas, then $\alpha \wedge \beta$ and $\alpha \vee \beta$ are Π-formulas.
4. If α is a Π-formula, and x is a variable that does not occur in the term t, then the following are Π-formulas: $(\forall x < t)\alpha$, $(\forall x \leq t)\alpha$, $(\exists x < t)\alpha$, $(\exists x \leq t)\alpha$.
5. If α is a Π-formula, x is a variable, then $(\forall x)\alpha$ is a Π-formula.

- The set of Σ-formulas is closed under bounded quantification and unbounded **existential** quantification, whereas the set of Π-formula is closed under bounded quantification and unbounded **universal** quantification.
- Gödel's First Incompleteness Theorem says that if given any consistent and decidable set of axioms, there exists a Π-formula σ such that σ is a true statement about the natural numbers but there is no deduction of the formula σ from our axioms.

Lemma 4.2.1. If α is a Σ-formula, then $\neg\alpha$ is logically equivalent to a Π-formula. If α is a Π-formula, then $\neg\alpha$ is logically equivalent to a Σ-formula.

Proof. (Think about the equivalence $(\exists v)\alpha \iff \neg(\forall v)(\neg\alpha)$.) $\square$

Definition 4.2.4. The collection of Δ-formulas is the **intersection** of the collection of Σ-formulas with the collection of Π-formulas.

- The collection of Δ-formulas is the intersection between Σ-formulas and Π-formulas.

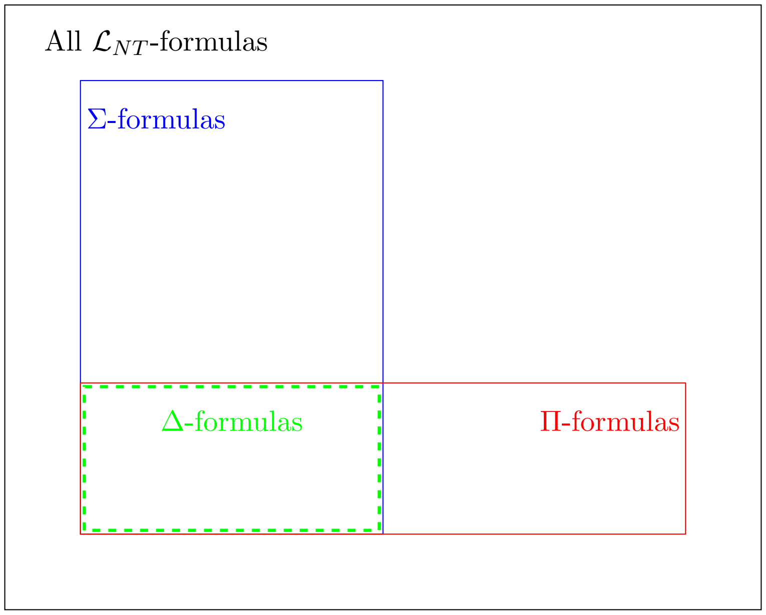

Figure 4.1: Σ-formulas, Π-formulas, and Δ-formulas

- All quantifiers in Δ-formulas are bounded.
- The set of axioms N in Example 2.8.3 can prove every Δ-formula that is true in $\mathfrak{N}$ and refute every Δ-formula that is false in $\mathfrak{N}$.

4.3 Two Roads to Incompleteness

4.3.1 By Formulas and Deduction

- For convenience, we want to "code" formulas and deductions into some natural numbers, and "decode" them back.
- By this, we can code statements about numbers that are actually codes for statements. For example, we can construct a statement: "24601 is a code for the statement [The number 42 is a code for deduction of $0 = 1$]".
- Notice not all natural numbers can be codes for statements.

4.3.2 By Theoretical Computation

- It focuses on function mapping the natural numbers to the natural numbers.
- A function f is *computable* on the input k: if there is a sequence of easy steps that leads to the correct output $f(k)$.
- We can show the existence of a formula that is true but yet not provable, by using the concept of computable function.
- It requires a way to code up statements too!

4.4 To Code a Sequence of Numbers

- We want to code a finite sequence of numbers, say $1, 0, 1$. The first way to do is to put the sequence as exponents of the first prime numbers and then multiple: $2^1 \cdot 3^0 \cdot 5^1 = 10$.
- One question here: 10 when reversed back, it becomes $2 \cdot 5$, it is not a product of the first few primes. The reason here is because of the existence of 0 in the sequence.
- To eliminate the issue of 0, we add 1 to each number before we put them to exponents of the first prime numbers: $1, 0, 1 \Rightarrow 2, 1, 2 \Rightarrow 2^2 \cdot 3^1 \cdot 5^2 = 300$.

Definition 4.4.1. The **function** p is the $IthPrime$ function mapping the natural numbers to prime numbers, where $p(0) = 1$ and $p(k)$ is the k^{th} prime for $k \geq 1$. Thus $p(0) = 1$, $p(1) = 2$, $p(2) = 3$, $p(3) = 5$ and so on. Often denote p_i instead of $p(i)$ in short.

Definition 4.4.2. Let $\mathbb{N}^{<\mathbb{N}}$ denote the **set of finite sequences of natural numbers** $(a_1, a_2, \ldots, a_k)$**,** $a_i \in \mathbb{N}$**,** $i = 1, \ldots, k$.

Definition 4.4.3. Define the **coding function** $\langle \cdot \rangle : \mathbb{N}^{<\mathbb{N}} \to \mathbb{N}$ by

$$\langle (a_1, a_2, \ldots, a_k) \rangle = \begin{cases} 1 & \text{if } k = 0 \\ \Pi_{i=1}^{k} p_i^{a_i+1} & \text{if } k > 0 \end{cases}$$

where p_i is the i-th prime number.

Denote $\langle a_1, a_2, \dots, a_k\rangle$ for $\langle (a_1, a_2, \dots, a_k)\rangle$ in short.

- We depend on the Fundamental Theorem of Arithmetic: every positive integer ≥ 1 can be represented in exactly one way apart from rearrangement as a product of one or more primes.
- A lot of natural numbers are not the code of sequences, we need to explicitly specify how to deal with these natural numbers.

Definition 4.4.4. Let $C = \{a \in \mathbb{N} \mid a = \langle s\rangle \text{ with } s \in \mathbb{N}^{<\mathbb{N}}\}$. Call C the set of **code numbers**.

It is easy to check if a is code number, i.e., if $a \in C$: Factor a to see if either $a = 1$ or if a is a product of the first few prime numbers.

Definition 4.4.5. The function $|\cdot| : \mathbb{N} \to \mathbb{N}$ is defined by:

$$|a| = \begin{cases} k & \text{if } a \in C \text{ and } a = \langle a_1, a_2, \dots, a_k\rangle \\ 0 & \text{otherwise.} \end{cases}$$

If a is a code number, call $|a|$ as **the length of** a.

Notice that $|a|$ and $|\langle a\rangle|$ are different: $|\langle a\rangle| = 2^{a+1} > a$ for $a > 1$.

Definition 4.4.6. For each $i \in \mathbb{N}$ with $i \geq 1$, let the **decoding function** $(\cdot)_i : \mathbb{N} \to \mathbb{N}$ as

$$(a)_i = \begin{cases} a_i & \text{if } a \in C \text{ and } a = \langle a_1, a_2, \dots, a_k\rangle \text{ and } 1 \leq i \leq k \\ 0 & \text{otherwise.} \end{cases}$$

Definition 4.4.7. The **sequence joining function** $^\frown : \mathbb{N} \to \mathbb{N}$ is defined by

$$a^\frown b = \begin{cases} \langle a_1, a_2, \dots, a_k, b_1, b_2, \dots, b_l\rangle & \text{if } a = \langle a_1, a_2, \dots, a_k\rangle \text{ and} \\ & \quad b = \langle b_1, b_2, \dots, b_l\rangle, \\ & \quad \text{with } (a_1, a_2, \dots, a_k) \in \mathbb{N}^{<\mathbb{N}} \text{ and} \\ & \quad (b_1, b_2, \dots, b_l) \in \mathbb{N}^{<\mathbb{N}} \\ 0 & \text{otherwise.} \end{cases}$$

The functions above allow us to code finite sequences as code number a and to decode a into finite sequences. Or, if $a \in C$ and $a = \langle a_1, a_2, \dots, a_k\rangle$, then for each i such that $1 \leq i \leq |a|$, $(a)_i = a_i$.

4.5 The Set of Nonlogical Axioms N

Here we reiterate the **nonlogical Axioms for number theory** N:

Suppose the language $\mathcal{L}_{NT} = \{0, S, +, \cdot, E, <\}$. The standard structure is $\mathfrak{N}$. The set of nonlogical axioms, called N, is a minimal set of assumptions to describe a bare-bones version of the usual operations on the set of natural numbers:

1. $(\forall x)\neg Sx = 0$.
2. $(\forall x)(\forall y)[Sx = Sy \rightarrow x = y]$.
3. $(\forall x)x + 0 = x$.
4. $(\forall x)(\forall y)x + Sy = S(x + y)$.
5. $(\forall x)x \cdot 0 = 0$.
6. $(\forall x)(\forall y)x \cdot Sy = (x \cdot y) + x$.
7. $(\forall x)xE0 = S0$.
8. $(\forall x)(\forall y)xE(Sy) = (xEy) \cdot x$.
9. $(\forall x)\neg x < 0$.
10. $(\forall x)(\forall y)[x < Sy \leftrightarrow (x < y \vee x = y)]$.
11. $(\forall x)(\forall y)[(x < y) \vee (x = y) \vee (y < x)]$.

Example 4.5.1. Recall that the collection of Δ-formulas contain no unbounded quantifiers. Consider $\phi(x) :\equiv (\exists y \leq x)\overline{2}y = x$, which states that x is an even number.

Consider $\phi(\overline{2})$ and $\phi(\overline{3})$. We agree $\phi(\overline{2})$ is true statement about $\mathfrak{N}$, while $\phi(\overline{3})$ is false.

Our set of nonlogical axioms N is strong enough to prove $\phi(\overline{2})$ and refute $\phi(\overline{3})$:

$$N \vdash \phi(\overline{2}), \quad N \vdash \neg\phi(\overline{3}).$$

As it will be shown in the following Proposition 5.3.3, this is a general fact about the relationship between Δ-formulas and N.

- Remind that in mathematics, **there are lots of structures of non-standard arithmetic where a property can be false of every natural number but still true of some other element of the universe.** So even if N can prove that ϕ is false for every natural number, we can not prove that ϕ is false of everything. In fact, it cannot.

- In short: N is strong enough to prove true Σ-sentence, but **not** strong enough to refute false Σ-sentence.

- Equivalently, N is strong enough to prove true Σ-sentence, but **not** strong enough to prove every true Π-sentence.

Example 4.5.2. Consider Goldbach Conjecture: every even number > 2 can be written as the sum of two primes.

This conjecture can be formally written as Π-sentence. Its denial is equivalent to a Σ-sentence. Currently, we do not know if this conjecture is true or false.

The nonlogical axioms N can prove Σ-sentence, which means that N can prove the denial of Goldbach Conjecture.

On the other hand, N is not strong enough to prove every true Π-sentence. Suppose Goldbach Conjecture is true, then there is no reason to believe that N is strong enough to prove this.

[pxl: the following reasoning is doubtful: "Or roughly equally to say, if we could prove that N is not strong enough to decide the Goldbach Conjecture, then the Goldbach Conjecture is true."

It is saying: "If we prove that N is not strong enough to decide the Goldbach Conjecture, then by '$False \to$ every statement is $True$' , Gold-Bach Conjecture is True"?]

Chapter 5

Syntactic Incompleteness - Groundwork

5.1 Introduction

- The proof of the First Incompleteness Theorem is to construct a certain sentence θ that is a true statement in $\mathfrak{N}$ but is unprovable from the collection of axioms N.
- We need to make sure this θ exists.
- Gödel's Self-Reference Lemma helps to construct the sentence θ.
- Then the First Incompleteness Theorem says that there is no decidable, consistent, complete set of axioms for $\mathfrak{N}$.
- The Second Incompleteness Theorem says that no reasonably strong set of axioms can prove its own consistency.

5.2 The Power of Robinson Arithmetic

- The language of number theory:

$$\mathcal{L}_{NT} = \{0, S, +, \cdot, E, <\}.$$

- The standard structure (model) of the natural numbers:

$$\mathfrak{N} = \{\mathbb{N}, 0, S, +, \cdot, E, <\},$$

where the functions and relations are the standard functions and relations on the natural numbers.

- The nonlogical axioms N (**Robinson Arithmetic**):

N_1: $(\forall x)\neg Sx = 0$.
N_2: $(\forall x)(\forall y)[Sx = Sy \to x = y]$.
N_3: $(\forall x)x + 0 = x$.
N_4: $(\forall x)(\forall y)x + Sy = S(x + y)$.
N_5: $(\forall x)x \cdot 0 = 0$.
N_6: $(\forall x)(\forall y)x \cdot Sy = (x \cdot y) + x$.
N_7: $(\forall x)xE0 = S0$.
N_8: $(\forall x)(\forall y)xE(Sy) = (xEy) \cdot x$.
N_9: $(\forall x)\neg x < 0$.
N_{10}: $(\forall x)(\forall y)[x < Sy \leftrightarrow (x < y \vee x = y)]$.
N_{11}: $(\forall x)(\forall y)[(x < y) \vee (x = y) \vee (y < x)]$.

- Recall Lemma 2.8.1:

Lemma 2.8.1. For natural numbers $a, b \in \mathbb{N}$.

1. If $a = b$ (holds in $\mathbb{N}$), then $N \vdash \overline{a} = \overline{b}$.
2. If $a \neq b$, then $N \vdash \overline{a} \neq \overline{b}$.
3. If $a < b$, then $N \vdash \overline{a} < \overline{b}$.
4. If $a \not< b$, then $N \vdash \overline{a} \not< \overline{b}$.
5. $N \vdash \overline{a} + \overline{b} = \overline{a + b}$.
6. $N \vdash \overline{a} \cdot \overline{b} = \overline{a \cdot b}$.
7. $N \vdash \overline{a}E\overline{b} = \overline{a^b}$.

- Recall by Example 2.8.4: N does **NOT** prove every sentence. As an example, N is *not* strong enough to prove the commutative law of addition: $N \nvdash (\forall x)(\forall y)\, x + y = y + x$.

- For $a \in \mathbb{N}$, use $\overline{a}$ as the abbreviation for the $\mathcal{L}_{NT}$ variable-free term $\underbrace{SSS\ldots S}_{a\,S's}0$.
- For a variable-free term t, then $t^{\mathfrak{N}} \in \mathbb{N}$ is the interpretation of t in $\mathfrak{N}$. Further on, $\overline{t^{\mathfrak{N}}}$ is a variable-free term that has the same interpretation of t in $\mathfrak{N}$.

  Example: Suppose the term $t :\equiv ESSS0SS0$, which is $SSS0^{SS0}$. Then $t^{\mathfrak{N}} \iff (SSS0^{SS0})^{\mathfrak{N}} \iff 9 \in \mathbb{N}$. $\overline{t^{\mathfrak{N}}} = \overline{9}$ which is the term $SSSSSSSSS0$.
- Robinson Arithmetic can prove that the variable-free term t is equivalent to $\overline{t^{\mathfrak{N}}}$: $N \vdash t = \overline{t^{\mathfrak{N}}}$. For example: that is to show N proves $SSS0^{SS0} = SSSSSSSSS0$, which is the same as to show $N \vdash \overline{3^2} = \overline{9}$.

Lemma 5.2.1. For each variable-free term t, $N \vdash t = \overline{t^{\mathfrak{N}}}$.

Proof. Prove by induction on the complexity of the term t.

- If t is the constant term 0, then $t^{\mathfrak{N}}$ is the natural number $0 \in \mathbb{N}$, and $\overline{t^{\mathfrak{N}}}$ is the term 0. We want to prove $N \vdash 0 = 0$:

  1. By Lemma 2.7.1, we find $N \vdash \theta \iff N \vdash \forall x\theta$. Take $\theta :\equiv x = x$, we have $N \vdash x = x \iff N \vdash (\forall x)x = x$.
  2. By logical axiom (Q1), $(\forall x)x = x \to 0 = 0$. Thus, $N \vdash (\forall x)x = x \to 0 = 0$
  3. By (PC) in Definition 2.4.5, we have $N \vdash 0 = 0$.

- If $t :\equiv S(u)$, where u is a variable-free term, then by induction hypothesis, we have $N \vdash u = \overline{u^{\mathfrak{N}}}$. By equality axiom (E2), $N \vdash S(u) = S(\overline{u^{\mathfrak{N}}})$. $\overline{t^{\mathfrak{N}}} = \overline{(S(u))^{\mathfrak{N}}} = S(\overline{u^{\mathfrak{N}}})$. [1] Thus, we find $N \vdash t = S(u) = S(\overline{u^{\mathfrak{N}}}) = \overline{t^{\mathfrak{N}}}$.
- If t is $u + v$, by induction hypothesis $N \vdash u = \overline{u^{\mathfrak{N}}}$ and $N \vdash v = \overline{v^{\mathfrak{N}}}$. By Lemma 2.8.1, we have

  $$N \vdash \overline{u^{\mathfrak{N}}} + \overline{v^{\mathfrak{N}}} = \overline{u^{\mathfrak{N}} + v^{\mathfrak{N}}}.$$

  This implies:

  $$N \vdash t = u + v = \overline{u^{\mathfrak{N}}} + \overline{v^{\mathfrak{N}}} = \overline{u^{\mathfrak{N}} + v^{\mathfrak{N}}} = \overline{t^{\mathfrak{N}}}.$$

[1] Think of the term u maps to $n \in \mathbb{N}$. Then $\overline{t^{\mathfrak{N}}} = \overline{(S(u))^{\mathfrak{N}}} = \overline{n+1}$, and $S(\overline{u^{\mathfrak{N}}}) = S(\overline{n}) = \overline{n+1}$.

- The arguments for terms of $u \cdot v$ and $uEv = u^v$ are similar.

$\square$

Lemma 5.2.2 (Rosser's Lemma)**.** If a is a natural number,

$$N \vdash (\forall x < \overline{a})[\perp \vee (x = \overline{0}) \vee (x = \overline{1}) \vee \ldots \vee (x = \overline{a-1})].$$

Proof. Prove by induction on a.

- For $a = 0$, we need to prove $N \vdash (\forall x)[x < 0 \rightarrow \perp]$. By ($N_9$ in Robinson Arithmetic), $N \vdash (\forall x)\neg x < 0$, which implies $N \vdash (\forall x) x < 0 \rightarrow \perp$.
- Suppose $a = b + 1$ and $N \vdash (\forall x)[x < \overline{b} \rightarrow x = \overline{0} \vee \ldots \vee x = \overline{b-1}]$. We want to show

  $$N \vdash (\forall x)[x < \overline{b+1} \rightarrow x = \overline{0} \vee \ldots \vee x = \overline{b-1} \vee x = \overline{b}].$$

  Notice $\overline{b+1} = S(\overline{b})$, this implies that we need to show:

  $$N \vdash (\forall x)[x < S(\overline{b}) \rightarrow x = \overline{0} \vee \ldots \vee x = \overline{b-1} \vee x = \overline{b}].$$

  By (N_{10} in Robinson Arithmetic): $(\forall x)(\forall y)[x < Sy \leftrightarrow (x < y \vee x = y)]$, we find

  $$N \vdash (\forall x)[x < S(\overline{b}) \rightarrow (x < \overline{b} \vee x = \overline{b})].$$

  By induction hypothesis, this is:

  $$N \vdash (\forall x)[x < S(\overline{b}) \rightarrow x = \overline{0} \vee \ldots \vee x = \overline{b-1} \vee x = \overline{b}].$$

$\square$

Corollary 5.2.1. If a is a natural number, then

$$N \vdash [(\forall x < \overline{a})\phi(x)] \leftrightarrow [\phi(\overline{0}) \wedge \phi(\overline{1}) \wedge \ldots \wedge \phi(\overline{a-1})].$$

Proof. We add a patch to suit for the case of $a = 0$. Let $\top :\equiv (\forall x)[x = x]$, and obviously $N \vdash \top$. We want to prove by two parts: (I) and (II).

$$N \vdash [(\forall x < \overline{a})\phi(x)] \rightarrow [\top \wedge \phi(\overline{0}) \wedge \ldots \wedge \phi(\overline{a-1})] \tag{I}$$

and

$$N \vdash [\top \wedge \phi(\overline{0}) \wedge \ldots \wedge \phi(\overline{a-1})] \rightarrow [(\forall x < \overline{a})\phi(x)] \tag{II}$$

- For (I). Pick an arbitrary $a \in \mathbb{N}$. Remind that $(\forall x < \overline{a})\phi(x) \iff (\forall x)[x < \overline{a} \to \phi(x)]$. Thus, by (Q1), we have for any $b \in \mathbb{N}$,

  $$N \vdash (\forall x)[x < \overline{a} \to \phi(x)] \to [\overline{b} < \overline{a} \to \phi(\overline{b})] \tag{i}$$

  Lemma 2.8.1 (3) states for $b < a$:

  $$N \vdash \overline{b} < \overline{a} \tag{3}$$

  Thus, by (3), (i) and (PC) in Definition 2.4.5, we have for any $b < a$:

  $$N \vdash [(\forall x < \overline{a})\phi(x)] \to \phi(\overline{b}) \tag{ii}$$

  Combine (ii) for all $b < a$ and by (PC) in Definition 2.4.5, we have:

  $$N \vdash [(\forall x < \overline{a})\phi(x)] \to [\top \wedge \phi(\overline{0}) \wedge \ldots \wedge \phi(\overline{a-1})]$$

  This shows (I) holds.

- For (II). Notice $\phi(\overline{b})$ is actually $\phi|^{x}_{\overline{b}}$, which implies that for any $b \in \mathbb{N}$,

  $$\vdash \phi(\overline{b}) \to (x = \overline{b} \to \phi(x)) \tag{iii}$$

  Notice that tautologically,

  $$\vdash [A_0 \to (B_0 \to C)] \wedge [A_1 \to (B_1 \to C)] \to [(A_0 \wedge A_1) \to ((B_0 \wedge B_1) \to C)]$$

  Thus, by (iii) and (PC) in Definition 2.4.5, we have:

  $$\vdash [\top \wedge \phi(\overline{0}) \wedge \ldots \wedge \phi(\overline{a-1})] \to [(x = \overline{0} \vee \ldots \vee x = \overline{a-1}) \to \phi(x)] \tag{iv}$$

  By Rosser's Lemma 5.2.2, we have:

  $$N \vdash x < \overline{a} \to (x = \overline{0} \vee \ldots \vee x = \overline{a-1}) \tag{v}$$

  By (iv) and (v) with (PC) in Definition 2.4.5, we have:

  $$N \vdash [\top \wedge \phi(\overline{0}) \wedge \ldots \wedge \phi(\overline{a-1})] \to [x < \overline{a} \to \phi(x)] \tag{vi}$$

  By (vi) and (QR) in Definition 2.4.6: $[\psi \to \phi] \to [\psi \to (\forall x)\phi]$, we have:

  $$N \vdash [\top \wedge \phi(\overline{0}) \wedge \ldots \wedge \phi(\overline{a-1})] \to (\forall x)[x < \overline{a} \to \phi(x)] \tag{vii}$$

  Notice $(\forall x < \overline{a})\phi(x)$ is shorthand for $(\forall x)[x < \overline{a} \to \phi(x)]$. This means (vii) is actually (II). So we just prove (II).

□

Now we are in the stage to deliver one major result that N is strong enough to prove all true Σ-sentences.

Proposition 5.2.1. Let $\phi(\underset{\sim}{x})$ be a Σ-formula with free variables $\underset{\sim}{x}$, $\underset{\sim}{t}$ be variable-free terms. If $\mathfrak{N} \models \phi(\underset{\sim}{t})$, then $N \vdash \phi(\underset{\sim}{t})$.

Proof. Prove by induction on the complexity of the formula ϕ by the Definition 1.2.2.

- Base case: $\phi(\underset{\sim}{x})$ is atomic or $\neg$ (atomic).

  Suppose for example $\phi(x, y) :\equiv x < y$. Let the terms be t and u. By assumption, we have $\mathfrak{N} \models t < u$. This means $t^{\mathfrak{N}} < u^{\mathfrak{N}}$.

  By Lemma 2.8.1, $N \vdash \overline{t^{\mathfrak{N}}} < \overline{u^{\mathfrak{N}}}$.

  By Lemma 5.2.1, $N \vdash t = \overline{t^{\mathfrak{N}}}$ and $N \vdash u = \overline{u^{\mathfrak{N}}}$.

  So by logical axiom (E3), $N \vdash t < u$.

- Suppose $\phi(x, y) :\equiv (x \vee y)$. Let the terms be α and β, so $\phi(\alpha, \beta) :\equiv (\alpha \vee \beta)$. Without loss of generality, assume $\mathfrak{N} \models \alpha$.

  By induction hypothesis, we have $N \vdash \alpha$.

  By (PC) in Definition 2.4.5, we have $N \vdash \alpha \vee \beta$.

- Suppose $\phi(x) :\equiv (\exists x)\alpha(x)$. Assume $\mathfrak{N} \models \phi(x)$. Then, there exists $a \in \mathbb{N}$ such that $\mathfrak{N} \models \alpha^{x}_{\overline{a}}$.

  $\alpha^{x}_{\overline{a}}$ has lower complexity than $\phi(x)$. So by induction hypothesis, $N \vdash \alpha^{x}_{\overline{a}}$.

  By logical axiom (Q2): $\vdash \alpha^{x}_{\overline{a}} \to (\exists x)\alpha$ since $\overline{a}$ is variable-free term and is substitutable for x in α.

  Thus, by (PC) rule in Definition 2.4.5, we have $N \vdash (\exists x)\alpha(x)$.

- Suppose $\mathfrak{N} \models (\forall x < u)\alpha(x)$ where u is a variable-free term.

  It follows that $\mathfrak{N} \models \alpha^{x}_{\overline{a}}$ for every $a < u^{\mathfrak{N}}$.

  By induction hypothesis, we have $N \vdash \alpha^{x}_{\overline{a}}$ for every $a < u^{\mathfrak{N}}$.

  By Corollary 5.2.1, we have:

  $$N \vdash [(\forall x < \overline{u^{\mathfrak{N}}})\alpha(x)] \leftrightarrow [\alpha(\overline{0}) \wedge \alpha(\overline{1}) \wedge \ldots \wedge \alpha(\overline{u^{\mathfrak{N}} - 1})]$$

  By (PC) rule in Definition 2.4.5, we have $N \vdash [(\forall x < \overline{u^{\mathfrak{N}}})\alpha(x)]$.

By Lemma 5.2.1, $N \vdash u = \overline{u^{\mathfrak{N}}}$.

Thus, we have $N \vdash [(\forall x < u)\alpha(x)]$

$\square$

5.3 Definable and Representable Sets and Functions

Definition 5.3.1. A set $A \subseteq \mathbb{N}^k$ is **$\Sigma/\Pi/\Delta$-definable** if there exists a $\Sigma/\Pi/\Delta$-formula $\phi(x_1, \ldots x_k)$ such that

- $\mathfrak{N} \models \phi(\overline{a_1}, \ldots, \overline{a_k})$ for all $(a_1, \ldots, a_k) \in A$.
- $\mathfrak{N} \models \neg\phi(\overline{b_1}, \ldots, \overline{b_k})$ for all $(b_1, \ldots, b_k) \notin A$.

Definition 5.3.2. A set $A \subseteq \mathbb{N}^k$ is **representable** if there exists a formula $\phi(x_1, \ldots, x_k)$ such that

- $N \vdash \phi(\overline{a_1}, \ldots, \overline{a_k})$ for all $(a_1, \ldots, a_k) \in A$.
- $N \vdash \neg\phi(\overline{b_1}, \ldots, \overline{b_k})$ for all $(b_1, \ldots, b_k) \notin A$.

Definition 5.3.3. A set $A \subseteq \mathbb{N}^k$ is **weakly representable** if there exists a formula $\phi(x_1, \ldots, x_k)$ such that

- $N \vdash \phi(\overline{a_1}, \ldots, \overline{a_k})$ for all $(a_1, \ldots, a_k) \in A$.
- $N \nvdash \neg\phi(\overline{b_1}, \ldots, \overline{b_k})$ for all $(b_1, \ldots, b_k) \notin A$.

Definition 5.3.4. [2] A function $f : A \to \mathbb{N}$ where $A \subseteq \mathbb{N}^k$ is **definable** or **representable** according to the corresponding set $\{(a_1, \ldots, a_k, b) : f(a_1, \ldots, a_k) = b\} \subseteq \mathbb{N}^{k+1}$.

[2]Specifically, we can start from separating between total function and partial function by the subtlety of the domain.

Definition 5.3.5 (Total Function and Partial Function)**.** Suppose $A \subseteq \mathbb{N}^k$ and $f : A \to \mathbb{N}$. If $A = \mathbb{N}^k$, then f is a **total function**. If $A \subset \mathbb{N}^k$, then f is a **partial function**.

Analogous to representable set and weakly representable set, we have representable function and weakly representable function.

Definition 5.3.6 (Representable Function)**.** Let $f : \mathbb{N}^k \to \mathbb{N}$ is a total function. Then f is a **representable function (in N)** if there is an $\mathcal{L}_{NT}$-formula $\phi(x_1, \ldots, x_k, x_{k+1})$ such that, for all $a_1, \ldots, a_k, a_{k+1} \in \mathbb{N}$,

$$\begin{array}{ll} \text{If } f(a_1, \ldots, a_k) = a_{k+1}, & \text{then } N \vdash \phi(\overline{a_1}, \ldots, \overline{a_k}, \overline{a_{k+1}}) \\ \text{If } f(a_1, \ldots, a_k) \neq a_{k+1}, & \text{then } N \vdash \neg\phi(\overline{a_1}, \ldots, \overline{a_k}, \overline{a_{k+1}}). \end{array}$$

Example 5.3.1. The function $f = x^2$ is Δ-definable, since it is defined by the Δ-formula $\phi(x, y) :\equiv (y = x \cdot x)$, or $\phi(x, y) :\equiv (y = xESS0)$.

Definition 5.3.8. ϕ is **provable (from N)** if $N \vdash \phi$. ϕ is **refutable (from N)** if $N \vdash \neg\phi$.

Recall that N is strong enough to prove all true Σ-sentences:

Proposition 5.2.1. Let $\phi(\underset{\sim}{x})$ be a Σ-formula with free variables $\underset{\sim}{x}$, $\underset{\sim}{t}$ be variable-free terms. If $\mathfrak{N} \models \phi(\underset{\sim}{t})$, then $N \vdash \phi(\underset{\sim}{t})$.

Suppose ϕ is a Δ-sentence. Let $\mathfrak{N} \models \phi$. Since a Δ-sentence is also a Σ-sentence, by proposition 5.2.1, we have $N \vdash \phi$.

If ϕ is false, then $\mathfrak{N} \not\models \phi$ or $\mathfrak{N} \models \neg\phi$. $\neg\phi$ is equivalent to a Δ-sentence, then again by proposition 5.2.1, we have $N \vdash \neg\phi$. The following proposition 5.3.3 sums up this finding.

Proposition 5.3.3. If $\phi(\underset{\sim}{x})$ is a Δ-formula with free variable $\underset{\sim}{x}$, if $\underset{\sim}{t}$ are variable-free terms, and if $\mathfrak{N} \models \phi(\underset{\sim}{t})$, then $N \vdash \phi(\underset{\sim}{t})$. If, on the other hand, $\mathfrak{N} \models \neg\phi(\underset{\sim}{t})$, then $N \vdash \neg\phi(\underset{\sim}{t})$.

By proposition 5.3.3 and Definition 5.3.2, we have the following Corollary 5.3.1. Notice the Corollary 5.3.1 has an implication: It gives us a convenient way to check whether a set is representable.

Corollary 5.3.1. Every Δ-definable set is representable.

Definition 5.3.7 (Weakly Representable Function). Let $f : A \subseteq \mathbb{N}^k \to \mathbb{N}$ is a (possibly) partial function. Then f is a **weakly representable function (in N)** if there is an $\mathcal{L}_{NT}$-formula $\phi(x_1, \ldots, x_k, x_{k+1})$ such that, for all $a_1, \ldots, a_k, a_{k+1} \in \mathbb{N}$,

$$\begin{array}{ll} \text{If } f(a_1, \ldots, a_k) = a_{k+1}, & \text{then } N \vdash \phi(\overline{a_1}, \ldots, \overline{a_k}, \overline{a_{k+1}}) \\ \text{If } f(a_1, \ldots, a_k) \neq a_{k+1}, & \text{then } N \not\vdash \phi(\overline{a_1}, \ldots, \overline{a_k}, \overline{a_{k+1}}). \end{array}$$

It is important to know if a function f is total function.

Proposition 5.3.1. If f is a total function $f : \mathbb{N}^k \to \mathbb{N}$. Then f is representable if and only if f is weakly representable.

Proposition 5.3.2. If f is a total function $f : \mathbb{N}^k \to \mathbb{N}$. Then the following are equivalent:

1. f is a representable function.
2. There exists an $\mathcal{L}_{NT}$-formula $\psi(x_1, \ldots, x_{k+1})$ such that for all $(a_1, \ldots, a_k) \in \mathbb{N}^k$,

$$N \vdash (\forall y)[\psi(\overline{a_1}, \ldots, \overline{a_k}, y) \leftrightarrow y = \overline{f(\overline{a_1}, \ldots, \overline{a_k}}].$$

Proof. Suppose $A \subseteq \mathbb{N}^k$ is defined by a Δ-formula $\phi(x_1, \ldots, x_n)$. Both $\phi(x_1, \ldots, x_n)$ and $\neg\phi(x_1, \ldots, x_n)$ are logically equivalent to Σ-formulas. Therefore, by Proposition 5.2.1, we have:

- $N \vdash \phi(\overline{a_1}, \ldots, \overline{a_n})$, since $\mathfrak{N} \models \phi(\overline{a_1}, \ldots, \overline{a_n})$ for every $(a_1, \ldots, a_n) \in A$.
- $N \vdash \neg\phi(\overline{b_1}, \ldots, \overline{b_n})$, since $\mathfrak{N} \models \neg\phi(\overline{b_1}, \ldots, \overline{b_n})$ for every $(b_1, \ldots, b_n) \notin A$.

□

The converse of Corollary 5.3.1 is partially covered in the following Proposition.

Proposition 5.3.4. Suppose $A \subseteq \mathbb{N}^k$ is representable. Then there is a Σ-formula that defines A.

From now on, if we want to verify a set is representable, we simply need to find the Δ-formula that defines the set.

Example 5.3.2 (The set of even numbers)**.** Define the set of even numbers, denoted by EVEN, by a $\mathcal{L}_{NT}$-formula:

$$\phi(x) :\equiv (\exists y)(x = y + y).$$

Or, we can define by a Δ-formula (Δ-definition of EVEN) in Table 5.1:

Table 5.1: *Even*(x)

Even(x) is: $$(\exists y \leq x)(x = y + y).$$

Now we have a Δ-definition of EVEN. By Corollary 5.3.1, we verify the set of even numbers EVEN is a representable subset of natural numbers.

Example 5.3.3. Suppose we have already written $\boxed{Prime(x)}$, a Δ-definition of PRIME, the set of prime numbers. Then we can define a set of prime pairs, denoted by PRIMEPAIR as: the set of pairs of numbers x and y such that x and y are primes and y is the next prime after x.

$$\text{PRIMEPAIR} := \{(p_i, p_{i+1}) : i \geq 1\} \subseteq \mathbb{N}^2$$

Table 5.2: $Primepair(x, y)$

$Primepair(x, y)$ is:
$Prime(x) \wedge Prime(y) \wedge (x < y) \wedge [(\forall z < y)(Prime(z) \rightarrow z \leq x)]$.

PRIMEPAIR has Δ-definition in Table 5.2.

Notice that, as PRIMEPAIR $\subseteq \mathbb{N}^2$, $Primepair(x, y)$ has two free variables.

Also notice that the quantifier $\forall z$ is bounded. $Prime(x)$ and $Prime(y)$ are Δ-definition, thus are also bounded. So $Primepair(x, y)$ is bounded as well, and thus is a Δ-formula, which means $Primepair(x, y)$ is a Δ-definition of PRIMEPAIR and thus PRIMEPAIR is representable.

The following definition will be used in later context.

Definition 5.3.9. Let $A \subseteq \mathbb{N}$. **The characteristic function of** A is $\chi_A : \mathbb{N} \rightarrow \mathbb{N}$ by

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

5.4 Representable Functions and Computer Programs

Three best-known models of computability:

1. Kurt Gödel's recursive functions (now often called computable functions).

2. The Turing Machines.

3. Alonzo Church's λ-calculus.

It is known that a function is Turing computable if and only if general recursive if and only if λ-computable. In other words, these three models of computability are equivalent.

Theorem 5.4.1 (Church's Thesis)**.** A total function f is calculable if and only if is representable.

Church's Thesis is a thesis in the sense that it is not that sort of "theorems" that could be proved. We simply admit the thesis by agreeing on the belief that the formal models of computation accurately represent the intuitive idea of a calculable function.

Definition 5.4.1 (Informal Definition of Calculable Function)**.**

- Informally speaking, a partial function $f : A \subseteq \mathbb{N} \to \mathbb{N}$ is **calculable** if there is an algorithm or computation that, given input $n \in \mathbb{N}$, does exactly one of the following:
  - If $f(n)$ is defined, the algoritheorem computes the correct value of $f(n)$, outputs $f(n)$, then halts;
  - If $f(n)$ is not defined, the algoritheorem runs without halting.
- If a function f is total and calculable, there is an algorithm that will compute $f(n)$ for all inputs n.
- If a function g is partial and calculable, then its algorithm will halt with $g(n)$ is defined, but will run forever if $g(n)$ is not defined.

Church's Thesis can be stated in terms of partial functions as well, if we consider the connection between representable and weakly representable by Proposition 5.3.1: If a total function f is calculable, then f is also a partial function and by Theorem 5.4.2, f is weakly representable. By Proposition 5.3.1, f as total and weakly representable function is representable as well. Thus, we have Theorem 5.4.1.

Theorem 5.4.2 (Church's Thesis in Partial Functions)**.** A partial function f is calculable if and only if f is weakly representable.

The infamous Church-Turing Thesis is stated here in Theorem 5.4.3.

Theorem 5.4.3 (The Church-Turing Thesis)**.** A function on the natural numbers is computable by a human being following an algorithm, ignoring resource limitations, if and only if it is computable by a Turing machine or any other equivalent notion (e.g., representability).

5.5 Coding is Representable

Recall:

- Let $p_1 = 2$, $p_2 = 3$, $p_3 = 5$, …, $p_i = i^{th}$ prime numbers.

- $\mathbb{N}^{<\mathbb{N}}$ is the set of finite sequence of natural numbers.
- Sequence-coding function by Definition 4.4.3 $\langle\rangle : \mathbb{N}^{<\mathbb{N}} \to \mathbb{N}$ as:

$$\langle a_1, \dots, a_k \rangle := \prod_{i=1}^{k} p_i^{a_i+1} = 2^{a_1+1} 3^{a_2+1} \dots p_k^{a_k+1}.$$

- Let $a = \langle a_1, \dots, a_k \rangle$ [3], then $|a| = k$ and $(a)_i = a_i$.

Theorem 5.5.1. All of the sequence-coding operations are Δ-definable, and thus representable.

Proof. We will prove by constructing Δ-formulas for all the sequence-coding operations in below. □

- Δ-definitions for Divides in Table 5.3:

Table 5.3: *Divides*(y, x)

$\mathit{Divides}(y, x) :\equiv (\exists z \leq x)[x = y \cdot z].$

- Δ-definitions for Prime number in Table 5.4:

Table 5.4: *Prime*(x)

$\mathit{Prime}(x) :\equiv S0 < x \wedge (\forall y \leq x)\big[\mathit{Divides}(y, x) \to (y = 1 \vee y = x)\big]$.

- The set PrimePair $:= \{(p_i, p_{i+1}) : i \geq 1\} \subseteq \mathbb{N}^2$ is defined as the Δ-formula in Table 5.2.
- Δ-definition for Codenumber the set of code numbers in Table 5.5:
- Define the set

$$\text{Yardstick} := \{\underbrace{\langle 0, 1, 2, \dots, k-1 \rangle}_{2^1 3^2 5^3 \dots (p_k)^k} : k \in \mathbb{N}\}.$$

Table 5.5: $Codenumber(c)$

$$Codenumber(c) :\equiv Divides(SS0, c) \wedge (\forall z < c)(\forall y < z)$$
$$\Big[\big(Primepair(y,z) \wedge Divides(z,c)\big) \to Divides(y,c)\Big].$$

Table 5.6: $Yardstick(x)$

$$Yardstick(x) :\equiv Divides(\overline{2}, x) \wedge \neg Divides(\overline{4}, x) \wedge$$
$$(\forall y \le x)(\forall z \le x)(\forall i < x)\ \Big[\big(Primepair(y,z) \wedge Divides(z,x)\big) \to$$
$$\big(Divides(\underbrace{yEi}_{y^i}, x) \leftrightarrow Divides(\underbrace{zESi}_{z^{i+1}}, x)\big)\Big].$$

It has the Δ-formula in Table 5.6.

- Notice that YARDSTICK is the collection of numbers a of the form $2^1 3^2 5^3 \ldots p_i^i$ for some i. The first few elements are $\{2, 18, 2250, \ldots\}$.
- $Yardstick(x)$ says that 2 divides x, 4 does not divide x, and if z is a prime such that z divides x, then the power of z into x is 1 more than the power of previous prime into x.

• Define the set

$$\text{ITHPRIME} := \{(i, p_i) : i \in \mathbb{N} \wedge i \le 1\}$$

by Δ-formula at Table 5.7:

- We want x to be the i-th yardstick number $(p_1)^1(p_2)^2 \ldots (p_i)^i$. And we have

$$(p_1)^1(p_2)^2 \ldots (p_i)^i \le (p_i)^1(p_i)^2 \ldots (p_i)^i = (p_i)^{1+2+\ldots+i} = (p_i)^{\frac{i(i+1)}{2}} \le (p_i)^{i^2}.$$

  Thus, if $y = p_i$, we have the boundary $\exists x \le y^{i^2}$, as in Table 5.7.

[3] $\langle a_1, \ldots, a_k \rangle$ is the shorthand of $\langle (a_1, \ldots, a_k) \rangle$.

Table 5.7: $IthPrime(i, y)$

$$IthPrime(i,y) :\equiv Prime(y) \wedge \\ (\exists x \leq y^{i^2})\big[Yardstick(x) \wedge Divides(\underbrace{yEi}_{y^i}, x) \wedge \neg Divides(\underbrace{yESi}_{y^{i+1}}, x)\big].$$

- The set IthPrime corresponds to the $IthPrime$ function at Definition 4.4.1. Therefore, the Δ-formula $IthPrime(i, y)$ at Table 5.7 defines the $IthPrime$ function at Definition 4.4.1.
- As an example, we know the 17-th prime is 59, we can then use the explicit $\mathcal{L}_{NT}$-formula to assert this fact by $N \vdash IthPrime(\overline{17}, \overline{59})$.

- Define the set

  $$\text{Length} := \{(\langle a_1, \ldots, a_k\rangle, k) : k \geq 1 \text{ and } (a_1, \ldots, a_k) \in \mathbb{N}^k\}$$

  by the Δ-formula at Table 5.5 with denoting $c = \langle a_1, \ldots, a_k\rangle$:

Table 5.8: $Length(c, l)$

$$Length(c,\ell) :\equiv Codenumber(c) \wedge \\ (\exists y \leq c)\Bigg[\Big(IthPrime(\ell, y) \wedge Divides(y, c) \wedge \\ (\forall z \leq c)\big[PrimePair(y, z) \rightarrow \neg Divides(z, c)\big]\Big)\Bigg]$$

- Define the set

  $$\text{IthElement} := \{(a_j, j, \langle a_1, \ldots, a_k\rangle) : 1 \leq j \leq k \text{ and } (a_1, \ldots, a_k) \in \mathbb{N}^k\}$$

  by the Δ-formula at Table 5.9:

  - $IthElement(e, i, c)$ is true if c is a code and e is the number at position i of the sequence coded by c. Example: $(7, 9, 11, 9)$ is coded by 1042492561137562500000000. Thus,

    $$IthElement(e = \overline{7}, i = \overline{1}, c = \overline{1042492561137562500000000}) = True.$$

Table 5.9: $IthElement(e, i, c)$

$$IthElement(e, i, c) :\equiv Codenumber(c) \wedge (\exists y \leq c)\Big[IthPrime(i, y) \wedge$$
$$Divides(y^{Se}, c) \wedge \neg Divides(y^{SSe}, c)\Big]$$

Here below we sum up some useful propositions that are derived from the Δ-formula constructions.

Proposition 5.5.1. The collection of code numbers for finite sequence is a representable set.

Proof. By the Δ-definition for Codenumber in Table 5.5 and by Corollary 5.3.1, Codenumber is a representable set.

□

Proposition 5.5.2. The function p that enumerates the primes is a representable function.

Proof. By the corresponding Δ-definition of $IthPrime(i, y)$, the function p is representable.

□

5.6 Gödel Numbers of Terms and Formulas

Definition 5.6.1. Assign **symbol numbers** to the symbols of $\mathcal{L}_{NT}$, given in Table 5.10.

- For an $\mathcal{L}_{NT}$-formula $s :\equiv s_1 \dots s_n$, we could encode s by the number $\langle \#(s_1), \dots, \#(s_n) \rangle$ where $\#(s_i)$ is the Gödel number corresponding to the symbol s_i.

- For example,

  Example 5.6.1. Consider the sentence $= 0S0$. The sequence of symbol numbers is
  $$(=, 0, S, 0) = (7, 9, 11, 9).$$
  The code for this sequence is
  $$2^{7+1}3^{9+1}5^{11+1}7^{9+1} = 2^8 3^{10} 5^{12} 7^{10} = 1042492561137562500000000.$$

Symbol	Symbol Number	Symbol	Symbol Number
$\neg$	1	$+$	13
$\vee$	3	$\cdot$	15
$\forall$	5	E	17
$=$	7	$<$	19
0	9	(	21
S	11	)	23
		v_i	$2i$

Table 5.10: Symbol Numbers for $\mathcal{L}_{NT}$

- But it would be better to encode s according to the inductive type of terms and formulas, see the Gödel numbering function in Definition 5.6.2.

Definition 5.6.2. For each term or formula s, the **Gödel numbering function** $\ulcorner s \urcorner$ is defined as:

$$\ulcorner s \urcorner = \begin{cases} \langle 1, \ulcorner \alpha \urcorner \rangle & \text{if } s :\equiv (\neg\alpha), \text{ where } \alpha \text{ is an } \mathcal{L}_{NT}\text{-formula} \\ \langle 3, \ulcorner \alpha \urcorner, \ulcorner \beta \urcorner \rangle & \text{if } s :\equiv (\alpha \vee \beta), \text{ where } \alpha \text{ and } \beta \text{ are } \mathcal{L}_{NT}\text{-formulas} \\ \langle 5, \ulcorner v_i \urcorner, \ulcorner \alpha \urcorner \rangle & \text{if } s :\equiv (\forall v_i)(\alpha), \text{ where } \alpha \text{ is an } \mathcal{L}_{NT}\text{-formula} \\ \langle 7, \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner \rangle & \text{if } s :\equiv = t_1 t_2, \text{ where } t_1 \text{ and } t_2 \text{ are terms} \\ \langle 9 \rangle & \text{if } s :\equiv 0 \\ \langle 11, \ulcorner t \urcorner \rangle & \text{if } s :\equiv St, \text{ where } t \text{ is a term} \\ \langle 13, \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner \rangle & \text{if } s :\equiv +t_1 t_2, \text{ where } t_1 \text{ and } t_2 \text{ are terms} \\ \langle 15, \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner \rangle & \text{if } s :\equiv \cdot t_1 t_2, \text{ where } t_1 \text{ and } t_2 \text{ are terms} \\ \langle 17, \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner \rangle & \text{if } s :\equiv E t_1 t_2, \text{ where } t_1 \text{ and } t_2 \text{ are terms} \\ \langle 19, \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner \rangle & \text{if } s :\equiv < t_1 t_2, \text{ where } t_1 \text{ and } t_2 \text{ are terms} \\ \langle 2i \rangle & \text{if } s \text{ is the variable } v_i \\ 3 & \text{otherwise.} \end{cases}$$

- Definition 5.6.2 has each symbol associated with its symbol number in Table 5.10.
- $\ulcorner s \urcorner$ is never divisible by 7: 7 is the 4-th prime number while $\ulcorner s \urcorner$ has only up to the 3-rd prime number in the Gödel numbering function at Definition 5.6.2.

- Notice $3 \neq \langle 3\rangle = 2^4 = 16$.

Example 5.6.2. Find the Gödel numbers for the following:

- $\ulcorner 0 \urcorner = \langle 9 \rangle = 2^{10} = 1024$.
- $\ulcorner 0 = 0 \urcorner = \ulcorner = 00 \urcorner = \langle 7, \ulcorner 0 \urcorner, \ulcorner 0 \urcorner \rangle = \langle 7, 1024, 1024\rangle = 2^8 3^{1025} 5^{1025}$.
- $\ulcorner = 0S0 \urcorner = \langle 7, \ulcorner 0 \urcorner, \ulcorner S0 \urcorner \rangle = 2^8 3^{1025} 5^{\ulcorner S0 \urcorner + 1}$.
  With $\ulcorner S0 \urcorner = \langle 11, \ulcorner 0 \urcorner \rangle = 2^{12} 3^{1025}$,
  $$\ulcorner = 0S0 \urcorner = 2^8 3^{1025} 5^{\ulcorner S0 \urcorner + 1} = 2^8 3^{1025} 5^{(2^{12}3^{1025}+1)}.$$
- Gödel number grows very fast:
  $$\ulcorner SSSS0 \urcorner = \langle 11, \langle 11, \langle 11, \langle 11, \langle 9 \rangle\rangle\rangle\rangle\rangle = 2^{12} 3^{2^{12}3^{2^{12}3^{2^{12}3^{2^{10}}}}}.$$

Next, we need the Δ-definition of the following sets:

- Term
  $$\text{Term} := \{\ulcorner t \urcorner : \text{ term } t\} = \{a \in \mathbb{N} : a = \ulcorner t \urcorner \text{ for some term } t\}.$$
- Formula
  $$\text{Formula} := \{\ulcorner \phi \urcorner : \text{ formula } \phi\} = \{a \in \mathbb{N} : a = \ulcorner \phi \urcorner \text{ for some formula } \phi\}.$$
- Δ-definition of Term $:= \{\ulcorner t \urcorner : t \text{ is a term}\}$ would be something like Table 5.11

Table 5.11: *Term*

$$\begin{array}{rr}
\ulcorner \neg\alpha \urcorner = \langle 1, \ulcorner \alpha \urcorner \rangle & \ulcorner (\alpha \vee \beta) \urcorner = \langle 3, \ulcorner \alpha \urcorner, \ulcorner \beta \urcorner \rangle \\
\ulcorner (\forall v_i)(\alpha) \urcorner = \langle 5, \ulcorner v_i \urcorner, \ulcorner \alpha \urcorner \rangle & \ulcorner = t_1 t_2 \urcorner = \langle 7, \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner \rangle \\
\ulcorner 0 \urcorner = \langle 9 \rangle & \ulcorner St \urcorner = \langle 11, \ulcorner t \urcorner \rangle \\
\ulcorner + t_1 t_2 \urcorner = \langle 13, \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner \rangle & \ulcorner \cdot t_1 t_2 \urcorner = \langle 15, \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner \rangle \\
\ulcorner E t_1 t_2 \urcorner = \langle 17, \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner \rangle & \ulcorner < t_1 t_2 \urcorner = \langle 19, \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner \rangle \\
\ulcorner = v_i \urcorner = \langle 2i \rangle &
\end{array}$$

- By inductive definition of $\mathcal{L}_{NT}$-term t at Definition 1.2.1, t is
  - a variable symbol v_i.
  - the constant symbol 0.
  - St_1 where t_1 is a term.
  - Functions $+t_1t_2$, $\cdot t_1t_2$, Et_1t_2 where t_1, t_2 are terms.
- Start with Δ-definition of

  $$\textsc{Variable} := \{\ulcorner v_i \urcorner : i = 1, 2, \ldots\} = \{2^{2i+1} : i = 1, 2, \ldots\}$$

  by the Δ-formula in Table 5.12

Table 5.12: $Variable(x)$

$$\boxed{Variable(x) :\equiv (\exists y < x)[Even(y) \wedge (0 < y) \wedge (x = 2^{Sy})].}$$

Lemma 5.6.1. The set

$$\textsc{Variable} := \{\ulcorner v_i \urcorner : i = 1, 2, \ldots\} = \{2^{2i+1} : i = 1, 2, \ldots\}$$

is representable.

Proof. Prove by the Δ-definition for Variable in Table 5.12. □

We look for writing $Term(x)$ as:

$$Term(x) :\equiv Variable(x) \vee \overbrace{(x = \overline{2^1 0})}^{\text{"}x \text{ is } \ulcorner 0 \urcorner\text{"}} \quad \vee \quad \overbrace{(\exists y < x)[Term(y) \wedge x = \underbrace{\overline{2^{12}} \cdot \overline{3}^{Sy}}_{\langle 11,y \rangle}}^{\text{"}x \text{ is } \ulcorner St_1 \urcorner \text{ for some term } t_1\text{"}}$$

$$\vee \quad \underbrace{\cdots}_{\text{"}x \text{ is } +t_1t_2, \text{ or } \cdot t_1t_2, \text{ or } Et_1t_2\text{"}}$$

However, there is a problem in this "Δ-formula": It is not a legitimate formula in first-order logic, see the recursive use of the subformula $Term(y)$, i.e., $Term(\cdot)$ shows up on both LHS and RHS.

We use a technique called **term construction sequence** to turn around from this issue.

Definition 5.6.3. A **term construction sequence** for a term t is a finite sequence of terms $(t_1, \ldots, t_\ell)$ such that $t_\ell :\equiv t$ and for each $k = 1, \ldots, \ell$, the term t_k is either

- a variable symbol,
- the constant symbol 0,
- St_j for some $j < k$,
- $+t_it_j$, or $\cdot t_it_j$, or Et_it_j for some $i, j < k$.

Example 5.6.3. Consider $t :\equiv +0Sv_1$. It has a term construction sequence $(0, v_1, Sv_1, +0Sv_1)$.

Lemma 5.6.2. Every term t has a term construction sequence of length at most the number of symbols in t.

The key idea of defining Term: to write a Δ-formula defining the set

$$\begin{aligned}\textsc{TermConSeq} = \ \{ \ & (c, a) : c = \langle \ulcorner t_1 \urcorner, \ldots, \ulcorner t_\ell \urcorner \rangle \text{ and } a = \ulcorner t_\ell \urcorner \text{ where} \\ & (t_1, \ldots, t_\ell) \text{ is a term construction sequence}\}.\end{aligned}$$

Table 5.13 states that $(c, a) \in \textsc{TermConSeq}$ if and onlyif c is a code of length ℓ, a is the last number of the sequence coded by c, and if e_i is an entry at position i of c, then e_i is either

- the Gödel number of a variable symbol,
- the Gödel number of the constant symbol 0,
- the Gödel number of Se_j where e_j is an earlier entry in c,
- the Gödel number of $+e_je_k$ where e_j and e_k are earlier entries in c,
- the Gödel number of $\cdot e_je_k$ where e_j and e_k are earlier entries in c,
- the Gödel number of Ee_je_k where e_j and e_k are earlier entries in c.

Then, we can define $Term(a)$ as:

$$Term(a) :\equiv (\exists c) TermConSeq(c, a)$$

To make this a Δ-formula, we need to find the upper bound of c as a function of a. The following lemmas help.

Table 5.13: $TermConSeq(c,a)$

$$
\begin{array}{l}
TermConSeq(c,a) :\equiv Codenumber(c)\wedge \\
(\exists \ell < c)\Big[Length(c,\ell) \wedge IthElement(a,\ell,c)\wedge \\
(\forall i \leq l)(\exists e_i < c)\big[IthElement(e_i,i,c)\wedge \\
\left(\begin{array}{l}
Variable(e_i) \\
\vee\, \overbrace{e_i = \overline{2}^{\overline{9+1}}}^{e_i \text{ is } \ulcorner 0 \urcorner} \\
\vee(\exists j < i)(\exists e_j < c)[IthElement(e_j,j,c) \wedge \overbrace{e_i = \overline{2}^{\overline{11+1}} \cdot \overline{3}^{Se_j}}^{e_i \text{ is } \ulcorner Se_j \urcorner}] \\
\vee(\exists j < i)(\exists e_j < c)(\exists k < i)(\exists e_k < c)\big[\\
\quad IthElement(e_j,j,c) \wedge IthElement(e_k,k,c)\wedge \\
\quad \big(e_i = \overline{2}^{\overline{13+1}} \cdot \overline{3}^{Se_j} \cdot \overline{5}^{Se_k} \vee \\
\quad\quad e_i = \overline{2}^{\overline{15+1}} \cdot \overline{3}^{Se_j} \cdot \overline{5}^{Se_k} \vee \\
\quad\quad e_i = \overline{2}^{\overline{17+1}} \cdot \overline{3}^{Se_j} \cdot \overline{5}^{Se_k}\big)\big]
\end{array}\right) \big]\Big]
\end{array}
$$

Lemma 5.6.3. Suppose $a = \ulcorner t \urcorner$. Then the number of symbols in t is less than a.

Lemma 5.6.4. Suppose u is a subterm of t. (In other words, u is a substring of t.), then $\ulcorner u \urcorner < \ulcorner t \urcorner$.

Lemma 5.6.5. If $a \geq 1$ is a natural number, then $p_a \leq 2^{a^a}$ where p_a is the a-th prime number.

Let $c := \langle \ulcorner t_1 \urcorner, \ldots, \ulcorner t_\ell \urcorner \rangle$. We have

$$
\begin{array}{rcl}
c & = & 2^{\ulcorner t_1 \urcorner + 1} 3^{\ulcorner t_2 \urcorner + 1} \ldots (p_\ell)^{\ulcorner t_\ell \urcorner + 1} \\
 & \leq & (p_\ell)^{\ulcorner t_1 \urcorner + \ulcorner t_2 \urcorner + \ldots + \ulcorner t_\ell \urcorner + \ell} \leq (p_\ell)^{\ell a + \ell} \\
 & \leq & (2^{a^a})^{\ell a + \ell} \leq (2^{a^a})^{a^2 + a}
\end{array}
$$

Therefore, we construct the Δ-definition of the $Term(a)$ as in Table 5.14:

Similarly, we use the concept of **formula construction sequence**, to get the Δ-definition of the set

$$\text{FORMULA} = \{\ulcorner \phi \urcorner : \phi \text{ is a formula }\}.$$

Table 5.14: $Term(a)$

$$Term(a) :\equiv \big(\exists c \leq (\overline{2}^{a^a})^{a^{\overline{2}}+a}\big)TermConSeq(c, a).$$

Definition 5.6.4. A **Formula Construction Sequence** for a formula ϕ is a finite sequence of terms $(\phi_1, \phi_2, \dots, \phi_\ell)$ such that $\phi_\ell :\equiv \phi$, and for each $k = 1, \dots, \ell$, the term ϕ_k is either

- $= t_1 t_2$ for some terms t_1 and t_2,
- $< t_1 t_2$ for some terms t_1 and t_2,
- $\neg\phi_j$ for some $j < k$,
- $()\phi_i \vee \phi_j$ for some $i, j < k$,
- $(\forall x)\phi_i$ for some $i < k$ and $x \in Vars$.

Remark 5.6.1. The idea is general: using an appropriate notion of *construction sequence*, we can construct Δ-definition of any recursively defined set or function.

5.7 Useful Δ-definable sets and Functions

In short, we need to use the following Δ-definable functions:

- $\mathrm{Num}(a) := \ulcorner \overline{a} \urcorner$,
- $\mathrm{TermSub}(\ulcorner u \urcorner, \ulcorner x \urcorner, \ulcorner t \urcorner) := \ulcorner u^x_t \urcorner$,
- $\mathrm{Sub}(\ulcorner \phi \urcorner, \ulcorner x \urcorner, \ulcorner t \urcorner) := \ulcorner \phi^x_t \urcorner$.

Also, the following sets are Δ-definable:

- $\textsc{Axiom}_N := \{\ulcorner N_1 \urcorner, \dots, \ulcorner N_{11} \urcorner\}$.
- $\textsc{LogicalAxiom} := \{\ulcorner \phi \urcorner : \phi \text{ is a logical axiom}\}$
- $\textsc{RofI} := \{(\langle \ulcorner \gamma_1 \urcorner, \dots, \ulcorner \gamma_n \urcorner \rangle, \ulcorner \phi \urcorner) : (\gamma_1, \dots, \gamma_n, \phi) \text{ is a rule of inference}\}$.
- $\textsc{Deduction}_N := \{(\langle \ulcorner \delta_1 \urcorner, \dots, \ulcorner \delta_n \urcorner \rangle, \ulcorner \phi \urcorner) : (\delta_1, \dots, \delta_n, \phi)$ is a deduction from N of $\phi\}$.

Num$(a) := \ulcorner \overline{a} \urcorner$**.** Recall $\overline{a}$ is the term representing the number $a \in \mathbb{N}$. For example, let $a = 2$. Then, $\overline{2} :\equiv SS0$.

$$\ulcorner \overline{a} \urcorner = \ulcorner SS0 \urcorner = \langle 11, \ulcorner S0 \urcorner \rangle = \langle 11, 2^{12}3^{2^{10}+1} \rangle = 2^{12}3^{2^{12}3^{1025}+1}.$$

In this case $\text{Num}(2) = \ulcorner \overline{2} \urcorner = 2^{12}3^{2^{12}3^{1025}+1}$.

We get a Δ-formula $Num(a, y)$ that defines the function $\text{Num}(a) = \ulcorner \overline{a} \urcorner$.

Table 5.15: $Num(a, y)$

$$Num(a, y) :\equiv \big(\exists c < \text{ some bound }\big) NumConSeq(c, a, y).$$

where $NumConSeq(c, a, y)$ is similar to $TermConSeq(c, a)$: it works if and only if c is the code for a construction sequence of length $a + 1$ with the last element $y = \ulcorner \overline{a} \urcorner$.

TermSub$(\ulcorner u \urcorner, \ulcorner x \urcorner, \ulcorner t \urcorner) := \ulcorner u^x_t \urcorner$**.**

Example 5.7.1. Suppose $u :\equiv +\cdot 0S0x$. $t :\equiv SS0$. Then, $u^x_t :\equiv +\cdot 0S0SS0$.

The construction sequence of u:

$$(0, x, S0, \cdot 0S0, + \cdot 0S0x).$$

If we replace x by t in the construction sequence of u, what we have is **not** a construction sequence any more (the second term of the sequence is illegal):

$$(0, SS0, S0, \cdot 0S0, + \cdot 0S0SS0).$$

To fix this, we can append the construction sequence of t to it, so we have

$$(0, S0, SS0, 0, SS0, S0, \cdot 0S0, + \cdot 0S0SS0).$$

where the first 3 items $(0, S0, SS0)$ are from the construction sequence of $t :\equiv SS0$.

Implied by this example, we need to have 3 pieces:

1. To change u's construction sequence by replacing x by t.

   By $TermReplace\ (c, u, d, x, t)$: c is the code for a construction sequence of a term with Gödel number u. Then d is the code of the sequence (probably not a construction sequence) by replacing each occurrence of the variable with Gödel number x by the term with Gödel number t.

2. To append one construction sequence in front of another.

   By $Append(b, d, a)$: b is the code for the term construction sequence for the term t. d is the code for the sequence obtained by replacing x by t. a is the code for the term construction sequence by adding the term construction sequence of b in front of the sequence coded by d.

   - The length of $a =$ the length of $b +$ the length of d,
   - The first ℓ_a elements of the a sequence are the same as the b sequence, while the rest of the a sequence are the same as the d sequence.

3. To find the boundary for all quantifiers so the formula in target is a Δ-formula in the end.

   The key element is to find the boundary for a at $Append(b, d, a)$. Notice a has the length no more than the sum of the length of b and the length of d, say ℓ_b and ℓ_d. By similar trick that we used in the boundary of $Term(a)$, we find the boundary of a:

   $$a \leq \left(\left[2^{\ell_b + {\ell_d}^{\ell_b + \ell_d}} \right]^y \right)^{\ell_b + \ell_d}$$

   where y is the last entry of the term construction sequence of a.

So, the Δ-definition of TermSub is in Table 5.16:

Table 5.16: $TermSub(u, x, t, y)$

$$TermSub(u, x, t, y) :\equiv$$
$$\left(\exists a \leq \left(\left[2^{\ell_b + {\ell_d}^{\ell_b + \ell_d}} \right]^y \right)^{\ell_b + \ell_d} \right) (\exists b < a)(\exists d < a) \left(\exists c \leq \left(\overline{2}^{u^u}\right)^{u^{\overline{2}} + u} \right)$$
$$\Big[TermConSeq(c, u) \wedge TermConSeq(b, t) \wedge$$
$$TermReplace(c, u, d, x, t) \wedge Append(b, d, a) \wedge TermConSeq(a, y) \Big]$$

Sub$(\ulcorner\phi\urcorner, \ulcorner x\urcorner, \ulcorner t\urcorner) := \ulcorner\phi_t^x\urcorner$**.** Similar to the $TermSub(u, x, t, y)$ where u is a term, we have Δ-definition of $Sub(f, x, t, y)$ where f is a formula in Table 5.17:

Table 5.17: $Sub(f, x, t, y)$

$$\begin{gathered}
Sub(f,x,t,y) :\equiv \\
\Big(\exists a \leq \text{some bound}\Big)(\exists b < a)(\exists d < a)\Big(\exists c \leq \text{some bound}\Big) \\
\Big[FormulaConSeq(c,f) \wedge TermConSeq(b,t) \wedge \\
FormulaReplace(c,f,d,x,t) \wedge Append(b,d,a) \wedge \\
FormulaConSeq(a,y)\Big]
\end{gathered}$$

Axiom$_N$**.** Recall the Robinson Arithmetic N in Example 2.8.3

N_1: $(\forall x)\neg Sx = 0$.

N_2: $(\forall x)(\forall y)[Sx = Sy \rightarrow x = y]$.

N_3: $(\forall x)x + 0 = x$.

N_4: $(\forall x)(\forall y)x + Sy = S(x + y)$.

N_5: $(\forall x)x \cdot 0 = 0$.

N_6: $(\forall x)(\forall y)x \cdot Sy = (x \cdot y) + x$.

N_7: $(\forall x)xE0 = S0$.

N_8: $(\forall x)(\forall y)xE(Sy) = (xEy) \cdot x$.

N_9: $(\forall x)\neg x < 0$.

N_{10}: $(\forall x)(\forall y)[x < Sy \leftrightarrow (x < y \vee x = y)]$.

N_{11}: $(\forall x)(\forall y)[(x < y) \vee (x = y) \vee (y < x)]$.

We have the set

$$\text{textscAxiom}_N := \{N_1, N_2, \ldots, N_{11}\}$$

It has the Δ-definition in Table :

Table 5.18: $Axiom_N(a)$

$$Axiom_N(a) :\equiv$$
$$a = \overline{\ulcorner (\forall x)\neg Sx = 0 \urcorner} \lor$$
$$a = \overline{\ulcorner (\forall x)(\forall y)[Sx = Sy \to x = y] \urcorner} \lor$$
$$a = \overline{\ulcorner (\forall x) x + 0 = x \urcorner} \lor$$
$$\vdots$$
$$a = \overline{\ulcorner (\forall x)(\forall y)[(x < y) \lor (x = y) \lor (y < x)] \urcorner}.$$

LogicalAxiom. Recall the set of logical axioms are:

$$\textsc{LogicalAxiom} := \{E1, E2, E3, Q1, Q2\}.$$

$$x = x \quad \text{for each variable } x. \tag{E1}$$

$$\begin{array}{rr} & [(x_1 = y_1) \land (x_2 = y_2) \land \ldots \land (x_n = y_n)] \\ \to & (f(x_1, x_2, \ldots, x_n) = f(y_1, y_2, \ldots, y_n)). \end{array} \tag{E2}$$

$$\begin{array}{rr} & [(x_1 = y_1) \land (x_2 = y_2) \land \ldots \land (x_n = y_n)] \\ \to & (R(x_1, x_2, \ldots, x_n) \to R(y_1, y_2, \ldots, y_n)). \end{array} \tag{E3}$$

$$(\forall x \phi) \to \phi^x_t, \text{ if } t \text{ is substitutable for } x \text{ in } \phi \tag{Q1}$$

$$\phi^x_t \to (\exists x \phi), \text{ if } t \text{ is substitutable for } x \text{ in } \phi \tag{Q2}$$

It has the Δ-definition as in Table 5.19, where $Subsitutable(t, x, f)$ means t is a Gödel number of the term, x is the Gödel number of a variable, f is the Gödel number of the formula, and with some abuse on the notation, t is substitutable for x in f.

Table 5.19: $LogicalAxiom(a)$

$$LogicalAxiom(a) :\equiv$$

$$\Big((\exists x < a)(Variable(x) \wedge a = \overline{\ulcorner x = x \urcorner})\Big) \vee$$

$$\Big((\exists x, y < a)\Big[Variable(x) \wedge Variable(y) \wedge$$

$$\underbrace{a = \overline{\ulcorner\big((\ulcorner\neg x = y\urcorner) \vee (\ulcorner Sx = Sy\urcorner)\big)\urcorner}}_{(x=y \to Sx=Sy)}\Big]\Big) \vee$$

$$\Big(\exists x_1, x_2, y_1, y_2 < a)\Big[Variable(x_1) \wedge \ldots \wedge Variable(y_2) \wedge$$

$$a = \overline{\ulcorner\big[(x_1 = y_1) \wedge (x_2 = y_2)\big] \to (x_1 + x_2 = y_1 + y_2)\urcorner}\Big]\Big) \vee$$

$$\vdots$$

(similar codes for (E2) and (E3) for $\cdot$, E, $=$, $<$)

$$\vdots$$

$$\vee(\exists f, x, t, y < a)\Big(Formula(f) \wedge Variable(x) \wedge Term(t) \wedge$$

$$Subsitutable(t, x, f) \wedge Sub(f, x, t, y) \wedge$$

$$a = \overline{\ulcorner(\forall x f) \to f_t^x\urcorner}\Big) \vee$$

(Similar codes for (Q2)).

RofI. Recall the **Gödel numbering function** in Definition 5.6.2:

Definition 5.6.2. For each term or formula s, the **Gödel numbering function** $\ulcorner s \urcorner$ is defined as:

$$\ulcorner s \urcorner = \begin{cases} \langle 1, \ulcorner \alpha \urcorner \rangle & \text{if } s :\equiv (\neg \alpha), \text{ where } \alpha \text{ is an } \mathcal{L}_{NT}\text{-formula} \\ \langle 3, \ulcorner \alpha \urcorner, \ulcorner \beta \urcorner \rangle & \text{if } s :\equiv (\alpha \vee \beta), \text{ where } \alpha \text{ and } \beta \text{ are } \mathcal{L}_{NT}\text{-formulas} \\ \langle 5, \ulcorner v_i \urcorner, \ulcorner \alpha \urcorner \rangle & \text{if } s :\equiv (\forall v_i)(\alpha), \text{ where } \alpha \text{ is an } \mathcal{L}_{NT}\text{-formula} \\ \langle 7, \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner \rangle & \text{if } s :\equiv = t_1 t_2, \text{ where } t_1 \text{ and } t_2 \text{ are terms} \\ \langle 9 \rangle & \text{if } s :\equiv 0 \\ \langle 11, \ulcorner t \urcorner \rangle & \text{if } s :\equiv St, \text{ where } t \text{ is a term} \\ \langle 13, \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner \rangle & \text{if } s :\equiv +t_1 t_2, \text{ where } t_1 \text{ and } t_2 \text{ are terms} \\ \langle 15, \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner \rangle & \text{if } s :\equiv \cdot t_1 t_2, \text{ where } t_1 \text{ and } t_2 \text{ are terms} \\ \langle 17, \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner \rangle & \text{if } s :\equiv E t_1 t_2, \text{ where } t_1 \text{ and } t_2 \text{ are terms} \\ \langle 19, \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner \rangle & \text{if } s :\equiv < t_1 t_2, \text{ where } t_1 \text{ and } t_2 \text{ are terms} \\ \langle 2i \rangle & \text{if } s \text{ is the variable } v_i \\ 3 & \text{otherwise.} \end{cases}$$

Here we use a new code sequences of formulas starting with smallest prime number 5 (instead of 2 in previous section). So if the sequence of formulas is

$$D = (\phi_1, \phi_2, \ldots, \phi_k)$$

Then the SequenceCode is $\ulcorner D \urcorner$:

$$\text{SequenceCode} := \{ 5^{\ulcorner \phi_1 \urcorner} 7^{\ulcorner \phi_2 \urcorner} \cdots p_{k+2}^{\ulcorner \phi_k \urcorner}$$

$$: (\phi_1, \phi_2, \ldots, \phi_k) \text{ is a sequence of formulas} \}$$

It has Δ-definition in Table 5.20.

Table 5.20: *SequenceCode*(c)

$SequenceCode(c) :\equiv \overline{5^{\ulcorner \phi_1 \urcorner} 7^{\ulcorner \phi_2 \urcorner} \cdots p_{k+2}^{\ulcorner \phi_k \urcorner}}$

Also recall the Rule of inference (QR) in Definition 2.4.6 and (PC) in Definition 2.4.5.

1. For (QR):
   - (QR1): $(\{\psi \to \phi\},\ \psi \to (\forall x \phi))$
   - (QR2): $(\{\phi \to \psi\},\ (\exists x \phi) \to \psi)$

   Write the Δ-definition for (QR1) in Table 5.21, where $QRRule1(c,e,i)$ says c is the code of the deduction, e is the entry at position i and is justified by (QR1). f refers to $\ulcorner \psi \urcorner$ and g refers to $\ulcorner \phi \urcorner$.

   Table 5.21: $QRRule1(c,e,i)$

$$QRRule1(c,e,i) :\equiv$$
$$SequenceCode(c) \land IthSequenceElement(e,i,c) \land$$
$$(\exists x, f, g < c)\Big[Formula(f) \land Formula(g) \land Variable(x) \land$$
$$\neg Free(x,f) \land e = \overline{\ulcorner \big(\ulcorner \neg \psi \urcorner\big) \lor \big(\ulcorner (\forall x) \phi \urcorner\big) \urcorner} \land$$
$$(\exists j < i)(\exists e_j < c)\big(IthSequenceElement(e_j, j, c) \land e_j = \overline{\ulcorner \psi \to \phi \urcorner}\big)\Big].$$

   Similar way can lead to $QRRule2(c,e,i)$.

   Then we have Δ-definition of QR rules in Table 5.22.

   Table 5.22: $QRRule(c,e,i)$

$$QRRule(c,e,i) :\equiv QRRule2(c,e,i) \lor QRRule2(c,e,i)$$

2. For (PC):

   Denote our deduction
$$D = \{\alpha_1, \alpha_2, \ldots, \alpha_i\}.$$
   α_i is a propositional consequence if and only if the formula
$$\beta :\equiv (\alpha_1 \land \alpha_2 \land \cdot \land \alpha_{i-1}) \to \alpha_i$$

is a tautology.

To decide if β is a tautology or not, we need to

- transform β into the propositional formula β_P:

$$\beta_P :\equiv \big[(\alpha_1)_P \wedge (\alpha_2)_P \wedge \cdot \wedge (\alpha_{i-1})_P\big] \to (\alpha_i)_P.$$

- show any truth assignment that makes $(\alpha_1)_P$ to $(\alpha_{i-1})_P$ true must also make $(\alpha_i)_P$ true.
- From the Procedure 2.4.1:
  (a) We need to find the **prime components** of β that are subformulas either universal (with universal quantifier $\forall$) or atomic.
  (b) We assign the truth values to these *prime components* to find out if β is tautology or not.

Definition 5.7.1. We say γ is a **prime component** of β if either:

(a) β is atomic and $\gamma = \beta$,

(b) β is universal and $\gamma = \beta$,

(c) β is $\neg\alpha$ and γ is a *prime component* of α,

(d) β is $\alpha_1 \vee \alpha_2$, and γ is a *prime component* of either α_1 or α_2.

We presume that the set

$$\textsc{PrimeComponent} := \big\{(u, f) | u = \ulcorner\gamma\urcorner \text{ and } f = \ulcorner\beta\urcorner,\ \gamma \text{ is a prime component of } \beta\ \big\}$$

is representable and has Δ-definition $\boxed{PrimeComponent(u, f)}$.

Then, we can create a sequence of all prime components of α_1 to α_i, denoted by PrimeList. It has a Δ-definition

$$\boxed{\text{PrimeList(c, i, r)}}$$

where r codes the PrimeList for the first i entries in deduction coded by c.

To code up an assignment of truth values to all the *prime components* of α_1 to α_i, we use the Δ-definition

$$\boxed{\text{TruthAssignment(c, i, r, v)}}$$

where v is the code number of a sequence with the same length of r from $PrimeList(c, i, r)$ and v assigns all the prime components either 0 (False) or 1 (True).

Given a truth assignment for the prime components, we are able to evaluate the truth value of a formula.

Example 5.7.2. Suppose $\alpha :\equiv ((\forall x)(0 < x \vee x < y)) \vee (\neg(0 < x))$. Its formula construction sequence is

$$\Big(\quad 0 < x,\ x < y,\ (0 < x \vee x < y),\ \neg(0 < x),$$
$$(\forall x)(0 < x \vee x < y),\ \underbrace{((\forall x)(0 < x \vee x < y)) \vee (\neg(0 < x))}_{\alpha} \Big)$$

Assume the PrimeList is

$$((\forall x)(0 < x \vee x < y), 0 < x)$$

with the truth assignment

$$(0, 1).$$

We go over the formula construction sequence from the beginning. If the entry is a prime component, then assign the corresponding truth value from truth assignment. If the entry is *not* a prime component, then

(a) if the entry is universal, then assign truth value 2 (for *undefined*),

(b) if the entry is an atomic formula that ends up inside the scopr of a quantifier in α, then assign truth value 2,

(c) if the entry is the denial ($\neg$) or disjunction ($\vee$), then use logical operations to find out its truth value, using 2 if any of the parts have truth value 2.

Then, the sequence of truth values is

$$(1, 2, 2, 0, 0, 0).$$

We can write a Δ-definition $\boxed{Evaluate(e, r, v, y)}$ where

- e is the Gödel number for a formula α,
- r is from $PrimeList(c, i, r)$,
- v is from $TruthAssignment(c, i, r, v)$,
- y is the truth value for α given the truth assignment v.

Now we can write the PC rule. What we need to check is whether any truth assignment that makes α_1 to α_{i-1} true also makes α_i true.

Table 5.23: $PCRule(c, e, i)$

$$\begin{gathered}
PCRule(c,e,i) :\equiv IthSequenceElement(e,i,c)\wedge \\
(\exists r < \text{some bound})(\forall v < \text{some bound}) \\
\Big(\big[PrimeList(c,i,r) \wedge TruthAssignment(ci,r,v)\big] \rightarrow \\
\big[((\forall j < i)(\exists e_j < c)(IthSequenceElement(e_j, j, c) \wedge \\
Evaluate(e_j, r, v, \overline{1})) \rightarrow Evaluate(e, r, v, \overline{1}))\big]\Big)
\end{gathered}$$

Deduction$_N$. As a summary, we can code up the set

$$\text{DEDUCTION}_N := \Big\{ \begin{array}{l} (c,a) : c = \langle \ulcorner\alpha_1\urcorner, \ldots, \ulcorner\alpha_i\urcorner\rangle \text{ and } a = \ulcorner\alpha_i\urcorner \\ \text{where } (\alpha_1, \ldots, \alpha_{i-1}) \text{ is a deduction from } N \text{ of } \alpha_i \Big\} \end{array}$$

by a Δ-definition in Table 5.24.

Σ-definable Thm$_N$. We want to study the set

$$\text{THM}_N := \{\ulcorner\psi\urcorner : N \vdash \psi\} \tag{5.1}$$

defined by the Σ-formula in Table 5.25.
This means that for $\forall a \in \mathbb{N}$,

- $\mathfrak{N} \models Thm(\overline{a})$ if $a = \ulcorner\psi\urcorner$ for some formula ψ such that $N \vdash \psi$.
- $\mathfrak{N} \models \neg Thm(\overline{a})$ if otherwise.

  However, we *cannot* conclude $N \vdash \neg Thm(\overline{a})$ since $\neg Thm(\overline{a})$ is a Π-sentence.

Table 5.24: $Deduction(c, f)$

$$\begin{gathered} Deduction(c,f) :\equiv SequenceCode(c) \wedge Formula(f) \wedge \\ (\exists \ell < c)\Big(Sequencelength(c,\ell) \wedge IthSequenceElement(f,\ell,c) \wedge \\ (\forall i \leq \ell)(\forall e < c)\Big[IthSequenceElement(e,i,c) \rightarrow \\ \Big(LogicalAxiom(e) \vee Axiom_N(e) \vee QRRule(c,e,i) \vee \\ PCRule(c,e,i)\Big)\Big]\Big) \end{gathered}$$

Table 5.25: $Thm_N(f)$

$$Thm_N(f) :\equiv (\exists c)(Deduction(c,f))$$

- There is no way to rewrite $Thm_N(f)$ as a Δ-sentence since we *cannot* figure out the boundary of c.

Chapter 6

The Incompleteness Theorems

6.1 Representable Set is Σ-definable

Previously, we showed that **every Δ-definable set is representable** in Corollary 5.3.1.

Now we show that **every representable set is Σ-definable**.

Proposition 6.1.1. If $A \subseteq \mathbb{N}^k$ is representable, then A is Σ-definable.

Proof. For simplicity, let $A \subseteq \mathbb{N}$. Assume that $\psi(v_1)$ represents A. By Definition 5.3.2, this means

$$\forall a \in A \quad N \vdash \psi(\overline{a}).$$

Let $\beta(x)$ be the following Σ-formula:

$$\beta(x) :\equiv \exists x \exists y \big[Num(x, y) \land Sub(\overline{\ulcorner \psi \urcorner}, \overline{\ulcorner v_1 \urcorner}, y, z) \land Thm_N(z) \big].$$

We claim: $\beta(x)$ defines A. That is, for every $n \in \mathbb{N}$, we have

$$n \in A \iff \mathfrak{N} \models \beta(\overline{n}).$$

- First, assume $n \in A$. Since $\psi(v_1)$ represents A, we have $N \vdash \psi(\overline{n})$. Therefore, by Definition in Equation (5.1), we have $\ulcorner \psi(\overline{n}) \urcorner \in \textsc{Thm}_N$.

  Since the Σ-formula $Thm_N(z)$ defines $\textsc{Thm}_N$, by Definition 5.3.1, we have
  $$\mathfrak{N} \models Thm_N(\overline{\ulcorner \psi(\overline{n}) \urcorner}).$$

Then, under a variable assignment function $s : Vars \to \mathbb{N}$ with

$$\begin{aligned} s(x) &= n \\ s(y) &= Num(n) = \ulcorner \overline{n} \urcorner \\ s(z) &= \ulcorner Sub(\ulcorner \psi \urcorner, \ulcorner v_1 \urcorner, \ulcorner \overline{n} \urcorner) \urcorner = \ulcorner \psi(\overline{n}) \urcorner \end{aligned}$$

we have

$$\begin{aligned} Num(x, y)[s] &= Num(s(x), s(y)) = Num(n, Num(n)) \\ &\quad\quad \text{which is True in } \mathfrak{N} \\ Sub(\overline{\ulcorner \psi \urcorner}, \overline{\ulcorner v_1 \urcorner}, y, z)[s] &= Sub(\overline{\ulcorner \psi \urcorner}, \overline{\ulcorner v_1 \urcorner}, s(y), s(z)) \\ &= Sub(\overline{\ulcorner \psi \urcorner}, \overline{\ulcorner v_1 \urcorner}, \ulcorner \overline{n} \urcorner, \ulcorner Sub(\ulcorner \psi \urcorner, \ulcorner v_1 \urcorner, \ulcorner \overline{n} \urcorner) \urcorner) \\ &\quad\quad \text{which is True in } \mathfrak{N} \end{aligned}$$

Therefore, we have

$$\mathfrak{N} \models \beta(s)[s] \equiv \beta(\overline{n}).$$

- Second, assume $\mathfrak{N} \models \beta(\overline{n})$. Then there exists a variable assignment function $s : Vars \to \mathbb{N}$ with $s(x) = n$ such that

$$\mathfrak{N} \models \beta(\overline{n})[s] \iff \mathfrak{N} \models \big[Num(x, y) \wedge Sub(\overline{\ulcorner \psi \urcorner}, \overline{\ulcorner v_1 \urcorner}, y, z) \wedge Thm_N(z)\big][s].$$

  - Since $\mathfrak{N} \models Num(x, y)[s]$, we have $s(y) = Num(s(x)) = Num(n) = \ulcorner \overline{n} \urcorner$.
  - Since $\mathfrak{N} \models Sub(\overline{\ulcorner \psi \urcorner}, \overline{\ulcorner v_1 \urcorner}, y, z)[s]$, we have

$$s(z) = \ulcorner Sub(\overline{\ulcorner \psi \urcorner}, \overline{\ulcorner v_1 \urcorner}, s(y)) \urcorner = \ulcorner \psi(\overline{n}) \urcorner.$$

  - Since $\mathfrak{N} \models Thm_N(z)[s]$, we have $\mathfrak{N} \models Thm_N(\overline{s(z)})$. By $s(z) = \ulcorner \psi(\overline{n}) \urcorner$, we have further $\mathfrak{N} \models Thm_N(\overline{\ulcorner \psi(\overline{n}) \urcorner})$.
  - Since $Thm_N(z)$ defines $\textsc{Thm}_N$, we have $\ulcorner \psi(\overline{n}) \urcorner \in \textsc{Thm}_N$. By Definition at Eq (5.1), we have $N \vdash \ulcorner \psi(\overline{n}) \urcorner$.
  - Since ψ represents A by assumption, from $N \vdash \ulcorner \psi(\overline{n}) \urcorner$, we derive $n \in A$.

$\square$

Alternative Proof of Proposition 6.1.1. For simplicity, let $A \subseteq \mathbb{N}$. Assume that $\psi(v_1)$ represents A. By Definition 5.3.2, this means

$$\forall a \in A \quad N \vdash \psi(\overline{a}).$$

Let $\beta(x)$ be the following Σ-formula:

$$\beta(x) :\equiv \exists d\big[Deduction_N\big(d, Sub(\overline{\ulcorner\psi\urcorner}, \overline{\ulcorner v_1 \urcorner}, Num(x))\big)\big].$$

We claim: $\beta(x)$ defines A. That is, for every $n \in \mathbb{N}$, we have

$$n \in A \iff \mathfrak{N} \models \beta(\overline{n}).$$

- First, let $n \in A$. Then since ψ represents A, we have $N \vdash \psi(\overline{n})$. So there exists a deduction $(\sigma_1, \ldots, \sigma_k)$ of $\psi(\overline{n})$ from N.

  Denote $d := \langle \ulcorner\sigma_1\urcorner, \ldots, \ulcorner\sigma_k\urcorner\rangle$. Then

  $$\begin{aligned}\mathfrak{N} \quad \models \quad & \text{"}Deduction_N(d, \ulcorner\psi^{v_1}_{\overline{n}}\urcorner)\text{"} \\ & \equiv Deduction_N\big(d, Sub(\overline{\ulcorner\psi\urcorner}, \overline{\ulcorner v_1\urcorner}, Num(n))\big) \\ & \equiv \exists y \exists z\big[Num(\overline{n}, y) \wedge Sub(\overline{\ulcorner\psi\urcorner}, \overline{\ulcorner v_1\urcorner}, y, z) \wedge Deduction_N(d, z)\big].\end{aligned}$$

  Therefore, we have $\mathfrak{N} \models \beta(\overline{n})$.

- Second, assume $\mathfrak{N} \models \beta(\overline{n})$, which means

  $$\mathfrak{N} \models \text{"}Deduction_N(d, \ulcorner\psi(v_1)\urcorner)\text{"}.$$

  Thus, these exists a deduction of $\psi(v_1)$ from N, take $v_1 = \overline{n}$, we have $N \vdash \psi(\overline{n})$. Then, since ψ represents A, we have $n \in A$.

□

6.1.1 Gödel's Self-Reference Lemma

Lemma 6.1.1 (Gödel's Self-Reference Lemma)**.** Let $\beta(x)$ be an $\mathcal{L}_{NT}$-formula with only one free variable x. There there exists a sentence θ such that

$$N \vdash \theta \leftrightarrow \beta\big(\overline{\ulcorner\theta\urcorner}\big).$$

Proof. To prove $N \vdash \theta \leftrightarrow \beta\big(\overline{\ulcorner\theta\urcorner}\big)$, we explicitly construct θ by defining $\gamma(v_1)$ by Equation (6.1).

$$\gamma(v_1) :\equiv \forall y \forall z \Big[\big(Numf(v_1, y) \wedge Subf(v_1, \overline{8}, y, z)\big) \rightarrow \beta(z)\Big] \tag{6.1}$$

- Here we have the formula $Numf$ that represents the function Num, and the formula $Subf$ that represents the function Sub such that

  $$\begin{aligned} Numf(x, y) \quad &:\equiv \quad Num(x, y) \wedge (\forall i < y)\big[\neg Num(x, i)\big] \\ Subf(x_1, x_2, x_3, y) \quad &:\equiv \quad Sub(x_1, x_2, x_3, y) \wedge (\forall i < y)\big[\neg Sub(x_1, x_2, x_3, i)\big] \end{aligned}$$

  with

  $$\begin{aligned} N \quad &\vdash \quad \Big[Numf(\overline{a}, y) \leftrightarrow y = \overline{\mathrm{Num}(a)}\Big] \\ N \quad &\vdash \quad \Big[Subf(\overline{a}, \overline{b}, \overline{c}, z) \leftrightarrow \overline{\mathrm{Sub}(a, b, c)}\Big] \end{aligned}$$

- Also, notice that Equation (6.1) uses $8 = \ulcorner v_1 \urcorner$.

Define $\theta = \gamma(\overline{m})$ where

$$m = \ulcorner\gamma(v_1)\urcorner$$

and

$$\overline{m} = \overline{\ulcorner\gamma(v_1)\urcorner}$$

With a small calculation:

$$\begin{aligned} & Sub(m, 8, \ulcorner\overline{m}\urcorner) \\ = \quad & Sub(\ulcorner\gamma(v_1)\urcorner, \ulcorner v_1 \urcorner, \ulcorner\overline{m}\urcorner) \\ = \quad & \ulcorner\gamma(v_1)^{v_1}_{\overline{m}}\urcorner \\ = \quad & \ulcorner\gamma(\overline{m})\urcorner \\ = \quad & \ulcorner\theta\urcorner \quad \text{by } \theta = \gamma(\overline{m}) \end{aligned}$$

Now we derive

$$\begin{aligned}
& \theta = \gamma(\overline{m}) \\
\iff \quad & \forall y \forall z \Big[\big(Numf(\overline{m}, y) \wedge Subf(\overline{m}, \overline{8}, y, z) \big) \to \beta(z) \Big] \\
\iff \quad & \forall y \forall z \Big[y = \overline{\mathrm{Num}(m)} \to \big(Subf(\overline{m}, \overline{8}, y, z) \to \beta(z) \big) \Big] \\
\iff \quad & \forall y \forall z \Big[y = \overline{\ulcorner \overline{m} \urcorner} \to \big(Subf(\overline{m}, \overline{8}, y, z) \to \beta(z) \big) \Big] \\
& \quad \text{by notice } y = \mathrm{Num}(n) = \ulcorner \overline{n} \urcorner \\
\iff \quad & \forall z \Big[Subf(\overline{m}, \overline{8}, \overline{\ulcorner \overline{m} \urcorner}, z) \to \beta(z) \Big] \\
\iff \quad & \forall z \Big[z = \overline{Sub(m, 8, \ulcorner \overline{m} \urcorner)} \to \beta(z) \Big] \\
\iff \quad & \Big[z = \ulcorner \theta \urcorner \to \beta(z) \Big] \quad \text{by small calculation} \\
\iff \quad & \beta(\ulcorner \theta \urcorner)
\end{aligned}$$

Therefore, we prove $N \vdash \big[\theta \leftrightarrow \beta(\ulcorner \theta \urcorner)\big]$

□

Theorem 6.1.1 (Tarski's Undefinability Theorem, 1936)**.** The set $\{\ulcorner \psi \urcorner : \mathfrak{N} \models \psi\}$ of Gödel numbers of formulas true in $\mathfrak{N}$ is not definable.

Proof. Prove by contradiction. Assume there is a formula $\alpha(x)$ that defines the set $\{\ulcorner \psi \urcorner : \mathfrak{N} \models \psi\}$. This means, for every formula ψ,

$$\mathfrak{N} \models \alpha(\overline{\ulcorner \psi \urcorner}) \iff \ulcorner \psi \urcorner \in \{\ulcorner \psi \urcorner : \mathfrak{N} \models \psi\} \iff \mathfrak{N} \models \psi.$$

Apply the Self-Reference Lemma 6.1.1 to the formula $\beta(x) :\equiv \neg\alpha(x)$, there exists a sentence θ such that

$$N \vdash \theta \leftrightarrow \beta(\overline{\ulcorner \theta \urcorner}), \text{ and therefore } \mathfrak{N} \models \theta \leftrightarrow \beta(\overline{\ulcorner \theta \urcorner}).$$

Here θ means "I am true in $\mathfrak{N}$ iff I am false in $\mathfrak{N}$".
Then, we have:

$$\mathfrak{N} \models \theta \iff \mathfrak{N} \models \beta(\overline{\ulcorner \theta \urcorner}) \iff \mathfrak{N} \not\models \alpha(\overline{\ulcorner \theta \urcorner}) \iff \mathfrak{N} \not\models \theta.$$

We observe contradiction.

□

Theorem 6.1.1 means: **truth is undefinable!**

The set $\{\ulcorner\psi\urcorner : \mathfrak{N} \models \psi\}$ is not definable, by Proposition 6.1.1, it is not representable. The following Corollary follows.

Corollary 6.1.1. The set $\{\ulcorner\psi\urcorner : \mathfrak{N} \models \psi\}$ is not representable. By Church's Thesis in Theorem 5.4.1, there is no computer program which, given an $\mathcal{L}_{NT}$-formula ψ as input, outputs "true" if $\mathfrak{N} \models \psi$ and outputs "false" if $\mathfrak{N} \not\models \psi$.

6.2 Gödel's First Incompleteness Theorem

Definition 6.2.1. A **theory** is a collection of formulas T that is closed under deduction: For every formula ψ, if $T \vdash \psi$, then $\psi \in T$.

- A theory T is **consistent** if $T \nvdash \perp$ (T has a model).
- A theory T is **complete** if, for every *sentence* σ, either $\sigma \in T$ or $\neg\sigma \in T$.
- If $\mathfrak{A}$ is a structure/model, the **theory of** $\mathfrak{A}$, denoted by $Th(\mathfrak{A})$ is the set of formulas that are true in $\mathfrak{A}$:

$$Th(\mathfrak{A}) = \{\psi : \mathfrak{A} \models \psi\}$$

By Completeness Theorem 3.2.1 and Soundness Theorem 2.5.3, the theory of $\mathfrak{A}$ is equivalent to

$$Th(\mathfrak{A}) = \{\psi : \mathfrak{A} \models \psi\} = \{\psi : \mathfrak{A} \vdash \psi\}.$$

Therefore,

- The theory $Th(\mathfrak{A})$ is complete and consistent.
- Recall we constructed a model for the consistent set of sentences Σ when we proved the Completeness Theorem 3.2.1, this implies that: every complete and consistent theory equals $Th(\mathfrak{A})$ for some structure $\mathfrak{A}$.

Example 6.2.1. Some examples of theory of models:

- $Th(\mathfrak{N}) = Th(\mathbb{N}, 0, 1, S, +, \cdot, E, <)$, the Theory of Arithmetic, sometimes called True Arithmetic.

- $Th(\mathbb{N}, 0, 1, +)$, Presburger Arithmetic
- $Th(\mathbb{R}, 0, 1, +, \cdot, <)$, the theory of real closed fields
- $Th(\mathbb{R}^2, \text{Between}, \text{Congruent})$, Euclidean Geometry where

$$\begin{aligned} \text{Between} &:= \{(a,b,c) \in (\mathbb{R}^2)^3 : b \in ac\} \\ \text{Congruent} &:= \{(a,b,c,d) \in (\mathbb{R}^2)^4 : |ab| = |cd|\}. \end{aligned}$$

here ab denotes the line segment between points $a, b \in \mathbb{R}^2$, and $|ab|$ is the length of ab.

Definition 6.2.2. If A is a set of formulas, then the theory of A, denoted by $Th(A)$, is defined as

$$Th(A) = \{\text{formula } \psi : \Sigma \vdash \psi\}$$

Notice that $Th(A)$ is the smallest theory that includes A.

Definition 6.2.3. A theory T is **axiomatized** by a set of formulas A if $T = Th(A)$.

Example 6.2.2. Suppose

$$A := \{x = y \lor x < y \lor y < x, (x < y \land y < z) \to x < z, \neg(x < x)\}.$$

Then A axiomatizes the theory

$$T = \{\mathcal{L}_{\{<\}} - \text{formulas that are true in all linear orders}\}$$

Example 6.2.3. Robinson Arithmetic $N = \{N_1, \ldots, N_{11}\}$ does not axiomatize $Th(\mathfrak{N})$:

N is finite! Axioms of N_1, …, N_{11} are true in $\mathfrak{N}$. However, N does not axiomatize $Th(\mathfrak{N})$, since for example N does not deduce commutative law:

$$\forall x \forall y(x + y = y + x) \in Th(\mathfrak{N}), \text{ and } \forall x \forall y(x + y = y + x) \notin Th(N).$$

We want to extend N by including more axioms, which allow us to prove more theorems in $Th(\mathfrak{N})$. We would like to find a set of axioms A that axiomatizes $Th(\mathfrak{N})$, i.e., for any formula ψ, $\mathfrak{N} \models \psi \iff A \vdash \psi$.

In other words, we want to have some "useful" axiomatization of $Th(\mathfrak{N})$: by "useful", we mean Σ should be decidable by an algorithm in the sense of "recursive".

Definition 6.2.4. Suppose A be a set of axioms of $\mathcal{L}_{NT}$. Then A is **recursive** if the set $\{\ulcorner\sigma\urcorner : \sigma \in A\}$ is representable.

By Church's Thesis 5.4.3: The set $B \subseteq \mathbb{N}$ is representable if and only if there exists a computer program that, given $n \in \mathbb{N}$ as input, after finite steps, outputs "yes" if $n \in B$ and "no" if $n \notin B$.

Therefore, a set of axiom A is recursive if and only if membership in A can be decided algorithmically in finite time.

Lemma 6.2.1. Let A be a recursive set of axioms, then the set $\textsc{Thm}_\Sigma :=$ $\{\ulcorner\psi\urcorner : A \vdash \psi\}$ is definable by a Σ-formula $Thm_A(x)$.

Proof. Since A is recursive, by definition, we have the set

$$\textsc{Axiom}_A := \{\ulcorner\alpha\urcorner : \alpha \in A\}$$

is representable and therefore, by Proposition 6.1.1, definable by a Σ-formula $Axiom_A(x)$.

Recall that $Deduction(y, z)$ is a Δ-formula, which has $Axiom_N(x)$ as a Δ-subformula. Replacing $Axiom_N(x)$ by $Axiom_A(x)$, we have a Σ-formula $Deduction_A(y, z)$ which defines the set

$$\begin{aligned}\textsc{Deduction}_A := \{(c, a) \quad : \quad & c = \langle\ulcorner\sigma_1\urcorner, \ldots, \ulcorner\sigma_n\urcorner\rangle \text{ and } a = \ulcorner\psi\urcorner \\ & \text{where } (\sigma_1, \ldots, \sigma_n) \text{ is a deduction in } A \text{ of } \psi\}\end{aligned}$$

It follows that the set $\textsc{Thm}_A$ is defined by a Σ-formula

$$Thm_A(x) :\equiv (\exists y) Deduction_A(y, x).$$

□

Theorem 6.2.1 (Gödel's First Incompleteness Theorem, 1931)**.** Suppose that A is a consistent and recursive set of axioms in the language $\mathcal{L}_{NT}$. Then there is a sentence θ such that $\mathfrak{N} \models \theta$ but $A \nvdash \theta$.

Proof. If $A \nvdash N_i$ for some axiom N_i of Robinson Arithmetic, then we finish the proof by using $\theta :\equiv N_i$. So we assume $A \vdash N$.

Applying the Self-Reference Lemma 6.1.1 to the formula

$$\beta(x) :\equiv \neg Thm_A(x),$$

we obtain the sentence θ such that

$$N \vdash \theta \leftrightarrow \neg Thm_A(\overline{\ulcorner\theta\urcorner}) \tag{*}$$

Here θ means "I am true if and only if I am not provable in A".

Since $\mathfrak{N} \models N$, Eq (*) implies that

$$\mathfrak{N} \models \theta \leftrightarrow \neg Thm_A(\overline{\ulcorner\theta\urcorner}).$$

Thus, we have

$$\begin{aligned}\mathfrak{N} \models \theta \quad &\iff \quad \mathfrak{N} \models \neg Thm_A(\overline{\ulcorner\theta\urcorner}) \\ &\iff \quad \ulcorner\theta\urcorner \notin \textsc{Thm}_A \quad \text{by } Thm_A \text{ defines } \textsc{Thm}_A \\ &\iff \quad A \nvdash \theta \quad \text{by Definition of } \textsc{Thm}_A\end{aligned}$$

In short, we just derived

$$\mathfrak{N} \models \theta \iff A \nvdash \theta \qquad (**)$$

So, θ is either true in $\mathfrak{N}$ but not provable from A, or θ is false in $\mathfrak{N}$ and provable from A.

Assume θ is false in $\mathfrak{N}$ and provable from A: $\mathfrak{N} \not\models \theta$, and (**) implies $A \vdash \theta$.

(**) also implies that $\mathfrak{N} \models Thm_A(\overline{\ulcorner\theta\urcorner})$. $Thm_A(\overline{\ulcorner\theta\urcorner})$ is a Σ-sentence which is true in $\mathfrak{N}$. By Proposition 5.2.1, we have $N \vdash Thm_A(\overline{\ulcorner\theta\urcorner})$. By (*) and the (PC) rule in Definition 2.4.3, we have $N \vdash \neg\theta$. Since $A \vdash N$, we have $A \vdash \neg\theta$.

We have observed $A \vdash \theta$ and $A \vdash \neg\theta$. This means $A \vdash \perp$, which is contradicting to our assumption that A is consistent.

□

Some remarks on the Gödel First Incompleteness Theorem 6.2.1:

- The condition of A being "recursive" can be relaxed to "A being Σ-definable".
- One application from the Gödel First Incompleteness Theorem 6.2.1: Any consistent theory extending N is undecidable, where "undecidable" means "not recursive".

  Theorem 6.2.2. If A is a consistent set of axioms extending N and in the language $\mathcal{L}_{NT}$. Then the set $\textsc{Thm}_A$ is not representable in A.

  Proof. Prove by contradiction. Assume $\gamma(x)$ represents $\textsc{Thm}_A$. We obtain the sentence θ such that

$$N \vdash \left[\theta \leftrightarrow \gamma(\overline{\ulcorner\theta\urcorner})\right].$$

As A extends N, we have

$$A \vdash \left[\theta \leftrightarrow \gamma(\overline{\ulcorner\theta\urcorner})\right].$$

Thus,

$$\begin{aligned} & A \vdash \theta \\ \Rightarrow\ & \ulcorner\theta\urcorner \in \textsc{Thm}_A \\ \Rightarrow\ & A \vdash \gamma(\overline{\ulcorner\theta\urcorner}) \quad \text{Since } \gamma \text{ represents } \textsc{Thm}_A \\ \Rightarrow\ & A \vdash \neg\theta \\ \Rightarrow\ & A \nvdash \theta \quad A \text{ is consistent} \\ \Rightarrow\ & \ulcorner\theta\urcorner \notin \textsc{Thm}_A \\ \Rightarrow\ & A \vdash \neg\gamma(\overline{\ulcorner\theta\urcorner}) \quad \text{Since } \gamma \text{ represents } \textsc{Thm}_A \\ \Rightarrow\ & A \vdash \theta \end{aligned}$$

We observe contradiction. □

Theorem 6.2.3 (Rosser's Theorem)**.** If A is a set of $\mathcal{L}_{NT}$-axioms that is recursive, consistent, and extends N, then A is incomplete.

Proof. Use the Self-Reference Lemma to construct a sentence θ which expresses: "I am true if and only if for every deduction of $A \vdash \theta$, there is a shorter deduction of $A \vdash \neg\theta$."

Notice that if $a := \ulcorner\theta\urcorner$, then $\ulcorner\neg\theta\urcorner = \langle 1, \ulcorner\theta\urcorner\rangle = 2^2 \cdot 3^{\ulcorner\theta\urcorner+1} = 4 \cdot 3^{a+1}$.

Let $\beta(x) :\equiv (\forall y)\big[\mathit{Deduction}_A(y, x) \rightarrow (\exists z < y)\mathit{Deduction}_A(z, \overline{4} \cdot \overline{3}^{Sx})\big]$.

By the Self-Reference Lemma,we get a sentence θ such that

$$\begin{aligned} N \ \vdash\ & \theta \leftrightarrow \beta(\overline{\ulcorner\theta\urcorner}) \\ \equiv\ & \theta \leftrightarrow (\forall y)\big[\mathit{Deduction}_A(y, \overline{\ulcorner\theta\urcorner}) \rightarrow (\exists z < y)\mathit{Deduction}_A(z, \overline{4} \cdot \overline{3}^{\overline{\ulcorner\theta\urcorner+1}})\big] \\ \equiv\ & \theta \leftrightarrow (\forall y)\big[\mathit{Deduction}_A(y, \overline{\ulcorner\theta\urcorner}) \rightarrow (\exists z < y)\mathit{Deduction}_A(z, \overline{\ulcorner\neg\theta\urcorner})\big] \end{aligned}$$

- First, we claim that $A \nvdash \theta$: By contradiction, assume $A \vdash \theta$. Let a be a number that codes up a deduction of θ, then we have

$$A \vdash \mathit{Deduction}_A(\overline{a}, \overline{\ulcorner\theta\urcorner}).$$

A extends N, which means $A \vdash N$, and thus $A \vdash \theta$ leads to

$$A \vdash (\forall y)\big[\mathit{Deduction}_A(y, \overline{\ulcorner\theta\urcorner}) \rightarrow (\exists z < y)\mathit{Deduction}_A(z, \overline{\ulcorner\neg\theta\urcorner})\big]$$

and

$$A \vdash \big[Deduction_A(\overline{a}, \overline{\ulcorner\theta\urcorner}) \to (\exists z < \overline{a}) Deduction_A(z, \overline{\ulcorner\neg\theta\urcorner})\big].$$

We already know $A \vdash Deduction_A(\overline{a}, \overline{\ulcorner\theta\urcorner})$, which leads to

$$A \vdash (\exists z < \overline{a}) Deduction_A(z, \overline{\ulcorner\neg\theta\urcorner}). \tag{*}$$

On the other hand, we have assumed that $A \vdash \theta$ and A is consistent. Therefore, we have for each $n \in \mathbb{N}$ that

$$A \vdash \neg Deduction_A(\overline{n}, \overline{\ulcorner\neg\theta\urcorner}).$$

Then, by Rosser's Lemma 5.2.2, we have

$$A \vdash \neg(\exists z < \overline{a}) Deduction_A(z, \overline{\ulcorner\neg\theta\urcorner}).$$

Combining with (*), this shows A is inconsistent, and we observe a contradiction. Thus, we have proved our claim that $A \nvdash \theta$.

- Also, we claim that $A \nvdash \neg\theta$. By contradiction, assume $A \vdash \neg\theta$. Let b be a code for a deduction of $\neg\theta$. Then we have

$$A \vdash Deduction_A(\overline{b}, \overline{\ulcorner\neg\theta\urcorner}).$$

Notice that

$$\begin{aligned} & (\forall x)(\alpha \to \beta) \\ \iff\ & \neg(\exists x)\big[\neg(\alpha \to \beta)\big] \\ \iff\ & \neg(\exists x)\big[\neg(\neg\alpha \vee \beta)\big] \\ \iff\ & \neg(\exists x)\big[\alpha \wedge \neg\beta)\big] \end{aligned}$$

Since $A \vdash \neg\theta$, this leads to

$$A \vdash (\exists y)\Bigg[Deduction_A(y, \overline{\ulcorner\theta\urcorner}) \wedge \neg(\exists z < y) Deduction_A(z, \overline{\ulcorner\neg\theta\urcorner})\Bigg]$$

which is equivalent to

$$A \vdash (\exists y)\Bigg[Deduction_A(y, \overline{\ulcorner\theta\urcorner}) \wedge \neg(\exists z)\big[(z < y) \wedge Deduction_A(z, \overline{\ulcorner\neg\theta\urcorner})\big]\Bigg].$$

Substitute $\bar{b}$ for z, we have

$$A \vdash (\exists y)\Big[\mathit{Deduction}_A(y, \overline{\ulcorner\theta\urcorner}) \land \neg\big[(\bar{b} < y) \land \mathit{Deduction}_A(\bar{b}, \overline{\ulcorner\neg\theta\urcorner})\big]\Big].$$

Since we already know $A \vdash \mathit{Deduction}_A(\bar{b}, \overline{\ulcorner\neg\theta\urcorner})$, this means we have to have

$$A \vdash (\exists y)\Big[\mathit{Deduction}_A(y, \overline{\ulcorner\theta\urcorner}) \land \neg\big[(\bar{b} < y)\big]\Big],$$

which is equivalent to

$$A \vdash (\exists y)\Big[y \leq \bar{b} \land \mathit{Deduction}_A(y, \overline{\ulcorner\theta\urcorner})\Big]. \qquad (**)$$

On the other hand, we have assumed that $A \vdash \neg\theta$ and A is consistent. Therefore, we have for each $n \in \mathbb{N}$ that

$$A \vdash \neg \mathit{Deduction}(\bar{n}, \overline{\ulcorner\theta\urcorner}).$$

Again, by Rosser's Lemma 5.2.2, we have

$$A \vdash \neg(\exists y)\Big[y \leq \bar{b} \land \mathit{Deduction}_A(y, \overline{\ulcorner\theta\urcorner})\Big].$$

Combining with (**), this shows A is inconsistent, and again, we observe a contradiction. Thus, we have proved the claim that $A \nvdash \neg\theta$.

- Therefore, we have $A \nvdash \theta$ and $A \nvdash \neg\theta$, hence A is incomplete.

$\square$

Rosser's Theorem 6.2.3 implies the First Incompleteness Theorem 6.2.1: Consider either $\mathfrak{N} \models \theta$ or $\mathfrak{N} \models \neg\theta$. We can find a sentence that is true in $\mathfrak{N}$ but not provable from A.

6.3 Gödel's Second Incompleteness Theorem

Definition 6.3.1. The axioms of **Peano Arithmetic** (1889), denoted by PA, consist of the eleven axioms of Robinson Arithmetic N together with axioms

$$\mathit{Induction}_{psi} :\equiv \Big[\psi(0) \land (\forall x)\big[\psi(x) \to \psi(Sx)\big]\Big] \to (\forall x)\psi(x)$$

for each $\mathcal{L}_{NT}$-formula $\psi(x)$ with one free variable.

Some properties of PA:

- Since $\mathfrak{N} \models Induction_\psi$ for each $\psi(x)$, we have $\mathfrak{N} \models PA$, and thus PA is **consistent**.
- PA is **recursive**: there exists a simple algorithm to decide the membership in $\{\ulcorner \alpha \urcorner : \alpha \in PA\}$.
- We can prove the set $\textsc{Axiom}_{PA}$ is representable, so PA is a recursively axiomatized extension of N.
- PA is strong enough to hold the following derivability conditions hold for all formulas ϕ:

  $$\text{If } PA \vdash \phi, \text{ then } PA \vdash Thm_{PA}(\overline{\ulcorner \phi \urcorner}). \tag{D1}$$

  $$PA \vdash \left[Thm_{PA}(\overline{\ulcorner \phi \urcorner}) \to Thm_{PA}(\overline{\ulcorner Thm_{PA}(\overline{\ulcorner \phi \urcorner}) \urcorner}) \right]. \tag{D2}$$

  $$PA \vdash \left[\left[Thm_{PA}(\overline{\ulcorner \phi \urcorner}) \wedge Thm_{PA}(\overline{\ulcorner \phi \to \psi \urcorner}) \right] \to Thm_{PA}(\overline{\ulcorner \psi \urcorner}) \right]. \tag{D3}$$

  (D1) means that if PA proves ϕ, then PA proves "PA proves ϕ".

  (D2) means that PA proves "if PA proves ϕ, then PA proves 'PA proves ϕ'".

  (D3) means that "if PA proves ϕ and PA proves $\phi \to \psi$, then PA proves ψ".
- By the First Incompleteness Theorem, there exists a sentence θ such that $\mathfrak{N} \models \theta$ but $PA \nvdash \theta$. In this sense, PA is not complete.

Definition 6.3.2. Let A be a recursive set of $\mathcal{L}_{NT}$-sentences. The set

$$\textsc{Thm}_A := \{\ulcorner \psi \urcorner : A \vdash \psi\}$$

is Σ-definable by a Σ-formula $Thm_A(x)$.

The sentence Con_A is then

$$Con_A :\equiv \neg Thm_A(\overline{\ulcorner \bot \urcorner}).$$

This sentence says "A is consistent": A is consistent if and only if $\mathfrak{N} \models Con_A$.

Theorem 6.3.1 (Gödel's Second Incompleteness Theorem)**.** If A is a consistent, recursive set of $\mathcal{L}_{NT}$-sentences which extends PA, then $A \nvdash Con_A$.

Proof. Let θ be a sentence such that

$$A \vdash \left[\theta \leftrightarrow \neg Thm_A(\overline{\ulcorner \theta \urcorner})\right] \tag{6.2}$$

By the proof of the First Incompleteness Theorem, we know $A \nvdash \theta$. We will show that

$$A \vdash (\theta \leftrightarrow Con_A).$$

Notice that $A \nvdash Con_A$, since if $A \vdash Con_A$, then we would have $A \vdash \theta$, a contradiction to what we have from the proof of the First Incompleteness Theorem that $A \nvdash \theta$.

- To prove $A \vdash (\theta \to Con_A)$: Since $\bot$ is the denial of a tautology, we have

  $$A \vdash (\bot \to \theta).$$

  Since A is a recursive extension of PA, A satisfies derivability conditions (D1), (D2), (D3) with respect to $Thm_A(x)$.

  So, by (D1), we have

  $$A \vdash Thm_A(\overline{\ulcorner \bot \to \theta \urcorner}).$$

  This implies, by (D3), that

  $$A \vdash Thm_A(\overline{\ulcorner \bot \urcorner}) \to Thm_A(\overline{\ulcorner \theta \urcorner}).$$

  It is equivalent to

  $$A \vdash \neg Thm_A(\overline{\ulcorner \theta \urcorner}) \to \neg Thm_A(\overline{\ulcorner \bot \urcorner}) \tag{6.3}$$

  Combining (6.2) and (6.3), we have

  $$A \vdash \theta \to \neg Thm_A(\overline{\ulcorner \bot \urcorner}),$$

  which is equivalent to

  $$A \vdash \theta \to Con_A.$$

- To prove $A \vdash (Con_A \to \theta)$: From (D2),

$$A \vdash Thm_A(\overline{\ulcorner\theta\urcorner}) \to Thm_A\big(\overline{\ulcorner Thm_A(\overline{\ulcorner\theta\urcorner})\urcorner}\big) \tag{6.4}$$

Since, by construction, we have $A \vdash \left[\theta \leftrightarrow \neg Thm_A(\overline{\ulcorner\theta\urcorner})\right]$, the sentence $A \vdash Thm_A(\overline{\ulcorner\theta\urcorner}) \to \neg\theta$ is a true Σ-sentence, and N suffices to prove true Σ-sentences, (D1) leads to

$$A \vdash Thm_A\left(\overline{\ulcorner Thm_A(\overline{\ulcorner\theta\urcorner}) \to \neg\theta\urcorner}\right) \tag{6.5}$$

Take (6.5) with (D3), we have

$$A \vdash Thm_A\left(\overline{\ulcorner Thm_A(\overline{\ulcorner\theta\urcorner})\urcorner}\right) \to Thm_A(\overline{\ulcorner\neg\theta\urcorner}) \tag{6.6}$$

Combine (6.4) and (6.6), we have

$$A \vdash Thm_A(\overline{\ulcorner\theta\urcorner}) \to Thm_A(\overline{\ulcorner\neg\theta\urcorner}) \tag{6.7}$$

Notice that the sentence $\theta \to \big[(\neg\theta) \to \perp\big]$ is a tautology, we have

$$A \vdash Thm_A\left(\overline{\ulcorner\theta \to \big[(\neg\theta) \to \perp\big]\urcorner}\right) \tag{6.8}$$

Using (D3) twice on (6.8), we have

$$A \vdash Thm_A(\overline{\ulcorner\theta\urcorner}) \to \big[Thm_A(\overline{\ulcorner\neg\theta\urcorner}) \to Thm_A(\overline{\ulcorner\perp\urcorner})\big].$$

Combining with (6.7), we obtain

$$A \vdash Thm_A(\overline{\ulcorner\theta\urcorner}) \to Thm_A(\overline{\ulcorner\perp\urcorner}).$$

This is equivalent to

$$A \vdash \neg Thm_A(\overline{\ulcorner\perp\urcorner}) \to \neg Thm_A(\overline{\ulcorner\theta\urcorner}).$$

This is

$$A \vdash Con_A \to \neg Thm_A(\overline{\ulcorner\theta\urcorner}).$$

Combining this with (6.2) and (PC) rule, we have $A \vdash Con_A \to \theta$.

□

Some insights from the Gödel Second Incompleteness Theorem 6.3.1:

- PA itself is consistent and recursive, therefore, $PA \nvdash Con_{PA}$.
- We can prove $\mathfrak{N}$ is a model of Con_{PA} using the axioms of ZFC (Zermelo-Frankl set theory with choice): $ZFC \vdash Con_{PA}$ by interpreting the sentence Con_{PA} in the language of set theory. However, we have $ZFC \vdash Con_{ZFC}$ as the Gödel Second Incompleteness Theorem 6.3.1 implies.
- the Gödel Second Incompleteness Theorem 6.3.1 answers a question asked by David Hilbert by showing: no "sufficiently powerful formal system" can prove its own consistency.
- Since $PA \nvdash Con_{PA}$, we have $PA \cup \{\neg Con_{PA}\}$ is consistent.
- An application from the Gödel Second Incompleteness Theorem 6.3.1: the sentence "I am provable in PA" is true and therefore provable.

Theorem 6.3.2 (Löb's Theorem)**.** If α is such that

$$PA \vdash Thm_{PA}(\overline{\ulcorner \alpha \urcorner}) \to \alpha.$$

Then $PA \vdash \alpha$.

Proof. The proof relies on the fact that PA is strong enough to prove the equivalence

$$\neg Thm_{PA}(\overline{\ulcorner \alpha \urcorner}) \leftrightarrow Con_{PA \cup \{\neg \alpha\}}.$$

Then, by our presumption, we have

$$PA \vdash \neg \alpha \to \neg Thm_{PA}(\overline{\ulcorner \alpha \urcorner}).$$

This means:

$$PA \cup \{\neg \alpha\} \vdash \neg Thm_{PA}(\overline{\ulcorner \alpha \urcorner}).$$

By our equivalence, we have

$$PA \cup \{\neg \alpha\} \vdash Con_{PA \cup \{\neg \alpha\}}.$$

By the Gödel Second Incompleteness Theorem 6.3.1, we have

$$PA \cup \{\neg\alpha\} \nvdash Con_{PA\cup\{\neg\alpha\}}.$$

So, we know $PA \cup \{\neg\alpha\}$ is inconsistent:

$$PA \cup \{\neg\alpha\} \vdash \perp .$$

This means $PA \vdash \alpha$, as needed.

□