Metamathematics

David Marker

Fall 2015

Part I
Truth and Proof

1 Languages and Structures

In mathematical logic, we use first-order languages to describe mathematical structures. Intuitively, a structure is a set that we wish to study equipped with a collection of distinguished functions, relations, and elements. We then choose a language where we can talk about the distinguished functions, relations, and elements and nothing more. For example, when we study the ordered field of real numbers with the exponential function, we study the structure $(\mathbb{R}, +, \cdot, \exp, <, 0, 1)$, where the underlying set is the set of real numbers, and we distinguish the binary functions addition and multiplication, the unary function $x \mapsto e^x$, the binary order relation, and the real numbers 0 and 1. To describe this structure, we would use a language where we have symbols for $+, \cdot, \exp, <, 0, 1$ and can write statements such as $\forall x \forall y\ \exp(x) \cdot \exp(y) = \exp(x+y)$ and $\forall x\ (x > 0 \rightarrow \exists y\ \exp(y) = x)$. We interpret these statements as the assertions "$e^x e^y = e^{x+y}$ for all x and y" and "for all positive x, there is a y such that $e^y = x$."

For another example, we might consider the structure $(\mathbb{N}, +, 0, 1)$ of the natural numbers with addition and distinguished elements 0 and 1. The natural language for studying this structure is the language where we have a binary function symbol for addition and constant symbols for 0 and 1. We would write sentences such as $\forall x \exists y\ (x = y + y\ \vee\ x = y + y + 1)$, which we interpret as the assertion that "every number is either even or 1 plus an even number."

Definition 1.1 A *language* $\mathcal{L}$ is given by specifying the following data:
i) a set of function symbols $\mathcal{F}$ and positive integers n_f for each $f \in \mathcal{F}$;
ii) a set of relation symbols $\mathcal{R}$ and positive integers n_R for each $R \in \mathcal{R}$;
iii) a set of constant symbols $\mathcal{C}$.

The numbers n_f and n_R tell us that f is a function of n_f variables and R is an n_R-ary relation.

Any or all of the sets $\mathcal{F}$, $\mathcal{R}$, and $\mathcal{C}$ may be empty. Examples of languages include:

i) the language of rings $\mathcal{L}_{\mathrm{r}} = \{+, -, \cdot, 0, 1\}$, where $+, -$ and $\cdot$ are binary function symbols and 0 and 1 are constants;

ii) the language of ordered rings $\mathcal{L}_{\mathrm{or}} = \mathcal{L}_{\mathrm{r}} \cup \{<\}$, where $<$ is a binary relation symbol;

iii) the language of pure sets $\mathcal{L} = \emptyset$;

iv) the language of graphs is $\mathcal{L} = \{R\}$ where R is a binary relation symbol.

Next, we describe the structures where $\mathcal{L}$ is the appropriate language.

Definition 1.2 An $\mathcal{L}$*-structure* $\mathcal{M}$ is given by the following data:

i) a nonempty set M called the *universe*, *domain*, or *underlying set* of $\mathcal{M}$;

ii) a function $f^{\mathcal{M}} : M^{n_f} \to M$ for each $f \in \mathcal{F}$;

iii) a set $R^{\mathcal{M}} \subseteq M^{n_R}$ for each $R \in \mathcal{R}$;

iv) an element $c^{\mathcal{M}} \in M$ for each $c \in \mathcal{C}$.

We refer to $f^{\mathcal{M}}$, $R^{\mathcal{M}}$, and $c^{\mathcal{M}}$ as the *interpretations* of the symbols f, R, and c. We often write the structure as $\mathcal{M} = (M, f^{\mathcal{M}}, R^{\mathcal{M}}, c^{\mathcal{M}} : f \in \mathcal{F}, R \in \mathcal{R},$ and $c \in \mathcal{C})$. We will use the notation $A, B, M, N, \ldots$ to refer to the underlying sets of the structures $\mathcal{A}, \mathcal{B}, \mathcal{M}, \mathcal{N}, \ldots$.

For example, suppose that we are studying groups. We might use the language $\mathcal{L}_{\mathrm{g}} = \{\cdot, e\}$, where $\cdot$ is a binary function symbol and e is a constant symbol. An $\mathcal{L}_{\mathrm{g}}$-structure $\mathcal{G} = (G, \cdot^{\mathcal{G}}, e^{\mathcal{G}})$ will be a set G equipped with a binary relation $\cdot^{\mathcal{G}}$ and a distinguished element $e^{\mathcal{G}}$. For example, $\mathcal{G} = (\mathbb{R}, \cdot, 1)$ is an $\mathcal{L}_{\mathrm{g}}$-structure where we interpret $\cdot$ as multiplication and e as 1; that is, $\cdot^{\mathcal{G}} = \cdot$ and $e^{\mathcal{G}} = 1$. Also, $\mathcal{N} = (\mathbb{N}, +, 0)$ is an $\mathcal{L}_{\mathrm{g}}$-structure where $\cdot^{\mathcal{N}} = +$ and $e^{\mathcal{G}} = 0$. Of course, $\mathcal{N}$ is not a group, but it is an $\mathcal{L}_{\mathrm{g}}$-structure.

Usually, we will choose languages that closely correspond to the structure that we wish to study. For example, if we want to study the real numbers as an ordered field, we would use the language of ordered rings $\mathcal{L}_{\mathrm{or}}$ and give each symbol its natural interpretation.

Formulas and Terms

We use the language $\mathcal{L}$ to create formulas describing properties of $\mathcal{L}$-structures. Formulas will be strings of symbols built using the symbols of $\mathcal{L}$, variable symbols $v_0, v_1, v_2, \ldots$, the equality symbol $=$, the Boolean connectives $\wedge$, $\vee$, and $\neg$, which we read as "and," "or," and "not", the quantifiers $\exists$ and $\forall$, which we read as "there exists" and "for all", and parentheses (,).

Definition 1.3 The set of $\mathcal{L}$*-terms* is the smallest set $\mathcal{T}$ such that

i) $c \in \mathcal{T}$ for each constant symbol $c \in \mathcal{C}$,

ii) each variable symbol $v_i \in \mathcal{T}$ for $i = 1, 2, \ldots$, and

iii) if $t_1, \ldots, t_{n_f} \in \mathcal{T}$ and $f \in \mathcal{F}$, then $f(t_1, \ldots, t_{n_f}) \in \mathcal{T}$.

For example, $\cdot(v_1, -(v_3, 1))$, $\cdot(+(v_1, v_2), +(v_3, 1))$ and $+(1, +(1, +(1, 1)))$ are $\mathcal{L}_{\mathrm{r}}$-terms. For simplicity, we will usually write these terms in the more standard notation $v_1(v_3 - 1)$, $(v_1 + v_2)(v_3 + 1)$, and $1 + (1 + (1 + 1))$ when no confusion arises. In the $\mathcal{L}_r$-structure $(\mathbb{Z}, +, \cdot, 0, 1)$, we think of the term $1 + (1 + (1 + 1))$ as a name for the element 4, while $(v_1 + v_2)(v_3 + 1)$ is a name for the function $(x, y, z) \mapsto (x + y)(z + 1)$. We will see below that we can do something similar for any term in any $\mathcal{L}$-structure.

We are now ready to define $\mathcal{L}$-formulas.

Definition 1.4 We say that ϕ is an *atomic $\mathcal{L}$-formula* if ϕ is either
i) $t_1 = t_2$, where t_1 and t_2 are terms, or
ii) $R(t_1, \ldots, t_{n_R})$, where $R \in \mathcal{R}$ and $t_1, \ldots, t_{n_R}$ are terms.
The set of *$\mathcal{L}$-formulas* is the smallest set $\mathcal{W}$ containing the atomic formulas such that
i) if ϕ is in $\mathcal{W}$, then $\neg\phi$ is in $\mathcal{W}$,
ii) if ϕ and ψ are in $\mathcal{W}$, then $(\phi \wedge \psi)$ and $(\phi \vee \psi)$ are in $\mathcal{W}$, and
iii) if ϕ is in $\mathcal{W}$, then $\exists v_i\ \phi$ and $\forall v_i\ \phi$ are in $\mathcal{W}$.

Here are three examples of $\mathcal{L}_{\rm or}$-formulas.
- $v_1 = 0 \vee v_1 > 0$.
- $\exists v_2\ v_2 \cdot v_2 = v_1$.
- $\forall v_1\ (v_1 = 0 \vee \exists v_2\ v_2 \cdot v_1 = 1)$.

Intuitively, the first formula asserts that $v_1 \geq 0$, the second asserts that v_1 is a square, and the third asserts that every nonzero element has a multiplicative inverse.

We want to define when a formula is true in a structure. The first example above already illustrates one problem we have to consider. Let $\mathbb{R}$ be the real numbers. Is the formula $v_1 \geq 0$ true? Of course the answer is "it depends". If $v_1 = 2$ then it is true, while if $v_1 = -7$, then it is false. Similarly, in the $\mathcal{L}_{\rm or}$-structure $(\mathbb{Z}, +, -, \cdot, <, 0, 1)$, the second formula would be true if $v_1 = 9$ but false if $v_1 = 8$. It should be clear that to decide if a formula is true or false we need to consider how we interpret the variables.

Definition 1.5 Let $V = \{v_0, v_1, \ldots\}$. If $\mathcal{M}$ is an $\mathcal{L}$- structure, an *assignment* is a function $\sigma : V \to M$.

We start by showing how to evaluate terms. Suppose $\mathcal{M}$ is an $\mathcal{L}$- structure and $\sigma : V \to M$ is an assignment. We inductively define $t^{\mathcal{M}}[\sigma] \in M$ as follows:
i) if $t = c \in \mathcal{C}$ is a constant, then $t^{\mathcal{M}}[\sigma] = c^{\mathcal{M}}$;
ii) if $t = v_i$ is a variable, then $t^{\mathcal{M}}[\sigma] = \sigma(v_i)$;
iii) if $t_1, \ldots, t_m$ are terms, f is an m-ary function symbol and $t = f(t_1, \ldots, t_m)$, then

$$t^{\mathcal{M}}[\sigma] = f^{\mathcal{M}}(t_1^{\mathcal{M}}[\sigma], \ldots, t_m^{\mathcal{M}}[\sigma]).$$

For example, let $\mathcal{L} = \{f, g, c\}$, where f is a unary function symbol, g is a binary function symbol, and c is a constant symbol. We will consider the $\mathcal{L}$-terms $t_1 = g(v_1, c)$, $t_2 = f(g(c, f(v_1)))$, and $t_3 = g(f(g(v_1, v_2)), g(v_1, f(v_2)))$. Let $\mathcal{M}$ be the $\mathcal{L}$-structure $(\mathbb{R}, \exp, +, 1)$; that is, $f^{\mathcal{M}} = \exp$, $g^{\mathcal{M}} = +$, and $c^{\mathcal{M}} = 1$.

Then

$$t_1^{\mathcal{M}}[\sigma] = \sigma(v_1) + 1,$$

$$t_2^{\mathcal{M}}[\sigma] = e^{1+e^{\sigma(v_1)}}, \text{ and}$$

$$t_3^{\mathcal{M}}[\sigma] = e^{\sigma(v_1)+\sigma(v_2)} + (\sigma(v_1) + e^{\sigma(v_2)}).$$

If $\sigma : V \to M$ is an assignment, $v \in V$ and $a \in M$ we let $\sigma[\frac{a}{v}]$ be the assignment

$$\sigma\left[\frac{a}{v}\right](v_i) = \begin{cases} \sigma(v_i) & \text{if } v_i \neq v \\ a & \text{if } v_i = v \end{cases}.$$

Satisfaction

Before defining truth for formulas, we need to isolate one other important concept.

Definition 1.6 We say that an occurence of a variable v in a formula ϕ is *free* it is not inside a $\exists v$ or $\forall v$ quantifier; otherwise, we say that it is *bound*.

For example in the formula

$$\forall v_2\ (v_0 > 0 \wedge \exists v_1\ v_1 \cdot v_2 = v_0)$$

v_0 occurs freely while v_1 and v_2 are bound. A more complicated example is the formula

$$v_0 > 0 \vee \exists v_0\ v_1 + v_0 = 0.$$

Clearly v_1 occurs freely, but v_0 has both free and bound occurences. The first occurence is free, while the second is bound.

Definition 1.7 Let $\mathcal{M}$ be an $\mathcal{L}$-structure. We inductively define $\mathcal{M} \models_\sigma \phi$ for all $\mathcal{L}$-formulas ϕ and all assignments σ. Intuitively, $\mathcal{M} \models_\sigma \phi$ means "ϕ is true in $\mathcal{M}$ under assignment σ."

i) If ϕ is $t_1 = t_2$, then $\mathcal{M} \models_\sigma \phi$ if $t_1^{\mathcal{M}}[\sigma] = t_2^{\mathcal{M}}[\sigma]$.
ii) If ϕ is $R(t_1, \ldots, t_{n_R})$, then $\mathcal{M} \models_\sigma \phi$ if $(t_1^{\mathcal{M}}[\sigma], \ldots, t_{n_R}^{\mathcal{M}}[\sigma]) \in R^{\mathcal{M}}$.
iii) If ϕ is $\neg\psi$, then $\mathcal{M} \models_\sigma \phi$ if $\mathcal{M} \not\models_\sigma \psi$.
iv) If ϕ is $(\psi \wedge \theta)$, then $\mathcal{M} \models_\sigma \phi$ if $\mathcal{M} \models_\sigma \psi$ and $\mathcal{M} \models_\sigma \theta$.
v) If ϕ is $(\psi \vee \theta)$, then $\mathcal{M} \models_\sigma \phi$ if $\mathcal{M} \models_\sigma \psi$ or $\mathcal{M} \models_\sigma \theta$.
vi) If ϕ is $\exists v_j \psi$, then $\mathcal{M} \models_\sigma \phi$ if there is $a \in M$ such that $\mathcal{M} \models_{\sigma[\frac{a}{v_j}]} \psi$.
vii) If ϕ is $\forall v_j \psi$, then $\mathcal{M} \models_\sigma \phi$ if $\mathcal{M} \models_{\sigma[\frac{a}{v_j}]} \psi$ for all $a \in M$.

If $\mathcal{M} \models_\sigma \phi$ we say that $\mathcal{M}$ with assignment σ *satisfies* ϕ or ϕ is *true* in $\mathcal{M}$ with assignment σ.

Remarks 1.8 • There are a number of useful abbreviations that we will use: $\phi \to \psi$ is an abbreviation for $\neg\phi \vee \psi$, and $\phi \leftrightarrow \psi$ is an abbreviation for $(\phi \to \psi) \wedge (\psi \to \phi)$. In fact, we did not really need to include the symbols $\vee$ and $\forall$. We could have considered $(\phi \vee \psi)$ as an abbreviation for $\neg(\neg\phi \wedge \neg\psi)$ and $\forall v\phi$ as an abbreviation for $\neg(\exists v \neg\phi)$. Viewing these as abbreviations will be an advantage when we are proving theorems by induction on formulas because it eliminates the $\vee$ and $\forall$ cases.

We also will use the abbreviations $\bigwedge_{i=1}^{n} \psi_i$ and $\bigvee_{i=1}^{n} \psi_i$ for $\psi_1 \wedge \ldots \wedge \psi_n$ and $\psi_1 \vee \ldots \vee \psi_n$, respectively.

• In addition to $v_1, v_2, \ldots,$ we will use $w, x, y, z, ...$ as variable symbols.

• It is important to note that the quantifiers $\exists$ and $\forall$ range only over elements of the model. For example the statement that an ordering is complete (i.e., every bounded subset has a least upper bound) cannot be expressed as a formula because we cannot quantify over subsets. The fact that we are limited to quantification over elements of the structure is what makes it "first-order" logic.

When proving results about satisfaction in models, we often must do an induction on the construction of formulas. As a first example of this method we show that $\mathcal{M} \models_\sigma \phi$ only depends on the restriction of σ to the variables occuring freely in ϕ.

Lemma 1.9 (Coincedence Lemma) *Suppose $\mathcal{M}$ is an $\mathcal{L}$-structure.*

i) Suppose t is an $\mathcal{L}$-term and $\sigma, \tau : V \to M$ are assignments that agree on all variables occuring in t. Then $t^{\mathcal{M}}[\sigma] = t^{\mathcal{M}}[\tau]$.

ii) Suppose ϕ is an $\mathcal{L}$-formula and $\sigma, \tau : V \to M$ are assignments that agree on all variables occuring freely in ϕ. Then $M \models_\sigma \phi$ if and only if $\mathcal{M} \models_\tau \phi$.

Proof i) We prove this by induction on terms.

If $t = c \in \mathcal{C}$ is a constant, then

$$t^{\mathcal{M}}[\sigma] = c^{\mathcal{M}} = t^{\mathcal{M}}[\tau].$$

If $t = v_i$ is a variable, then

$$t^{\mathcal{M}}[\sigma] = \sigma(v_i) = \tau(v_i) = t^{\mathcal{M}}[\tau].$$

Suppose the lemma is true for $t_1, \ldots, t_m$, f is an m-ary function symbol and $t = f(t_1, \ldots, t_m)$. Then

$$t^{\mathcal{M}}[\sigma] = f^{\mathcal{M}}(t_1^{\mathcal{M}}[\sigma], \ldots, t_m^{\mathcal{M}}[\sigma]) = f^{\mathcal{M}}(t_1^{\mathcal{M}}[\tau], \ldots, t_m^{\mathcal{M}}[\tau]) = t^{M}[\tau].$$

ii) We prove this by induction on formulas.

Suppose ϕ is $t_1 = t_2$ where t_1 and t_2 are $\mathcal{L}$-terms. Then

$$\begin{aligned} \mathcal{M} \models_\sigma \phi &\Leftrightarrow t_1^{\mathcal{M}}[\sigma] = t_2^{\mathcal{M}}[\sigma] \\ &\Leftrightarrow t_1^{\mathcal{M}}[\tau] = t_2^{\mathcal{M}}[\tau] \\ &\Leftrightarrow \mathcal{M} \models_\tau \sigma. \end{aligned}$$

Suppose R is an m-ary relation symbol, $t_1, \ldots, t_m$ are $\mathcal{L}$- terms, and ϕ is $R(t_1, \ldots, t_m)$. Then

$$\begin{aligned} \mathcal{M} \models_\sigma \phi &\Leftrightarrow (t_1^{\mathcal{M}}[\sigma], \ldots, t_m^{\mathcal{M}}[\sigma]) \in R^{\mathcal{M}} \\ &\Leftrightarrow (t_1^{\mathcal{M}}[\tau], \ldots, t_m^{\mathcal{M}}[\tau]) \in R^{\mathcal{M}} \\ &\Leftrightarrow \mathcal{M} \models_\tau \phi. \end{aligned}$$

Suppose the claim is true for ψ and ϕ is $\neg\psi$. Then

$$\begin{aligned}\mathcal{M} \models_\sigma \phi &\Leftrightarrow \mathcal{M} \not\models_\sigma \psi\\ &\Leftrightarrow \mathcal{M} \not\models_\tau \psi\\ &\Leftrightarrow \mathcal{M} \models_\tau \phi.\end{aligned}$$

Suppose the claim is true for ψ and θ and ϕ is $\psi \wedge \theta$. Then

$$\begin{aligned}\mathcal{M} \models_\sigma \phi &\Leftrightarrow \mathcal{M} \not\models_\sigma \psi \text{ and } \mathcal{M} \models_\sigma \theta\\ &\Leftrightarrow \mathcal{M} \not\models_\tau \psi \text{ and } \mathcal{M} \models_\tau \theta\\ &\Leftrightarrow \mathcal{M} \models_\tau \phi.\end{aligned}$$

Suppose the claim is true for ψ, ϕ is $\exists v_i \psi$ and $\mathcal{M} \models_\sigma \phi$. Then there is $a \in M$ such that $\mathcal{M} \models_{\sigma[\frac{a}{v_i}]} \psi$. The assignments $\sigma[\frac{a}{v_i}]$ and $\tau[\frac{a}{v_i}]$ agree on all variables free in ψ. Thus, by induction, $\mathcal{M} \models_{\tau[\frac{a}{v_i}]} \psi$ and $\mathcal{M} \models_\tau \phi$. Symmetricly, if $\mathcal{M} \models_\tau \phi$, then $\mathcal{M} \models_\sigma \phi$.

Thus, by induction, $\mathcal{M} \models_\sigma \phi$ if and only if $\mathcal{M} \models_\tau \phi$.

Definition 1.10 We say that an $\mathcal{L}$-formula ϕ is a *sentence* if ϕ has no freely occuring variables.

Corollary 1.11 *Suppose ϕ is an $\mathcal{L}$-sentence and $\mathcal{M}$ is an $\mathcal{L}$-structure. The following are equivalent:*
i) $\mathcal{M} \models_\sigma \phi$ for some assignment σ;
ii) $\mathcal{M} \models_\sigma \phi$ for all assignments σ.

Definition 1.12 If ϕ is a sentence, we write $\mathcal{M} \models \phi$ if $\mathcal{M} \models_\sigma \phi$ for all assignments $\sigma : V \to M$.

Suppose ϕ is a formula with free variables from $v_1, \ldots, v_n$. For notational simplicity, if $a_1, \ldots, a_n \in M$ we write $t^{\mathcal{M}}(\overline{a})$ for the common value of $t^{\mathcal{M}}[\sigma]$ where σ is an assignment with $\sigma(v_i) = a_i$ for $i = 1, \ldots, n$. Similarly, we write and $t\mathcal{M} \models \phi(a_1, \ldots, a_n)$ if $\mathcal{M} \models_\sigma \phi$ for any such σ. By the Coincedence Lemma, this is well defined.

$\mathcal{L}$-embeddings and Substructures

We will also study maps that preserve the interpretation of $\mathcal{L}$.

Definition 1.13 Suppose that $\mathcal{M}$ and $\mathcal{N}$ are $\mathcal{L}$-structures with universes M and N, respectively. An $\mathcal{L}$-*embedding* $\eta : \mathcal{M} \to \mathcal{N}$ is a one-to-one map $\eta : M \to N$ that preserves the interpretation of all of the symbols of $\mathcal{L}$. More precisely:

i) $\eta(f^{\mathcal{M}}(a_1, \ldots, a_{n_f})) = f^{\mathcal{N}}(\eta(a_1), \ldots, \eta(a_{n_f}))$ for all $f \in \mathcal{F}$ and $a_1, \ldots, a_n \in M$;

ii) $(a_1, \ldots, a_{m_R}) \in R^{\mathcal{M}}$ if and only if $(\eta(a_1), \ldots, \eta(a_{m_R})) \in R^{\mathcal{N}}$ for all $R \in \mathcal{R}$ and $a_1, \ldots, a_{m_j} \in M$;

iii) $\eta(c^{\mathcal{M}}) = c^{\mathcal{N}}$ for $c \in \mathcal{C}$.

A bijective $\mathcal{L}$-embedding is called an *$\mathcal{L}$-isomorphism.* If $M \subseteq N$ and the inclusion map is an $\mathcal{L}$-embedding, we say either that $\mathcal{M}$ is a *substructure* of $\mathcal{N}$ or that $\mathcal{N}$ is an *extension* of $\mathcal{M}$.

For example:

i) $(\mathbb{Z}, +, 0)$ is a substructure of $(\mathbb{R}, +, 0)$.

ii) If $\eta : \mathbb{Z} \to \mathbb{R}$ is the function $\eta(x) = e^x$, then η is an $\mathcal{L}_{\text{g}}$-embedding of $(\mathbb{Z}, +, 0)$ into $(\mathbb{R}, \cdot, 1)$.

The next proposition asserts that if a formula without quantifiers is true in some structure, then it is true in every extension and substructre. It is proved by induction on quantifier-free formulas.

Proposition 1.14 *Suppose that $\mathcal{M}$ is a substructure of $\mathcal{N}$, $\overline{a} \in M$, and $\phi(\overline{v})$ is a quantifier-free formula. Then, $\mathcal{M} \models \phi(\overline{a})$ if and only if $\mathcal{N} \models \phi(\overline{a})$.*

Proof

Claim If $t(\overline{v})$ is a term and $\bar{b} \in M$, then $t^{\mathcal{M}}(\bar{b}) = t^{\mathcal{N}}(\bar{b})$. This is proved by induction on terms.

If t is the constant symbol c, then $c^{\mathcal{M}} = c^{\mathcal{N}}$.

If t is the variable v_i, then $t^{\mathcal{M}}(\bar{b}) = b_i = t^{\mathcal{N}}(\bar{b})$.

Suppose that $t = f(t_1, \ldots, t_n)$, where f is an n-ary function symbol, $t_1, \ldots, t_n$ are terms, and $t_i^{\mathcal{M}}(\bar{b}) = t_i^{\mathcal{N}}(\bar{b})$ for $i = 1, \ldots, n$. Because $\mathcal{M} \subseteq \mathcal{N}$, $f^{\mathcal{M}} = f^{\mathcal{N}} | M^n$. Thus,

$$\begin{aligned} t^{\mathcal{M}}(\bar{b}) &= f^{\mathcal{M}}(t_1^{\mathcal{M}}(\bar{b}), \ldots, t_n^{\mathcal{M}}(\bar{b})) \\ &= f^{\mathcal{N}}(t_1^{\mathcal{M}}(\bar{b}), \ldots, t_n^{\mathcal{M}}(\bar{b})) \\ &= f^{\mathcal{N}}(t_1^{\mathcal{N}}(\bar{b}), \ldots, t_n^{\mathcal{N}}(\bar{b})) \\ &= t^{\mathcal{N}}(\bar{b}). \end{aligned}$$

We now prove the proposition by induction on formulas.

If ϕ is $t_1 = t_2$, then

$$\mathcal{M} \models \phi(\overline{a}) \Leftrightarrow t_1^{\mathcal{M}}(\overline{a}) = t_2^{\mathcal{M}}(\overline{a}) \Leftrightarrow t_1^{\mathcal{N}}(\overline{a}) = t_2^{\mathcal{N}}(\overline{a}) \Leftrightarrow \mathcal{N} \models \phi(\overline{a}).$$

If ϕ is $R(t_1, \ldots, t_n)$, where R is an n-ary relation symbol, then

$$\begin{aligned} \mathcal{M} \models \phi(\overline{a}) &\Leftrightarrow (t_1^{\mathcal{M}}(\overline{a}), \ldots, t_n^{\mathcal{M}}(\overline{a})) \in R^{\mathcal{M}} \\ &\Leftrightarrow (t_1^{\mathcal{M}}(\overline{a}), \ldots, t_n^{\mathcal{M}}(\overline{a})) \in R^{\mathcal{N}} \\ &\Leftrightarrow (t_1^{\mathcal{N}}(\overline{a}), \ldots, t_n^{\mathcal{N}}(\overline{a})) \in R^{\mathcal{N}} \\ &\Leftrightarrow \mathcal{N} \models \phi(\overline{a}). \end{aligned}$$

Thus, the proposition is true for all atomic formulas.

Suppose that the proposition is true for ψ and that ϕ is $\neg\psi$. Then,

$$\mathcal{M} \models \neg\phi(\overline{a}) \Leftrightarrow \mathcal{M} \not\models \psi(\overline{a}) \Leftrightarrow \mathcal{N} \not\models \psi(\overline{a}) \Leftrightarrow \mathcal{N} \models \phi(\overline{a}).$$

Finally, suppose that the proposition is true for ψ_0 and ψ_1 and that ϕ is $\psi_0 \wedge \psi_1$. Then,

$$\begin{array}{rcl} \mathcal{M} \models \phi(\overline{a}) & \Leftrightarrow & \mathcal{M} \models \psi_0(\overline{a}) \text{ and } \mathcal{M} \models \psi_1(\overline{a}) \\ & \Leftrightarrow & \mathcal{N} \models \psi_0(\overline{a}) \text{ and } \mathcal{N} \models \psi_1(\overline{a}) \\ & \Leftrightarrow & \mathcal{N} \models \phi(\overline{a}). \end{array}$$

We have shown that the proposition holds for all atomic formulas and that if it holds for ϕ and ψ, then it also holds for $\neg\phi$ and $\phi \wedge \psi$. Because the set of quantifier-free formulas is the smallest set of formulas containing the atomic formulas and closed under negation and conjunction, the proposition is true for all quantifier-free formulas.

Elementary Equivalence and Isomorphism

We next consider structures that satisfy the same sentences.

Definition 1.15 We say that two $\mathcal{L}$-structures $\mathcal{M}$ and $\mathcal{N}$ are *elementarily equivalent* and write $\mathcal{M} \equiv \mathcal{N}$ if

$$\mathcal{M} \models \phi \text{ if and only if } \mathcal{N} \models \phi$$

for all $\mathcal{L}$-sentences ϕ.

We let $\mathrm{Th}(\mathcal{M})$, the *full theory of* $\mathcal{M}$, be the set of $\mathcal{L}$-sentences ϕ such that $\mathcal{M} \models \phi$. It is easy to see that $\mathcal{M} \equiv \mathcal{N}$ if and only if $\mathrm{Th}(\mathcal{M}) = \mathrm{Th}(\mathcal{N})$.

Our next result shows that $\mathrm{Th}(\mathcal{M})$ is an isomorphism invariant of $\mathcal{M}$. The proof uses the important technique of "induction on formulas."

Theorem 1.16 *Suppose that $j : \mathcal{M} \to \mathcal{N}$ is an isomorphism. Then, $\mathcal{M} \equiv \mathcal{N}$.*

Proof We show by induction on formulas that $\mathcal{M} \models \phi(a_1, \ldots, a_n)$ if and only if $\mathcal{N} \models \phi(j(a_1), \ldots, j(a_n))$ for all formulas ϕ.

We first must show that terms behave well.

Claim Suppose that t is a term and the free variables in t are from $\overline{v} = (v_1, \ldots, v_n)$. For $\overline{a} = (a_1, \ldots, a_n) \in M$, we let $j(\overline{a})$ denote $(j(a_1), \ldots, j(a_n))$. Then $j(t^{\mathcal{M}}(\overline{a})) = t^{\mathcal{N}}(j(\overline{a}))$. More formally, we are showing that $j(t^{\mathcal{M}}[\sigma]) = t^{\mathcal{N}}[j \circ \sigma]$ for any assignment σ

We prove this by induction on terms.

i) If $t = c$, then $j(t^{\mathcal{M}}(\overline{a})) = j(c^{\mathcal{M}}) = c^{\mathcal{N}} = t^{\mathcal{N}}(j(\overline{a}))$.

ii) If $t = v_i$, then $j(t^{\mathcal{M}}(\overline{a})) = j(a_i) = t^{\mathcal{N}}(j(a_i))$.

iii) If $t = f(t_1, \ldots, t_m)$, then

$$\begin{array}{rcl} j(t^{\mathcal{M}}(\overline{a})) & = & j(f^{\mathcal{M}}(t_1^{\mathcal{M}}(\overline{a}), \ldots, t_m^{\mathcal{M}}(\overline{a}))) \\ & = & f^{\mathcal{N}}(j(t_1^{\mathcal{M}}(\overline{a})), \ldots, j(t_m^{\mathcal{M}}(\overline{a}))) \\ & = & f^{\mathcal{N}}(t_1^{\mathcal{N}}(j(\overline{a})), \ldots, t_m^{\mathcal{N}}(j(\overline{a}))) \\ & = & t^{\mathcal{N}}(j(\overline{a})). \end{array}$$

We proceed by induction on formulas.
i) If $\phi(\overline{v})$ is $t_1 = t_2$, then

$$\begin{aligned}
\mathcal{M} \models \phi(\overline{a}) &\Leftrightarrow t_1^{\mathcal{M}}(\overline{a}) = t_2^{\mathcal{M}}(\overline{a}) \\
&\Leftrightarrow j(t_1^{\mathcal{M}}(\overline{a})) = j(t_2^{\mathcal{M}}(\overline{a})) \text{ because } j \text{ is injective} \\
&\Leftrightarrow t_1^{\mathcal{N}}(j(\overline{a})) = t_2^{\mathcal{N}}(j(\overline{a})) \\
&\Leftrightarrow \mathcal{N} \models \phi(j(\overline{a})).
\end{aligned}$$

ii) If $\phi(\overline{v})$ is $R(t_1, \ldots, t_n)$, then

$$\begin{aligned}
\mathcal{M} \models \phi(\overline{a}) &\Leftrightarrow (t_1^{\mathcal{M}}(\overline{a}), \ldots, t_n^{\mathcal{M}}(\overline{a})) \in R^{\mathcal{M}} \\
&\Leftrightarrow (j(t_1^{\mathcal{M}}(\overline{a})), \ldots, j(t_n^{\mathcal{M}}(\overline{a}))) \in R^{\mathcal{N}} \\
&\Leftrightarrow (t_1^{\mathcal{N}}(j(\overline{a})), \ldots, t_n^{\mathcal{N}}(j(\overline{a}))) \in R^{\mathcal{N}} \\
&\Leftrightarrow \mathcal{N} \models \phi(j(\overline{a})).
\end{aligned}$$

iii) If ϕ is $\neg\psi$, then by induction

$$\mathcal{M} \models \phi(\overline{a}) \Leftrightarrow \mathcal{M} \not\models \psi(\overline{a}) \Leftrightarrow \mathcal{N} \not\models \psi(j(\overline{a})) \Leftrightarrow \mathcal{N} \models \phi(j(\overline{a})).$$

iv) If ϕ is $\psi \wedge \theta$, then

$$\begin{aligned}
\mathcal{M} \models \phi(\overline{a}) &\Leftrightarrow \mathcal{M} \models \psi(\overline{a}) \text{ and } \mathcal{M} \models \theta(\overline{a}) \\
&\Leftrightarrow \mathcal{N} \models \psi(j(\overline{a})) \text{ and } \mathcal{N} \models \theta(j(\overline{a})) \Leftrightarrow \mathcal{N} \models \phi(j(\overline{a})).
\end{aligned}$$

v) If $\phi(\overline{v})$ is $\exists w\ \psi(\overline{v}, w)$, then

$$\begin{aligned}
\mathcal{M} \models \phi(\overline{a}) &\Leftrightarrow \mathcal{M} \models \psi(\overline{a}, b) \text{ for some } b \in M \\
&\Rightarrow \mathcal{N} \models \psi(j(\overline{a}), j(b)) \\
&\Rightarrow \mathcal{N} \models \phi(j(\overline{a})).
\end{aligned}$$

On the other hand,

$$\begin{aligned}
\mathcal{N} \models \phi(j(\overline{a})) &\Leftrightarrow \mathcal{N} \models \psi(j(\overline{a}), c) \text{ for some } c \in N \\
&\Rightarrow \mathcal{M} \models \psi(\overline{a}, j^{-1}(c)) \text{ because } j \text{ is surjective.} \\
&\Rightarrow \mathcal{M} \models \phi(\overline{a}).
\end{aligned}$$

2 Theories

Let $\mathcal{L}$ be a language. An $\mathcal{L}$-*theory* T is simply a set of $\mathcal{L}$-sentences. We say that $\mathcal{M}$ is a *model* of T and write $\mathcal{M} \models T$ if $\mathcal{M} \models \phi$ for all sentences $\phi \in T$.

The set $T = \{\forall x\ x = 0, \exists x\ x \neq 0\}$ is a theory. Because the two sentences in T are contradictory, there are no models of T. We say that a theory is *satisfiable* if it has a model.

We say that a class of $\mathcal{L}$-structures $\mathcal{K}$ is an *elementary class* if there is an $\mathcal{L}$-theory T such that $\mathcal{K} = \{\mathcal{M} : \mathcal{M} \models T\}$.

One way to get a theory is to take $\mathrm{Th}(\mathcal{M})$, the full theory of an $\mathcal{L}$-structure $\mathcal{M}$. In this case, the elementary class of models of $\mathrm{Th}(\mathcal{M})$ is exactly the class of $\mathcal{L}$-structures elementarily equivalent to $\mathcal{M}$. More typically, we have a class of structures in mind and try to write a set of properties T describing these structures. We call these sentences *axioms* for the elementary class.

We give a few basic examples of theories and elementary classes that we will return to frequently.

Example 2.1 *Infinite Sets*

Let $\mathcal{L} = \emptyset$.

Consider the $\mathcal{L}$-theory where we have, for each n, the sentence ϕ_n given by

$$\exists x_1 \exists x_2 \ldots \exists x_n \bigwedge_{i<j\leq n} x_i \neq x_j.$$

The sentence ϕ_n asserts that there are at least n distinct elements, and an $\mathcal{L}$-structure $\mathcal{M}$ with universe M is a model of T if and only if M is infinite.

Example 2.2 *Linear Orders*

Let $\mathcal{L} = \{<\}$, where $<$ is a binary relation symbol. The class of linear orders is axiomatized by the $\mathcal{L}$-sentences
$\forall x\ \neg(x < x)$,
$\forall x \forall y \forall z\ ((x < y \wedge y < z) \rightarrow x < z)$,
$\forall x \forall y\ (x < y \vee x = y \vee y < x)$.

There are a number of interesting extensions of the theory of linear orders. For example, we could add the sentence

$$\forall x \forall y\ (x < y \rightarrow \exists z\ (x < z \wedge z < y))$$

to get the theory of dense linear orders, or we could instead add the sentence

$$\forall x \exists y\ (x < y \wedge \forall z (x < z \rightarrow (z = y \vee y < z)))$$

to get the theory of linear orders where every element has a unique successor. We could also add sentences that either assert or deny the existence of top or bottom elements.

Example 2.3 *Equivalence Relations*

Let $\mathcal{L} = \{E\}$, where E is a binary relation symbol. The theory of equivalence relations is given by the sentences
$\forall x\ E(x, x)$,
$\forall x \forall y (E(x, y) \rightarrow E(y, x))$,
$\forall x \forall y \forall z ((E(x, y) \wedge E(y, z)) \rightarrow E(x, z))$.

If we added the sentence

$$\forall x \exists y (x \neq y \wedge E(x,y) \wedge \forall z\ (E(x,z) \rightarrow (z = x \vee z = y)))$$

we would have the theory of equivalence relations where every equivalence class has exactly two elements. If instead we added the sentence

$$\exists x \exists y (\neg E(x,y) \wedge \forall z (E(x,z) \vee E(y,z)))$$

and the infinitely many sentences

$$\forall x \exists x_1 \exists x_2 \ldots \exists x_n \left(\bigwedge_{i<j\leq n} x_i \neq x_j \wedge \bigwedge_{i=1}^{n} E(x, x_i) \right)$$

we would axiomatize the class of equivalence relations with exactly two classes, both of which are infinite.

Example 2.4 *Graphs*

Let $\mathcal{L} = \{R\}$ where R is a binary relation. We restrict our attention to irreflexive graphs. These are axiomatized by the two sentences
$\forall x\ \neg R(x,x)$,
$\forall x \forall y\ (R(x,y) \rightarrow R(y,x))$.

Example 2.5 *Groups*

Let $\mathcal{L} = \{\cdot, e\}$, where $\cdot$ is a binary function symbol and e is a constant symbol. We will write $x \cdot y$ rather than $\cdot(x,y)$. The class of groups is axiomatized by
$\forall x\ e \cdot x = x \cdot e = x$,
$\forall x \forall y \forall z\ x \cdot (y \cdot z) = (x \cdot y) \cdot z$,
$\forall x \exists y\ x \cdot y = y \cdot x = e$.
We could also axiomatize the class of Abelian groups by adding

$$\forall x \forall y\ x \cdot y = y \cdot x.$$

Let $\phi_n(x)$ be the $\mathcal{L}$-formula

$$\underbrace{x \cdot x \cdots x}_{n-\text{times}} = e;$$

which asserts that $x^n = e$.

We could axiomatize the class of torsion-free groups by adding

$$\{\forall x\ (x = e \vee \neg \phi_n(x)) : n \geq 2\}$$

to the axioms for groups. Alternatively, we could axiomatize the class of groups where every element has order at most N by adding to the axioms for groups the sentence

$$\forall x \bigvee_{n \leq N} \phi_n(x).$$

Note that similar ideas will not work to axiomatize the class of torsion groups because the corresponding sentence would be infinitely long. In the next chapter, we will see that the class of torsion groups is not elementary.

Let $\psi_n(x,y)$ be the formula

$$\underbrace{x \cdot x \cdots x}_{n\text{-times}} = y;$$

which asserts that $x^n = y$. We can axiomatize the class of divisible groups by adding the axioms $\{\forall y \exists x\ \psi_n(x,y) : n \geq 2\}$.

It will often be useful to deal with additive groups instead of multiplicative groups. The class of additive groups is the collection structures in the language $\mathcal{L} = \{+, 0\}$, axiomatized as above replacing $\cdot$ by $+$ and e by 0.

Example 2.6 *Ordered Abelian Groups*

Let $\mathcal{L} = \{+, <, 0\}$, where $+$ is a binary function symbol, $<$ is a binary relation symbol, and 0 is a constant symbol. The axioms for ordered groups are
the axioms for additive groups,
the axioms for linear orders, and
$\forall x \forall y \forall z (x < y \rightarrow x + z < y + z)$.

Example 2.7 *Left R-modules*

Let R be a ring with multiplicative identity 1. Let $\mathcal{L} = \{+, 0\} \cup \{r : r \in R\}$ where $+$ is a binary function symbol, 0 is a constant, and r is a unary function symbol for $r \in R$. In an R-module, we will interpret r as scalar multiplication by R. The axioms for left R-modules are
the axioms for additive commutative groups,
$\forall x\ r(x + y) = r(x) + r(y)$ for each $r \in R$,
$\forall x\ (r + s)(x) = r(x) + s(x)$ for each $r, s \in R$,
$\forall x\ r(s(x)) = rs(x)$ for $r, s \in R$,
$\forall x\ 1(x) = x$.

Example 2.8 *Rings and Fields*

Let $\mathcal{L}_{\mathrm{r}}$ be the language of rings $\{+, -, \cdot, 0, 1\}$, where $+$, $-$, and $\cdot$ are binary function symbols and 0 and 1 are constants. The axioms for rings are given by
the axioms for additive commutative groups,
$\forall x \forall y \forall z\ (x - y = z \leftrightarrow x = y + z)$,
$\forall x\ x \cdot 0 = 0$,
$\forall x \forall y \forall z\ (x \cdot (y \cdot z) = (x \cdot y) \cdot z)$,
$\forall x\ x \cdot 1 = 1 \cdot x = x$,
$\forall x \forall y \forall z\ x \cdot (y + z) = (x \cdot y) + (x \cdot z)$,
$\forall x \forall y \forall z\ (x + y) \cdot z = (x \cdot z) + (y \cdot z)$.

The second axiom is only necessary because we include $-$ in the language (this will be useful later). We axiomatize the class of fields by adding the axioms

$\forall x \forall y\ x \cdot y = y \cdot x$,
$\forall x\ (x \neq 0 \to \exists y\ x \cdot y = 1)$.

We axiomatize the class of algebraically closed fields by adding to the field axioms the sentences

$$\forall a_0 \ldots \forall a_{n-1} \exists x\ x^n + \sum_{i=0}^{n-1} a_i x^i = 0$$

for $n = 1, 2, \ldots$. Let ACF be the axioms for algebraically closed fields.

Let ψ_p be the $\mathcal{L}_{\mathrm{r}}$-sentence $\forall x \underbrace{x + \ldots + x}_{p-\text{times}} = 0$, which asserts that a field has characteristic p. For $p > 0$ a prime, let $\mathrm{ACF}_p = \mathrm{ACF} \cup \{\psi_p\}$ and $\mathrm{ACF}_0 = \mathrm{ACF} \cup \{\neg\psi_p : p > 0\}$, be the theories of algebraically closed fields of characteristic p and characteristic zero, respectively.

Example 2.9 *Ordered Fields*

Let $\mathcal{L}_{\mathrm{or}} = \mathcal{L}_{\mathrm{r}} \cup \{<\}$. The class of ordered fields is axiomatized by the axioms for fields,
the axioms for linear orders,
$\forall x \forall y \forall z\ (x < y \to x + z < y + z)$,
$\forall x \forall y \forall z\ ((x < y \wedge z > 0) \to x \cdot z < y \cdot z)$.

Example 2.10 *Differential Fields*

Let $\mathcal{L} = \mathcal{L}_{\mathrm{r}} \cup \{\delta\}$, where δ is a unary function symbol. The class of differential fields is axiomatized by
the axioms of fields,
$\forall x \forall y\ \delta(x + y) = \delta(x) + \delta(y)$,
$\forall x \forall y\ \delta(x \cdot y) = x \cdot \delta(y) + y \cdot \delta(x)$.

Example 2.11 *Peano Arithmetic*[1]

Let $\mathcal{L} = \{+, \cdot, s, 0\}$, where $+$ and $\cdot$ are binary functions, s is a unary function, and 0 is a constant. We think of s as the successor function $x \mapsto x + 1$. The Peano axioms for arithmetic are the sentences
$\forall x\ s(x) \neq 0$,
$\forall x\ (x \neq 0 \to \exists y\ s(y) = x)$,
$\forall x\ x + 0 = x$,
$\forall x\ \forall y\ x + (s(y)) = s(x + y)$,
$\forall x\ \ x \cdot 0 = 0$,
$\forall x \forall y\ x \cdot s(y) = (x \cdot y) + x$,
and the axioms $\mathrm{Ind}(\phi)$ for each formula $\phi(v, \overline{w})$, where $\mathrm{Ind}(\phi)$ is the sentence
$\forall \overline{w}\ [(\phi(0, \overline{w}) \wedge \forall v\ (\phi(v, \overline{w}) \to \phi(s(v), \overline{w}))) \to \forall x\ \phi(x, \overline{w})]$.

The axiom $\mathrm{Ind}(\phi)$ formalizes an instance of induction. It asserts that if $\overline{a} \in M$, $X = \{m \in M : \mathcal{M} \models \phi(m, \overline{a})\}$, $0 \in X$, and $s(m) \in X$ whenever $m \in X$, then $X = M$.

[1] This axiomatization is traditional, but in §10 we give a different axiomatization of Peano Arithmetic.

Logical Consequence

Definition 2.12 Let T be an $\mathcal{L}$-theory and ϕ an $\mathcal{L}$-sentence. We say that ϕ is a *logical consequence* of T and write $T \models \phi$ if $\mathcal{M} \models \phi$ whenever $\mathcal{M} \models T$.

We give several examples.

Example 2.13 *Let $\mathcal{L} = \{\cdot, 1\}$ be the language groups and let T be the theory of groups. Then*

$$T \models \forall x \forall y \forall z\ (x \cdot z = y \cdot z \rightarrow x = y).$$

Proof Suppose $G \models T$ is a group and $a, b, c \in G$ and $ac = bc$. There is $d \in G$ such that $cd = 1$.

$$\begin{aligned} (ac)d &= (bc)d \\ a(cd) &= b(cd) \\ a \cdot 1 &= b \cdot 1 \\ a &= b \end{aligned}$$

Example 2.14 *Let $\mathcal{L} = \{+, <, 0\}$ and let T be the theory of ordered Abelian groups. Then $\forall x (x \neq 0 \rightarrow x + x \neq 0)$ is a logical consequence of T.*

Proof Suppose that $\mathcal{M} = (M, +, <, 0)$ is an ordered Abelian group. Let $a \in M \setminus \{0\}$. We must show that $a + a \neq 0$. Because $(M, <)$ is a linear order $a < 0$ or $0 < a$. If $a < 0$, then $a + a < 0 + a = a < 0$. Because $\neg(0 < 0)$, $a + a \neq 0$. If $0 < a$, then $0 < a = 0 + a < a + a$ and again $a + a \neq 0$.

Example 2.15 *Let T be the theory of groups where every element has order 2. Then, $T \not\models \exists x_1 \exists x_2 \exists x_3 (x_1 \neq x_2 \wedge x_2 \neq x_3 \wedge x_1 \neq x_3)$.*

Proof Clearly, $\mathbb{Z}/2\mathbb{Z} \models T \wedge \neg \exists x_1 \exists x_2 \exists x_3 (x_1 \neq x_2 \wedge x_2 \neq x_3 \wedge x_1 \neq x_3)$.

Exercise 2.16 Show that if T is unsatisfiable, then $T \models \phi$ for all ϕ.

In general, to show that $T \models \phi$ we give an informal mathematical proof as above that $\mathcal{M} \models \phi$ whenever $\mathcal{M} \models T$. To show that $T \not\models \phi$, we usually construct a counterexample, i.e., we construct $\mathcal{M} \models T \cup \{\neg \phi\}$.

The following observation will be useful. It formalizes the usual way we prove a universal statement by naming a generic element and proving it for that element.

Lemma 2.17 *Suppose c is a constant not occurring in T or $\phi(v)$ where ϕ is a formula with free variable ϕ. and $T \models \phi(c)$. Then $\mathcal{M} \models \forall v\ \phi(v)$.*

Proof Suppose $\mathcal{M} \models T$. Let a be any element of $\mathcal{M}$. We construct $\mathcal{M}^*$ by changing the interpretation to make $c^{\mathcal{M}} = a$. Since c does not occur in T and we have changed the interpretation of no other symbols $\mathcal{M}^* \models T$. But then $\mathcal{M}^* \models \phi(c)$ and $\mathcal{M} \models \phi(a)$. Thus $\mathcal{M} \models \forall v\ \phi(v)$.

In the next sections we will also need a notion of logical consequence for formulas.

Definition 2.18 If Γ is a set of $\mathcal{L}$-formulas and ϕ is an $\mathcal{L}$-formula, we say that ϕ is a *logical consequnce* of Γ and write $\Gamma \models \phi$ if $\mathcal{M} \models_\sigma \phi$, whenever $\mathcal{M}$ is an $\mathcal{L}$-structure, $\sigma : V \to M$ is an assignment and $\mathcal{M} \models_\sigma \psi$ for all $\psi \in \Gamma$.

Exercise 2.19 Suppose Γ is a set of formulas, $\phi(v)$ is a formula where v is free, ψ is a formula and w is either a variable or constant symbol not occurring in either Γ , ϕ or ψ.

a) Show that if $\Gamma \models \phi(w)$, then $\Gamma \models \forall v\ \phi(v)$.

b) Show that if $\Gamma \cup \{\phi(w)\} \models \psi$, then $\Gamma \cup \{\exists v\ \phi(v)\} \models \psi$.

3 Formal Proofs

A priori to show $\Gamma \models \phi$ we must examine all structures $\mathcal{M}$ and all assignments $\sigma : V \to M$ where $\mathcal{M} \models_\sigma \Gamma$ and show that $\mathcal{M} \models_\sigma \phi$. This is in general an impossible task. In mathematics we show that $\Gamma \models \phi$ by giving a proof. In this section we will give one example of a formal proof system. We will write $\Gamma \vdash \phi$ if there is a formal proof of ϕ from Γ. We will demand two properties of our proof system.

• SOUNDNESS: If $\Gamma \vdash \phi$, then $\Gamma \models \phi$.
Thus anything that is provable is a logical consequence.

• COMPLETENESS: If $\Gamma \models \phi$, then $\Gamma \vdash \phi$.
Thus every logical consequence is provable.

Soundness of our system will be routine. Gödel's Completeness Theorem will be proved in the next section.

In addition we will demand that proof are finite. Any proof will be a finite collection of symbols. Moreover, it should be easy to check that a proported proof is correct.

Our proof system is a variant of the sequent calculus.

Definition 3.1 A *proof* will be a finite sequence of assertions of the form

1. $\Gamma_1 \vdash \phi_1$
2. $\Gamma_2 \vdash \phi_2$

$\vdots \quad \vdots$

$n.$ $\Gamma_n \vdash \phi_n$

where each Γ_i is a **finite** set of formulas (possibly empty), ϕ_i is a formula and each assertion $\Gamma_i \vdash \phi_i$ can be derived from the assertions $\Gamma_1 \vdash \phi_1, \ldots, \Gamma_{i-1} \vdash \phi_{i-1}$ by one of the inference rules that we will shortly describe.

We think of "$\Gamma \vdash \phi$" as the assertion that ϕ is derivable from Γ. We will write $\Gamma, \psi \vdash \phi$ to abbreviate $\Gamma \cup \{\psi\} \vdash \phi$.

Our inference rules will have the form

$$\frac{\Gamma_1 \vdash \phi_1 \ \ldots \ \ \Gamma_n \vdash \phi_n}{\Delta \vdash \psi.}$$

This means that if have already established $\Gamma_1 \vdash \phi_1, \ldots, \Gamma_n \vdash \phi_n$, then we can conlclude that $\Delta \vdash \psi$.

We begin to give the rules of our calculus.

Structural Rules:

S1. (Assumption) If $\phi \in \Gamma$, then

$$\Gamma \vdash \phi$$

S2. (Monotonicity) If $\Gamma \subseteq \Delta$, then

$$\frac{\Gamma \vdash \phi}{\Delta \vdash \phi}$$

S3. (Proof by cases)

$$\frac{\Gamma, \psi \vdash \phi \quad \Gamma, \neg\psi \vdash \phi}{\Gamma \vdash \phi}$$

Connective Rules

C1. (Contradiction Rule)

$$\frac{\Gamma, \neg\phi \vdash \psi \quad \Gamma, \neg\phi \vdash \neg\psi}{\Gamma \vdash \phi}$$

C2. (Left $\vee$-rule)

$$\frac{\Gamma, \phi \vdash \theta \quad \Gamma, \psi \vdash \theta}{\Gamma, (\phi \vee \psi) \vdash \theta}$$

C3. (Right $\vee$-rules)

$$\frac{\Gamma \vdash \phi}{\Gamma \vdash (\phi \vee \psi)} \qquad\qquad \frac{\Gamma \vdash \phi}{\Gamma \vdash (\psi \vee \phi)}$$

Before giving the inference rules for quantifiers and equality we give some sample derivations and prove some useful inference rules which are consequences of the rules above.

Example 3.2 $\vdash (\phi \vee \neg\phi)$

1.	$\phi \vdash \phi$	S1
2.	$\phi \vdash (\phi \vee \neg\phi)$	C3
3.	$\neg\phi \vdash \neg\phi$	S1
4.	$\neg\phi \vdash (\phi \vee \neg\phi)$	C3
5.	$\vdash (\phi \vee \neg\phi)$	S3

Example 3.3 $\neg\neg\phi \vdash \phi$

1.	$\neg\neg\phi, \neg\phi \vdash \neg\neg\phi$	S1
2.	$\neg\neg\phi, \neg\phi \vdash \neg\phi$	S1
3.	$\neg\neg\phi \vdash \phi$	C1

Lemma 3.4 (Second Contradiction Rule)

$$\frac{\Gamma \vdash \psi \qquad \Gamma \vdash \neg\psi}{\Gamma \vdash \phi}$$

Proof

1.	$\Gamma \vdash \psi$	Premise
2.	$\Gamma, \neg\phi \vdash \psi$	S2
3.	$\Gamma \vdash \neg\psi$	Premise
4.	$\Gamma, \neg\phi \vdash \neg\psi$	S2
5.	$\Gamma \vdash \phi$	C1

Lemma 3.5 (Chain Rule)

$$\frac{\Gamma \vdash \phi \qquad \Gamma, \phi \vdash \psi}{\Gamma \vdash \psi}$$

Proof

1.	$\Gamma \vdash \phi$	Premise
2.	$\Gamma, \neg\phi \vdash \phi$	S2
3.	$\Gamma, \neg\phi \vdash \neg\phi$	S1
4.	$\Gamma, \neg\phi \vdash \psi$	Apply 3.4 to 2,3
5.	$\Gamma, \phi \vdash \psi$	Premise
6.	$\Gamma \vdash \psi$	apply S3 to 4,5

Having proved the Second Contradiction Rule, we are now free to use it as if it was an inference rules.

Lemma 3.6 (Contraposition)

$$\frac{\Gamma, \phi \vdash \psi}{\Gamma, \neg\psi \vdash \neg\phi}$$

Proof

1.	$\Gamma, \phi \vdash \psi$	Premise
2.	$\Gamma, \neg\psi, \phi \vdash \psi$	S2
3.	$\Gamma, \neg\psi, \phi \vdash \neg\psi$	S1
4.	$\Gamma, \neg\psi, \phi \vdash \neg\phi$	apply 3.4 to 2,3
5.	$\Gamma, \neg\psi, \neg\phi \vdash \neg\phi$	S1
6.	$\Gamma, \neg\psi \vdash \neg\phi$	apply S3 to 4,5

Exercise 3.7 We can similarly prove the following versions of the contraposition law.

$$\frac{\Gamma, \neg\phi \vdash \neg\psi}{\Gamma, \psi \vdash \phi} \qquad \frac{\Gamma, \neg\phi \vdash \psi}{\Gamma, \neg\psi \vdash \phi} \qquad \frac{\Gamma, \phi \vdash \neg\psi}{\Gamma, \psi \vdash \neg\phi}$$

Lemma 3.8 (Modus ponens)

$$\frac{\Gamma \vdash (\phi \to \psi) \qquad \Gamma \vdash \phi}{\Gamma \vdash \psi}$$

Proof

Recall that $(\phi \to \psi)$ is an abbreviation for $(\neg\phi \vee \psi)$.

1.	$\Gamma \vdash \phi$	Premise
2.	$\Gamma, \neg\phi \vdash \phi$	S2
3.	$\Gamma, \neg\phi \vdash \neg\phi$	S1
4.	$\Gamma, \neg\phi \vdash \psi$	3.4 applied to 2,3
5.	$\Gamma, \psi \vdash \psi$	S1
6.	$\Gamma, (\neg\phi \vee \psi) \vdash \psi$	C2
7.	$\Gamma \vdash (\neg\phi \vee \psi)$	Premise
8.	$\Gamma \vdash \psi$	3.5 applied to 6,7

Equality Rules:

E1.(Reflexivity) Let t be any term.

$$\vdash t = t$$

E2. (Substitution) Let $\phi(v)$ be a formula in which v occurs freely Let t_0, t_1 be terms and let $\phi(t_i)$ be the formula obtained by substituting t_i for all free occurences of v in $\phi(v)$.

$$\frac{\Gamma \vdash \phi(t_0)}{\Gamma, t_0 = t_1 \vdash \phi(t_1)}$$

We give two sample derivations.

Example 3.9 $t_0 = t_1 \vdash t_1 = t_0$.

Let $\phi(v)$ be "$v = t_0$".

1.	$\vdash t_0 = t_0$	E1
2.	$t_0 = t_1 \vdash t_0 = t_0$	S2
3.	$t_0 = t_1 \vdash t_1 = t_0$	E2 applied to $\phi(v)$

Example 3.10 $t_0 = t_1, t_1 = t_2 \vdash t_0 = t_2$

Substitute t_2 for t_1 in $t_0 = t_1$.

We conclude our list of inference rules with rules for manipulating quantifiers.

Quantifier Rules

Q1. (right $\exists$-introduction) Let $\phi(v)$ be a formula in which v is a free variable (there may be others). Suppose t is a term and $\phi(t)$ is the formula obtained by replacing all free occurences of v by t.

$$\frac{\Gamma \vdash \phi(t)}{\Gamma \vdash \exists v \phi(v)}$$

Q2. (left $\exists$-introduction) Let $\phi(v)$ be a formula in which v is a free variable. Let y be either i) a constant symbol not occuring in Γ or ψ or ii) a variable not occuring freely in Γ or ψ.

$$\frac{\Gamma, \phi(y) \vdash \psi}{\Gamma, \exists v \phi(v) \vdash \psi}$$

Q2. expresses the usual way that we prove ψ from $\exists v \phi(v)$. We assume that $\phi(v)$ holds for some v and show that $\phi(v) \vdash \psi$. We then conclude ψ follows from $\exists v\ \phi(v)$. See Exercise 2.19

This completes our list of inference rules. We give one more useful lemma and two sample derivations.

Example 3.11 $\vdash \exists x\ x = x$

Let t be a term. Let $\phi(v)$ be $v = v$.

1.	$\vdash t = t$	E1
2.	$\vdash \exists x\ x = x$	Q1

Lemma 3.12 (Right $\forall$-introduction) *Suppose v does not occur freely in Γ then*

$$\frac{\Gamma \vdash \phi(v)}{\Gamma \vdash \forall v\ \phi(v).}$$

Proof

Let ψ be any sentence. Recall that $\forall v\ \phi(v)$ is an abbreviation for $\neg \exists v\ \neg \phi(v)$.

1.	$\Gamma \vdash \phi(v)$	Premise
2.	$\Gamma, \neg\phi(v) \vdash \phi(v)$	S2
3.	$\Gamma, \neg\phi(v) \vdash \neg\phi(v)$	S1
4.	$\Gamma, \neg\phi(v) \vdash \psi$	apply 3.4 to 2,3
5.	$\Gamma, \exists v \neg\phi(v) \vdash \psi$	Q2
6.	$\Gamma, \neg\psi \vdash \neg\exists v \neg\phi(v)$	apply 3.6 to 5
7.	$\Gamma, \neg\phi(v) \vdash \neg\psi$	apply 3.4 to 2,3
8.	$\Gamma, \exists v \neg\phi(v) \vdash \neg\psi$	Q2
9.	$\Gamma, \psi \vdash \neg\exists v \neg\phi(v)$	apply 3.7 to 8
10.	$\Gamma \vdash \neg\exists v \neg\phi(v)$	by S2 from 6,9

Example 3.13 $\exists x \forall y\ \phi(x,y) \vdash \forall y \exists x\ \phi(x,y)$.

1.	$\neg\phi(x,y) \vdash \neg\phi(x,y)$	S1
2.	$\neg\phi(x,y) \vdash \exists y\ \neg\phi(x,y)$	Q1
3.	$\neg\exists y\ \neg\phi(x,y) \vdash \phi(x,y)$	apply 3.7 to 2.
4.	$\neg\exists y\ \neg\phi(x,y) \vdash \exists x \phi(x,y)$	Q1
5.	$\neg\exists y\ \neg\phi(x,y) \vdash \forall y \exists x \phi(x,y)$	3.12
6.	$\exists x \neg\exists y\ \neg\phi(x,y) \vdash \forall y \exists x \phi(x,y)$	Q2

Theorem 3.14 (Soundness Theorem) *Suppose that the assertion $\Gamma \vdash \phi$ can be derived using the inference rules given above. Then $\Gamma \models \phi$.*

Proof

Recall that $\Gamma \models \phi$ if for any $\mathcal{L}$-structure $\mathcal{M}$ and any assignment $\sigma : V \to M$, if $\mathcal{M} \models_\sigma \Gamma$, then $\mathcal{M} \models_\sigma \phi$.

We prove the Soundness Theorem by induction on proofs.

Base cases:

S1. Clearly if $\phi \in \Gamma$, then $\Gamma \models \phi$.

E1. Clearly $\mathcal{M} \models_\sigma t = t$ for any assignment σ.

Inference rules: If we have an inference rule

$$\frac{\Gamma_1 \vdash \phi_1\ \ldots\ \Gamma_n \vdash \phi_n}{\Delta \vdash \psi}$$

then we must show that if $\Gamma_i \models \phi_i$ for all i, then $\Delta \models \psi$.

This is obvious for S2, C2, C3, E2, and Q1.

S3. Suppose $\Gamma, \phi \models \psi$ and $\Gamma, \neg\phi \models \psi$. If $\mathcal{M} \models \Gamma$, then $\mathcal{M} \models \phi$ or $\mathcal{M} \models \neg\phi$. In either case $\mathcal{M} \models \psi$.

C1. Suppose $\Gamma, \neg\phi \models \psi$ and $\Gamma, \neg\phi \models \neg\psi$. Let $\mathcal{M} \models \Gamma$. Since we can't have $\mathcal{M} \models \psi$ and $\mathcal{M} \models \neg\psi$ we must have $\mathcal{M} \models \phi$.

Q2. See Exercise 2.19

Since all of the inference rules preserve truth the soundness theorem holds.

Definition 3.15 Suppose Γ is a (possibly infinite) set of sentences. We say that ϕ is *provable* from Γ if for some finite $\Delta \subseteq \Gamma$ the assertion $\Delta \vdash \phi$ is derivable in our calculus. If ϕ is provable from Γ we write $\Gamma \vdash \phi$.

This is a slight abuse of notation, but should cause no confusion.

Corollary 3.16 *If $\Gamma \vdash \phi$, then $\Gamma \models \phi$.*

Proof Let Δ be a finite subset of Γ such that $\Delta \vdash \phi$ is derivable. Then $\Delta \models \phi$. Since any model of Γ is a model of Δ, $\Gamma \models \phi$.

Definition 3.17 : We say that Γ is *consistent* if there is no sentence ϕ such that $\Gamma \vdash \phi$ and $\Gamma \vdash \neg\phi$.

Proposition 3.18 *i) Γ is inconsistent if and only if $\Gamma \vdash \psi$ for every formula ψ.*

ii) If Γ is satisfiable, then Γ is consistent.

iii) If Γ is consistent, then for any formula ϕ either $\Gamma \cup \{\phi\}$ is consistent or $\Gamma \cup \{\neg\phi\}$ is consistent (or both).

iv) If $\Gamma \not\vdash \phi$, then $\Gamma \cup \{\neg\phi\}$ is consistent.

Proof i) If $\Gamma \vdash \phi$ and $\Gamma \vdash \neg\phi$, then $\Gamma \vdash \psi$ by Lemma 3.4. Certainly if every sentence is derivable from Γ, then Γ is inconsistent.

ii) If $\mathcal{A} \models \Gamma$ either $\mathcal{A} \not\models \phi$ or $\mathcal{A} \not\models \neg\phi$. Thus by the Soundness Theorem, $\Gamma \not\vdash \phi$ or $\Gamma \not\vdash \neg\phi$.

iii) Suppose not. Let ψ be any sentence. By i) $\Gamma, \phi \vdash \psi$ and $\Gamma, \neg\phi \vdash \psi$. By S3, $\Gamma \vdash \psi$. Thus Γ is inconsistent.

iv) Suppose $\Gamma \cup \{\neg\phi\}$ is inconsistent. Then $\Gamma \cup \{\neg\phi\} \vdash \phi$. Since $\Gamma \cup \{\phi\} \vdash \phi$, by S3, $\Gamma \vdash \phi$.

In §4 we will prove the converse of 3.18 ii). We will see that the converse is just a restatement of Gödel's Completeness Theorem.

4 Gödel's Completeness Theorem

In this section we will prove one of the central theorems of mathematical logic

Theorem 4.1 (Gödel's Completeness Theorem) *Let Γ be a set of $\mathcal{L}$-sentences. If $\Gamma \models \phi$, then $\Gamma \vdash \phi$.*

To prove the Completeness Theorem we will infact prove the following converse to 3.18 ii).

(*) If Γ is consistent, then Γ is satisfiable.

Proof (*) $\Rightarrow$ Completeness

Suppose $\Gamma \not\vdash \phi$, then, by 3.18, $\Gamma \cup \{\neg\phi\}$ is consistent. By (*) $\Gamma \cup \{\neg\phi\}$ has a model $\mathcal{M}$. But then $\Gamma \not\models \phi$.

To prove (*) we must actually construct a model of Γ. The method of proof we give here is due to Leon Henkin.

Definition 4.2 We say that a consistent set of $\mathcal{L}$-sentences Σ is *maximal consistent* if for all $\mathcal{L}$-sentences ϕ either $\phi \in \Sigma$ or $\neg\phi \in \Sigma$ (as Σ is consistent exactly one of ϕ and $\neg\phi$ is in Σ).

Lemma 4.3 *i) If Σ is maximal consistent and $\Sigma \vdash \phi$, then $\phi \in \Sigma$.*
ii) If Σ is maxiaml consistent and $\phi \vee \psi \in \Sigma$, then $\phi \in \Sigma$ or $\psi \in \Sigma$.

Proof

i) If not, $\neg\phi \in \Sigma$ and Σ is inconsistent.

ii) Otherwise $\neg\phi$ and $\neg\psi$ are both in Σ and hence $\neg(\phi \vee \psi) \in \Sigma$, contradicting consistency.

Definition 4.4 We say that Σ has the *witness property* if for any $\mathcal{L}$-formula $\phi(v)$ there is a constant c such that

$$\Sigma \vdash (\exists v \phi(v) \ \rightarrow \ \phi(c)).$$

Theories with this property are sometimes called *Henkinized.*

The proof of (*) comes in two steps:

STEP 1. Show that if Γ is consistent, there is $\Sigma \supseteq \Gamma$ which is maximal consistent and Henkinized. (Note: In general we will have to expand the language to get a theory with the witness property.)

STEP 2. Show that if Σ is maximal consistent and has the witness property, then there is a model of Σ.

We will examine STEP 2 first. Let $\mathcal{L}$ denote the language of Σ. Let C be the constants of $\mathcal{L}$. The universe of our model will be equivalence classes of elements of C. If c_1 and c_2 are constants we say that $c_1 E c_2$ iff and only if $c_1 = c_2 \in \Sigma$.

Lemma 4.5 *E is an equivalence relation.*

Proof
Let $c_1, c_2, c_3 \in C$. By E1, E2, and the examples following them

$$\Sigma \vdash c_1 = c_1$$

$$\Sigma, c_1 = c_2 \vdash c_2 = c_1$$

and

$$\Sigma, c_1 = c_2, c_2 = c_3 \vdash c_1 = c_3.$$

Thus, by 4.3, E is an equivalence relation.

For $c \in C$ let $[c]$ denote the equivalence class of c. We now begin to build a structure $\mathcal{A}$ which we call the *canonical structure* for Σ. The underlying set of $\mathcal{A}$ will be

$$A = \{[c] : c \in C\}.$$

The next lemma will allow us to interpret the relation and function symbols of $\mathcal{L}$.

Lemma 4.6 *i) If R is an n-ary relation symbol of $\mathcal{L}$, $c_1, \ldots, c_n, d_1, \ldots, d_n \in C$ and $c_i E d_i$ for all i, then*

$$R(c_1, \ldots, c_n) \in \Sigma \Leftrightarrow R(d_1, \ldots, d_n) \in \Sigma.$$

ii) Let f be an n-ary function symbol of $\mathcal{L}$ and let $c_1, \ldots, c_n \in C$, there is $d \in C$ such that $f(c_1, \ldots, c_n) = d \in \Sigma$.

iii) Let f be an n-ary function symbol of $\mathcal{L}$ and let $c_0, \ldots, c_n, d_0, \ldots, d_n \in C$ such that $c_i E d_i$ for $i \geq 0$, $f(c_1, \ldots, c_n) = c_0 \in \Sigma$ and $f(d_1, \ldots, d_n) = d_0 \in \Sigma$. Then $c_0 = d_0 \in \Sigma$.

Proof

i) By repeated applications of E2,

$$c_1 = d_1, \ldots, c_n = d_n \vdash \ R(c_1, \ldots, c_n) \leftrightarrow R(d_1, \ldots, d_n)$$

ii) By E1

$$\vdash f(c_1, \ldots, c_n) = f(c_1, \ldots, c_n)$$

where $\phi(v)$ is $f(c_1, \ldots, c_n) = v$. Thus by Q1

$$\vdash \exists v \ f(c_1, \ldots, c_n) = v.$$

Thus $\exists v \ f(c_1, \ldots, c_n) = v$ is in Σ. Since Σ has the witness proterty, there is a constant symbol d such that $f(c_1, \ldots, c_n) = d \in \Sigma$.

iii) By repeated application of E2,

$$c_1 = d_1, \ldots, c_n = d_n, f(c_1, \ldots, c_n) = c_0 \vdash f(d_1, \ldots, d_n) = c_0$$

Thus $\Sigma \vdash f(d_1, \ldots, d_n) = c_0$ and $\Sigma \vdash f(d_1, \ldots, d_n) = d_0$. By the examples in §3, $\Sigma \vdash c_0 = d_0$.

We can now give the interpretation of $\mathcal{L}$ in $\mathcal{A}$.

- The universe of $\mathcal{A}$ is A.
- For each constant symbol c of $\mathcal{L}$, let $c^{\mathcal{A}} = [c]$.
- If R is an n-ary relation symbol let $R^{\mathcal{A}} \subseteq A^n$ be defined by

$$R^{\mathcal{A}} = \{([c_1], \ldots, [c_n]) \in A^n : R(c_1, \ldots, c_n) \in \Sigma\}.$$

By 4.6 i) $R^{\mathcal{A}}$ is well defined.

- If f is an n-ary function symbol define $f^{\mathcal{A}} : A^n \to A$ by

$$f^{\mathcal{A}}([c_1], \ldots, [c_n]) = [d] \Leftrightarrow f(c_1, \ldots, c_n) = d \in \Sigma.$$

By 4.6 ii) and iii) $f^{\mathcal{A}}$ is well defined and $f^{\mathcal{A}} : A^n \to A$.

Lemma 4.7 *Suppose $t(v_1, \ldots, v_n)$ is a term (some of the variables may not occur) and $c_0, \ldots, c_n \in C$ such that $t(c_1, \ldots, c_n) = c_0 \in \Sigma$. If σ is an assignment where $\sigma(v_i) = [c_i]$, then $t^{\mathcal{A}}[\sigma] = [c_0]$. Moreover if $d_0, \ldots, d_n \in C$, $t(d_1, \ldots, d_n) = d_0 \in \Sigma$ and $d_i E c_i$ for $i > 0$, then $c_0 E d_0$.*

Proof The moreover is clear since

$$t(c_1, \ldots, c_n) = c_0, t(d_1, \ldots, d_n) = d_0, c_1 = d_1, \ldots, c_n = d_n \vdash c_0 = d_0$$

so $c_0 = d_0 \in \Sigma$.

The main assertion is proved by induction on the complexity of t.

If t is a constant symbol c, then $t^{\mathcal{A}}[\sigma] = [c]$. Since $c = c_0 \in \Sigma$, $[c] = [c_0]$.

If t is the variable v_i, then $t^{\mathcal{A}}[\sigma] = [c_i]$ and $c_i = c_0 \in \Sigma$, thus $[c_0] = t^{\mathcal{A}}[\sigma]$.

Suppose t is $f(t_1, \ldots, t_m)$ and the claim holds for $t_1, \ldots, t_m$. For each i,

$$\exists w\ t_i(c_1, \ldots, c_n) = w \in \Sigma.$$

Thus since Σ has the witness property, for each i there is $b_i \in C$ such that $t_i(c_1, \ldots, c_n) = b_i \in \Sigma$. By our inductive assumption $t_i^{\mathcal{A}}[\sigma] = [b_i]$. Clearly $t(c_1, \ldots, c_n) = f(b_1, \ldots, b_m) \in \Sigma$, thus $f(b_1, \ldots, b_m) = c_0 \in \Sigma$. But then

$$t^{\mathcal{A}}[\sigma] = f([b_1], \ldots, [b_m]) = [c_0]$$

as desired.

Thus the claim holds for all terms.

Theorem 4.8 *If Σ is a maximal, consistent theory with the witness property and $\mathcal{A}$ is the canonical structure for Σ, then $\mathcal{A} \models \Sigma$.*

Proof

We will prove that for all formulas $\phi(v_1, \ldots, v_n)$ and constants $c_1, \ldots, c_n$,

$$\mathcal{A} \models \phi([c_1], \ldots, [c_n]) \quad \text{if and only if} \quad \phi(c_1, \ldots, c_n) \in \Sigma.$$

This will be proved by induction on the complexity of ϕ.

1) ϕ is $t_1(v_1, \ldots, v_n) = t_2(v_1, \ldots, v_n)$

Since Σ has the witness property there are $d_1, d_2 \in C$ such that $t_i(c_1 \ldots, c_n) = d_i \in \Sigma$. By Lemma 4.7 $t_i([c_1], \ldots, [c_n]) = [d_i]$. Thus

$$\begin{aligned} \mathcal{A} \models t_1([c_1], \ldots, [c_n]) = t_2([c_1], \ldots, [c_n]) &\Leftrightarrow [d_1] = [d_2] \\ &\Leftrightarrow t_1(\overline{c}) = t_2(\overline{c}) \in \Sigma. \end{aligned}$$

2) ϕ is $R(t_1, \ldots, t_m)$ where R is an m-ary relation symbol.

Since Σ has the witness property there are $d_1, \ldots, d_m \in C$ such that $t_i(c_1, \ldots, c_n) = d_i \in \Sigma$. By Lemma 4.7, $t_i([c_1], \ldots, [c_n]) = [d_i]$.

$$\begin{aligned} \mathcal{A} \models \phi([c_1], \ldots, [c_n]) &\Leftrightarrow ([d_1], \ldots, [d_m]) \in R^{\mathcal{A}} \\ &\Leftrightarrow R(d_1, \ldots, d_m) \in \Sigma \\ &\Leftrightarrow R(t_1(\overline{c}), \ldots, t_m(\overline{c})) \in \Sigma. \end{aligned}$$

3) ϕ is $\neg\psi$

Then

$$\begin{aligned} \mathcal{A} \models \phi(\overline{[c]}) &\Leftrightarrow \mathcal{A} \not\models \psi(\overline{[c]}) \\ &\Leftrightarrow \psi(\overline{c}) \notin \Sigma \quad \text{(by induction)} \\ &\Leftrightarrow \phi(\overline{c}) \in \Sigma \text{ since } \Sigma \text{ is maximal.} \end{aligned}$$

4) ϕ is $\psi \vee \theta$

$$\begin{aligned}\mathcal{A} \models \phi(\overline{[c]}) &\Leftrightarrow \mathcal{A} \models \psi(\overline{[c]}) \text{ or } \mathcal{A} \models \theta(\overline{[c_i]})\\ &\Leftrightarrow \psi(\overline{c}) \in \Sigma \text{ or } \theta(\overline{c}) \in \Sigma \text{ by induction}\\ &\Leftrightarrow \phi(\overline{c}) \in \Sigma \text{ by 4.3 ii).}\end{aligned}$$

5) $\phi(\overline{v})$ is $\exists w\ \psi(w, \overline{v})$

If $\mathcal{A} \models \exists w\ \psi(w, \overline{[c]})$, then there is $d \in C$ such that $\mathcal{A} \models \psi([d], \overline{[c]})$. By induction $\psi(d, \overline{c}) \in \Sigma$, and by maximality $\exists w\ \psi(w, \overline{c}) \in \Sigma$.

On the other hand if $\exists w\ \psi(w, \overline{c}) \in \Sigma$, then, since Σ has the witness property, there is $d \in C$, such that $\psi(d, \overline{c}) \in \Sigma$. By induction $\mathcal{A} \models \psi([d], \overline{[c]})$ and $\mathcal{A} \models \phi(\overline{[c]})$.

We have now completed STEP 2. That is, we have shown that if Σ is maximal, consistent theory with the witness property, then there is $\mathcal{A} \models \Sigma$. The Completeness Theorem will now follow from the following result.

Theorem 4.9 *Let Γ be a consistent $\mathcal{L}$-theory. There is $\mathcal{L}^* \supseteq \mathcal{L}$ and $\Sigma \supseteq \Gamma$ a maximal consistent $\mathcal{L}^*$-theory with the witness property.*

Let $\mathcal{L}_0$=$\mathcal{L}$, let C_0 be the constants of $\mathcal{L}$, and let $\Gamma_0 = \Gamma$. Let F_n be the set of all $\mathcal{L}_n$-formulas in one free variable v.

Let $\mathcal{L}_{n+1} = \mathcal{L}_n \cup \{c_\phi : \phi(v) \in F_n\}$, where each c_ϕ is a new constant symbol. For $\phi(v) \in F_n$ let θ_ϕ be the formula

$$(\exists v \phi(v) \ \rightarrow \ \phi(c_\phi)).$$

Let

$$\Gamma_{n+1} = \Gamma_n \cup \{\theta_\phi : \phi \in F_n\},$$

$$\Gamma^* = \bigcup_{n \geq 0} \Gamma_n \text{ and } \mathcal{L}^* = \bigcup_{n \geq 0} \mathcal{L}_n.$$

The following Lemma is the key step to proving the consistency of Γ^*.

Lemma 4.10 *Suppose Δ is a consistent $\mathcal{L}$-theory, $\phi(v)$ is an $\mathcal{L}$-formula with free variable v, c a constant symbol not in $\mathcal{L}$ and θ is the formula*

$$\exists v \phi \rightarrow \phi(c).$$

If ψ is an $\mathcal{L}$-sentence and $\Delta, \theta \vdash \psi$, then $\Delta \vdash \psi$.

In particular, if Δ is consistent, then $\Delta \cup \{\theta\}$ is consistent.

Proof

1.	$\Delta, \neg\exists v\phi(v) \vdash \neg\exists v\phi(v)$	S1
2.	$\Delta, \neg\exists v\phi(v) \vdash \theta$	C3 since θ is $(\neg\exists v\phi(v) \vee \phi(c))$
3.	$\Delta, \theta \vdash \psi$	Premise
4.	$\Delta, \neg\exists v\phi(v), \theta \vdash \psi$	S2
5.	$\Delta, \neg\exists v\phi(v) \vdash \psi$	apply Lemma 3.5 to 2,4
6.	$\Delta, \phi(c) \vdash \phi(c)$	S1
7.	$\Delta, \phi(c) \vdash \theta$	C3 since θ is $(\neg\exists v\phi(v) \vee \phi(c))$
8.	$\Delta, \phi(c), \theta \vdash \psi$	S2 to 2,4
9.	$\Delta, \phi(c) \vdash \psi$	by Lemma 3.5
10.	$\Delta, \exists v\phi(v) \vdash \psi$	Q2 (as c does not occur in ψ)
11.	$\Delta \vdash \psi$	S3 applied to 5,10

Lemma 4.11 *i) If $\Sigma \supseteq \Gamma^*$ is an $\mathcal{L}^*$-theory, then Σ has the witness property.*
ii) Each Γ_n is consistent.
iii) Γ^* is consistent.

Proof
i) For any $\mathcal{L}^*$ formula $\phi(v)$ in one free variable v, there is an n, such that $\phi(v) \in F_n$. Then $(\exists v\phi(v) \rightarrow \phi(c_\phi)) \in \Gamma_{n+1} \subseteq \Sigma$. Thus Σ has the witness property.

ii) We prove this by induction on n. Since $\Gamma_0 = \Gamma$ it is conistent. Suppose Γ_n is consistent, but Γ_{n+1} is inconsistent. Since the proofs of contradictions are finite, there are $\phi_1, \ldots, \phi_m \in F_n$ such that $\Gamma_n, \theta_{\phi_1}, \ldots, \theta_{\phi_m}$ is inconsistent. By choosing m-minimal we may assume that $\Delta = \Gamma_n, \theta_{\phi_1}, \ldots, \theta_{\phi_{m-1}}$ is consistent. By Lemma 4.10 $\Delta \cup \theta_{\phi_m}$ is still consistent, a contradiction.

iii) In general suppose we have consistent theories

$$\Sigma_0 \subseteq \Sigma_1 \subseteq \ldots$$

and $\Sigma = \bigcup_n \Sigma_n$. If Σ is inconsistent, there is ϕ such that $\Sigma \vdash \phi \wedge \neg\phi$. Since the proof of $\phi \wedge \neg\phi$ uses only finitely many premises from Σ, there is an n such that $\Sigma_n \vdash \phi \wedge \neg\phi$, a contradiction.

We have one lemma remaining.

Lemma 4.12 *If Δ is a consistent $\mathcal{L}$-theory, there is a maximal consistent $\mathcal{L}$-theory $\Sigma \supseteq \Delta$.*

If we apply Lemma 4.12 to Γ^* from Lemma 4.11 we obtain a maximal consistent $\Sigma \supseteq \Gamma$ with the witness property.

We first prove Lemma 4.12 in the special case that the language $\mathcal{L}$ is countable. We let $\phi_0, \phi_1, \ldots$ list all $\mathcal{L}$-sentences. We build a sequence of consistent $\mathcal{L}$-theories

$$\Delta = \Delta_0 \subseteq \Delta_1 \subseteq \Delta_2 \subseteq \ldots$$

as follows: We assume that Δ_n is consistent. If $\Delta_n \cup \{\phi_n\}$ is consistent, let $\Delta_{n+1} = \Delta_n \cup \{\phi_n\}$. If not, let $\Delta_{n+1} = \Delta_n \cup \{\neg\phi_n\}$. By Lemma 3.18 iii), Δ_{n+1} is consistent.

Let $\Sigma = \bigcup_n \Delta_n$. As in Lemma 4.11 iii), Σ is a consistent $\mathcal{L}$-theory. For any ϕ, either ϕ or $\neg\phi$ is in Σ. Thus Σ is maximal consistent.

In the general case when $\mathcal{L}$ is uncountable we need to use Zorn's Lemma.

Definition 4.13 Let P be a set and let $<$ be a partial order of P. We say that $X \subseteq P$ is a *chain* if for all $x, y \in X$ $x = y$ or $x < y$ or $x > y$ (ie. $<$ linearly orders X). We say that $z \in P$ is an *upper bound* for X if for all $x \in X$, $x \leq z$. We say that $z \in P$ is *maximal* for $<$ if there is no $z^* \in P$, with $z < z^*$.

Lemma 4.14 (Zorn's Lemma) *Let $(P, <)$ be a partial order such that every chain has an upper bound. Then there is $z \in P$ maximal for $<$.*

Zorn's Lemma is equivalent to the Axiom of Choice.

Proof of Lemma 4.12

Let $P = \{\Gamma \supseteq \Delta : \Gamma \text{ is a consistent } \mathcal{L}\text{-theory}\}$. We order P by $\Gamma_0 < \Gamma_1$ if and only if $\Gamma_0 \subset \Gamma_1$.

Claim If $X \subset P$ is a chain, then X has an upper bound.

Let

$$\Gamma^* = \bigcup_{\Gamma \in X} \Gamma.$$

Clearly for all $\Gamma \in X$, $\Gamma \subseteq \Gamma^*$ thus Γ^* is an upper bound. We need only show that $\Gamma^* \in P$ (ie. Γ^* is consistent).

Suppose Γ^* is inconsistent. Since proofs are finite, there are $\theta_1, \ldots, \theta_m \in \Gamma^*$ such that $\{\theta_1, \ldots, \theta_m\}$ is inconsistent. For each i, there is n_i, such that $\theta_i \in \Gamma_{n_i}$. Since X is a chain, there is $k \leq m$ such that for all i, $\Gamma_{n_i} \subseteq \Gamma_{n_k}$. Thus all $\theta_i \in \Gamma_{n_k}$ and Γ_{n_k} is inconsistent, a contradiction. Hence $\Gamma^* \in P$.

Thus we may apply Zorn's Lemma to obtain $\Sigma \in P$ which is maximal for $<$. Since $\Sigma \in P$, $\Sigma \supseteq \Delta$ and Σ is consistent. Let ϕ be any $\mathcal{L}$-sentence, By 3.18 iii) one of $\Sigma \cup \{\phi\}$ or $\Sigma \cup \{\neg\phi\}$ is consistent. Say $\Sigma \cup \{\phi\}$ is consistent. Then $\phi \in \Sigma$ for otherwise $\Sigma \cup \{\phi\}$ would contradict the maximality of Σ. Thus Σ is maximal.

We can now summarize the proof of the Completeness Theorem. Suppose Γ is a consistent $\mathcal{L}$-theory. By Lemma 4.11 there is $\mathcal{L}^* \supseteq \mathcal{L}$ and $\Gamma^* \supseteq \Gamma$ a consistent $\mathcal{L}^*$-theory such that every $\mathcal{L}^*$-theory extending Γ^* has the witness property. By Lemma 4.12 there is a maximal consistent $\mathcal{L}^*$-theory $\Sigma \supseteq \Gamma^*$. By construction Σ has the witness property. By Theorem 4.8 there is $\mathcal{A} \models \Sigma$. Clearly $\mathcal{A} \models \Gamma$.

Our proof gives some information about the size of the model obtained. For $\mathcal{L}$ any language, $|\mathcal{L}|$ is the cardinality of the set of constant, function and relation symbols of $\mathcal{L}$. The *cardinality of $\mathcal{M}$* is $|M|$, the cardinality of the universe of $\mathcal{M}$.

Corollary 4.15 *Suppose Γ is a consistent $\mathcal{L}$-theory. Then Γ has a model $\mathcal{A} = (A, \ldots)$ with $|A| \leq \max(|\mathcal{L}|, \aleph_0)$.*

Proof The model of Γ that we build above as cardinality at most $|C|$, where C is the set of constant symbols of $\mathcal{L}^*$. We argue inductively that $\mathcal{L}_n$ has at most $|\mathcal{L}| + \aleph_0$ constant symbols. This is because $\mathcal{L}_{n+1}$ has at most one new constant symbol for each $\mathcal{L}_n$-formula. In general if a language has κ symbols, there are $\max(\kappa, \aleph_0)$ possible formulas (formulas are finite strings of symbols).]

5 Basic Model Theory

Our first result is deceptively simple but suprisingly powerful consequence of the Completeness Theorem.

Theorem 5.1 (Compactness Theorem) *Suppose Γ is a set of sentences and every finite subset of Γ is satisfiable. Then Γ is satisfiable. Indeed Γ has a model of cardinality at most $\max(|\mathcal{L}|, \aleph_0)$.*

Proof If Γ is not satisfiable, then, by the Completeness Theorem, Γ is inconsistent. Thus for some ϕ, $\Gamma \vdash \phi$ and $\Gamma \vdash \neg\phi$. But then there is a finite $\Delta \subseteq \Gamma$ such that $\Delta \vdash \phi$ and $\Delta \vdash \neg\phi$. By the Soundness Theorem, Δ is not satisfiable.

Corollary 5.2 *Suppose Γ has arbitrarily large finite models, then Γ has an infinite model.*

Proof Let ϕ_n be the sentence:

$$\exists v_1 \ldots \exists v_n \bigwedge_{i<j\leq n} v_i \neq v_j.$$

Let $\Gamma^* = \Gamma \cup \{\phi_n : n = 1, 2, \ldots\}$. Clearly any model of Γ^* is an infinite model of Γ. If $\Delta \subset \Gamma^*$ is finite, then for some N, $\Delta \subset \Gamma \cup \{\phi_1, \ldots, \phi_N\}$. There is $\mathcal{A} \models \Gamma$ with $|\mathcal{A}| \geq N$, thus $\mathcal{A} \models \Delta$. By the Compactness Theorem, Γ^* has a model.

Corollary 5.3 *Let $\mathcal{L} = \{+, \cdot, 0, 1, <\}$ and let $\mathrm{Th}(\mathbb{N})$, be the complete theory of the natural numbers. There is $\mathcal{A} \models \mathrm{Th}(\mathbb{N})$ with $a \in \mathcal{A}$ infinite.*

Proof Let $\mathcal{L}^* = \mathcal{L} \cup \{c\}$, where c is a new constant symbol. Let $\Gamma = \mathrm{Th}(\mathbb{N}) \cup \{c > 0, c > 1, c > 1 + 1, c > 1 + 1 + 1, \ldots\}$. If $\Delta \subset \Gamma$ is finite, then

$$\Delta \subseteq \mathrm{Th}(\mathbb{N}) \cup \{c > 0, \ldots, c > \underbrace{1 + \ldots + 1}_{N-\text{times}}\}$$

for some N. But then we can find a model of Δ by taking the natural numbers and interpreting c as $N + 1$. Thus by the Compactness Theorem Γ has a model. In this model the interpretation of c is greater that every natural number.

Example: Let $G = (V, E)$ be a graph such that every finite subgraph can be four colored.[2] We claim that G can be four colored. Let $\mathcal{L} = \{R, B, Y, G\} \cup \{c_v : v \in V\}$. Let Γ be the $\mathcal{L}$-theory with axioms:

[2]That is, we can color the vertices with four colors so that no adjacent vertices have the same color. For example, the Four Color Theorem says that every finite planar graph can be four colored.

i) $\forall x\ [(R(x)\wedge\neg B(x)\wedge\neg Y(x)\wedge\neg G(x))\vee\ldots\vee(\neg R(x)\wedge\neg B(x)\wedge\neg Y(x)\wedge G(x))]$

ii) if $(v,w)\in E$ add the axiom: $\neg(R(c_v)\wedge R(c_w))\wedge\ldots\wedge\neg(G(c_v)\wedge G(c_w))$.

If Δ is a finite subset of Γ, let V_Δ be the verticies such that c_v is used in Δ. Since the restriction of G to V_Δ is four colorable, Δ is consistent. Thus Γ is consistent. Let $\mathcal{A}\models\Gamma$.

Color G by coloring v as $\mathcal{A}$ colors c_v.

Theorem 5.4 (Upward Löwenheim–Skolem Theorem) *Suppose Γ is an $\mathcal{L}$-theory. If Γ has an infinite model, then it has a model of cardinality κ for every $\kappa\geq\max(|\mathcal{L}|,\aleph_0)$.*

Proof Let I be a set of cardinality κ. Let $\mathcal{L}^*=\mathcal{L}\cup\{c_\alpha:\alpha\in I\}$. Let

$$\Gamma^*=\Gamma\cup\{c_\alpha\neq c_\beta:\alpha\neq\beta\}.$$

If Δ is a finite subset of Γ^*, then in any infinite model $\mathcal{A}$ of Γ we can interpret the constants such that $\mathcal{A}\models\Delta$. Thus Γ has a model of size at most κ. But certainly any model of Γ^* has size at least κ (the map $\alpha\mapsto\widehat{c}_\alpha$ is one to one).

Definition 5.5 A consistent theory Γ is *complete* if $\Gamma\models\phi$ or $\Gamma\models\neg\phi$ for all $\mathcal{L}$-sentences ϕ.

It is easy to see that Γ is complete if and only if $\mathcal{M}\equiv\mathcal{N}$ for any $\mathcal{M},\mathcal{N}\models\Gamma$. If $\mathcal{M}$ is an $\mathcal{L}$-structure, then $\mathrm{Th}(\mathcal{M})$ is a complete theory, but it may be difficult to figure out if $\phi\in\mathrm{Th}(\mathcal{M})$. We will give one useful test to decide if a theory is complete.

We know from the Löwenheim-Skolem theorem that if a theory has an infinite model it has arbitrarily large models. Thus the theory of an infinite structure can not capture the structure up to isomorphism. Sometimes though knowing the theory and the cardinality determines the structure.

Definition 5.6 Γ is κ-categorical if and only if any two models of Γ of cardinality κ are isomorphic.

• Let $\mathcal{L}$ be the empty language. Then the theory of an infinite set is κ-categorical for all cardinals κ.

• Let $\mathcal{L}=\{E\}$, where E is a binary relation, and let T be the theory of an equivalence relation with exactly two classes, both of which are infinite. It is easy to see that any two countable models of T are isomorphic. On the other hand, T is not κ-categorical for $\kappa>\aleph_0$. To see this, let $\mathcal{M}_0$ be a model where both classes have cardinality κ, and let $\mathcal{M}_1$ be a model where one class has cardinality κ and the other has cardinality $\aleph_0$. Clearly, $\mathcal{M}_0$ and $\mathcal{M}_1$ are not isomorphic.

Let $\mathcal{L}=\{+,0\}$ be the language of additive groups and let T be the $\mathcal{L}$-theory of nontrivial torsion-free divisible Abelian groups. The axioms of T are the axioms for Abelian groups together with the axioms

$$\exists x\ x\neq 0,$$

$$\forall x(x \neq 0 \rightarrow \underbrace{x + \ldots + x}_{n-\text{times}} \neq 0)$$

and

$$\forall y \exists x \ \underbrace{x + \ldots + x}_{n-\text{times}} = y$$

for $n = 1, 2, \ldots$.

Proposition 5.7 *The theory of torsion-free divisible Abelian groups is κ-categorical for all $\kappa > \aleph_0$.*

Proof We first argue that models of T are essentially vector spaces over the field of rational numbers $\mathbb{Q}$. Clearly, if V is any vector space over $\mathbb{Q}$, then the underlying additive group of V is a model of T. On the other hand, if $G \models T$, $g \in G$, and $n \in \mathbb{N}$ with $n > 0$, we can find $h \in G$ such that $nh = g$. If $nk = g$, then $n(h - k) = 0$. Because G is torsion-free there is a unique $h \in G$ such that $nh = g$. We call this element g/n. We can view G as a $\mathbb{Q}$-vector space under the action $\frac{m}{n} g = m(g/n)$.

Two $\mathbb{Q}$-vector spaces are isomorphic if and only if they have the same dimension. Thus, models of T are determined up to isomorphism by their dimension. If G has dimension λ, then $|G| = \lambda + \aleph_0$. If κ is uncountable and G has cardinality κ, then G has dimension κ. Thus, for $\kappa > \aleph_0$ any two models of T of cardinality κ are isomorphic.

Note that T is not $\aleph_0$-categorical. Indeed, there are $\aleph_0$ nonisomorphic models corresponding to vector spaces of dimension $1, 2, 3, \ldots$ and $\aleph_0$.

A similar argument applies to the theory of algebraically closed fields. Let ACF_p be the theory of algebraically closed fields of characteristic p, where p is either 0 or a prime number.

Proposition 5.8 ACF_p *is κ-categorical for all uncountable cardinals κ.*

Proof Two algebraically closed fields are isomorphic if and only if they have the same characteristic and transcendence degree (see, for example Lang's *Algebra* X §1). An algebraically closed field of transcendence degree λ has cardinality $\lambda + \aleph_0$. If $\kappa > \aleph_0$, an algebraically closed field of cardinality κ also has transcendence degree κ. Thus, any two algebraically closed fields of the same characteristic and same uncountable cardinality are isomorphic.

Theorem 5.9 (Vaught's Test) *Suppose every model of Γ is infinite, $\kappa \geq \max(|\mathcal{L}|, \aleph_0)$ and Γ is κ-categorical. Then Γ is complete.*

Proof Suppose not. Let ϕ be an $\mathcal{L}$-sentence such that $\Gamma \not\models \phi$ and $\Gamma \not\models \neg\phi$. Let $\Gamma_0 = \Gamma \cup \{\phi\}$ and $\Gamma_1 = \Gamma \cup \{\neg\phi\}$. Each Γ_i has a model, thus since Γ has only infinite models, each Γ_i has an infinite model. By the Löwenheim-Skolem theorem there is $\mathcal{A}_i \models \Gamma_i$ where $\mathcal{A}_i$ has cardinality κ. Since Γ is κ-categorical, $\mathcal{A}_0 \cong \mathcal{A}_1$ and hence by 1.16, $\mathcal{A}_0 \prec \mathcal{A}_1$. But $\mathcal{A}_0 \models \phi$ and $\mathcal{A}_1 \models \neg\phi$, a contradiction.

The assumption that T has no finite models is necessary. Suppose that T is the $\{+, 0\}$-theory of Abelian groups, where every element has order 2. In the exercises, we will show that T is κ-categorical for all $\kappa \geq \aleph_0$. However, T is not complete. The sentence $\exists x \exists y \exists z\ (x \neq y \wedge y \neq z \wedge z \neq x)$ is false in the two-element group but true in every other model of T.

Vaught's Test implies that all of the categorical theories discussed above are complete. In particular, the theory of algebraically closed fields of a fixed characteristic is complete. This result of Tarski has several immediate interesting consequences.

The next definition is, for the moment, imprecise. In later chapters we will make the concepts precise.

Definition 5.10 We say that an $\mathcal{L}$-theory T is *decidable* if there is an algorithm that when given an $\mathcal{L}$-sentence ϕ as input decides whether $T \models \phi$.

Lemma 5.11 *Let T be a recursive complete satisfiable theory in a recursive language $\mathcal{L}$. Then T is decidable.*

Proof Start enumerating all finite sequence of strings of $\mathcal{L}$-symbols. For each one, check to see if it is a derivation of $\Delta \vdash \phi$ or $\Delta \vdash \neg\phi$. If it is then check to see if all of the sentences in Δ are in Γ. If so output yes if $\Delta \vdash \phi$ and no if $\Delta \vdash \neg\phi$. If not, goon to the next string. Since Γ is complete, the Completeness Theorem implies there is a finite $\Delta \subseteq \Gamma$ such that $\Delta \vdash \phi$ or $\Delta \vdash \neg\phi$. Thus our search will halt at some stage.

Informally, to decide whether ϕ is a logical consequence of a complete satisfiable recursive theory T, we begin searching through possible proofs from T until we find either a proof of ϕ or a proof of $\neg\phi$. Because T is satisfiable, we will not find proofs of both. Because T is complete, we will eventually find a proof of one or the other.

Corollary 5.12 *For $p = 0$ or p prime, ACF_p is decidable. In particular, $\mathrm{Th}(\mathbb{C})$, the first-order theory of the field of complex numbers, is decidable.*

The completeness of ACF_p can also be thought of as a first-order version of the Lefschetz Principle from algebraic geometry.

Corollary 5.13 *Let ϕ be a sentence in the language of rings. The following are equivalent.*

i) ϕ is true in the complex numbers.

ii) ϕ is true in every algebraically closed field of characteristic zero.

iii) ϕ is true in some algebraically closed field of characteristic zero.

iv) There are arbitrarily large primes p such that ϕ is true in some algebraically closed field of characteristic p.

v) There is an m such that for all $p > m$, ϕ is true in all algebraically closed fields of characteristic p.

Proof The equivalence of i)–iii) is just the completeness of ACF_0 and v)$\Rightarrow$ iv) is obvious.

For ii) $\Rightarrow$ v) suppose that $\mathrm{ACF}_0 \models \phi$. There is a finite $\Delta \subset \mathrm{ACF}_0$ such that $\Delta \vdash \phi$. Thus, if we choose p large enough, then $\mathrm{ACF}_p \models \Delta$. Thus, $\mathrm{ACF}_p \models \phi$ for all sufficiently large primes p.

For iv) $\Rightarrow$ ii) suppose $\mathrm{ACF}_0 \not\models \phi$. Because ACF_0 is complete, $\mathrm{ACF}_0 \models \neg\phi$. By the argument above, $\mathrm{ACF}_p \models \neg\phi$ for sufficiently large p; thus, iv) fails.

Ax found the following striking application of Corollary 5.13.

Theorem 5.14 *Every injective polynomial map from $\mathbb{C}^n$ to $\mathbb{C}^n$ is surjective.*

Proof Remarkably, the key to the proof is the simple observation that if k is a finite field, then every injective function $f : k^n \to k^n$ is surjective. From this observation it is easy to show that the same is true for $\mathbb{F}_p^{\mathrm{alg}}$, the algebraic closure of the p-element field.

Claim Every injective polynomial map $f : (\mathbb{F}_p^{\mathrm{alg}})^n \to (\mathbb{F}_p^{\mathrm{alg}})^n$ is surjective.

Suppose not. Let $\overline{a} \in \mathbb{F}_p^{\mathrm{alg}}$ be the coefficients of f and let $\overline{b} \in (\mathbb{F}_p^{\mathrm{alg}})^n$ such that $\overline{b}$ is not in the range of f. Let k be the subfield of $\mathbb{F}_p^{\mathrm{alg}}$ generated by $\overline{a}, \overline{b}$. Then $f|k^n$ is an injective but not surjective polynomial map from k^n into itself. But $\mathbb{F}_p^{\mathrm{alg}} = \bigcup_{n=1}^{\infty} \mathbb{F}_{p^n}$ is a locally finite field. Thus k is finite, a contradiction.

Suppose that the theorem is false. Let $X = (X_1, \ldots, X_n)$. Let $f(X) = (f_1(X), \ldots, f_n(X))$ be a counterexample where each $f_i \in \mathbb{C}[X]$ has degree at most d. There is an $\mathcal{L}$-sentence $\Phi_{n,d}$ such that for K a field, $K \models \Phi_{n,d}$ if and only if every injective polynomial map from K^n to K^n where each coordinate function has degree at most d is surjective. We can quantify over polynomials of degree at most d by quantifying over the coefficients. For example, $\Phi_{2,2}$ is the sentence

$$\forall a_{0,0}\forall a_{0,1}\forall a_{0,2}\forall a_{1,0}\forall a_{1,1}\forall a_{2,0}\forall b_{0,0}\forall b_{0,1}\forall b_{0,2}\forall b_{1,0}\forall b_{1,1}\forall b_{2,0}$$
$$\Big[\big(\forall x_1\forall y_1\forall x_2\forall y_2((\textstyle\sum a_{i,j}x_1^i y_1^j = \sum a_{i,j}x_2^i y_2^j \wedge \sum b_{i,j}x_1^i y_1^j = \sum b_{i,j}x_2^i y_2^j) \rightarrow$$
$$(x_1 = x_2 \wedge y_1 = y_2))\big) \rightarrow \forall u\forall v\exists x\exists y \textstyle\sum a_{i,j}x^i y^j = u \wedge \sum b_{i,j}x^i y^j = v\Big].$$

By the claim $\mathbb{F}_p^{\mathrm{alg}} \models \Phi_{n,d}$ for all primes p. By Corollary 5.13, $\mathbb{C} \models \Phi_{n,d}$, a contradiction.

Back-and-Forth

We give two examples of $\aleph_0$-categorical theories. The proofs use the "back-and-forth" method, a style of argument that has many interesting applications. We start with Cantor's proof that any two countable dense linear orders are isomorphic.

Let $\mathcal{L} = \{<\}$ and let DLO be the theory of dense linear orders without endpoints. DLO is axiomatized by the axioms for linear orders plus the axioms

$$\forall x\forall y\ (x < y \rightarrow \exists z\ x < z < y)$$

and

$$\forall x\exists y\exists z\ y < x < z.$$

Theorem 5.15 *The theory DLO is $\aleph_0$-categorical and complete.*

Proof Let $(A,<)$ and $(B,<)$ be two countable models of DLO. Let $a_0, a_1, a_2, \dots$ and $b_0, b_1, b_2, \dots$ be one-to-one enumerations of A and B. We will build a sequence of partial bijections $f_i : A_i \to B_i$ where $A_i \subset A$ and $B_i \subset B$ are finite such that $f_0 \subseteq f_1 \subseteq \dots$ and if $x, y \in A_i$ and $x < y$, then $f_i(x) < f_i(y)$. We call f_i a *partial embedding*. We will build these sequences such that $A = \bigcup A_i$ and $B = \bigcup B_i$. In this case, $f = \bigcup f_i$ is the desired isomorphism from $(A,<)$ to $(B,<)$.

At odd stages of the construction we will ensure that $\bigcup A_i = A$, and at even stages we will ensure that $\bigcup B_i = B$.

<u>stage 0</u>: Let $A_0 = B_0 = f_0 = \emptyset$.

<u>stage $n+1 = 2m+1$</u>: We will ensure that $a_m \in A_{n+1}$.

If $a_m \in A_n$, then let $A_{n+1} = A_n, B_{n+1} = B_n$ and $f_{n+1} = f_n$. Suppose that $a_m \notin A_n$. To add a_m to the domain of our partial embedding, we must find $b \in B \setminus B_n$ such that

$$\alpha < a_m \Leftrightarrow f_n(\alpha) < b$$

for all $\alpha \in A_n$. In other words, we must find $b \in B$, which is in the image under f_n of the cut of a_m in A_n. Exactly one of the following holds:

i) a_m is greater than every element of A_n, or

ii) a_m is less than every element of A_n, or

iii) there are α and $\beta \in A_n$ such that $\alpha < \beta$, $\gamma \le \alpha$ or $\gamma \ge \beta$ for all $\gamma \in A_n$ and $\alpha < a_m < \beta$.

In case i) because B_n is finite and $B \models$ DLO, we can find $b \in B$ greater than every element of B_n. Similarly in case ii) we can find $b \in B$ less than every element of B_n. In case iii), because f_n is a partial embedding, $f_n(\alpha) < f_n(\beta)$ and we can choose $b \in B \setminus B_n$ such that $f_n(\alpha) < b < f_n(\beta)$. Note that

$$\alpha < a_m \Leftrightarrow f_n(\alpha) < b$$

for all $\alpha \in A_n$.

In any case, we let $A_{n+1} = A_n \cup \{a_m\}$, $B_{n+1} = B_n \cup \{b\}$, and extend f_n to $f_{n+1} : A_{n+1} \to B_{n+1}$ by sending a_m to b. This concludes stage n.

<u>stage $n+1 = 2m+2$</u>: We will ensure that $b_m \in B_{n+1}$.

Again, if b_m is already in B_n, then we make no changes and let $A_{n+1} = A_n, B_{n+1} = B_n$ and $f_{n+1} = f_n$. Otherwise, we must find $a \in A$ such that the image of the cut of a in A_n is the cut of b_m in B_n. This is done as in the odd case.

Clearly, at odd stages we have ensured that $\bigcup A_n = A$ and at even stages we have ensured that $\bigcup B_n = B$. Because each f_n is a partial embedding, $f = \bigcup f_n$ is an isomorphism from A onto B.

Because there are no finite dense linear orders, Vaught's test implies that DLO is complete.

The proof of Theorem 5.15 is an example of a *back-and-forth* construction. At odd stages, we go forth trying to extend the domain, whereas at even stages we go back trying to extend the range. We give another example of this method.

The Random Graph

Let $\mathcal{L} = \{R\}$, where R is a binary relation symbol. We will consider an $\mathcal{L}$-theory containing the graph axioms $\forall x\ \neg R(x,x)$ and $\forall x \forall y\ R(x,y) \to R(y,x)$. Let ψ_n be the "extension axiom"

$$\forall x_1 \ldots \forall x_n \forall y_1 \ldots \forall y_n \left(\bigwedge_{i=1}^{n} \bigwedge_{j=1}^{n} x_i \neq y_j \to \exists z \bigwedge_{i=1}^{n} (R(x_i,z) \wedge \neg R(y_i,z)) \right).$$

We let T be the theory of graphs where we add $\{\exists x \exists y\ x \neq y\} \cup \{\psi_n : n = 1,2,\ldots\}$ to the graph axioms. A model of T is a graph where for any finite disjoint sets X and Y we can find a vertex with edges going to every vertex in X and no vertex in Y.

Theorem 5.16 *T is satisfiable and $\aleph_0$-categorical. In particular, T is complete and decidable.*

Proof We first build a countable model of T. Let G_0 be any countable graph.

Claim There is a graph $G_1 \supset G_0$ such that G_1 is countable and if X and Y are disjoint finite subsets of G_0 then there is $z \in G_1$ such that $R(x,z)$ for $x \in X$ and $\neg R(y,z)$ for $y \in Y$.

Let the vertices of G_1 be the vertices of G_0 plus new vertices z_X for each finite $X \subseteq G_0$. The edges of G_1 are the edges of G together with new edges between x and z_X whenever $X \subseteq G_0$ is finite and $x \in X$. Clearly, G_1 is countable and has the desired property.

We iterate this construction to build a sequence of countable graphs $G_0 \subset G_1 \subset \ldots$ such that if X and Y are disjoint finite subsets of G_i, then there is $z \in G_{i+1}$ such that $R(x,z)$ for $x \in X$ and $\neg R(y,z)$ for $y \in Y$. Then, $G = \bigcup G_n$ is a countable model of T.

Next we show that T is $\aleph_0$-categorical. Let G_1 and G_2 be countable models of T. Let $a_0, a_1, \ldots$ list G_1, and let $b_0, b_1, \ldots$ list G_2. We will build a sequence of finite partial one-to-one maps $f_0 \subseteq f_1 \subseteq f_2 \subseteq \ldots$ such that for all x, y in the domain of f_s,

$$G_1 \models R(x,y) \quad \text{if and only if} \quad G_2 \models R(f_s(x), f_s(y)). \qquad (*)$$

Let $f_0 = \emptyset$.

<u>stage $s+1 = 2i+1$</u>: We make sure that a_i is in the domain.

If a_i is in the domain of f_s, let $f_{s+1} = f_s$. If not, let $\alpha_1, \ldots, \alpha_m$ list the domain of f_s and let $X = \{j \leq m : R(\alpha_j, a_i)\}$ and let $Y = \{j \leq m : \neg R(\alpha_j, a_i)\}$. Because $G_2 \models T$, we can find $b \in G_2$ such that $G_2 \models R(f_s(\alpha_j), b)$ for $j \in X$ and $G_2 \models \neg R(f_s(\alpha_j), b)$ for $j \in Y$. Let $f_{s+1} = f_s \cup \{(a_i, b)\}$. By choice of b and induction, f_{s+1} satisfies $(*)$.

<u>stage $s+1 = 2i+2$</u>: By a similar argument, we can ensure that f_{s+1} satisfies $(*)$ and b_i is in the image of f_{s+1}.

Let $f = \bigcup f_s$. We have ensured that f maps G_1 onto G_2. By $(*)$, f is a graph isomorphism. Thus, $G_1 \cong G_2$ and T is $\aleph_0$-categorical.

Because all models of T are infinite, T is complete. Because T is recursively axiomatized, T is decidable.

The theory T is very interesting because it gives us insights into random finite graphs. Let $\mathcal{G}_N$ be the set of all graphs with vertices $\{1, 2, \ldots, N\}$. We consider a probability measure on $\mathcal{G}_N$ where we make all graphs equally likely. This is the same as constructing a random graph where we independently decide whether there is an edge between i and j with probability $\frac{1}{2}$. For any $\mathcal{L}$-sentence ϕ,

$$p_N(\phi) = \frac{|\{G \in \mathcal{G}_N : G \models \phi\}|}{|\mathcal{G}_N|}$$

is the probability that a random element of $\mathcal{G}_N$ satisfies ϕ.

We argue that large graphs are likely to satisfy the extension axioms.

Lemma 5.17 $\lim\limits_{N\to\infty} p_N(\psi_n) = 1$ *for* $n = 1, 2, \ldots$.

Proof Fix n. Let G be a random graph in $\mathcal{G}_N$ where $N > 2n$. Fix $x_1, \ldots, x_n, y_1, \ldots, y_n, z \in G$ distinct. Let q be the probability that

$$\neg\left(\bigwedge_{i=1}^{n} (R(x_i, z) \wedge \neg R(y_i, z))\right).$$

Then $q = 1 - 2^{-2n}$. Because these probabilities are independent, the probability that

$$G \models \neg\exists z \neg\left(\bigwedge_{i=1}^{n} (R(x_i, z) \wedge \neg R(y_i, z))\right)$$

is q^{N-2n}. Let M be the number of pairs of disjoint subsets of G of size n. Thus

$$p_N(\neg\psi_n) \le M q^{N-2n} < N^{2n} q^{N-2n}.$$

Because $q < 1$,

$$\lim_{N\to\infty} p_N(\neg\psi_n) = \lim_{N\to\infty} N^{2n} q^N = 0,$$

as desired.

We can now use the fact that T is complete to get a good understanding of the asymptotic properties of random graphs.

Theorem 5.18 (Zero-One Law for Graphs) *For any $\mathcal{L}$-sentence ϕ either* $\lim\limits_{N\to\infty} p_N(\phi) = 0$ *or* $\lim\limits_{N\to\infty} p_N(\phi) = 1$. *Moreover, T axiomatizes* $\{\phi : \lim\limits_{N\to\infty} p_N(\phi) = 1\}$, *the* almost sure theory of graphs. *The almost sure theory of graphs is decidable and complete.*

Proof If $T \models \phi$, then there is n such that if G is a graph and $G \models \psi_n$, then $G \models \phi$. Thus, $p_N(\phi) \ge p_N(\psi_n)$ and by Lemma 5.17, $\lim\limits_{N\to\infty} p_N(\phi) = 1$. On the other hand, if $T \not\models \phi$, then, because T is complete, $T \models \neg\phi$ and $\lim\limits_{N\to\infty} p_N(\neg\phi) = 1$ so $\lim\limits_{N\to\infty} p_N(\phi) = 0$.

Ultraproducts

Let I be an infinite set. We let

$$\mathcal{P}(I) = \{A : A \subseteq I\}$$

be the *power set* of I.

Definition 5.19 We say that $\mathcal{F} \subseteq \mathcal{P}(I)$ is a *filter* if
i) $I \in \mathcal{F}$, $\emptyset \notin \mathcal{F}$;
ii) If $A \in \mathcal{F}$ and $A \subseteq B$, then $B \in \mathcal{F}$;
iii) If $A, B \in \mathcal{F}$ then $A \cap B \in \mathcal{F}$.

We say that $\mathcal{F}$ is an *ultrafilter* if in addition,
iv) for all $A \subseteq I$ either $A \in \mathcal{F}$ or $I \setminus A \in \mathcal{F}$.

Example 5.20 *Cof* $= \{A \subseteq I : I \setminus A$ *is finite*$\}$ *is a filter.*

Example 5.21 *Let* $I = \mathbb{R}$ *then* $\mathcal{F} = \{A : \mathbb{R} \setminus A$ *has Lebesgue measure zero*$\}$, *is a filter.*

Lemma 5.22 *If* $\mathcal{F} \subseteq \mathcal{P}(I)$ *is a filter,* $A \subseteq I$ *and* $I \setminus A \notin \mathcal{F}$, *then*

$$\mathcal{F}' = \{C : \text{ there is } B \in \mathcal{F}, C \supseteq A \cap B\}$$

is an ultrafilter. Note that $A \in \mathcal{F}'$.

Proof Since $I \supseteq I \cap A$, $I \in \mathcal{F}'$.

If $\emptyset \in \mathcal{F}'$, then there is $B \in \mathcal{F}$ such that $A \cap B = \emptyset$. But then $B \subseteq I \setminus A$ and $I \setminus A \in \mathcal{F}$, a contradiction.

It is easy to see that $\mathcal{F}'$ is closed under superset.

If $C_1, C_2 \in \mathcal{F}'$ there are $B_1, B_2 \in \mathcal{F}$ such that $C_i \supseteq B_i \cap A$. Then $C_1 \cap C_2 \supseteq B_1 \cap B_2 \cap A$, so $C_1 \cap C_2 \in \mathcal{F}'$.

Corollary 5.23 *If* $\mathcal{F} \subseteq \mathcal{P}(I)$ *is a filter, then there is an ultrafilter* $\mathcal{U} \supseteq \mathcal{F}$.

Proof Let $\mathcal{I} = \{\mathcal{F}' : \mathcal{F} \subseteq \mathcal{F}' \subseteq \mathcal{P}(I)$ is a filter$\}$.

If $(X, <)$ is a linearly ordered set, $\mathcal{F}_x \in \mathcal{I}$ for $x \in X$ and $\mathcal{F}_x \subseteq \mathcal{F}_y$ for $x < y$, then $\mathcal{F}^* = \bigcup_{x \in X} \mathcal{F}_x$ is a filter. Thus we can apply Zorn's Lemma to find $\mathcal{U} \in \mathcal{I}$ maximal. Suppose $A \subseteq I$. If $I \setminus A \notin \mathcal{U}$, then, by the Lemma and the maximality of $\mathcal{U}$, $A \in U$.

Corollary 5.24 *There are non-principal ultrafilters.*

Proof Let $\mathcal{U} \supseteq$ Cof be an ultrafilter. Then $\mathcal{U}$ contains no finite sets.

Our proof of the existence of non-priniciple ultrafilters is non-constructive as it depends heavily on the Axiom of Choice. Unfortunately, some use of choice is unavoidable.

We will use ultrafilters to give a new construction of models. Let $\mathcal{L}$ be a first order language. Suppose that $\mathcal{M}_i$ is an $\mathcal{L}$-structure for all $i \in I$ with universe M_i. Let $\mathcal{U} \subseteq \mathcal{P}(\mathcal{I})$ be an ultrafilter.

We define $\sim$ on $\prod_{i\in I} M_i$ by

$$f \sim g \Leftrightarrow \{i \in I : f(i) = g(i)\} \in \mathcal{U}.$$

Lemma 5.25 *$\sim$ is an equivalence relation*

Proof Let $f, g, h \in \prod_{i\in I} M_i$. Clearly $f \sim f$ and if $f \sim g$, then $g \sim f$.

Suppose $f \sim g$ and $g \sim h$. Since

$$\{i : f(i) = h(i)\} \supseteq \{i : f(i) = g(i)\} \cap \{i : g(i) = h(i)\} \in \mathcal{U},$$

$f \sim h$.

For $f \in \prod_{i\in I}$, let $[f]$ be the $\sim$-equivalence class of f and let

$$M = \left\{[f] : f \in \prod_{i\in I} M_i\right\}.$$

We will interpret the symbols of $\mathcal{L}$ in M to construct an $\mathcal{L}$-structure $\mathcal{M}$, which we also denote $\prod M_i/\mathcal{U}$.

If c is a constant symbol of $\mathcal{L}$, let $f \in \prod M_i$ be the function $f(i) = c^{\mathcal{M}_i}$ and let $c^{\mathcal{M}} = [f]$.

Let R be an n-ary relation symbol of $\mathcal{L}$.

Lemma 5.26 *$f_1, \ldots, f_n, g_1, \ldots, g_n \in \prod M_i$ such that $f_j \sim g_j$ for all $j = 1, \ldots, n$. Then*

$$\{i \in I : (f_1(i), \ldots, f_n(i)) \in R^{\mathcal{M}_i}\} \in \mathcal{U} \Leftrightarrow \{i \in I : (g_1(i), \ldots, g_n(i)) \in R^{\mathcal{M}_i}\} \in \mathcal{U}.$$

Proof Suppose $\{i \in I : (f_1(i), \ldots, f_n(i)) \in R^{\mathcal{M}_i}\} \in U$. Then $\{i \in I : (g_1(i), \ldots, g_n(i)) \in R^{\mathcal{M}_i}\}$ contains

$$\{i \in I : (f_1(i), \ldots, f_n(i)) \in R^{\mathcal{M}_i}\} \cap \{i \in I : g_1(i) = f_1(i)\} \cap \ldots \cap \{i \in I : g_n(i) = f_n(i)\}.$$

Since $\mathcal{U}$ is a filter this later set is in $\mathcal{U}$.

The other direction is symmetric.

We define

$$R^{\mathcal{M}} = \{([f_1], \ldots, [f_n]) : \{i \in I : (f_1(i), \ldots, f_n(i)) \in \mathbb{R}^{\mathcal{M}_i}\} \in \mathcal{U}\}.$$

By the Lemma, this is well-defined and does not depend on the choice of representatives for the equivalence classes.

Let F be an n-ary function symbol of $\mathcal{L}$. Let $f_1, \ldots, f_n, g_1, \ldots, g_n \in \prod M_i$ with $f_j \sim g_j$ for $j = 1, \ldots, n$. Define $f_{n+1}, g_{n+1} \in \prod M_i$ by

$$f_{n+1}(i) = F(f_1(i), \ldots, f_n(i)) \text{ and } g_{n+1}(i) = F(g_1(i), \ldots, g_n(i)).$$

Exercise 5.27 Argue as in Lemma 5.26 that $f_{n+1} \sim g_{n+1}$.

We define $F^{\mathcal{M}} : M^n \to M$ by

$$F([f_1], \ldots, [f_n]) = [g]$$

where $g(i) = F(f_1(i), \ldots, f_n(i))$. By Exercise 5.27 this is well defined and does not depend on choice of representatives.

We have now completely defined the structure $\mathcal{M} = \prod M_i/U$. We call $\mathcal{M}$ an *ultraproduct* of $(\mathcal{M}_i : i \in I)$

The following exercise is an easy induction on terms.

Exercise 5.28 If t is an $\mathcal{L}$-term, then $t^{\mathcal{M}}(f_1, \ldots, f_n) = [g]$ where $g(i) = t^{\mathcal{M}_i}(f_1(i), \ldots, f_n(i))$.

We can now state the Fundamental Theorem of Ultraproducts.

Theorem 5.29 (łos's Theorem) *Let $\phi(v_1, \ldots, v_n)$ be any $\mathcal{L}$-formula Then*

$$\mathcal{M} \models \phi([f_1], \ldots, [f_n]) \Leftrightarrow \{i : \mathcal{M}_i \models \phi(f_1(i), \ldots, f_n(i))\} \in \mathcal{U}.$$

Proof We prove this by induction on complexity of formulas

1) Suppose ϕ is $t_1 = t_2$ where t_1 and t_2 are terms.

Define $g_j(i) = t_j^{\mathcal{M}_i}(f_1(i), \ldots, f_n(i))$. Then

$$\mathcal{M} \models t_1([f_1], \ldots, [f_n]) = t_2([f_1], \ldots, [f_n]) \Leftrightarrow [g_1] = [g_2]$$

$$\Leftrightarrow \{i : t_1^{M_i}(f_1(i), \ldots, f_n(i)) = t_2^{\mathcal{M}_i}(f_1(i), \ldots, f_n(i)\} \in \mathcal{U}$$

as desired.

2) Suppose ϕ is $R(t_1, \ldots, t_m)$.

For $j = 1, \ldots, m$ let $g_j(i) = t_i^{\mathcal{M}_i}(f_1(i), \ldots, f_n(i))$. Then

$$\begin{aligned} \mathcal{M} \models \phi([f_1], \ldots, [f_n]) &\Leftrightarrow \{i : (g_1(i), \ldots, g_n(i)) \in R^{\mathcal{M}_i}\} \in \mathcal{U} \\ &\Leftrightarrow \{i : \mathcal{M}_i \models \phi(f_1(i), \ldots, f_n(i))\} \in \mathcal{U} \end{aligned}$$

3) Suppose the theorem is true for θ and ψ, and ϕ is $\theta \wedge \psi$. (We suppress the parameters $[f_1], \ldots, [f_n]$)

Then

$$\begin{aligned} \mathcal{M} \models \phi &\Leftrightarrow \mathcal{M} \models \psi \text{ and } \mathcal{M} \models \theta \\ &\Leftrightarrow \{i : \mathcal{M}_i \models \psi\} \in \mathcal{U} \text{ and } \{i : \mathcal{M}_i \models \psi\} \in \mathcal{U} \\ &\Leftrightarrow \{i : \mathcal{M}_i \models \psi \wedge \theta\} \in \mathcal{U} \end{aligned}$$

4) Suppose the theorem is true for ψ and ϕ is $\neg\psi$ Then

$$\begin{aligned} \mathcal{M} \models \phi &\Leftrightarrow \mathcal{M} \not\models \psi \\ &\Leftrightarrow \{i : \mathcal{M}_i \models \psi\} \notin \mathcal{U} \\ &\Leftrightarrow \{i : \mathcal{M}_i \models \neg\psi\} \in \mathcal{U} \end{aligned}$$

5) Suppose the theorem is true for $\psi(v)$ and ϕ is $\exists v\ \psi(v)$.

If $\mathcal{M} \models \exists v\ \psi(v)$, then there is g such that $\mathcal{M} \models \psi([g])$. But then

$$\{i : \mathcal{M}_i \models \exists v\ \psi(v)\} \supseteq \{i : \mathcal{M}_i \models \psi(g(i))\} \in \mathcal{U}$$

On the other hand if $A = \{i : \mathcal{M}_i \models \exists v\ \psi(v)\} \in \mathcal{U}$ define $g \in \prod M_i$ such that $\mathcal{M}_i \models \psi(g(i))$ for all $i \in A$. Then $\mathcal{M} \models \psi([g])$, so $\mathcal{M} \models \phi$.

Note that step 4) is the only place in the construction that we used that $\mathcal{U}$ is an ultrafilter rather than just a filter.

Exercise 5.30 Let $\mathcal{U}$ be a non-princpal ultrafilter on the set of prime numbers. For each prime p, let $\mathbb{F}_p^{\text{alg}}$ be the algebraic closure of $\mathbb{F}_p$ the field with p elements. Prove that $\prod \mathbb{F}_p/\mathcal{U}$ is an algebraically closed field of characteristic 0.

Another Proof of Compactness

We can use łos's Theorem to give a proof of the Compactness Theorem that avoids Henkin arguments and the Completeness Theorem.

Let Γ be an $\mathcal{L}$-theory such that every finite $\Delta \subseteq \Gamma$ has a model. Let I be the collection of finite subsets of Γ.

For $\phi \in \Gamma$ let

$$X_\phi = \{\Delta \in I : \Delta \models \phi\}$$

and let

$$\mathcal{F} = \{Y \subseteq I : X_\phi \subseteq Y \text{ for some } \phi \in \Gamma\}.$$

We claim that $\mathcal{F}$ is a filter. It is easy to see that $I \in \mathcal{F}$, $\emptyset \notin \mathcal{F}$ and $\mathcal{F}$ is closed under superset. Also if $Y_1, Y_2 \in \mathcal{F}$ there are ϕ_1, ϕ_2 such that $X_{\phi_i} \subseteq Y_i$. Then $X_{\phi_1 \wedge \phi_2} = X_{\phi_1} \cap X_{\phi_2}$, so

$$X_{\phi_1 \wedge \phi_2} \subseteq Y_1 \cap Y_2$$

and $Y_1 \cap Y_2 \in \mathcal{F}$

Let $\mathcal{U} \supseteq \mathcal{F}$ be an ultrafilter. For $\Delta \in I$, let $\mathcal{M}_\Delta \models \Delta$ and let $\mathcal{M} = \prod \mathcal{M}_\Delta/\mathcal{U}$. Since $X_\phi \in \mathcal{U}$ for all $\phi \in \Gamma$, by łos's Theorem $\mathcal{M} \models \Gamma$.

Ultrapowers and Elementary Extensions

Fix $\mathcal{M}$ and $\mathcal{L}$ structure and let $\mathcal{U}$ be an ultrafilter on an infinite set I. An interesting special case of the ultraproduct construction is when we take all of the $\mathcal{M}_i = \mathcal{M}$. In this case we let $\mathcal{M}^* = \mathcal{M}^I/U$.

Exercise 5.31 Prove that if $\mathcal{M}$ is finite or $\mathcal{U}$ is principal, then $\mathcal{M} \cong \mathcal{M}^*$.

For each $a \in M$, let $f_a : I \to M$ be the constant function $f_a(i) = a$. If $a \neq b$, then $[f_a] \neq [f_b]$. By łos's Theorem if $a_1, \ldots, a_n \in \mathcal{M}$ and ϕ is an $\mathcal{L}$-formula, then

$$\mathcal{M} \models \phi(a_1, \ldots, a_n) \Leftrightarrow \mathcal{M}^* \models \phi([f_{a_1}], \ldots, [f_{a_n}])$$

Identifying $\mathcal{M}$ and it's image under the embedding $a \mapsto [f_a]$ we can think of $\mathcal{M}$ as substructure of $\mathcal{M}^*$. Then for $a_1, \ldots, a_n \in M$.

$$\mathcal{M} \models \phi(a_1, \ldots, a_n) \Leftrightarrow \mathcal{M}^* \models \phi(a_1, \ldots, a_n).$$

Definition 5.32 If $\mathcal{M} \subseteq \mathcal{N}$ we say that $\mathcal{N}$ is an *elementary extension* of $\mathcal{M}$ and write $\mathcal{M}\ ee \mathcal{N}$ if

$$\mathcal{M} \models \phi(\overline{a}) \Leftrightarrow \mathcal{N} \models \phi(\overline{a})$$

for all $\overline{a} \in M$.

We have argued that $\mathcal{M}^*$ is an elementary extension of $\mathcal{M}$. This is only interesting if we can also prove $\mathcal{M}^*$ properly extends $\mathcal{M}$.

Proposition 5.33 *If $|I| \leq |\mathcal{M}|$ and $\mathcal{U}$ is a non-principal ultrafilter, then $\mathcal{M}^*$ is a proper extension of $\mathcal{M}$.*

Proof Let $f : I \to M$ be injective. Then for all $a \in M$, $|\{i : f(i) = f_a(i)\}| \leq 1$. Since $\mathcal{U}$ is non-principal, $f \not\sim f_a$. Thus $[f] \in M^* \setminus M$.

Cardinalities of Ultraproducts

Suppose we have $(\mathcal{M}_i : i \in I)$ and an ultrafilter $\mathcal{U} \subseteq \mathcal{P}(I)$.

Exercise 5.34 Suppose $\{i \in I : |\mathcal{M}_i| = n\} \in \mathcal{U}$, then $|\prod \mathcal{M}_i/\mathcal{U}| = n$

Exercise 5.35 If we also have $(\mathcal{N}_i : i \in I)$ and $\{i : |\mathcal{M}_i| = |\mathcal{N}_i|\} \in \mathcal{U}$, then $|\prod \mathcal{M}_i/U| = |\prod \mathcal{N}_i/U|$.

Exercise 5.36 If $\lambda \leq |\mathcal{M}_i| \leq \kappa$ for all $i \in I$, then

$$\lambda \leq \prod \mathcal{M}_i/\mathcal{U} \leq \kappa^{|I|}.$$

For the rest of these Exercises we will assume $I = \mathbb{N}$.

Exercise 5.37 Suppose that for all $n \in \mathbb{N}$, $\{i : |\mathcal{M}_i| = n\} \notin \mathcal{U}$ and $\mathcal{U}$ is non-principal.

a) Show there is a family X of functions $f : \mathbb{N} \to \mathbb{N}$ such that:

i) $|X| = 2^{\aleph_0}$

ii) for each $f \in X$ $f(n) < 2^n$

iii)$f \neq g \in X$, then $\{n : f(n) = g(n)\}$ is finite.

[Hint: For $\alpha : \mathbb{N} \to \{0, 1\}$ let $f_\alpha(n) = \sum_{i=0}^{n-1} \alpha(i) 2^i$].

b) Show there is a partition $I = \bigcup_{n=0}^{\infty} A_n$ such that

i) each $A_n \notin \mathcal{U}$

ii) if $i \in A_n$, then $|\mathcal{M}_i| \geq 2^i$.

[Hint: Let $A_n = \{i : 2^n \leq |M_i| < 2^{n+1}$ or $i = n$ and $|\mathcal{M}_i| \geq \aleph_0\}$.]

For $i \in I$ let $n(i)$ be unique such that $i \in A_{n(i)}$. For $i \in I$ choose $(m_{i,j} : 0 \leq j < 2^{n(i)})$ distinct elements of M_i. For $f \in X$, let $\alpha_f \in \prod M_i$ such that $\alpha_f(i) = m_{i,f(n(i))}$.

c) Prove that if $f \neq g \in X$, then $\alpha_f \not\sim \alpha_g$. Conclude that $|\prod \mathcal{M}_i/U| \geq 2^{\aleph_0}$.

Corollary 5.38 *Suppose that $\mathcal{U}$ is a non-prinicple ultrafilter on $\mathbb{N}$, $|\mathcal{M}_n| \le \aleph_0$ for all n, and $\{n : |\mathcal{M}_n| = m\} \notin U$ for any m, Then $|\prod \mathcal{M}_i/U| = 2^{\aleph_0}$.*

Exercise 5.39 Let $\mathcal{U}$ be a non-principal ultrafilter on the set of primes. Prove $\prod \mathbb{F}_p^{\mathrm{alg}}/\mathcal{U}$ is isomorphic to $\mathbb{C}$ the field of complex numbers.

Part II
Computability

6 Models of Computation

What is a computable function? Our modern intuition is that a function $f : \mathbb{N} \to \mathbb{N}$ is computable if there is a a program in a computer language like C^{++}, PASCAL or LISP such that if we had an idealized computer, with unlimited memory, and we ran the program on input n it would eventually halt and output $f(n)$.

While this definition could be made precise, it is rather unwieldy to try to analyze a complex modern programming language and an idealized computer. We begin this section by presenting two possible candidates for the class of computable functions. The first will be based on register machines, a very primitive version of a computer. The second class will be defined more mathematically. We will then prove that these two classes of functions are the same. *Church's Thesis* will assert that this class of functions is exactly the class of all computable functions.

Church's Thesis should be though of as a statement of philosophy or physics rather than a mathematical conjecture. It asserts that the definitions we have given completely capture our intuition of what can be computed. There is a great deal of evidence for Church's Thesis. In particular, there is no know notion of deterministic computation, including C^{++}-computable or the modern notion of quantum computable, that gives rise to computable functions that are not included in our simple classes. On the other hand issues like time and space complexity or feasability may vary as we change the model.

Register Machines

We will take register machines as our basic model of computations. The programming language for register machines will be a simple type of assembly language. This choice is something of a compromise. Register machines are not as simple as Turing machines, but they are much easier to program. It is not as easy to write complicated programs as it would be in a modern programming language like C^{++} or PASCAL, but it will be much easier to analyze the basic steps of a computation.

In our model of computation we have infinitely many registers $R_1, R_2, \ldots$. At any stage of the computation register R_i will store a nonnegative integer r_i.

Definition 6.1 A *register machine program* is a finite sequence $I_1, \ldots, I_n$ where each I_j is one of the following:
i) Z(n): set R_n to zero; $r_n \leftarrow 0$;
ii) S(n): increment R_n by one; $r_n \leftarrow r_n + 1$;
iii) T(n,m): transfer contents of R_n to R_m; $r_m \leftarrow r_n$;

iv) J(n,m,s), where $1 \leq s \leq n$: if $r_n = r_m$, then go to I_s otherwise go to the next instruction;
v) HALT
and I_n is HALT.

A register machine must be provided with both a program and an initial configuration of the registers. A *computation* procedes by sequentially following the instructions. Note that for any program P there is a number N such, no matter what the initial configuration of the registers is, any computation with P will use at most registers $R_1, \ldots, R_N$.

Example 6.2 *We give a program which, if we start with n in R_1, ends with R_1 containing $n-1$ if $n > 0$ and 0 if $n = 0$.*

```
1)   Z(2)
2)   J(1,2,10)
3)   Z(3)
4)   S(2)
5)   J(1,2,9)
6)   S(2)
7)   S(3)
8)   J(1,1,4)
9)   T(3,1)
10)  HALT
```

We first test to see if R_1 contains 0. If it does we halt. If not, we make $r_2 = 1$ and $r_3 = 0$ and test to see if $r_1 = r_2$ have the same contents. If they do, we move r_3 to R_1 and halt. Otherwise, we increment r_2 and r_3 until $r_1 = r_2$. Since r_3 will always be one less than r_2, this produces the desired result.

Example 6.3 *We give a program which adds the contents of R_1 and R_2 and leaves the sum in R_1.*

```
1)  Z(3)
2)  J(2,3,6)
3)  S(1)
4)  S(3)
5)  J(1,1,2)
6)  HALT
```

We set $r_3 \leftarrow 0$. We increment R_3 and R_1 until $r_3 = r_2$.

Example 6.4 *We give a program to multiply the contents of R_1 and R_2 and leave the product in R_1.*

```
1)   Z(3)
2)   Z(4)
3)   J(2,4,11)
4)   Z(5)
5)   J(1,5,9)
6)   S(3)
7)   S(5)
8)   J(1,1,5)
9)   S(4)
10)  J(1,1,3)
11)  T(3,1)
12)  HALT
```

The main idea is that we will add r_1 to itself r_2 times using R_3 to store the intermediate results. R_4 will be a counter to tell us how many times we have already added r_1 to itself. We add r_1 to r_3 by incrementing R_3, r_1 times. We use R_5 to count how many times we have incremented R_3.

Note that lines 4)–8) are just a slightly modified version of Example 6.3. We add r_1 to r_3 storing the result in R_3. We can think of this as a "subroutine". It is easy to see that we could add a command to our language A(n,m,s) that does:

$$r_s \leftarrow r_n + r_m.$$

Any program written with this additional command could be rewritten in our original language. Similarly, using Example 6.2 we could add a command D(n) that decrements R_n if $r_n > 0$ and leaves R_n unchange if $r_n = 0$.

We give one more example of a program that, on some initial configurations, runs for ever.

Example 6.5 *We give a program such that if r_1 is even halts with $r_1/2$ in R_1 and otherwise never halts.*

```
1)  Z(2)
2)  Z(3)
3)  J(1,2,8)
4)  S(2)
5)  S(2)
6)  S(3)
7)  J(1,1,3)
8)  T(3,1)
9)  HALT
```

We next define what it means for a function $f : \mathbb{N}^k \to \mathbb{N}$ to be computable by a register machine. The last example shows that we need to take partial functions into account.

Suppose P is a register machine program. If $\overline{x} = (x_1, \ldots, x_k)$ we consider the computation where we begin with initial configuration $r_1 = x_1, \ldots, r_k = x_k$ and $r_n = 0$ for $n > k$. If this computation halts we say that P halts on input $\overline{x}$.

Definition 6.6 Suppose $A \subseteq \mathbb{N}^k$. We say $f : A \to \mathbb{N}$ is an *RM-computable* partial function if there is a register machine program P such that:
i) if $\overline{x} \notin A$, then P does not halt on input $\overline{x}$;
ii) if $\overline{x} \in A$, then P halts on input $\overline{x}$ with $f(\overline{x})$ in register R_1.

Alternatively, we could think of a register machine as computing $f : \mathbb{N}^k \to \mathbb{N} \cup \{\uparrow\}$ where $f(\overline{x}) = \uparrow$ means the machine does not halt on input $\overline{x}$.

We could start showing more and more functions are RM-computable by writing more complicated programs. Instead we will give mathematically define an interesting class of fuctions and prove it is exactly the class of RM-computable functions.

Primitive Recursive Functions

Definition 6.7 The class of *primitive recursive functions* is the smallest class C of functions such that:

i) the zero function, $z(x) = 0$ is in C,

ii) the sucessor function $s(x) = x + 1$ is in C,

iii) for all n and all $i \leq n$ the projection function $\pi_i^n(x_1 \ldots x_n) = x_i$, is in C (in particular the identity function on $\mathbb{N}$ is in C),

iv) (Composition Rule) If $g_1 \ldots g_m, h \in C$, where $g_i : \mathbb{N}^n \to \mathbb{N}$ and $h : \mathbb{N}^m \to \mathbb{N}$, then

$$f(\overline{x}) = h(g_1(\overline{x}) \ldots g_m(\overline{x}))$$

is in C,

v) (Primitive Recursion) If $g, h \in C$ where $g : \mathbb{N}^{n-1} \to \mathbb{N}$ and $h : \mathbb{N}^{n+1} \to \mathbb{N}$, then $f \in C$ where:

$$\begin{aligned} f(\overline{x}, 0) &= g(\overline{x}) \\ f(\overline{x}, y+1) &= h(\overline{x}, y, f(\overline{x}, y)). \end{aligned}$$

We now give a large number of examples of primitive recursive functions with derivations showing that they are primitive recursive. A derivation is a sequence of functions $f_1 \ldots f_m$ such that each f_i is either z, s or π_i^n or is obtained from earlier functions by compostion or primitive recursion.

1) the n-ary zero function: $(x_1 \ldots x_n) \mapsto 0$
$f_1 = \pi_i^n$, $f_2 = z$, $f_3 = f_2 \circ f_1$.

2) the constant function $x \mapsto 2$
$f_1 = s$, $f_2 = z$, $f_3 = s \circ z$, $f_4 = s \circ f_3$.

3) $(x, y) \mapsto x + y$
$f_1 = \pi_1^1$, $f_2 = \pi_3^3$, $f_3 = s$, $f_4 = f_3 \circ f_2$ is $(x, y, z) \mapsto z + 1$, and f_5 is $(x, y) \mapsto x + y$ (by primitive recursion using $g = f_1$ and $h = f_4$).

The formal derivations are not very inlightening so we give an informal primitive recursive defintion of addition (and henceforth only give informal defintions):
$x + 0 = x$
$x + (y + 1) = s(x + y)$.

4) multiplication
$x \cdot 0 = 0$
$x \cdot (y + 1) = xy + x$.

5) exponentiation
$x^0 = 1$
$x^{y+1} = x^y \cdot x$.

6) predecesor:

$$\text{pr}(x) = \begin{cases} 0, & \text{if } x = 0; \\ x - 1, & \text{otherwise.} \end{cases}$$

$\text{pr}(0) = 0$
$\text{pr}(y + 1) = y$.

7) sign

$$\operatorname{sgn}(x) = \begin{cases} 0, & \text{if } x = 0; \\ 1, & \text{otherwise.} \end{cases}$$

$\operatorname{sgn}(0) = 0$
$\operatorname{sgn}(y + 1) = 1$

8) $\dot{-}$

$$x \dot{-} y = \begin{cases} 0, & \text{if } x \le y; \\ x - y, & \text{otherwise.} \end{cases}$$

$x \dot{-} 0 = x$
$x \dot{-} (y + 1) = \operatorname{pr}(x \dot{-} y)$

9) Factorials
$0! = 1$
$(n + 1)! = n!(n + 1)$

If $f(\overline{x}, y)$ is primitive recursive then so is

$$(\overline{x}, n) \mapsto \sum_{y \le n} f(\overline{x}, y).$$

$F(\overline{x}, 0) = f(\overline{x}, 0)$
$F(\overline{x}, y + 1) = F(\overline{x}, y) + f(\overline{x}, y + 1)$.

Similary $(\overline{x}, n) \mapsto \prod_{y \le n} f(\overline{x}, y)$ is primitive recursive.

We say that $R(\overline{x})$ is a *primitive recursive predicate* if it is a 0-1 valued primitive recursive function. If P and Q are primitive recursive predicates then so are:
$P \wedge Q(\overline{x}) = P(\overline{x}) \cdot Q(\overline{x})$
$P \vee Q(\overline{x}) = \operatorname{sgn}(P(\overline{x}) + Q(\overline{x}))$
$\neg P(\overline{x}) = 1 \dot{-} P(\overline{x})$

10) $x = y$ is a primitive recursive relation.
The characteristic function of $x = y$ is $1 \dot{-} (\operatorname{sgn}(x \dot{-} y) + \operatorname{sgn}(y \dot{-} x))$

Also if $P(\overline{x}, y)$ is a primitive recursive relation and $g(\overline{x})$ is primitive recursive, then

$$\exists y \le g(\overline{x}) P(\overline{x}, y) = \operatorname{sgn}\left(\sum_{y \le g(\overline{x})} P(\overline{x}, y) \right), \text{ and}$$

$$\forall y \le g(\overline{x}) P(\overline{x}, y) = \operatorname{sgn}\left(\prod_{y \le g(\overline{x})} P(\overline{x}, y) \right)$$

are primitive recursive relations.

For example:

11) $x|y = \exists z \leq y\ xz = y$ is primitive recursive.

Exercise 6.8 Show that $x \leq y$ and $x < y$ are primitive recursive relations.

Exercise 6.9 (Definition by cases): Suppose g and h are primitive recursive functions and P is a primitive recursive predcate. Then f is primitive recursive where:

$$f(\overline{x}) = \begin{cases} g(\overline{x}), & \text{if } P(\overline{x}); \\ h(\overline{x}) & \text{otherwise.} \end{cases}$$

Exercise 6.10 Suppose $f(\overline{x}, y)$ is primitive recursive. Let $g(\overline{x}, z) = \max\{f(\overline{x}, y) : y \leq z\}$ and $h(\overline{x}, z) = \min\{f(\overline{x}, y) : y \leq z\}$. Show that g and h are primitive recursive.

If $P(\overline{x}, y)$ is a primitive recursive function define $\mu y\ P(\overline{x}, y)$ to be the least y such that $P(\overline{x}, y)$ if such a y exists and otherwise $\mu y\ P(\overline{x}, y)$ is undefined. In general $\overline{x} \mapsto \mu y\ P(\overline{x}, y)$ is only a partial function. Even if it is total, it will not in general be primitive recursive. The next excercise gives the best we can do primitive recursively.

Exercise 6.11 Let $P(\overline{x}, y)$ be a primitive recursive predicate and $g(\overline{x})$ a primitive recursive function. Let

$$f(\overline{x}) = \begin{cases} 0, & \text{if } \forall y \leq g(x)\ \neg P(\overline{x}, y); \\ \mu y\ P(\overline{x}, y), & \text{otherwise.} \end{cases}$$

Then f is primitive recursive.

We next show that coding and decoding of sequences is primitive recursive.

12) "x is prime" is a primitive recursive predicate.
x is prime if and only if $x \neq 0 \wedge x \neq 1 \wedge \forall y \leq \mathrm{pr}(x)\ \neg(y|x)$.

13) We next show that the function $n \mapsto p_n$ is primitive recursive, where p_n is the n^{th} prime number (note we set $p_0 = 2, p_1 = 3, \ldots$). To show this we use the following consequence of Euclid's proof that there are infinitely many primes. For any number $n \geq 1$ there is a prime number p wich that $n < p \leq n! + 1$. Thus:
$p_0 = 1$
$p_{n+1} = \mu x \leq p_n!\ Prime(x)$.

We code the sequence $(n_1, \ldots, n_m)$ by $x = \prod_{i=1}^{n} p_i^{n_i+1}$. We say that 1 codes the empty sequence.

14) x codes a sequence is a primitive recursive predicate.
$Seq(x)$ if and only if $x \neq 0 \wedge \forall p \leq x \forall q \leq p\ [(Prime(p) \wedge Prime(q) \wedge p|x) \rightarrow q|x]$.

15) Define $l(x)$ to be 0 if x does not code a sequence, otherwise let $l(x)$ be the length of the sequence coded by x.

$$l(x) = \begin{cases} 0, & \neg Seq(x), \\ max\ m(p_m|x), & \text{otherwise.} \end{cases}$$

16) Define $(x)_i$ to be the i^{th} element of the sequence coded by x if x codes a sequence of length at least i, otherwise it is zero.

$$(x)_i = \begin{cases} max\ n(p_i^{n+1}|x), & Seq(x) \wedge i \leq l(x), \\ 0, & \text{otherwise.} \end{cases}$$

We next show that we can simultaneously define several functions by primitive recursion.

Lemma 6.12 *Suppose* $g_1, \ldots, g_n : \mathbb{N}^k \to \mathbb{N}$, $h_1, \ldots, h_n : \mathbb{N}^{k+n+1} \to \mathbb{N}$ *are primitive recursive and we define* $f_1, \ldots, f_n : \mathbb{N}^{k+1} \to \mathbb{N}$ *by*

$$\begin{aligned} f_i(\overline{x}, 0) &= g_i(\overline{x}) \\ f_i(\overline{x}, m+1) &= h_i(\overline{x}, m, f_1(\overline{x}, m), \ldots, f_n(\overline{x}, m)). \end{aligned}$$

Then $f_1, \ldots, f_n$ *are primitive recursive.*

Proof We define a primitive recursive function $F : \mathbb{N}^{k+1} \to \mathbb{N}$ such that $F(\overline{x}, m) = \prod_{i=1}^n p_i^{f_i(\overline{x}, m)}$. Then we will have $f_i(\overline{x}, m) = v(p_i, F(\overline{x}, m))$. Let

$$\begin{aligned} F(\overline{x}, 0) &= \prod_{i=1}^n p_i^{g_i(\overline{x})} \\ F(\overline{x}, m+1) &= \prod_{i=1}^n p_i^{h_i(\overline{x}, m, v(p_1, F(\overline{x}, m)), \ldots, v(p_n, F(\overline{x}, m)))}. \end{aligned}$$

Then F is primitive recursive and $f_1, \ldots, f_m$ are primitive recursive.

The primitive recursive functions do not exhaust the functions computable by algorithms. Each primitive recursive function has a derivation. As usual we can code each derivation by a natural number. We give a listing of all the primitive recursive functions. Define F_n to be z if n is does not code a derivation, otherwise F_n is the function with derivation coded by n. Intuitively we can do this on such a way that if $G(n, x) = F_n(x)$ is "computable". If this function is primitive recurive, then so is the function $f(x) = G(x, x) + 1$. But f can not be primitive recursive, for if $f = F_n$, then

$$f(n) = G(n, n) + 1 = F_n(n) + 1 = f(n) + 1,$$

a contradiction. Thus G is "computable", but not primitive recursive.

This argument shows that for any class of total computable functions, we can not give an exhaustive listing $H_1, H_2, \ldots$ such that the function $(x, y) \mapsto H_x(y)$ is computable. We will see later this is possible when we consider partial computable functions.

Exercise 6.13 We can give a more concrete example of a "computable" non-primitive recursive function. For any function F we define the n^{th} iterate of F as follows:

$F^{(0)}(x) = x$
$F^{(n+1)}(x) = F(F^{(n)}(x))$

We now define a sequence of functions $f_0, f_1 \ldots$.

$f_0(x) = x + 1$
$f_{n+1}(x) = f_n^{(x)}(x)$.

Define the Ackermann function, $A(x) = f_x(x)$.

a) Show that each f_i is primitive recursive.

b) We say $f \ll g$ if there is a number n such that for all $m > n$, $f(m) < g(m)$. Show that for any primitive recursive function g there is an n such that $g \ll f_n$.

c) Show that for all n, $f_n \ll A$. Thus the Ackermann function is not primitive recursive.

The Recursive Funcitons

We add on more construction to expand th

Definition 6.14 The class of *recursive functions* is the smallest class C of partial functions, containing the the zero function, succesor, and all projection functions and closed under composition, primitive recursion and

vi) (Unboundend Search) If $f(\overline{x}, y)$ is in C, then so is F where $F(\overline{x})$ is the least y such that $f(\overline{x}, y) = 0$ and for all $z < y f(\overline{x}, z)$ is defined. As above we denote F as $\mu y\ f(\overline{x}, y) = 0$.

We use $\uparrow$ to denote "undefined". We need to be careful about composition and primitive recursion for partial functions. If $f = h(g_1, \ldots, g_n)$, then if any $g_i(\overline{x}) \uparrow$, then $f(\overline{x}) \uparrow$.

Similarly, if f is defined by primitive recursion, then then $f(\overline{x}, s) \uparrow$ if $f(\overline{x}, t) \uparrow$ for some $t < s$.

Our intuition tells us that every partial recursive function is computable. We will prove that the RM-computable functions are exactly the partial recursive functions.

Theorem 6.15 *Every recursive function is RM-computable.*

Proof Clearly the basic functions z, s and π_i^n are RM-computable. Thus we need only show that the RM-computable functions are closed under composition, primitive recursion and unbounded search.

claim 1 Suppose $f_1, \ldots, f_n : \mathbb{N}^m \to \mathbb{N}$ and $g : \mathbb{N}^n \to \mathbb{N}$ are RM-computable. Let $h(\overline{x}) = g(f_1(\overline{x}), \ldots, f_n(\overline{x}))$, then h is RM-computable.

Suppose the computation of P_i is a program to compute f_i. By modifying the program slightly we may assume that:

- P_i does not destroy the input (i.e,. does not alter registers $R_1, \ldots, R_m$)

• uses only registers $R_{n+i+1}, R_{n+i+2}, \ldots$

• halts with $f_i(\overline{x})$ in R_{n+i}.

[If necessary we modify P_i to P_i^* which starts by copying R_j into R_{n+i_j} for $j \leq n$, and then is identical to P_i except that for all j the role of R_j is played by R_{n+i+j}.]

The program for computing h begins by running the programs $P_1, \ldots, P_m$ (except that HALTS are replaced by jumping to the begining of the next program). Once we run these programs the registers contain $a_1, \ldots, a_n, f_1(\overline{a}), \ldots, f_m(\overline{a})$.

We next write $f_1(\overline{a}), \ldots, f_m(\overline{a})$ into the first m-registers and erase all of the other registers which were used in the earlier computations. We now run the program to compute g.

claim 2 Suppose $g : \mathbb{N}^m \to \mathbb{N}$ and $h : \mathbb{N}^{m+2} \to \mathbb{N}$ are RM-computatble (possibly partial) functions. Let f be defined from g and h by primitive recursion. Then f is RM-computable.

step 0:

We start with $\overline{x}, y$ in the first $m+1$-registers.

• let $r_{m+2} \leftarrow 0$; this will be a counter

• copy $\overline{x}$ into $R_{m+3}, \ldots, R_{2m+2}$

• run the program for g suitably modified such that we end with the configuration

$$(\overline{x}, y, 0, g(\overline{x}), 0, 0, \ldots).$$

In general at the end of step s we will have $(\overline{x}, y, s, f(\overline{x}, s), 0, 0, \ldots)$.

step $s+1$

• if $r_{m+2} = r_{m+1}$ we are done, fiddle with things so that the configuration is

$$(f(\overline{x}, s), 0, 0, \ldots)$$

and halt, otherwise;

• increment r_{m+2}. Move things around so that we have configuration

$$(\overline{x}, y, s+1, \overline{x}, y, f(\overline{x}, s), 0, 0, \ldots).$$

Run the program for h suitably modified so that we end with configuration

$$(\overline{x}, y, s+1, f(\overline{x}, s+1), 0, 0, \ldots).$$

• Go to next step.

This program computes f

claim 3 If $f(\overline{x}, y)$ is RM-computable, then $\mu y \; f(\overline{x}, y) = 0$ is RM-computable.

Consider the following program:

• Start with configuration $(\overline{x}, 0, 0, 0, \ldots)$.

<u>stage s</u>:

• At the beging of stage s we will have configuration.

$$(\overline{x}, s, 0, 0, \ldots)$$

• Change configuration to $(\overline{x}, s, \overline{x}, s, 0, \ldots)$.

• Run modified version of program for f if this halts we will have configuration

$$(\overline{x}, s, f(\overline{x}, s), 0, 0 \ldots).$$

• If $f(\overline{x}, s) = 0$ halt with configuration $(s, 0, 0, \ldots)$. If not change to configuration $(\overline{x}, s+1, 0, 0, \ldots)$ and go to next stage.

If there is an s such that $f(\overline{x}, s) = 0$ and for all $t < s$, $f(\overline{t}, s) \downarrow \neq 0$, then we will eventually halt and output s. Otherwise the search will continue forever.

Thus every recursive function is RM-computable.

Theorem 6.16 *Every RM-computable function is recursive.*

Proof Let $f : \mathbb{N}^m \to \mathbb{N}$ be RM-computable (possibly partial). Let $I_1, \ldots, I_m$ be a program which computes f. Suppose this program uses only registers $R_1, \ldots, R_N$. We define primitive recursive functions $g_1, \ldots, g_N : \mathbb{N}^{m+1} \to \mathbb{N}$ and $j : \mathbb{N}^{m+1} \to \mathbb{N}$ such that:

$$g_i(\overline{x}, s) = \text{contents of } R_i \text{ at stage } s \text{ on input } \overline{x}$$

and

$$j(\overline{x}, s) = \begin{cases} 0 & \text{if the machine on input } x \text{ has halted by stage } s \\ j & \text{if } I_j \text{ is the next instruction to be executed.} \end{cases}$$

Let $h(x) = \mu s\ j(x, s) = 0$. Then $f(x) = g_1(x, h(x))$.

The construction of g_i and j are routine but tedious primitive recursions. We define them simultaneously and use Lemma 6.12.

We give one example. Consider the program to compute

$$f(x, y) = \begin{cases} x - y & y \leq x \\ \uparrow & y > x. \end{cases}$$

```
1)  Z(3)
2)  J(1,2,6)
3)  S(2)
4)  S(3)
5)  J(1,1,2)
6)  T(3,1)
7)  HALT
```

$$j(x, y, s) = \begin{cases} 1 & s = 0 \\ 2 & s = 1 \text{ or } j(x, y, s-1) = 5 \\ 3 & j(x, y, s-1) = 2 \\ 4 & j(x, y, s-1) = 3 \\ 5 & j(x, y, s-1) = 4 \\ 6 & j(x, y, s-1) = 2 \text{ and } g_2(x, y, s-1) = g_1(x, y, s-1) \\ 7 & j(x, y, s-1) = 6 \\ 0 & j(x, y, s-1) \geq 7 \text{ or } j(x, y, s-1) = 0. \end{cases}$$

$$\begin{array}{rcl} g_1(x,y,0) & = & x \\ g_1(x,y,s+1) & = & \begin{cases} g_1(x,y,s) & j(x,y,s) \neq 6 \\ g_3(x,y,s) & j(x,y,z) = 6 \end{cases} \end{array}$$

$$\begin{array}{rcl} g_2(x,y,0) & = & y \\ g_2(x,y,s+1) & = & \begin{cases} g_2(x,y,s) & j(x,y,s) \neq 3 \\ g_2(x,y,s)+1 & \text{otherwise.} \end{cases} \end{array}$$

$$\begin{array}{rcl} g_3(x,y,0) & = & 0 \\ g_3(x,y,s+1) & = & \begin{cases} g_3(x,y,s) & j(x,y,s) \neq 4 \\ g_3(x,y,s)+1 & \text{otherwise.} \end{cases} \end{array}$$

These functions are clearly primitive recursive.

Church's Thesis

Church's Thesis *A partial function is computable if and only if it is partial recursive*

Church's Thesis asserts that the partial recursive functions, or the RM-computable functions, completely capture our intuitve notion of computability.

We will use Church's Thesis frequently in arguments by giving an intuitive argument that a function is computable and then asserting that therefore it is recursive or RM-computable. Whenever we make such an argument, we are asserting that, if challenged, we could produce the RM-machine code that would compute the function.

There is a great deal of evidence for Church's Thesis. Any reasonable notion of "computabile function" has been shown to be equivalent to "partial recursive" or "RM-computable". Indeed, Church first stated the conjecture for functions definable in λ-calculus, an ancestor of the LISP programming language.

Random Access Machines

We give one more argument towards the plausibility of Church's thesis. One aspect of modern computing that is missing in register machines is dynamic access to memory. In a modern computer language we can compute a number n and then store another number in memory cell n. We will describe a generalization of register machines that allows this kind of dynamic access and prove that they do not allow us to compute new functions.

Definition 6.17 A *Random Access Machine* is one where we have memory locations $M_0, M_1, M_2, M_3, \ldots$. Let m_i be the contents of M_i. A program for

a random access machine is a finite sequence of instructions $I_1, \ldots, I_m$. Where the allowable instructions are:
i) Z(n); set m_n to zero
ii) S(n); increment m_n
iii) J(i,j,l); if $m_i = m_j$, go to instruction l.
iv) T(i,j); transfer the contents of M_{m_i} to M_{m_j}
v) HALT

The key difference is that we are allowed to specifiy in the program what address we want to store something in.

A function f is said to be *RAM-computable* if there is a random access machine program which given initial configuration $(x, 0, 0, \ldots)$ halts with $f(x)$ in M_0 if $x \in \text{dom}(f)$ and does not halt if $x \notin \text{dom}(f)$.

Exercise 6.18 Every RM-computable function is RAM-compuable.

We next out line the proof that every RAM-computable function is RM-computable. The key idea is to code configurations of the RAM as a single number. Suppose at some stage s, n is the largest memory location that we have used. Then the configuration of the machine is given by the sequence $(m_1, \ldots, m_n, 0, 0, 0, \ldots)$.

We code this configuration with the number $\prod p_i^{m_i}$. All of the operations of the machine correspond to simple arithmetic operations on the code. Let $v(p, x) =$ largest power of p dividing x. Note that $v(p_i, x)$ extracts the contents of M_i from the code x.

For example: • Z(n): corresponds to the operation

$$x \mapsto \frac{x}{p_n^{v(p_n, x)}}.$$

• S(n): corresponds to

$$x \mapsto x p_n.$$

• T(i,j): Let $l = v(p_j, x)$ and $k = v(p_i, x)$. The new configuration is coded by

$$\frac{x}{p_l^{v(p_l, x)}} p_l^{v(p_k, x)}$$

Exercise 6.19 Using the above idea show that any RAM computable function is RM computable.

Henceforth we will usually use Church's thesis blindly. We will say that a partial function is *computable* if it is RM-computable with full confidence that anything which is intuitively computable can be done with a register machine.

7 Universal Machines and Undecidability

Our main goal in this section is to prove that there is a computable partial function $\Psi : \mathbb{N}^2 \to \mathbb{N}$ such that if ϕ_n is the function

$$\phi_n(x) = \Psi(n, x)$$

then $\phi_0, \phi_1, \ldots$ is an enumeration of all computable partial functions. This is in sharp contrast with the observation in the previous section, that there is no such enumeration of the total recursive functions.

We will code register machine programs by natural numbers and we will arange the coding so that each number codes a program. If P_n is the program with code n, then $\phi_n(x)$ will be the result of running P_n on input x.

The register machine computing Ψ is a *universal register machine.* It behaves like a modern compiler. If f is a computable function we find e such that $f = \phi_e$ and compute $f(x)$ by computing $\Psi(e, x)$.

Our first task is to code register machine programs. We will use a more subtle coding than the one of §6 to insure that every natural number codes a program.

Let $\pi : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ by $\pi(m, n) = 2^m(2n + 1) - 1$.

Lemma 7.1 *π is a bijection and both π and π^{-1} are computable (indeed primitive recursive).*

Proof Clearly π is primitive recursive. To calculate $\pi^{-1}(x)$, factor $x + 1 = yz$ where y is a power of 2 and z is odd. Then $m = \log_2 y$ and $n = \frac{z-1}{2}$.

Once we can code pairs it is easy to code triples. We view (a, b, c) as $((a, b), c)$. Let $\psi : \mathbb{N}^3 \to \mathbb{N}$ by

$$\psi(p, q, r) = \pi(\pi(p, q), r).$$

Let I be the set of all instructions for register machines. There is $\beta : I \to \mathbb{N}$ a computable bijections.

$\beta(\text{HALT}) = 0$
$\beta(\text{Z}(n)) = 4(n - 1) + 1$
$\beta(\text{S}(n)) = 4(n - 1) + 2$
$\beta(\text{T}(m, n)) = 4(\pi(m - 1, n - 1)) + 3$
$\beta(\text{J}(m, n, r)) = 4(\psi(m - 1, n - 1, r - 1)) + 4$

β is easy to decode. For example for what i is $\beta(i) = 47$? Since $47 \equiv 3 (\text{mod } 4)$, i must code T(m,n) for some m and n, where $\pi(m - 1, n - 1) = \frac{47-3}{4} = 11$. Since $11 + 1 = 2^2(2 \cdot 1 + 1)$, $\pi(2, 1) = 11$. Thus i codes the instruction $T(3, 2)$.

We also want

$$\tau : \bigcup_{k>0} \mathbb{N}^k \to \mathbb{N}$$

a computable bijection with computable inverse. We let

$$\tau(a_1,\ldots,a_k) = 2^{a_1} + 2^{a_1+a_2+1} + 2^{a_1+a_2+a_3+2} \ldots + 2^{a_1+\ldots+a_k+k-1} - 1.$$

Given x we calculate $\tau^{-1}(x)$ as follows:
i) find the binary expansion of $x+1 = 2^{b_1} + \ldots + 2^{b_k}$ where $b_1 < \ldots < b_k$
ii) Let $a_1 = b_1$ and $a_{i+1} = b_{i+1} - b_i - 1$ for $1 \leq i < k$.

For example we calculate $\tau^{-1}(45)$: $45 + 1 = 2 + 2^2 + 2^3 + 2^5$. Thus $a_1 = 1$, $a_2 = 0$, $a_3 = 0$,and $a_4 = 1$. Thus $\tau^{-1}(45) = (1,0,0,1)$ [note: $46 = 2^1 + 2^{1+0+1} + 2^{1+0+0+2} + 2^{1+0+0+1+3} - 1$]

We now give a method for coding all register machine programs. [3] Let P be the program $I_1,\ldots,I_m$ by

$$\gamma(P) = \tau(\beta(I_1),\ldots,\beta(I_m)).$$

For $m \in \mathbb{N}$, let $P_m = \gamma^{-1}(m)$. Let $\phi_m^{(n)}$ be the n-ary function computed by program P_m. Clearly $\phi_0^{(n)}, \phi_1^{(n)}, \ldots$ is a list of all partial recursive functions in n-variables. [We will supress the superscript if it is clear]
If f is computable we say that n is an *index* for f if $f = \phi_n$. There will usually be many indicies for f.

Consider the partial function $\Psi^{(n)} : \mathbb{N}^{n+1} \to \mathbb{N}$ by $\Psi^{(n)}(e,\overline{x}) = \phi_e^{(n)}(\overline{x})$.

Theorem 7.2 *The functions $\Psi^{(n)}$ are computable.*

Proof For notational simplicity we will consider only the case $n = 1$.
Informally we compute $\Psi(e,x)$ by decoding e to obtain the program $P_e = I_1,\ldots,I_N$. Simulate program P_e on input x.
We use one number to store the register configuration in the simulation. Suppose we are using registers $R_1,\ldots,R_m$ and R_i contains r_i. We will code this configuration by

$$c = \prod_{i=1}^{m} p_i^{r_i}.$$

We call c the *configuration code* of the machine. The *current state* of the machine will be $\sigma = \pi(c,j)$ where j is the next instruction to be executed (and if we have halted $j = 0$) [here π is the pairing function].
Define $c(e,x,t)$ = configuration after t steps of program P_e on input x if we have not yet halted. If we have halted let $c(e,x,t)$ be the final configuration.
Let $j(e,x,t)$ = number of the next instruction if the computation of P_e on input x has not halted by step t and let it be 0 otherwise.
Let $\sigma(e,x,t) = \pi(c(e,x,t), j(e,x,t))$.

[3]We will also code up some nonsense programs that contain $\mathrm{J}(i,j,n)$ where n is greater than the number of instructions. By convention, whenever we reach such an instruction we halt.

claim c, j and σ are computable (indeed they are primitive recursive).

• $c(e,x,0) = 2^x$ and $j(e,x,0) = 1$.

• Given $c = c(e,x,t)$ and $j = j(e,x,t)$, we compute $j(e,x,t+1)$ and $c(e,x,t+1)$.

• If $j = 0$, then $c(e,x,t+1) = c$ and $j(e,x,t+1) = j$.

• If $N \leq j > 0$, then decode e to find I_j.

• If I_j is $I(m)$ then $c(e,x,t+1) = c \cdot p_m$ and $j(e,x,t+1) = j+1$.

• If I_j is $Z(m)$ then $c(e,x,t+1) = \frac{c}{p_m^l}$ where l is the largest such that p_m^l divides c, and $j(e,x,t+1) = j+1$.

• If I_j is $T(n,m)$ then $c(e,x,t+1) = c \cdot p_m^{l-k}$ where l is largest such that p_n^l divides c and k is largest such that p_m^l divides c. Let $j(e,x,t+1) = j+1$.

• If I_j is $J(n,m,i)$ then $c(e,x,t+1) = c$ and $j(e,x,t+1) = i$ if the largest k such that p_m divides c is equal to the largest l such that p_n divides c, and otherwise $j(e,x,t+1) = j+1$.

• If I_j is HALT or $j > N$, then $c(e,x,t+1) = c$ and $j(e,x,t) = 0$.

Once we know that c and j are computable (indeed primitive recursive), we obtain a general recursive $h(e,x) = \mu\, tj(e,x,t) = 0$. Then $\Psi(e,x)$ is the largest n such that 2^n divides $c(e,x,h(e,x))$. Clearly Ψ is computable.

The machine that computes Ψ is called the *Universal Register Machine.*

Definition 7.3 Let $T = \{(e,x,s) : P_e$ on input x halts by stage $s\}$. This is called *Kleene's T-predicate.* The arguments above show that T is primitive recursive as $(e,x,s) \in T$ if and only if $j(e,x,s) = 0$.

The following theorem is often useful. (For some reason it is often refered to as the *s-m-n theorem*).

Lemma 7.4 (Parameterization Lemma) *If $f(x,y)$ is a computable partial function then there is a total computable function $k(x)$ such that for all x, $k(x)$ is an index for the function $y \mapsto f(x,y)$. Indeed the function $k(x)$ can be choosen one to one.*

Proof Let P be a program computing $f(x,y)$ [starting with x in R_1 and y in R_2. Consider the following program Q_n. Start with y in register 1.

```
1)     T(2,1)     $r_2 \leftarrow r_1$
2)     Z(1)       $r_1 \leftarrow 0$
3)     S(1)       $r_1 \leftarrow 1$
4)     S(1)       $r_1 \leftarrow 2$
⋮      ⋮
n+2)   S(1)       $r_1 \leftarrow n$
       $P$
```

If we start with input y, after step $n+2$ we will have n in R_1 and y in R_2. Running the program P will compute $f(n,y)$.

Thus the program Q_n is a program to compute $\lambda y[f(n,y)]$. The function k is the function which takes us from n to a code for the program P_m. k is easily seen to be one to one.

Definition 7.5 We say that a set $A \subseteq \mathbb{N}^m$ is *recursive* if it's characteristic function

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

is computable.

Since there are $2^{\aleph_0}$ subsets of $\mathbb{N}$ and only $\aleph_0$ possible algorithms, most subsets of $\mathbb{N}$ are not computable. Turing gave an important natural example.

Let $H = \{(e,x) : \phi_e(x) \downarrow\}$. We call H the *halting problem.*
Let $K = \{e : \phi_e(e) \downarrow\}$.

Theorem 7.6 (Unsolvability of the Halting Problem) *Neither H nor K is not recursive.*

Proof If H were recursive then K would be recursive so it suffices to show that K is not recursive. Suppose K is recursive. Let P be a program computing the characteristic function of K. Consider the following program $\widehat{P}$.

• On input x, run program P. If P outputs 0, then halt. If P outputs 1, then go into an infinite loop.

Suppose $I_1, \ldots, I_m$ is the program P. Let $\widehat{I_1}, \ldots \widehat{I_m}$ be the same program where every HALT has been replaced by J(1,1,$m+1$), then $\widehat{P}$ is

1)	$\widehat{I_1}$
$\vdots$	$\vdots$
m)	$\widehat{I_m}$
m+1)	Z(2)
m+2)	J(1,2,$m+4$)
m+3)	J(1,1,$m+2$)
m+4)	HALT

For some e, $\widehat{P} = P_e$. Then

$$\phi_e(x) = \begin{cases} 0 & x \notin K \\ \uparrow & x \in K. \end{cases}$$

Is $e \in K$?

$$e \in K \Leftrightarrow \phi_e(e) \downarrow \Leftrightarrow e \notin K$$

a contradiction. Thus K is not recursive.

Definition 7.7 Let $Tot = \{e : \phi_e \text{ is total}\}$.

We argue that Tot is not recursive. Suppose it were, let g be the characteristic function of Tot. Let

$$f(x) = \begin{cases} \phi_x(x) + 1 & \text{if } g(x) = 1 \\ 0 & \text{if } g(x) = 0. \end{cases}$$

If g is computable, then f is computable. In fact

$$f(x) = \begin{cases} \psi(x,x)+1 & \text{if } g(x)=1 \\ 0 & \text{otherwise} \end{cases}.$$

Thus for some e, $f = \phi_e$. Also f is easily seen to be total. But then $\phi_e(e)\downarrow$ and $f(e) = \phi_e(e)+1$, a contradiction.

We will give other natural examples in §8.

We will finish this section with an application to logic.

Theorem 7.8 (Church) *The set of valid sentences of first order logic is not recursive.*[4]

Proof For any P and any natural number n we will give a sentence θ_n^P such that θ_n^P is valid if and only if P halts on input n. If we had a program to decide if a sentence is valid, then we would have an algorithm to decide the halting problem.

Suppose P uses registers $R_1,\ldots,R_m$. Let $P = I_1,\ldots,I_s$. Let $\mathcal{L} = \{0,s,R\}$ where s is a unary function symbol and R is an $m+1$-ary predicate. We use $s^n(x)$ to denote

$$\underbrace{s(s(\ldots(x)\ldots))}_{n \text{ times}}.$$

The intended interpretation is that $s^n(0) = n$ and $R(s^{n_1}(0),\ldots,s^{n_m}(0),s^j(0))$ holds iff and only if one possible configuration of the machine is that R_i is n_i and the next instruction is j.

For each instruction I_i we write down an axiom τ_i where:

i) If I_i is $Z(l)$, then τ_i is

$$\forall x_1,\ldots,x_m\ (R(x_1,\ldots,x_m,s^i(0)) \to R(x_1,\ldots,x_{l-1},0,x_{l+1},\ldots,x_m,s^{i+1}(0))).$$

ii) If I_i is $S(l)$, then τ_i is

$$\forall x_1,\ldots,x_m\ (R(x_1,\ldots,x_m,s^i(0)) \to R(x_1,\ldots,x_{l-1},s(x_l),x_{l+1},\ldots,x_m,s^{i+1}(0))).$$

iii) If I_i is $T(i,l)$, then τ_i is

$$\forall x_1,\ldots,x_m\ (R(x_1,\ldots,x_m,s^i(0)) \to R(x_1,\ldots,x_{l-1},x_i,x_{l+1},\ldots,x_m,s^{i+1}(0))).$$

iv) If I_i is $J(i,l,j)$, then τ_i is

$$\forall x_1,\ldots,x_m\ (R(\overline{x},s^i(0)) \to ((x_i = x_l \to R(\overline{x},s^j(0)) \wedge ((x_i \neq x_l \to R(\overline{x},s^{i+1}(0))$$

v) If I_i is HALT, then τ_i is

$$\forall \overline{x}\ R(\overline{x},s^i(0)) \to R(\overline{x},0).$$

[4] To make this precise we need the machinery of Gödel codes from §11.

The sentence

$$R(s^n(0), 0, \ldots, 0, s(0))$$

corresponds to the initial configuration on input n.

Let θ_n^P be

$$(R(s^n(0), 0, \ldots, 0, s(0)) \wedge \bigwedge_{i=1}^{s} \tau_i) \to \exists x \ R(\overline{x}, 0)$$

Suppose P halts on input n. Suppose

$$\mathcal{M} \models R(s^n(0), 0, \ldots, 0, s(0)) \wedge \bigwedge_{i=1}^{s} \tau_i.$$

An easy induction shows that if at some stage in the computation of P on input n our register machine has configuration $(k_1, \ldots, k_m)$ and the next instruction is I_j then

$$\mathcal{M} \models R(s^{k_1}(0), \ldots, s^{k_m}(0), s^j(0).$$

In particular, we reach the HALT instruction so $\mathcal{M} \models R(s^{l_1}, \ldots, s^{l_m}, 0)$ where $(l_1, \ldots, l_m)$ are the contents of the registers when P halts. Thus $\mathcal{M} \models \theta_n^P$. It follows that θ_n^P is valid.

On the other hand, suppose θ_n^P is valid. Let $\mathcal{M}$ be the $\mathcal{L}$-structure with universe $\mathbb{N}$ where $s^{\mathcal{M}}(n) = n + 1$ and $R(k_1, \ldots, k_m, j)$ if and only $j > 0$ and at some stage in the computation the registers hold $(k_1, \ldots, k_m)$ and the next instruction is j or $j = 0$ and the halting configuration is $(k_1, \ldots, k_m)$. Then

$$\mathcal{M} \models R(s^n(0), 0, \ldots, 0, s(0)) \wedge \bigwedge_{i=1}^{s} \tau_i.$$

So

$$\mathcal{M} \models \exists \overline{x} \ R(\overline{x}, 0)$$

and P halts on input n. Thus P halts on input n.

Thus P halts on input n if and only if θ_n^P is valid. If validity were recursive then we could decide the halting problem.

8 Recursively Enumerable and Arithmetic Sets

Definition 8.1 A set $X \subseteq \mathbb{N}$ is *recursively enumerable* if X is the domain of a partial recursive function.

The next proposition gives equivalent characterizations of recursively enumerable. The second justifies the intuition behind the name. A nonempty set is recursively enumerable if there is a total recursive function f such that $f(0), f(1), \ldots$ enumerates f.

Proposition 8.2 *Let $X \subseteq \mathbb{N}$. The following are equivalent:*
i) X is recursively enumerable;
ii) $X = \emptyset$ or X is the range of a total recursive function.
iii) there is a recursive $Y \subseteq \mathbb{N}^{m+1}$ such that $X = \{y : \exists \overline{x}\ (\overline{x}, y) \in Y\}$;
iii) X is the domain of a partial recursive function;

Proof
i)$\Rightarrow$ii) Suppose $X \neq \emptyset$ is the range of the partial recursive function f. Let $x_0 \in X$. Let $g : \mathbb{N}^2 \to \mathbb{N}$ by

$$g(x, s) = \begin{cases} f(x) & \text{if } T(e, x, s) \\ x_0 & \text{otherwise} \end{cases}$$

where $T(e, x, s)$ is the Kleene T-predicate asserting P_e halts on input x by stage s. Then g is total recursive and the range of X is equal to the range of g. If $\sigma : \mathbb{N} \to \mathbb{N}^2$ is a recursive bijection, then $\widehat{g} = g \circ \sigma$ is the desired function.

ii)$\Rightarrow$iii) Let X be the range of f. Let $Y = \{(x, y) : f(x) = y\}$. Then Y is recursive and $X = \{y : \exists x\ f(x) = y\}$.

iii)$\Rightarrow$v) Let $Y \subset \mathbb{N}^{m+1}$. Let $\sigma : \mathbb{N} \to \mathbb{N}^m$ be a recursive bijection. Let $f : \mathbb{N} \to \mathbb{N}$, by $f(n) = \mu x\ (\sigma(x), n) \in Y$. f is partial recursive and X is the domain of f.

iii)$\Rightarrow$iv) Let X be the domain of f. Let

$$g(x) = \begin{cases} x & f(x) \downarrow \\ \uparrow & \text{otherwise.} \end{cases}$$

Then g is partial recursive and the range of g is the domain of f.

We next fix an enumeration of the recursively enumerable sets.

Definition 8.3 Let $W_e = \{x : \phi_e(x) \downarrow\} = \text{dom } \phi_e$. Then $W_0, W_1, W_2, \ldots$ is an enumeration of the recursively enumerable sets.

The Halting set $H = \{(e, x) : \phi_e(x) \downarrow\}$ is the domain of the universal function Ψ. Thus H is recursively enumerable. Similarly $K = \{e : \phi_e(e) \downarrow\}$ is the domain of $e \mapsto \psi(e, e)$ and hence recursively enumerable. Thus there are recursively enumerable sets which are not recursive.

Recursively enumerable sets arise naturally in logic when we take the set of logical consequences of a theory. For the moment this will be informal (since we are talking about sets of sentences rather than natural numbers). They will me made precise in §11 when we talk about coding formulas.

Suppose T is a recusive set of sentences. Then $Cn(T) = \{\phi : T \vdash \phi\}$ is recursively enumerable as $Cn(T) = \{\phi : \exists p\ p \text{ is a proof of } \phi \text{ from } T\}$. By ii) $Cn(T)$ is recursively enumerable.

Proposition 8.4 *If A and B are recursively enumerable, then $A \cup B$ and $A \cap B$ are recursively enumerable.*

Proof We give intuitive arguments which can easily be made precise.

Suppose we have programs enumerating A and B. To enumerate $A \cup B$, we enumerate x whenever we see x appear in either the enumeration of A or the enumeration of B.

To enumerate $A \cap B$, we enumerate x once we see x appear in the enumeration of both A and B.

Proposition 8.5 *Every recursive set is recursively enumerable.*

Proof Let f be the characteristic function for A and let

$$g(x) = \begin{cases} 1 & f(x) = 1 \\ \uparrow & f(x) \neq 0. \end{cases}$$

Then $A = \text{dom } g$.

Proposition 8.6 *A is recursive if and only if A and $\mathbb{N} \setminus A$ are recursively enumerable.*

Proof If A is recusive, then $\neg A$ is recursive. Thus, by Proposition 8.5 both A and $\neg A$ are recursively enumerable.

If A and $\neg A$ are recursively enumerable, then we can decide if $x \in A$ as follows: start enumerating A and $\neg A$. We will eventually find x in one of the two lists. If x is enumerated into A, then output x. If x is enumerated into $\neg A$, output no.

Corollary 8.7 *$\neg K$ and $\neg H$ are not recursively enumerable.*

Proof Otherwise K and H are recursive by 8.6.

Definition 8.8 $A \leq_m B$ (A is many-one reducible to B) if there is a total recursive $f : \mathbb{N} \to \mathbb{N}$ such that $x \in A \Leftrightarrow f(x) \in A$.

If $A \leq_m B$ then B is at least as complicated as A. We can reduce problems about A to probelms about B. We next show that the Halting Problem is the most complicated recursively enumerable set.

Lemma 8.9 *Suppose $A \leq_m B$. If B is recursive, then so is A. Also if B is recursively enumerable so is A.*

Proof If B is recursive this is clear. Suppose B is recursively enumerable. Suppose g is partial recursive and $B = \text{dom} g$. Suppose f is total recursive and $n \in A$ iff $f(n) \in B$. Then $A = \{n : g(f(n)) \downarrow\}$ a recursively enumerable set.

Lemma 8.10 *If A is recursively enumerable, then $A \leq_m H$.*

Proof Suppose A is the domain of ϕ_e. Let $f(n) = (e, n)$. Then

$$\begin{aligned} n \in A &\Leftrightarrow \phi_e(n) \downarrow \\ &\Leftrightarrow \Psi(e, n) \downarrow \\ &\Leftrightarrow f(e, n) \in H. \end{aligned}$$

Lemma 8.11 *If A is recursively enumerable $A \leq_m K$.*

Proof If suffices to show $H \leq_m K$. There is a total recursive function g such that for all e, x, y, $\phi_{g(e,x)}(y) = \phi_e(x)$. Intuitively g is a function which on input e and x outputs a program P, such that on any input y, P runs P_e on input x.

More formally let $G(e,x,y) = \Psi(e,x)$. Apply the Parameterization Lemma to obtain a total recursive $g(e,x)$ such that $\phi_{g(e,x)}(y) = G(e,x,y) = \phi_e(x)$. Then $(e,x) \in H$ if and only if for all y, $\phi_{g(e,x)}(y)\downarrow$ if and only if $\phi_{g(e,x)}(g(e,x))\downarrow$.

Thus $(e,x) \in H$ if and only if $g(e,x) \in K$, so $H \leq_m K$.

Thus A is recursively enumerable if and only if $A \leq_m H$ if and only if $A \leq_m K$.

Recall that $Tot = \{e : \phi_e \text{ is total}\}$. We will show that

Lemma 8.12 *i) $K \leq_m Tot$*
ii) $\neg K \leq_m Tot$
iii) Neither Tot nor $\mathbb{N} \setminus Tot$ is recursively enumerable.

Proof

i) Define a total recursive function $f(x)$ such that for all e, $\phi_{f(e)}(y) = \phi_e(e)$. (The existence of such an f follows from the parameterization lemma.) Then $e \in K \Leftrightarrow f(e) \in Tot$.

ii) Define a total recursive function $f(x)$ such that

$$\phi_{f(e)}(s) = \begin{cases} 1 & \phi_e(e) \text{ has not halted by stage } s \\ \uparrow & \text{otherwise.} \end{cases}$$

Let

$$G(e,s) = \begin{cases} 1 & \neg T(e,e,s) \\ \uparrow & \text{otherwise} \end{cases}$$

and apply the paramterization lemma to obtain a total recursive g such that $\phi_{g(e)}(s) = G(e,s)$. Then $e \notin K$ if and only if there is an s such that $T(e,e,s)$ if and only if there is an s such that $\phi_{g(e)}(s)\uparrow$. Thus $e \in \neg K \Leftrightarrow g(e) \in K$.

iii) If Tot were recursively enumerable, then since $\neg K \leq_m Tot$, $\neg K$ would be recursively enumerable and K would be recursive.

Note that if $x \in A \Leftrightarrow f(x) \in B$, then $x \notin A \Leftrightarrow f(x) \notin B$. So $A \leq_m B \Leftrightarrow \neg A \leq_m \neg B$. Thus since $K \leq_m Tot$, $\neg K \leq_m \neg Tot$. If $\neg Tot$ were recursively enumerable then $\neg K$ would be recursively enumerable, a contradiction.

Definition 8.13 We say that $X \subseteq \mathbb{N}^m$ is Σ^0_1 if and only if there is a recursive $Y \subseteq \mathbb{N}^{m+n}$ such that

$$X = \{\overline{x} \in \mathbb{N}^m : \exists \overline{y}\ (\overline{x}, \overline{y}) \in Y\}.$$

We say that $X \subseteq \mathbb{N}^m$ is Π^0_n if and only if $\mathbb{N} \setminus X$ is Σ^0_n. X is Σ^0_{n+1} if and only if there is a Π^0_n set $Y \subset \mathbb{N}^{m+k}$ such that

$$X = \{\overline{x} : \exists \overline{y}\ (\overline{x}, \overline{y}) \in Y\}.$$

We say that X is Δ_n if and only if X is Σ_n^0 and X is Π_n^0.

By 8.2 the Σ_1^0 sets are exactly the recursively enumerable sets. Note that the Δ_1 sets are the recursive sets. It is easy to see that $\Sigma_n^0 \cup \Pi_n^0 \subseteq \Delta_{n+1}$.

Definition 8.14 We say that X is *arithmetic* if $X \in \cup_n \Sigma_n^0$.

Proposition 8.15 *i) If A_0 and A_1 are Σ_n^0 (respectively, Π_n^0), then $A_0 \cap A_1$ and $A_0 \cup A_1$ are Σ_n^0 (Π_n^0).*

ii) If $A \subset \mathbb{N}^{m+1}$ is Σ_n^0, then $\{\overline{x} : \exists y\ (\overline{x}, y) \in A\}$ is Σ_n^0.

iii) If $A \subset \mathbb{N}^{m+1}$ is Π_n^0, then $\{\overline{x} : \forall y\ (\overline{x}, y) \in A\}$ is Π_n^0.

iv) If $A \subset \mathbb{N}^{m+1}$ is Σ_n^0 and $f : \mathbb{N}^m \to \mathbb{N}$ is total recursive, then $\{\overline{x} : \forall y < f(\overline{x})\ (\overline{x}, y) \in A\}$ is Σ_n^0.

v) If $A \subset \mathbb{N}^{m+1}$ is Π_n^0 and $f : \mathbb{N}^m \to \mathbb{N}$ is total recursive, then $\{\overline{x} : \exists y < f(\overline{x})\ (\overline{x}, y) \in A\}$ is Π_n^0.

vi) If A is Σ_n^0 (respectiely Π_n^0) and $B \leq_m A$, then B is Σ_n^0 (Π_n^0).

Proof

i) Let $A_i = \{\overline{x} : \exists \overline{y}\ (\overline{x}, \overline{y}) \in B_i\}$ where B_i is Π_{n-1}^0 (or recursive if $n = 1$. Then $A_0 \cup A_1 = \{\overline{x} : \exists \overline{y}\ ((\overline{x}, \overline{y}) \in B_0 \cup B_1)\}$. By induction $B_0 \cup B_1$ is Π_{n-1}^0. Thus $A_0 \cup A_1$ is Σ_n^0.

Similarly $A_0 \cap A_1 = \{\overline{x} : \exists \overline{y_0} \exists \overline{y_1}\ ((\overline{x}, \overline{y_0}) \in B_0 \wedge (\overline{x}, \overline{y_1}) \in B_1\}$.

ii) and iii) are similar.

iv) Suppose $A = \{(\overline{x}, y) : \exists \overline{z} (\overline{x}, y, \overline{z}) \in B\}$. Then $\forall y < f(\overline{x}) \exists \overline{z} (\overline{x}, y, \overline{z}) \in B$ iff and only if $\exists \sigma (\overline{x}, y, \sigma) \in B^*$, where we think of σ as coding a finite sequence $(\overline{z_0}, \ldots, \overline{z_{f(\overline{x})-1}})$ and B^* asserts that forall $y < f(\overline{x})$, $(\overline{x}, y, \overline{z_y}) \in B$. Since Π_{n-1}^0 sets are closed under $\forall y$, B^* is Π_{n-1}^0. Thus our set is Σ_n^0.

v) is similar

vi) Suppose A is Σ_n^0. Let f be a total recursive function such that

$$x \in B \Leftrightarrow f(x) \in A.$$

Let

$$Y = \{(x, y) : y \in A \wedge f(x) = y\}.$$

Then $Y \in \Sigma_n^0$ and $B = \{x : \exists y\ (x, y) \in A\}$ is Σ_n^0.

Exercise 8.16 Show that every subset of $\mathbb{N}^k$ definable in the language of arithmetic $\mathcal{L} = \{+, \cdot, <, 0, 1\}$ is arithmetic. We will see in §10 that the converse is true.

Examples

Below let $W_e^s = \{x : \phi_e(x) \downarrow$ by stage $s\}$. Clearly W_e^s is recursive.

- $Tot = \{e : \phi_e$ is total$\}$ is Π_2^0 as

$$e \in Tot \Leftrightarrow \forall n \exists s x \in W_e^s.$$

• $Fin = \{e : W_e \text{ is finite}\}$ is Σ^0_2 as

$$e \in Fin \Leftrightarrow \exists n \forall y \forall s \ (y < x \vee y \notin W_e^s).$$

• $\{(a, b, c, d, e) : \exists x, y \forall z \ az^3 - bxz = cx^2 - dxy^2 + ey^3\}$ is Σ^0_2.

• $\{e : W_e \text{ is recursive}\}$ is Σ^0_3 as W_e is recursive if and only there is an i such that $\neg W_e = W_i$. Thus W_e is recursive iff and only if

$$\exists i \forall x \ ((x \in W_e \vee x \in W_i) \wedge (x \notin W_e \vee x \notin W_i)).$$

This is equivalent to

$$\exists i \forall x (\underbrace{\underbrace{\exists s(x \in W_e^s \vee x \in W_i^s)}_{\Sigma^0_1} \wedge \underbrace{\forall s(x \notin W_e^s \vee x \notin W_i^s)}_{\Pi^0_1}}_{\Pi^0_2}).$$

Thus $\{e : W_e \text{ is recursive}\}$ is Σ^0_3.

Complete Sets

Definition 8.17 For Γ be Σ^0_n or Π^0_n. We say that X is Γ-*complete* if $X \in \Gamma$ and for all $Y \in \Gamma$, $Y \leq_m X$.

By 8.11 K and H are Σ^0_1-complete.

Proposition 8.18 *Tot is Π^0_2-complete.*

Proof Let X be Π^0_2. Then there is a recurisve $R(x, y, z)$ such that

$$x \in X \Leftrightarrow \forall y \exists z \ R(x, y, z).$$

Let $f(x, y) = \begin{cases} 1 & \exists z \ R(x, y, z) \\ \uparrow & \text{otherwise} \end{cases}$. Clearly f as computable as on input x, y we search for a z such that $R(x, y, z)$. If there is one we will evenutally find it and halt. If not we will search forever.

By the parameterization theorem there is a recursive function $k(x)$ such that

$$\phi_{k(x)}(y) = f(x, y).$$

But then $x \in X$ if and only if $\phi_{k(x)}$ is total.

Proposition 8.19 *Fin is Σ^0_2-complete.*

Proof

Let $X \in \Sigma^0_2$. Suppose $x \in X$ if and only if $\exists y \forall z \ R(x, y, z)$ where R is recursive.

Let

$$f(x,y) = \begin{cases} 1 & \forall w \leq y \exists z \ \neg R(x,w,z) \\ \uparrow & \text{otherwise.} \end{cases}$$

By the parameterization theorem there is a total recursive g such that $\phi_{g(x)}(y) = f(x,y)$.

Then $W_{g(x)} = \{y : \forall w < y \exists z \ \neg R(x,w,z)\}$. Thus $x \in X$ if and only if $g(x) \in Fin$.

Definition 8.20 Let $U \subset \mathbb{N}^2$. For $e \in \mathbb{N}$, let $U_e = \{x : (e,x) \in U\}$. We say that U is Γ-*universal* if $U \in \Gamma$ and for any $X \in \Gamma$, there is an e such that $X = U_e$.

Clearly every Γ-universal set is Γ-complete

Lemma 8.21 *For $\Gamma = \Sigma^0_n$ or Π^0_n, there is U_Γ which is Γ-universal.*

Proof Let $U_{\Sigma^0_1} = \{(e,n) : n \in W_e\} = \{(e,n) : \Psi(e,n) \downarrow\}$ is Σ^0_1 and clealy universal.

If $U_{\Sigma^0_n}$ is universal for Σ^0_n then $\mathbb{N} \setminus U_{\Sigma^0_n}$ is universal for Π^0_n.

Let $U_{\Pi^0_n}$ be universal Π^0_n. Let $\pi : \mathbb{N}^2 \to \mathbb{N}$ be a recursive bijection. Then

$$\{(e,n) : \exists y (e, \pi(x,y)) \in U_{\Pi^0_n}\}.$$

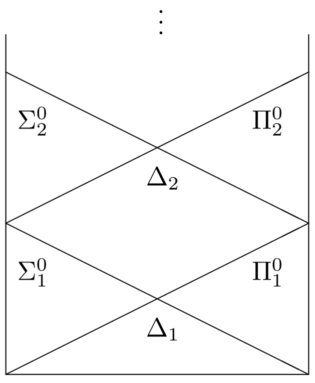
is universal Σ^0_{n+1}.

Proposition 8.22 *The universal Σ^0_n set is not Π^0_n.*

Proof Let U be the universal Σ^0_n set. Let $V = \{e : (e,e) \notin U\}$. If U were Π^0_n then V would be Σ^0_n. In that case there would be an e_0 such that $V = U_{e_0}$. But then

$$e_0 \in V \Leftrightarrow (e_0,e_0) \notin U \Leftrightarrow e_0 \notin U_{e_0} \Leftrightarrow e_0 \notin V.$$

Thus $\Sigma^0_n \supset \Delta_n$ and $\Pi^0_n \supset \Delta_n$. This gives the following picture of the arithemtic hierarchy.



9 Further Topics in Computability Theory

In this section we will take a quick look at several other important ideas in computability theory.

Rice's Theorem

Definition 9.1 We say that $X \subset \mathbb{N}$ is an *index set* if whenever $\phi_i = \phi_j$, $i \in X \Leftrightarrow j \in X$.

Tot is an index set. We will show below that K is not an index set.

Theorem 9.2 (Rice's Theorem) *If X is an index set then either $X = \emptyset$, $X = \mathbb{N}$, $K \leq_m X$ or $\mathbb{N} \setminus K \leq_m X$. In particular the only recursive index sets are $\mathbb{N}$ and $\emptyset$.*

Proof Suppose $X \neq \emptyset$ and $X \neq \mathbb{N}$. Choose e_0 such that for all x, $\phi_{e_0}(x) \uparrow$.
case 1 $e_0 \notin X$.

Let $e_1 \in X$. Then $\phi_{e_1} \neq \phi_{e_0}$. There is a total recursive f such that forall x, y,

$$\phi_{f(x)}(y) = \begin{cases} \phi_{e_1}(y) & x \in K \\ \uparrow & x \notin K. \end{cases}$$

Let $G(x, y)$ be the partial function computed as follows, enumerate K until we see that $x \in K$ (if $x \notin K$, this search will never terminate), once we see that $x \in K$ start computing $\phi_{e_1}(y)$. Apply the Parameterization Lemma to G to get g such that $\phi_{g(x)}(y) = G(x, y)$.

If $x \in K$, then $\phi_{g(x)} = \phi_{e_1}$ while if $x \notin K$, then $\phi_{g(x)} = \phi_{e_0}$, the everywhere undefined function. Since X is an index set if $\phi_{g(x)} = \phi_{e_i}$, then $g(x) \in X \Leftrightarrow e_i \in X$. Thus $x \in K \Leftrightarrow g(x) \in X$.

case 2: $e_0 \notin X$.

Use the fact that $\mathbb{N} \setminus X$ is an index set. By case 1, $K \leq_m \mathbb{N} \setminus X$. Thus $\mathbb{N} \setminus K \leq_m X$.

The Recursion Theorem

Suppose you are given the task of writing a computer program Q which we will call a "modifier". The program Q will compute a total recursive function f. The goal of Q is to insure that for $\phi_e \neq \phi_{f(e)}$ for any input e. Intuitively Q takes as input a program P_e and outputs a modified program $P_{f(e)}$ and Q's goal is to insure that these programs do not compute the same partial recursive function. Is there such a program Q?

One at first might think this is easy as Q could do something like output $P_{f(e)}$ where we first run P_e and then add one to the output. This almost works. If there is any x such that P_e halts on input x, then $\phi_e(x) \neq \phi_{f(e)}(x)$. However suppose we choose e an index for the everywhere divergent function. Then $P_{f(e)}$ is also the everywhere divergent function. Perhaps you would expect that if one were a little more clever one could avoid this problem. The Recursion Theorem says that this is not the case.

Theorem 9.3 (Kleene's Recursion Theorem) *Suppose $f : \mathbb{N} \to \mathbb{N}$ is a total recurstive function. There is a number e such that $\phi_e = \phi_{f(e)}$. In particular there is an e such $W_e = W_{f(e)}$.*

Proof Consider the partial recursive function

$$F(x,y) = \begin{cases} \phi_{\phi_x(x)}(y) & \text{if } \phi_x(x)\downarrow \\ \uparrow & \text{otherwise.} \end{cases}$$

By the Parameterization Lemma there is a total recursive function d such that

$$\phi_{d(x)}(y) = F(x,y).$$

Choose n such that $\phi_n = f \circ d$. Let $e = d(n)$. Since d and f are total, ϕ_n is total. Thus $\phi_n(n)$ converges and $\phi_{d(n)} = \phi_{\phi_n(n)}$. Hence

$$\phi_n = \phi_{d(n)} = \phi_{\phi_n(n)} = \phi_{f(d(n))} = \phi_{f(e)}$$

as desired.

The following is typical of the many odd corollaries of the recursion theorem.

Corollary 9.4 *There is an e such that $W_e = \{e\}$.*

Proof By the Parameterization Lemma there is a total recursive function f such that

$$\phi_{f(x)}(y) = \begin{cases} 1 & y = x \\ \uparrow & \text{otherwise.} \end{cases}$$

Thus $W_{f(x)} = \{x\}$ for all x. By the Recursion Theorem there is an e such that

$$W_e = W_{f(e)} = \{e\}.$$

Thus there is a program P_e which on input x checks "Is x a code for my own program?" and halts if and only if it is. Such a program can be written in **any** programming language.

We can now answer a question raised above.

Corollary 9.5 *K is not an index set.*

Proof Suppose e as in Corollary 9.4. Since $W_e = \{e\}$, $e \in K$. On the other hand if $\phi_i = \phi_e$ and $i \neq e$, then $\phi_i(i)\uparrow$. Thus $i \notin K$.

Recursively Inseparable r.e. Sets

Definition 9.6 Suppose A and B are recursively enumerable and $A \cap B = \emptyset$. We say that A and B are *recursively inseparable* if there is no recursive set C such that $A \subseteq C$ and $B \cap C = \emptyset$.

Theorem 9.7 *There is a pair of recursively inseparable recursively enumerable sets.*

Proof Let $A = \{e : \phi_e(e) = 0\}$ and let $B = \{e : \phi_e(e) = 1\}$. Suppose C is recursive, $A \subseteq C$ and $C \cap B = \emptyset$. Let ϕ_n be the characteristic function of C. Then

$$n \in C \Rightarrow \phi_n(n) = 1 \Rightarrow n \in B \Rightarrow n \notin C.$$

On the other hand

$$n \notin C \Rightarrow \phi_n(n) = 0 \Rightarrow n \in A \Rightarrow n \in C.$$

Thus we have a contradiction.

Simple Sets

We will give an example of a non-recursive recursively enumerable set which is not Σ_1-complete.

Definition 9.8 We say that a recursively enumerable set A is *simple* if
i) $\mathbb{N} \setminus A$ is infinite but
ii) $\mathbb{N} \setminus A$ contains no infinite recusively enumerable set. Thus A is simple if and only if $\mathbb{N} \setminus A$ is infinite and for any e if W_e is infinite, then $A \cap W_e$ is nonempty.

Theorem 9.9 (Post) *There is a simple recusively enumerable set.*

Proof Let $B = \{(e, s, x) : x \in W_e^s \wedge x > 2e\}$. Let f be a partial recursive function

$$f(e, s) = \mu x \ (e, s, x) \in B$$

ie. $f(e, s)$ is the least x such that $(e, s, x) \in b$ and $f(e, s) \uparrow$ if no such x exists. Note that if $f(e, s) \downarrow$ and $t > s$, then $f(e, s) = f(e, t)$. We call this common value m_e.

Let A be the range of f. Then A is recursively enumerable and $A = \{m_e : f(e, s) \downarrow \text{ for some } s\}$. If $n \in A$ and $n \leq N$, then $n = m_e$ for some $e < N/2$. Thus $\mathbb{N} \setminus A$ is infinite.

Suppose W_e is infinite. There is $x \in W_e$ such that $x > 2e$, thus $f(e, s) \downarrow$ for large enough s. But $f(e, s) \in W_e \cap A$. Thus $W_e \not\subseteq A$. So $\mathbb{N} \setminus A$ contains no infinite recursively enumerable sets.

Definition 9.10 A recursively enumerable set A is *creative* if and only if there is a total recursive F such that $F(e) \in A$ if and only if $F(e) \in W_e$ for all e.

Proposition 9.11 *i) If A is creative, then A is not recursive.*

ii) K is creative.

iii) Any complete recursively enumerable set is creative.

Proof

i) If $W_e = \mathbb{N} \setminus A$, then

$$F(e) \in A \Leftrightarrow F(e) \in W_e \Leftrightarrow F(e) \notin A$$

a contradiction.

ii) Let $F(e) = e$. Then $F(e) \in K$ if and only if $e \in W_e$.

iii) Since $K \leq_m A$, there is a total recursive f such that $e \in K$ if and only if $f(e) \in A$. By the Parameterization Lemma, there is a total recursive g such that

$$\phi_{g(e)}(x) = \phi_e(f(x))$$

for all e and x. Then

$$\begin{array}{rcl} f(g(e)) \in A & \Leftrightarrow & g(e) \in K \\ & \Leftrightarrow & \phi_{g(e)}(g(e)) \downarrow \\ & \Leftrightarrow & \phi_e(f(g(e)) \downarrow \\ & \Leftrightarrow & f(g(e)) \in W_e. \end{array}$$

Thus $F = f \circ g$ shows that A is creative.

Proposition 9.12 *If A is creative, then $\mathbb{N} \setminus A$ contains an infinite recursively enumerable set.*

Proof Suppose F is a total recursive function such that $F(e) \in A$ if and only if $F(e) \in W_e$. There is a total recursive function f such that

$$W_{f(n)} = W_n \cup \{F(n)\}$$

for all n.

Suppose $W_e \subseteq \mathbb{N} \setminus A$. Since $F(e) \in W_e$ if and only if $F(e) \in A$, $F(e) \notin W_e$ and $W_{f(e)} \subseteq \mathbb{N} \setminus A$. Thus $W_e \subset W_{f(e)} \subseteq \mathbb{N} \setminus A$.

Choose e_0 such that $W_{e_0} = \emptyset$. Let

$$\begin{array}{rcl} h(0) & = & e_0 \\ h(n+1) & = & f(h_n) \end{array}$$

Then

$$W_{h_0} \subset W_{h_1} \subset W_{h_2} \subset \ldots \mathbb{N} \setminus A$$

and

$$\bigcup_{n=0}^{\infty} W_{h(n)} = \{x : \exists n \; x \in W_{h(n)}\}$$

is an infinite recursively enumerable subset of $\mathbb{N} \setminus A$.

Corollary 9.13 *If A is simple, then A is not complete.*

Proof A is not creative and, hence, not complete.

Kolmogorov Randomness

For $x \in \mathbb{N}$ let $|x|$ be the length of the binary expansion of x. Then $|x| = \lceil \log_2(x+1) \rceil$.

We say that $\langle n, m \rangle$ is a *description* of x if $\phi_n(m) = x$. We say that k codes a description of x if $\pi(n,m) = k$ where $\pi(n,m) = 2^n(2m+1) - 1$ is our usual pairing function $\pi : \mathbb{N}^2 \to \mathbb{N}$.

Definition 9.14 The *Kolmogorov complexity* of x is

$$K(x) = \min\{|k| : k \text{ codes a description of } x\}.$$

We say that x is random if $K(x) \geq |x|$.

Proposition 9.15 *$\{x : x$ is not random $\}$ is recursively enumerable.*

Proof x is not random if and only if

$$\exists n, m\ (|\pi(n,m)| < |x| \wedge \phi_n(m) = x\}.$$

Proposition 9.16 *There are random x.*

Proof The key observation is that

$$|\{x : |x| \leq M\}| = 2^M$$

for any M. Thus for any $M \in \mathbb{N}$, there are at most 2^{M-1} descriptions with codes k where $|k| < M$. Thus

$$\{x : |x| \leq M \text{ and } K(x) < M\}| \leq 2^{M-1}$$

and at least half the numbers of length at most M are random!

Proposition 9.17 *$\{x : x$ is not random$\}$ is simple.*

Proof Suppose A is an infinite recursively enumerable set of random numbers. Let $f : \mathbb{N} \to A$ be the function $f(m) =$ first x enumerated into A with $|x| \geq m$. Let $f = \phi_n$. Pick $m > 2^n$.

$$m \leq |f(m)| \leq K(f(m)) \leq |\pi(n,m)| \leq |2^n(2m+1)| \approx n + |m| < 2|m|$$

a contradiction.

The simplicity of the set of non-random elements has an amazing metamathematical consequence. We know there are infinitely many random numbers. Consider

$$\{n \in \mathbb{N} : \text{ we can prove in set theory that } n \text{ is random}\}.$$

This is a recursively enumerable subset of the complement of a simple set. Hence must be finite! Thus for most random numbers there is no proof that they are random.

Part III

Incompleteness

10 Gödel's Incompleteness Theorem

The following three problems can be considered the basic problems in the foundations of mathematics.

1) Does every mathematical truth about the natural numbers have a meaningful finitistic proof? While it is arguable what a "finitistic proof" is, one precise way of saying this is that there is a natural set of axioms from which we can derive all truths about the natural numbers. For example Peano Arithmetic (PA) would be a good candidate. Is every sentence that is true in $\mathbb{N}$ provable in PA?

2) **Hilbert's Conservation Program**: If a mathematical truth can be proved by strong methods (say using set theoretic methods), then it can be proved by finitistic methods.

3) **Hilbert's Consistency Program**: We should be able to give a finitistic proof of the consistency of our methods.

Gödel showed that all of this is impossible.

Theorem 10.1 (First Incompleteness Theorem) *There is a sentence ϕ such that $\mathbb{N} \models \phi$ and $\mathrm{PA} \not\vdash \phi$. Indeed, if T is a recursive theory such that $T \supseteq \mathrm{PA}$ and $\mathbb{N} \models T$, then there is a sentence ϕ such that $T \not\vdash \phi$ and $T \not\vdash \neg\phi$.*[5]

(Second Incompleteness Theorem) *Let T be as above. Then T does not prove the consistency of T.*

The First Incompleteness theorem shows that i) will fail. The Second Incompleteness theorem shows that iii) fails. Since using set theory we can show that Peano Arithmetic is consistent, iii) shows that the conservation program fails as well.

Let $\mathcal{L} = \{+, \cdot, <, 0, 1\}$ be the language of arithmetic.

For $n \in \mathbb{N}$, let $\widehat{n}$ denote the $\mathcal{L}$-term $\underbrace{1 + \ldots + 1}_{n-\text{times}}$.

Definition 10.2 We say that an $\mathcal{L}$- theory T *represents* $A \subseteq \mathbb{N}^k$ if there is an $\mathcal{L}$-formula $\phi_A(v_1, \ldots, v_k)$ such that:

i) if $(n_1, \ldots, n_k) \in A$, then $T \vdash \phi(\widehat{n}_1, \ldots, \widehat{n}_k)$;

and

ii) if $(n_1, \ldots, n_k) \notin A$, then $T \vdash \neg\phi(\widehat{n}_1, \ldots, \widehat{n}_k)$.

[5]There is an issue here of what we mean for a set of sentences to be recursive (or later in this section, recursively enumerable or arithmetic). To make these precise we need the idea of Gödel codes from §11, but for the moment, as we have done in earlier sections, we use the informal notion that there is an algorithm that decides if $\phi \in T$.

Further we say that T represents $f : \mathbb{N}^k \to \mathbb{N}$ if there is an $\mathcal{L}$-formula $\phi_f(v_1, \ldots, v_k, w)$ such that:

i) if $f(n_1, \ldots, n_k) = m$ then $T \vdash \phi_f(\widehat{n}_1, \ldots, \widehat{n}_k, \widehat{m})$

and

ii) if $f(n_1, \ldots, n_k) \neq m$ then $T \vdash \neg\phi_f(\widehat{n}_1, \ldots, \widehat{n}_k, \widehat{m})$

The key step is to show that Peano Arithmetic represents all primitive recursive functions–indeed, all recursive functions. In fact, we need only a very weak fragment of Peano arithmetic, that we will call PA$^-$. Let PA$^-$ be the $\mathcal{L}$-theory asserting the following basic properties of the natural numbers:

- addition and multiplication are commutative and associative;
- the distributive law;
- $\forall x\ (x + 0 = x \wedge x \cdot 0 = 0 \wedge x \cdot 1 = x)$;
- $<$ is a linear order;
- $\forall x\ (x = 0 \vee 0 < x)$;
- $\forall x \neg(0 < x \wedge x < 1)$;
- $\forall x \forall y \forall z\ (x < y \to (x + z < y + z)$;
- $\forall x \forall y \forall z\ [(x < y \wedge 0 < z) \to x \cdot z < y \cdot z]$;
- $\forall x \forall y\ [x < y \to \exists z\ x + z = y]$.

Models of PA$^-$ are the nonnegative parts of discrete ordered rings.

Lemma 10.3 (Representation Lemma) PA$^-$ *represents every primitive recursive function.*

If $f : \mathbb{N}^k \to \{0, 1\}$ is the characteristic function of $A \subseteq \mathbb{N}^k$ and $\phi_f(\overline{v}, w)$ represents f, then $\phi_f(\overline{v}, \widehat{1})$ represents A. Thus PA$^-$ also represents all primitive recursive relations.

For some applications, we can get by with a simpler corollary.

Corollary 10.4 (Weak Representation Lemma) *The graph of every primitive recursive function is definable in $\mathcal{N}$.*

Proof If ϕ represents f, then

$$f(\overline{m}) = n \Leftrightarrow \mathbb{N} \models \phi(\overline{m}, n).$$

So, ϕ defines the graph of f.

We will postpone the proof of the Representation Lemma for the moment and first show how we can apply it and the results of §7 to prove the First Incompleteness Theorem.

Suppose $A \subseteq \mathbb{N}$ is Σ^0_n. Say

$$x \in A \Leftrightarrow \exists y_1 \forall y_2 \ldots Q y_n\ R(x, \overline{y})$$

where R is recursive and Q is $\exists$ if n is odd and $\forall$ if n is even. Let $R = W_e$. Then

$$\overline{x} \in R \Leftrightarrow \exists s\ T(e, \overline{x}, s)$$

where T Kleene's T-predicate. Then

$$x \in A \Leftrightarrow \exists y_1 \forall y_2 \ldots Q y_n \exists s\ T(e, x, \overline{y}, s)$$

Let $\psi_A(v)$ be the formula

$$\exists y_1 \forall y_2 \ldots Q y_n \exists s\ \phi_T(\widehat{e}, v, \overline{y}, s)$$

where ϕ_T is the formula which represents T.

It is clear that

$$n \in A \Leftrightarrow \mathbb{N} \models \psi_A(\widehat{n}),$$

i.e., ψ_A defines A. We have proved that every arithmetic set is definable. Note that we proved the converse in Exercise 8.16.

Proposition 10.5 *$A \subseteq \mathbb{N}^k$ is arithmetic if and only if it is definable in $\mathbb{N}$.*

Recall that $\mathrm{Th}(\mathbb{N}) = \{\psi : \mathbb{N} \models \psi\}$. The function

$$n \mapsto \psi_A(\widehat{n})$$

gives a many-one reduction of A to $\mathrm{Th}(\mathbb{N})$.

We can now deduce a strong form of the First Incompleteness Theorem.

Theorem 10.6 $\mathrm{Th}(\mathbb{N})$ *is not arithmetic*

Proof Suppose $\mathrm{Th}(\mathbb{N})$ were Σ^0_n. By Proposition 8.22, there is a Σ^0_{n+1} set A which is not Σ^0_n. By the above arguments, $A \leq_m \mathrm{Th}(\mathbb{N})$ and by Proposition 8.15 vi), $A \in \Sigma^0_n$, a contradiction.

Corollary 10.7 *If $T \supseteq \mathrm{PA}^-$ is a recursive $\mathcal{L}$-theory such that $\mathbb{N} \models T$, there is ϕ such that $\mathbb{N} \models \phi$ and $T \not\vdash \phi$. In particular,* PA *is incomplete.*

Proof If T is recursive, then $\{\phi : T \vdash \phi\}$ is a recursively enumerable subset of $\mathrm{Th}(\mathbb{N})$. Since $\mathrm{Th}(\mathbb{N})$ is not recursively enumerable, there is a sentence ϕ such that $\mathbb{N} \models \phi$ but $T \not\vdash \phi$.

Note that these results needed only the Weak Representation Lemma. The next version of the Incompleteness Theorem weakens the assumption that $\mathbb{N} \models T$, but assumes $T \supseteq \mathrm{PA}$ and uses more of the force of the Representation Lemma.

Let $T \supseteq \mathrm{PA}$ be recursively axiomatized. Let

$$P(T) = \{\phi : T \vdash \phi\} \text{ and } R(T) = \{\phi : T \vdash \neg\phi\}$$

be the sentences provable and refutable from T. Note that since T is recursively axiomatized, $P(T)$ and $R(T)$ are recursively enumerable. Moreover, if T is consistent, then $P(T)$ and $R(T)$ are disjoint.

Theorem 10.8 (Rosser's Incompleteness Theorem) *If $T \supseteq \mathrm{PA}$ is consistent and recursively axiomatizable, then $P(T)$ and $R(T)$ are recursively inseparable. It follows that T is incomplete.*

Proof We argue that later point first. If T is complete and ϕ is a sentence, then

$$\phi \notin P(T) \Leftrightarrow \phi \in R(T).$$

Thus $P(T)$ is co-recursively enumerable and hence recursive. But this contradicts the recursive inseparability of $P(T)$ and $R(T)$.

Note that to prove the recursive inseparability of $P(T)$ and $R(T)$ it suffices to show that $P(\mathrm{PA})$ and $R(\mathrm{PA})$ are recursively inseparable since

$$P(\mathrm{PA}) \subseteq P(T), R(\mathrm{PA}) \subseteq R(T) \text{ and } P(T) \cap R(T) = \emptyset.$$

Thus any set that separates $P(T)$ and $R(T)$ also separates $P(\mathrm{PA})$ and $R(\mathrm{PA})$.

For $i = 0, 1$ let $A_i(e, x, s)$ be the primitive recursive predicate asserting that "$\phi_e(x)$ halts in at most s steps with output i" and let $\psi_i(u, v, w)$ be an $\mathcal{L}$-formula representing A_i in PA^-.

Let $\theta_0(x)$ be the formula

$$\exists y \ \ (\psi_0(x, x, y) \wedge \forall z < y \neg\psi_1(x, x, z))$$

and let $\theta_1(x)$ be the formula

$$\exists y \ \ (\psi_1(x, x, y) \wedge \forall z \leq y \neg\psi_0(x, x, z)).$$

Intuitively, $\theta_0(e)$ says that we find a witness that $\phi_e(e) = 0$ at least as soon as we find a witness that $\phi_e(e) = 1$ and $\theta_1(e)$ says that we see $\phi_e(e) = 1$ before we see $\phi_e(e) = 0$.[6]

Note that $\mathrm{PA}^- \vdash \forall \neg(\theta_0(x) \wedge \theta_1(x))$.

Claim $\mathrm{PA} \vdash \forall x \, [\exists y \ \ (\psi_0(x, x, y) \vee \psi_1(x, x, y)) \rightarrow (\theta_0(x) \vee \theta_1(x))]$

If $\exists y \ \ (\psi_0(x, x, y) \vee \psi_1(x, x, y))$ then, using induction in PA, there is a least y_0 such that

$$\psi_0(x, x, y_0) \vee \psi_1(x, x, y_0).$$

If $\psi_0(x, x, y_0)$, then $\theta_0(x)$. Otherwise $\theta_1(x)$.

Let $A = \{e \in \mathbb{N} : \phi_e(e) = 0\}$ and $B = \{e \in \mathbb{N} : \phi_e(e) = 1\}$. By Theorem 9.7 A and B are recursively inseparable.

Suppose for purposes contradiction that there is a recursive set of sentences C such that $P(\mathrm{PA}) \subseteq C$ and $C \cap R(\mathrm{PA}) = \emptyset$. Let

$$D = \{e \in \mathbb{N} : \theta_0(\widehat{e}) \in C\}.$$

Clearly D is recursive. We claim that D separates A and B.

[6]Note that if $\phi_e(e)$does not halt it is possible that in a nonstandard model $\mathcal{M}$ we might find nonstandard s and t with

$$\mathcal{M} \models \psi_0(e, e, s) \wedge \psi_1(e, e, t).$$

Let $e \in A$. There is $s \in \mathbb{N}$ such that $A_0(e, e, s)$ and $\neg A_1(e, e, t)$ for all $t \leq s$. Since ψ_i represents A_i,

$$\mathrm{PA}^- \vdash \psi_0(e, e, s) \text{ and for all } t \leq s \ \mathrm{PA}^- \vdash \psi_1(e, e, t).$$

It follows that $\mathrm{PA}^- \vdash \theta_0(e)$ and $e \in D$.

A similar argument shows that if $e \in B$, then $\mathrm{PA}^- \vdash \theta_1(e)$. Hence $\mathrm{PA} \not\vdash \theta_0(e)$, so $e \notin D$. Thus D separates A and B, a contradiction since they are recursively inseparable.

Σ_1-Formulas

We now start doing the preparation we need to prove the Representation Lemma.

Definition 10.9 We say that an $\mathcal{L}$-formula $\phi(\overline{v})$ is a Δ_0-formula if it is in the smallest collection of formulas $\mathcal{C}$ such that:

i) every atomic formula is in $\mathcal{C}$;

ii) if $\phi \in \mathcal{C}$, then $\neg\phi \in \mathcal{C}$;

iii) if $\phi \in \mathcal{C}$ and $\psi \in \mathcal{C}$, then $\phi \wedge \psi \in \mathcal{C}$;

iv) if $\phi \in \mathcal{C}$,v is variable and t is an $\mathcal{L}$-term not involving v, then $\exists v\ (v < t \wedge \phi) \in \mathcal{C}$ and $\forall v\ (v < t \rightarrow \phi) \in \mathcal{C}$.

We abbreviate the later two formulas as $\exists v < t\ \phi$ and $\forall v < t\ \phi$.

Exercise 10.10 Prove that if $\phi(v_1, \ldots, v_k) \in \Delta_0$, then

$$\{(n_1, \ldots, n_k) : \mathbb{N} \models \phi(\widehat{n}_1, \ldots, \widehat{n}_k)\}$$

is a primitive recursive predicate.

Definition 10.11 We say $\phi(\overline{x})$ is a Σ_1-formula, if there is a Δ_0-formula $\psi(\overline{x}, \overline{y})$ such that $\phi(\overline{x})$ is $\exists \overline{y}\ \psi(\overline{x}, \overline{y})$.

As a first step toward proving the Representation Lemma, we will prove that for every primitive recursive function there is a Σ_1-formula defining its graph. That is for each primitive recursive $f : \mathbb{N} \to \mathbb{N}$ there is a Σ_1 formula $\phi(x_1, \ldots, x_k, y)$ such that $f(n_1, \ldots, n_k) = m \Leftrightarrow \mathbb{N} \models \phi(n_1, \ldots, n_k, m)$.

This will be proved by induction. For the basic functions this is easy.

i) The graph of $x \mapsto 0$ is defined by the formula $y = 0$.

ii) The graph of $x \mapsto x + 1$ is defined by the formula $y = x + 1$.

iii) The graph of $(x_1, \ldots, x_m) \mapsto x_i$ is defined by $y = x_i$.

Lemma 10.12 *Suppose that the graphs of the functions the functions $g_1, \ldots, g_m : \mathbb{N}^k \to \mathbb{N}$ and $h : \mathbb{N} \to \mathbb{N}$ have Σ_1-definable graphs and $f : \mathbb{N}^k \to \mathbb{N}$ is defined by $f(\overline{x}) = h(g_1(\overline{x}), \ldots, g_m(\overline{x}))$. Then f has a Σ_1-definable graph.*

Proof Let $\exists \overline{z}_i \phi_i(\overline{x}, y, \overline{z}_i)$ define the graph of g_i and let $\exists \overline{w} \psi(\overline{u}, y, \overline{w})$ define the graph of h. Then $f(\overline{x}) = y \Leftrightarrow$

$$\exists \overline{z}_1 \ldots \exists \overline{z}_m \exists w_1 \ldots \exists w_m \exists \overline{u}\ \left[\phi_1(\overline{x}, w_1, \overline{z}_1) \wedge \ldots \wedge \phi_m(\overline{x}, w_m, \overline{z}_m) \wedge \psi(\overline{w}, y, u)\right].$$

We have shown that the class of functions with Σ_1-definable graphs is closed under composition. We need to show it is closed under primitive recursion. To do that we will need a new method of coding sequences where it is easy to show the graph is Δ_0.

Gödel's β-function

Let *Seq* be the set of finite sequences of elements of $\mathbb{N}$. We define $\beta : \mathbb{N}^3 \to Seq$, such that $\beta(u, v, w)$ is the sequence $(a_0, \dots, a_{w-1})$ where

$$a_i = u \bmod (i+1)v + 1 \text{ for } i = 0, \dots, w-1$$

and $\beta(u, v, w)$ is the empty sequence for $w = 0$.

Let $\Psi(u, v, w, i, x)$ be the formula

$$i < w \wedge 0 \leq x < (i+1)v + 1 \wedge \exists y \leq u \; y((i+1)v+1) + x = u.$$

Then $\Psi(u, v, w, i, x)$ expresses that x is the i^{th} element in the sequence $\beta(u, v, w)$. Note that Ψ is Δ_0. We will write

$$\beta(u, v, w)_i = x$$

for $\Psi(u, v, w, i, x)$. While it is easy to express that $\beta(u, v, w)_i = x$, it is not so easy to see that every sequence is coded in this way.

We need the following standard lemma from number theory.

Lemma 10.13 (Chinese Remainder Theorem) *Suppose $m_1, \dots, m_n$ are relatively prime. Then for any $a_1, \dots, a_n$ there is an x such that $x \equiv a_i (\text{mod } m_i)$ for $i = 1, \dots, n$.*

Proof Let

$$M_i = \prod_{j \neq i} m_j.$$

Since M_i and m_i are relatively prime, we can find b_i such that $b_i M_i \equiv 1 (\text{mod } m_i)$. Let

$$x = \sum_{i=1}^{n} a_i b_i M_i.$$

Since $m_i | M_j$ for $j \neq i$, $x \equiv a_i b_i M_i (\text{mod } m_i)$. Thus $x \equiv a_i (\text{mod } m_i)$ for $i = 1, \dots, n$.

Lemma 10.14 *For any sequence $\sigma = (a_0, \dots, a_{w-1})$ there are u and v such that $\beta(u, v, w) = \sigma$.*

Proof Let $n = \max(w, a_0, \ldots, a_{w-1})$ and let $v = n!$. We claim that

$$v+1, 2v+1, \ldots, wv+1$$

are relatively prime. Suppose p is prime and $p|iv+1$ and $p|jv+1$ where $j > i > 0$ Then $p|(j-i)v$ Thus $p|(j-i)$ or $p|v$ and, since $(j-i)|v$, $p|v$. But then $p \not| iv+1$. Thus $v+1, \ldots, wv+1$ are relatively prime.

By the Chinese Remainder Theorem there is a number u such that

$$u \equiv a_i (\text{mod } (i+1)v+1) \text{ for } i = 0, \ldots, w-1.$$

Corollary 10.15 *Suppose ϕ is a Σ_1-formula, v is a variable and t is an $\mathcal{L}$-term not involving v. There is a Σ_1-formula ψ such that*

$$\mathbb{N} \models \psi \leftrightarrow \forall v < t\ \phi.$$

Proof Suppose $\forall v < t\ \phi$ is

$$\forall v < t \exists y_1 \ldots \exists y_n\ \theta(\overline{x}, \overline{y}, v)$$

where θ is Δ_0. This is clearly equivalent to

$$\forall v < t \exists y \exists y_1 < y \ldots \exists y_n < y\ \theta(\overline{x}, \overline{y}, v).$$

Thus without loss of generality we may assume $\forall v < t\phi$ is

$$\forall v < t \exists y \theta(\overline{x}, y, v)$$

where θ is Δ_0.

Also note that for any formula ϕ,

$$\forall v < t\ \phi \Leftrightarrow \exists z\ (z = t \wedge \forall v < z\phi).$$

Thus it will suffice to prove that if θ is Δ_0, then $\forall v < z \exists y\ \theta(\overline{x}, y, v)$ is equivalent in $\mathbb{N}$ to a Σ_1-formula.

Informally,

$$\forall v < z \exists y\ \theta(\overline{x}, y, v) \Leftrightarrow \exists \sigma \in Seq\ \forall v < z\ \theta(\overline{x}, \sigma_v, v)$$

We can make this formal using Gödel's β-function. In $\forall v < z \exists y\ \theta(\overline{x}, y, v)$ is equivalent to

$$\exists u \exists v \forall v < z \exists y < u\ \left(\beta(u, v, z)_i = y \wedge \theta(\overline{x}, y, v)\right).$$

Because $\beta(u, v, z)_i = y$ has a Δ_0-definition, this is a Σ_1-formula.

Corollary 10.16 *Let $\mathcal{C}$ be the smallest class of formulas containing the Δ_0 formulas and closed under $\wedge$, $\vee$, bounded quantification (i.e., $\forall v < t$ and $\exists v < t$), and existential quantification. If $\phi \in \mathcal{C}$ there is $\psi \in \Sigma_1$ such that*

$$\mathbb{N} \models \phi \leftrightarrow \psi.$$

We can now complete the proof that primitive recursive functions have Σ_1-definable graphs.

Corollary 10.17 *If $f : \mathbb{N}^k \to \mathbb{N}$ is primitive recursive, then the graph of f is Σ_1-definable.*

Proof We have shown that the basic functions are Σ_1-definable and that the class of Σ_1-definable functions is closed under composition. We need only show that it is closed under primitive recursion.

Suppose $g : \mathbb{N}^k \to \mathbb{N}$ and $h : \mathbb{N}^{k+2} \to \mathbb{N}$ have graphs defined by the Σ_1-formulas ϕ and ψ respectively and $f : \mathbb{N}^{k+1} \to \mathbb{N}$ is defined by

$$\begin{array}{rcl} f(\overline{x}, 0) & = & g(\overline{x}) \\ f(\overline{x}, y+1) & = & h(\overline{x}, y, f(\overline{x}, y)). \end{array}$$

Then $f(\overline{x}, y) = z$ if and only if

$$\exists \sigma \in Seq \ [g(\overline{x}) = \sigma_0 \wedge \forall i < y \sigma_{i+1} = h(\overline{x}, i, \sigma_i) \wedge \sigma_y = z] .$$

Using the β-function and the Σ_1-definitions of g and h, this is equivalent in $\mathbb{N}$ to

$$\begin{array}{l} \exists u \exists v \big[(\exists w < u(\beta(u, v, y+1)_0 = y \wedge \phi(\overline{x}, w)) \wedge \forall i \leq y+1 \\ \big(\exists w_1 < u \exists w_2 < u(\beta(u, v, y+1)_i = w_1 \wedge \beta(u, v, y+1)_{i+1} = w_2 \wedge \psi(\overline{x}, i, w_1, w_2)\big) \\ \wedge \beta(u, v, y+1)_y = z \big]. \end{array}$$

Using the Δ_0-definition of the graph of the β-function and Corollary 10.16, we see that the graph of f is Σ_1-definable.

We can extend this result to all partial recursive functions.

Corollary 10.18 *Suppose $f : \mathbb{N}^k \to \mathbb{N}$ is partial recursive. Then the graph of f is Σ_1-definable.*

Proof Let $f = \phi_e$. Then $f(\overline{x}) = y$ if and only if

$$\exists s \ \phi_e(\overline{x}) \text{ halts at stage } s \text{ with output } y$$

Since "$\phi_e(\overline{x})$ halts at stage s with output y" is a primitive recursive predicate it has a Σ_1-definition. Thus so does $f(\overline{x}) = y$.

The following exercise gives an alternative proof of the last corollary.

Exercise 10.19 Show that the collection of partial functions with Σ_1-definable graphs is closed under the minimization operator μ. Note that the arguments above about closure under composition and primitive recursion work for partial functions and conclude that every partial recursive function has a Σ_1-definable graph.

Σ_1-completeness of PA^-

We have shown that the graph of any primitive recursive $f(x_1,\dots,x_k)$ has a Σ_1-definition $\phi(\overline{x},y)$. Suppose $f(n_1,\dots,n_k)=m$. Thus $\mathbb{N}\models\phi(\widehat{n}_1,\dots,\widehat{n}_k,\widehat{m})$. To prove the Representation Lemma we will need to know that

$$\mathrm{PA}^-\vdash\phi(\widehat{n}_1,\dots,\widehat{n}_k,\widehat{m}).$$

In fact, we will prove that any Σ_1-sentence that is true in $\mathbb{N}$ is provable in PA^-.

Suppose $\mathcal{M}\models\mathrm{PA}^-$. There is a natural map $j:\mathbb{N}\to\mathbb{N}$ by $n\mapsto\widehat{n}^{\mathcal{M}}$, i.e., $j(n)$ is the interpretation of $\widehat{n}$ in $\mathcal{M}$.

Using the fact that $0<1$ and addition is order preserving we see that

$$\mathcal{M}\models\widehat{n}<\widehat{n+1}$$

for all n. Thus j is order preserving and hence injective.

We also argue that j preserves addition and multiplication.

Lemma 10.20 *$j(n+m)=j(n)+j(m)$ and $j(nm)=j(n)j(m)$.*

Proof $j(n+m)=\widehat{n+m}$ and $\mathcal{M}\models\widehat{n}+\widehat{m}=\widehat{n+m}$. Similar for multiplication.

We have shown that $j(\mathbb{N})$ is a substructure of $\mathcal{M}$. We identify $\mathbb{N}$ and $j(\mathbb{N})$ and think of $\mathbb{N}$ as a substructure of $\mathcal{M}$ We next argue that $j(\mathbb{N})$ is an initial segment of $\mathcal{M}$. This is easy since the following argument shows that the ordering of $\mathcal{M}$ is discrete.

Lemma 10.21 $\mathrm{PA}^-\vdash\forall z\neg\exists x\ z<x<z+1$.

Proof Suppose $\mathcal{M}\models\mathrm{PA}^-$ and there are $a,b\in\mathcal{M}$ such that $a<b<a+1$. Since $a<b$ there is a $c\in\mathcal{M}$ such that $b=a+c$. Since $a\neq b$ we must have $c\neq 0$. But then $c\geq 1$ and $b=a+c\geq a+1$, a contradiction.

By the Completeness Theorem, $\mathrm{PA}^-\vdash\forall z\neg\exists x\ z<x<z+1$.

It follows inductively that if $a\in\mathcal{M}$ and $a<\widehat{n}^{\mathcal{M}}$, then $a=\widehat{m}$ for some $m<n$. Thus for any we can view $\mathbb{N}$ as an initial segment of any $\mathcal{M}\models\mathrm{PA}^-$.

Lemma 10.22 *Suppose $\mathcal{M}\models\mathrm{PA}^-$ and $\phi(v_1,\dots,v_k)$ is a Δ_0-formula and $n_1,\dots,n_k\in\mathbb{N}$. Then*

$$\mathcal{M}\models\phi(n_1,\dots,n_k)\Leftrightarrow\mathcal{N}\models\phi(n_1,\dots,n_k).$$

Moreover, if ϕ is Σ_1 and $\mathbb{N}\models\phi(n_1,\dots,n_k)$, then so does $\mathcal{M}$.

Proof We prove this by induction on the complexity of ϕ. If ϕ is atomic, this follows as in Proposition 1.14. Also the proof that if the claim is true for ϕ and for ψ, then it is true for $\phi\wedge\psi$ and $\neg\phi$ is exactly as in Proposition 1.14.

Suppose ϕ is $\forall v<t\,\psi(x_1,\dots,x_k,v)$ where t is a term using the variables from $\overline{x}$ but not v. Since $\{a\in\mathcal{M}:a<t(n_1,\dots,n_k)\}=\{n\in\mathbb{N}:n<t(n_1,\dots,n_k)$,

$$\mathcal{M}\models\phi(n_1,\dots,n_k)\Leftrightarrow\mathbb{N}\models\phi(n_1,\dots,n_k).$$

The argument is similar for bounded existential quantifiers–or follows from the bounded universal case using negations.

Now assume ϕ is Σ_1 say ϕ is

$$\exists \overline{y}\ \theta(\overline{x}, \overline{y})$$

where θ is Δ_0. If $\mathbb{N} \models \phi(\overline{n})$ then there are $\overline{m} \in \mathbb{N}$ such that $\mathbb{N} \models \theta(\overline{n}, \overline{m})$. But then $\mathcal{M} \models \theta(\overline{n}.\overline{m})$ and $\mathcal{M} \models \exists \overline{y}\ \theta(\overline{n}, \overline{y})$, as desired.

Exercise 10.23 Given an example showing that it is possible to have ϕ is Σ_1 and $\mathcal{M} \models \mathrm{PA}^- + \phi(n_1, \ldots, n_k)$ but $\mathbb{N} \models \neg\phi(n_1, \ldots, n_k)$.

Corollary 10.24 (Σ_1-Completeness) *If $\phi(v_1, \ldots, v_m)$ is Σ_1 and $\mathbb{N} \models \phi(n_1, \ldots, n_m)$, then $PA^- \vdash \phi(\widehat{n}_1, \ldots, \widehat{n}_m)$.*

Proof If $\mathcal{N} \models \phi(n_1, \ldots, n_k)$. Then $\mathcal{M} \models \phi(\widehat{n}_1, \ldots, \widehat{n}_k)$ for every $\mathcal{M} \models \mathrm{PA}^-$. By the Completeness Theorem, $\mathrm{PA}^- \vdash \phi(\widehat{n}_1, \ldots, \widehat{n}_k)$.

The Representation Lemma

We can now finish the proof of the Representation Lemma. We restate it in a more precise form.

Lemma 10.25 (Σ_1-Representation Lemma) *For any primitive recursive $f : \mathbb{N}^k \to \mathbb{N}$ there is a Σ_1-formula ϕ_f such that:*
i) if $f(n_1, \ldots n_k) = m$, then $\mathrm{PA}^- \vdash \phi_f(\widehat{n}_1, \ldots, \widehat{n}_k, \widehat{m})$;
and
ii) if $f(n_1, \ldots, n_k) \neq m$, then $\mathrm{PA}^- \vdash \neg\phi_f(\widehat{n}_1, \ldots, \widehat{n}_k, \widehat{m})$.

Proof If we only wanted i) this would follow immediately from the fact that primitive recursive functions have Σ_1-definable graphs and the Σ_1-Completeness of PA^-.

A little more care is needed to guarantee ii). Suppose the graph $f(\overline{x}) = y$ is defined by $\exists z_1, \ldots, \exists z_s\ \theta(\overline{x}, y, \overline{z})$. Where θ is Δ_0. It is easily seen that this is equivalent to

$$\exists z \exists z_1 < z \ldots \exists z_m < z\ \theta(\overline{x}, y, \overline{z}).$$

Thus, without loss of generality, we may assume that the graph of f is defined by

$$\exists z\ \theta(\overline{x}, y, z)$$

where θ is Δ_0.

Next consider the formula ϕ_f asserting

$$\exists z\ \ [\theta(\overline{x}, y, z) \wedge \forall u < z \forall v < z \neg\theta(\overline{x}, u, v))\,.$$

Note that ϕ_f is Σ_1.

If $f(\overline{n}) = m$, then for all $m_1 < m$ and for all z, $\neg\theta(\overline{n}, m_1, z)$. Thus $\phi_f(\overline{n}, m)$. By Σ_1-completeness $\mathrm{PA}^- \vdash \phi_f(\overline{n}_1, \ldots, \overline{n}_k, \widehat{m})$.

Suppose $f(\overline{n}) = s \neq m$ where $\overline{n}, s, m \in \mathbb{N}$. We need to show $\mathrm{PA}^- \vdash \neg\phi_f(\widehat{n}_1, \ldots, \widehat{n}_k, \widehat{m})$. Suppose not. Then, by the Completeness Theorem, there is $\mathcal{M} \models \mathrm{PA}^-$ and $a \in \mathcal{M}$ such that

$$\mathcal{M} \models \theta(\overline{n}, m, a) \wedge \forall u < a \forall v < a\ \neg\theta(\overline{n}, u, v)$$

Since $f(\overline{n}) \neq m$ we must have $a > n$ for all $n \in \mathbb{N}$. But $f(\overline{n}) = s$. Thus there is $t \in \mathbb{N}$ such that $\mathbb{N} \models \theta(\overline{n}.s, t)$. Since θ is Δ_0, $\mathcal{M} \models \theta(\overline{n}.s.t)$. But $t \in \mathbb{N}$, so $s, t < a$, a contradiction. Thus $\mathrm{PA}^- \vdash \neg\phi_f(\widehat{n}_1, \ldots, \widehat{n}_k, \widehat{m})$.

Exercise 10.26 Modify ϕ_f so that in addition $\mathrm{PA}^- \vdash \exists! y \phi_f(\widehat{n}_1, \ldots, \widehat{n}_k, y)$ for all $n_1, \ldots, n_k \in \mathbb{N}$

Exercise 10.27 Argue that we could replace "primitive recursive" by "total recursive" in the Representation Lemma, but not by "partial recursive".

For a deeper study of Peano Arithmetic–and indeed for complete proofs of some of the results in the next section. We would need to have finer versions of some of the technical results in this section. For example, we would need to know that the Chinese Remainder Theorem is provable in PA. We also used the fact that the we can code any sequence with the β-function. One consequence of that that we use is that if we can code a sequence σ, then for any a we can code the sequence σ, a obtained by adding a to then end of σ. We need to show this is provable in PA. More formally, we need

$\mathrm{PA} \vdash \forall m \forall n \forall w \forall a \exists m' \exists n' [\forall i < w \beta(m, n, w)_i = \beta(m', n', w+1)_i \wedge$
$\beta(m', n', w+1)_w = a]$.

Using the above remarks we can prove sharper versions of the Representation Lemma.

Exercise 10.28 Let f be a primitive recursive formula. Show that we can find a Σ_1 formula ϕ_f representing f such that

$$\mathrm{PA} \vdash \forall \overline{x} \exists! y\ \phi_f(\overline{x}, y)$$

This says that the function f is *provably total* in PA. In fact not all total recursive functions are provably total and there are interesting bounds on the growth rates of provably total functions.

11 Arithmetization of Syntax

In §10 we gave a proof of the First Incompleteness Theorem based on basic recursion theoretic ideas. In this section we give a second proof which follows Gödel more closely. We will also sketch the ideas behind the proof of the Second Incompleteness Theorem. The new idea of this section is the idea of Gödel codes for formulas.

We will assign a number $\ulcorner\phi\urcorner$ to each formula ϕ. We call $\ulcorner\phi\urcorner$ the *Gödel code* for ϕ. Gödel coding allows us to talk about properties of formulas in the language of arithmetic. Gödel showed that there are amazing possibilities for self reference. In particular he proved the following striking lemma.

Lemma 11.1 (Diagonalization Lemma) *Let $\phi(v)$ be a formula in the language of arithmetic with one free variable v. There is a sentence ψ such that*

$$\mathrm{PA}^- \vdash \ \psi \leftrightarrow \phi(\ulcorner \psi \urcorner).$$

Intuitively the sentence ψ says "My code has property ϕ". Strictly speaking we should write $\mathrm{PA} \vdash \psi \leftrightarrow \phi(\widehat{\ulcorner \phi \urcorner})$ but we will drop the $\widehat{\ }$ when no confusion arises.

We will begin shortly the work needed to prove the Diagonalization Lemma and deduce the Incompleteness Theorem from it, but first let us deduce one simple and important corollary.

A formula $\Gamma(v)$ is called a *truth definition* if and only if

$$\mathbb{N} \models \psi \ \text{ if and only if } \ \mathbb{N} \models \Gamma(\ulcorner \psi \urcorner).$$

for all sentences ψ

Corollary 11.2 (Tarski's Undefinability of Truth) *There are no truth definitions.*

Proof Suppose $\Gamma(v)$ is a truth definition. Apply the Diagonalization Lemma to $\neg\Gamma$ to obtain a sentence ψ such that $PA \vdash \ \psi \leftrightarrow \neg\Gamma(\ulcorner \psi \urcorner)$. Clearly ψ shows that Γ is not a truth definition.

We now begin the mechanics of coding. We fix a primitive recursive method of coding finite sequences. We let $\langle a_1, \ldots, a_m \rangle$ be the code for the sequence $(a_1, \ldots, a_m)$. We choose the coding so that:
i) every natural number codes a sequence,
ii) $n \mapsto l(n)$ is primitive recursive, where $l(n)$ is the length of the sequence coded by n, and
iii) $(n, i) \mapsto (n)_i$ is primitive recursive, where $(n)_i$ is the i^{th}-element of the sequence coded by n if $i \leq l(n)$ and $(n)_i = 0$ if $i > l(n)$.

For example we could use the coding τ described in §7 or the β-function from §10.

Let us assume that our language is $\mathcal{L} = \{+, \cdot, <, 0, 1\}$ and that we use only the connectives $\wedge$ and $\neg$, the quantifier $\exists$ and variables $v_0, v_1, \ldots$. We assign each symbol a code as follows.

$\ulcorner 0 \urcorner = \langle 0, 0 \rangle$	$\ulcorner 1 \urcorner = \langle 0, 1 \rangle$	$\ulcorner v_i \urcorner = \langle 1, i \rangle$
$\ulcorner + \urcorner = \langle 2, 0 \rangle$	$\ulcorner \cdot \urcorner = \langle 2, 1 \rangle$	$\ulcorner < \urcorner = \langle 3, 0 \rangle$
$\ulcorner = \urcorner = \langle 3, 1 \rangle$	$\ulcorner \wedge \urcorner = \langle 4, 0 \rangle$	$\ulcorner \neg \urcorner = \langle 4, 1 \rangle$
$\ulcorner \exists \urcorner = \langle 5, 0 \rangle$		

We inductively define coding of terms as follows. If t_1 and t_2 are terms then
$\ulcorner t_1 + t_2 \urcorner = \langle \ulcorner + \urcorner, \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner \rangle$ and
$\ulcorner t_1 \cdot t_2 \urcorner = \langle \ulcorner \cdot \urcorner, \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner \rangle$.

If t_1 and t_2 are terms, we code atomic formulas as follows.
$\ulcorner t_1 = t_2 \urcorner = \langle \ulcorner = \urcorner, \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner \rangle$ and
$\ulcorner t_1 < t_2 \urcorner = \langle \ulcorner < \urcorner, \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner \rangle$.

Finally if ϕ and ψ are formulas then
$\ulcorner \neg\phi \urcorner = \langle \ulcorner \neg \urcorner, \ulcorner \phi \urcorner \rangle$,
$\ulcorner \phi \wedge \psi \urcorner = \langle \ulcorner \wedge \urcorner, \ulcorner \phi \urcorner, \ulcorner \psi \urcorner \rangle$ and
$\ulcorner \exists v_i \phi \urcorner = \langle \ulcorner \wedge \urcorner, \ulcorner v_i \urcorner, \ulcorner \phi \urcorner \rangle$.

We will see that all basic syntactic properties of formulas are primitive recursive. It is easy to see for example that the maps $\ulcorner \phi \urcorner \mapsto \ulcorner \neg\phi \urcorner$ and $(\ulcorner \phi \urcorner, \ulcorner \psi \urcorner) \mapsto \ulcorner \phi \wedge \psi \urcorner$ are primitive recursive.

Lemma 11.3 *The predicates "n codes a term" and "n codes a formula" are primitive recursive.*

Proof Let

$$T(x) = \begin{cases} 1 & x = \ulcorner 0 \urcorner > \text{ or } x = \ulcorner 1 \urcorner \\ 1 & l(x) = 3, (x)_1 = \langle 2, 0 \rangle \text{ or } \langle 2, 1 \rangle,\ T((x)_2) = 1 \text{ and } T((x)_3) = 1 \\ 0 & \text{otherwise} \end{cases}.$$

Clearly T is primitive recursive and $T(n) = 1$ if and only if n codes a term. Let

$$F(x) = \begin{cases} 1 & l(x) = 3, (x)_1 = \ulcorner = \urcorner \text{ or } \ulcorner < \urcorner,\ T((x)_1) = 1 \text{ and } T((x)_2) = 1 \\ 1 & l(x) = 2, (x)_1 = \ulcorner \neg \urcorner \text{ and } F((x)_2) = 1 \\ 1 & l(x) = 3, (x)_1 = \ulcorner \wedge \urcorner \text{ and } F((x)_2) = F((x)_3) = 1 \\ 1 & l(x) = 3, (x)_1 = \ulcorner \exists \urcorner,\ \exists i < x\ (x)_2 = \langle 1, i \rangle \text{ and } F((x)_3) = 1 \\ 0 & \text{otherwise} \end{cases}.$$

Then F is primitive recurisive and $F(n) = 1$ if and only if n is the code for a formula

The next lemmas will be the key to proving the Diagonalization Lemma.

Lemma 11.4 *There is a primitive recursive function s such that if t is a term and $i, y \in \mathbb{N}$, then $s(\ulcorner t \urcorner, i, y)$ is the code for the term obtained by replacing all occurences of v_i in t by the term $\widehat{y}$ (where $\widehat{y}$ is the term $\underbrace{1 + \ldots + 1}_{y-\text{times}}$).*

Proof We define s by:

$$s(x, i, y) = \begin{cases} x & x = \ulcorner 0 \urcorner, x = \ulcorner 1 \urcorner \text{ or } x = \ulcorner v_j \urcorner \text{ where } i \neq j \\ \ulcorner \widehat{y} \urcorner & x = \ulcorner v_i \urcorner \\ \langle +, s(t_1, i, y), s(t_2, i, y) \rangle & x = \langle +, t_1, t_2 \rangle \\ \langle \cdot, s(t_1, i, y), s(t_2, i, y) \rangle & x = \langle \cdot, t_1, t_2 \rangle \\ 0 & \text{otherwise} \end{cases}$$

Clearly s is primitive recursive and s is the desired function.

Lemma 11.5 *There is a primitive recursive function sub such that* $sub(\ulcorner\phi\urcorner, i, y) = \ulcorner\psi\urcorner$ *where* ψ *is the formula obtained by substituting* $\widehat{y}$ *for each free occurence of* v_i *in* ϕ.

Proof We may define *sub* by

$$sub(x,i,y) = \begin{cases} \langle\ulcorner=\urcorner, s(t_1,i,y), s(t_2,i,y)\rangle & x = \langle\ulcorner=\urcorner, t_1, t_2\rangle \\ \langle\ulcorner<\urcorner, s(t_1,i,y), s(t_2,i,y)\rangle & x = \langle\ulcorner<\urcorner, t_1, t_2\rangle \\ \langle\ulcorner\neg\urcorner, sub(\ulcorner\phi\urcorner,i,y)\rangle & x = \langle\ulcorner\neg\urcorner, \ulcorner\phi\urcorner\rangle \\ \langle\ulcorner\wedge\urcorner, sub(\ulcorner\phi\urcorner,i,y), sub(\ulcorner\psi\urcorner,i,y)\rangle & x = \langle\ulcorner\wedge\urcorner, \ulcorner\phi\urcorner, \ulcorner\psi\urcorner\rangle \\ \langle\ulcorner\exists\urcorner, \ulcorner v_j\urcorner, sub(\ulcorner\phi\urcorner,i,y)\rangle & x = \langle\ulcorner\exists\urcorner, \ulcorner v_j\urcorner, \ulcorner\phi\urcorner\rangle \text{ and } i \neq j \\ \langle\ulcorner\exists\urcorner, \ulcorner v_i\urcorner, \ulcorner\phi\urcorner\rangle & x = \langle\ulcorner\exists\urcorner, \ulcorner v_i\urcorner, \ulcorner\phi\urcorner\rangle \\ 0 & \text{otherwise.} \end{cases}$$

We can now prove the Diagonalization Lemma.

Proof of 10.1 Let $\phi(v_0)$ be an $\mathcal{L}$-formula with one free variable v_0. Let $S(x,y,z,w)$ be an $\mathcal{L}$-formula representing the primitive function *sub*.

Let $\theta(v_0) = \exists y\ (S(v_0,0,v_0,y) \wedge \phi(y))$ That is, $\theta(v_0)$ asserts $\phi(sub(v_0,0,v_0))$. Let $m = \ulcorner\theta(v_0)\urcorner$ and let $\psi = \theta(m)$.

Then

$$\begin{array}{rcl} \mathrm{PA}^- \vdash \psi & \leftrightarrow & \theta(m) \\ & \leftrightarrow & \exists y\ S(m,0,m,y) \wedge \phi(y) \\ & \leftrightarrow & \exists y\ S(\ulcorner\theta(v_0)\urcorner,0,m,y) \wedge \phi(y) \\ & \leftrightarrow & \exists y\ y = \ulcorner\theta(m)\urcorner \wedge \phi(y) \\ & \leftrightarrow & \phi(\ulcorner\theta(m)\urcorner) \\ & \leftrightarrow & \phi(\ulcorner\psi\urcorner) \end{array}$$

In some of the arguments below we will work with theories $T \supseteq \mathrm{PA}$. While we stated the First Incompleteness Theorem for recursively axiomatized theories. It is technically easier to deal with primitively recursively axiomatized theories. The next lemma shows this is no loss of generality.

Lemma 11.6 (Craig's Trick) *Suppose* T *is a* $\mathcal{L}$*-theory with a recursively enumerable axiomatization. Then there is a primitive recursively axiomatized* $\mathcal{L}$*-theory* T^* *such that* T *and* T^* *have the same consequences (ie.,* $T \vdash \phi \Leftrightarrow T^* \vdash \phi$ *for every* $\mathcal{L}$*-sentence* ϕ*).*

Proof Suppose $W_e = \{\ulcorner\phi\urcorner : \phi \in T\}$. Let

$$T^* = \{\underbrace{\phi \wedge \ldots \wedge \phi}_{s-\text{times}} : \ulcorner\phi\urcorner \in W_e^s\}.$$

It is easy to see that T and T^* have the same logical consequences. On the other hand, since "$x \in W_e^s$" is a primitive recursive predicate, $\{\ulcorner\psi\urcorner : \psi \in T^*\}$ is primitive recursive.

Lemma 11.7 *Let T be a primitive recursive $\mathcal{L}$-theory. Let $Prov_T(x,y)$ be the predicate "x is a proof from T if the formula with Gödel code y". The predicate $Prov_T$ is primitive recursive.*

Sketch of Proof Basicly, $Prov_T(x,y)$ if and only if $\forall i \le l(x) \ \ (x)_i \in T$ or $(x)_i$ follows from previous $(x)_j$ be an inference rule.

This can be coded in a primitive recursive way. We leave the details to the reader.

Let $\psi_{Prov_T}(x,y)$ be an $\mathcal{L}$-formula representing $Prov_T$ in PA and let

$$Pr_T(y) \Leftrightarrow \exists x \ \psi_{Prov_T}(x,y)$$

Using Pr_T we can give Gödel's proof of the First Incompleteness Theorem. Let T be a consistent primitive recursive theory extending PA$^-$. By the Diagonalization Lemma there is a sentence ϕ such that PA$^- \vdash \phi \leftrightarrow \neg Pr_T(\phi)$. We call ϕ the *Gödel sentence* for T.

Theorem 11.8 (First Incompleteness Theorem) *Let $T \supseteq$ PA$^-$ be consistent. Let ϕ be the Gödel sentence for T. Then $T \not\vdash \phi$. Moreover if $\mathbb{N} \models T$, then $T \not\vdash \neg\phi$.*

Proof If $T \vdash \phi$, then there is an $n \in \mathbb{N}$ such that $Prov_T(n, \ulcorner\phi\urcorner)$. But then

$$\text{PA}^- \vdash \psi_{Prov_T}(n, \ulcorner\phi\urcorner)$$

and PA$^- \vdash Pr_T(\ulcorner\phi\urcorner)$ and PA$- \vdash \neg\phi$. Thus $T \vdash \neg\phi$, contradicting the consistency of T. Hence $T \not\vdash \phi$.

If $\mathbb{N} \not\models \phi$, then $\mathbb{N} \models Pr_T(\ulcorner\phi\urcorner)$ and hence there is $m \in \mathbb{N}$ such that $\mathbb{N} \models \psi_{Prov_T}(m, \ulcorner\phi\urcorner)$ and m really is the code for a proof of ϕ from T. But then $T \vdash \phi$ and $\mathbb{N} \models \phi$ a contradiction, thus $\mathbb{N} \models \phi$. Hence if $\mathbb{N} \models T$, then $T \not\vdash \neg\phi$.

A slightly different diagonalization gives a second proof of Roesser's Incompleteness Theorem.

Theorem 11.9 *Let T be a recursively axiomatized consistent extension of* PA$^-$*, then T is incomplete.*

Proof Let $\theta(x,y)$ be an $\mathcal{L}$-formula representing the primitive recursive relation "x and y are Gödel codes for formulas and x codes the negation of the formula coded by y".

Let $Pr_T^*(v)$ be the formula

$$\exists y \ (Prov_T(y,v) \wedge \forall z \ (\theta(z,v) \rightarrow \forall x < y \ \neg Prov_T(x,z))).$$

Thus $Pr_T^*(\ulcorner\phi\urcorner)$ asserts that there is x coding a proof of ϕ and no $y < x$ codes a proof of $\neg\phi$.

By the Diagonalization Lemma there is a sentence ϕ such that

$$\mathrm{PA}^- \vdash \phi \leftrightarrow \neg Pr_T^*(\ulcorner \phi \urcorner).$$

We call ϕ a *Rosser sentence.* Intuitively, ϕ says "for any proof of me there is a shorter proof of my negation".

Suppose $T \vdash \phi$. Then there is a natural number n coding a proof of ϕ and since T is consistent if $m < n$, then m does not code a proof of $\neg\phi$. But then if $\mathcal{M} \models T$, then $\mathcal{M} \models Pr_T^*(\ulcorner \phi \urcorner))$ and $\mathcal{M} \models \neg\phi$ a contradiction. Thus $T \not\vdash \phi$.

Suppose $T \vdash \neg\phi$. Then there is a natural number n coding a proof of $\neg\phi$ and if $m < n$, then m does not code a proof of ϕ. Thus if $\mathcal{M} \models T$, then $\mathcal{M} \models \neg Pr_T^*(\ulcorner \phi \urcorner)$, so $\mathcal{M} \models \phi$, a contradiction. Thus $T \not\vdash \neg\phi$.

The next Lemma summarized the facts about provability that one must verify in PA to prove the Second Incompleteness Theorem.

Theorem 11.10 *Let $T \supseteq$ PA be a primitive recursive theory and let ϕ and ψ be $\mathcal{L}$-sentences. Then the following* Derivability Conditions *hold.*

D1. *If $T \vdash \phi$, then* PA $\vdash Pr_T(\ulcorner \phi \urcorner)$.

D2. *If* PA $\vdash Pr_T(\ulcorner \phi \urcorner) \to Pr_T(\ulcorner Pr_T(\ulcorner \phi \urcorner) \urcorner)$.

D3. PA $\vdash (Pr_T(\ulcorner \phi \urcorner) \wedge Pr_T(\ulcorner \phi \to \psi \urcorner)) \to Pr_T(\ulcorner \psi \urcorner)$

D4. PA $\vdash (Pr_T(\ulcorner \phi \urcorner) \wedge Pr_T(\ulcorner \psi \urcorner)) \to Pr_T(\ulcorner \phi \wedge \psi \urcorner)$

D5. PA $\vdash Pr_{T+\psi}(\ulcorner \phi \urcorner) \leftrightarrow Pr_T(\ulcorner \psi \to \phi \urcorner)$.

We will not prove the Derivability Conditions in this course. Note that D1 follows from the Σ^1-Completeness of PA$^-$. D3-D5 are relatively routine and follow like the remarks at the end of §10 by arguing in PA that certain primitive recursive functions, like the one that take proofs of ϕ and ψ and gives a proof of $\phi \wedge \psi$ do what we expect them to. D2 is a bit more subtle. It is a formal version of Σ^1-Completeness requiring us to redo that analysis in PA.

Theorem 11.11 (Second Incompleteness Theorem) *Let T be a consistent recursively axiomatized theory such that $T \supseteq PA$. Let $Con(T)$ be the sentence $\neg Pr_T(\ulcorner 0 = 1 \urcorner)$. Then $T \not\vdash Con(T)$.*

Proof

Let ϕ be the Gödel sentence such that $PA \vdash \phi \leftrightarrow \neg Pr_T(\ulcorner \phi \urcorner)$.

We will show that PA $\vdash \phi \leftrightarrow Con(T)$. Then, by **??**0.8 , $T \not\vdash \phi$, so $T \not\vdash Con(T)$.

Since we can derive anything from a contradiction, $T \vdash 0 = 1 \to \phi$. Thus by D1,

$$\mathrm{PA} \vdash Pr_T(\ulcorner 0 = 1 \to \phi \urcorner).$$

Thus by D3, PA $\vdash \neg Con(T) \to Pr_T(\ulcorner \phi \urcorner)$. By choice of ϕ, we have

$$\mathrm{PA} \vdash \neg Con(T) \to \neg\phi$$

and taking the contrapositive

$$\text{PA} \vdash \phi \rightarrow Con(T).$$

On the otherhand, by D2

$$\text{PA} \vdash Pr_T(\ulcorner\phi\urcorner) \rightarrow Pr_T(\ulcorner Pr_T(\ulcorner\phi\urcorner)\urcorner).$$

By choice of ϕ, PA $\vdash Pr_T(\phi) \rightarrow \neg\phi$. Thus by D1 and D3

$$\text{PA} \vdash Pr_T(\ulcorner Pr_T(\ulcorner\phi\urcorner)\urcorner) \rightarrow Pr_T(\ulcorner\neg\phi\urcorner).$$

Thus

$$\text{PA} \vdash Pr_T(\ulcorner\phi\urcorner) \rightarrow Pr_T(\ulcorner\neg\phi\urcorner).$$

Using D4 we see that

$$\text{PA} \vdash Pr_T(\ulcorner\phi\urcorner) \rightarrow Pr_T(\ulcorner\phi \wedge \neg\phi\urcorner).$$

But then by D1 and D2

$$\text{PA} \vdash Pr_T(\ulcorner\phi\urcorner) \rightarrow Pr_T(\ulcorner 0 = 1\urcorner).$$

So PA $\vdash Con(T) \rightarrow \phi$, as desired.

By the Diagonalization Lemma there are sentences ϕ such that $PA \vdash \phi \leftrightarrow Pr_T(\phi)$. Henkin asked if such a sentence is provable? The following result shows that it is.

Corollary 11.12 (Löb's Theorem) *Let T be a consisitent recursively axiomatized theory extending PA and let ϕ be any sentence. Then*

$$T \vdash Pr_T(\ulcorner\phi\urcorner) \rightarrow \phi \quad \Leftrightarrow \quad T \vdash \phi.$$

Proof ($\Leftarrow$) This is clear since if $T \vdash \phi$, then $T \vdash \psi \rightarrow \phi$ for any sentence ψ.

($\Rightarrow$). Suppose $T \nvdash \phi$. Then $T + \neg\phi$ is consistent and by the Second Incompleteness Theorem

$$T + \neg\phi \nvdash Con(T + \neg\phi)$$

i.e.,

$$T + \neg\phi \nvdash Pr_{T+\neg\phi}(\ulcorner 0 = 1\urcorner).$$

By D5,

$$T + \neg\phi \nvdash \neg Pr_T(\ulcorner\neg\phi \rightarrow 0 = 1\urcorner).$$

Using the Derivability Conditions (*perhaps suitably modified?*) we know that

$$T \vdash Pr_T(\ulcorner\phi\urcorner) \leftrightarrow Pr_T(\ulcorner\neg\phi \rightarrow 0 = 1\urcorner).$$

Thus

$$T + \neg\phi \nvdash \neg Pr_T(\ulcorner\phi\urcorner).$$

Thus

$$T \nvdash \neg\phi \rightarrow \neg Pr_T(\ulcorner\phi\urcorner)$$

and

$$T \nvdash Pr_T(\ulcorner\phi\urcorner)$$

a contradiction.