Logic and Computability

Daniel Naylor

December 4, 2024

Contents

Lecture 1

1 Non-classical Logic

1.1 Intuitionistic Logic

Idea: a proof of $\varphi \to \psi$ is a "procedure" that comments a proof of φ into a proof of ψ.

In particular, $\neg\neg\varphi$ is not always the same as φ.

Fact: The law of excluded middle ($\varphi \vee \neg\varphi$) is not generally intuitionistically valid.

Moreover, the Axiom of Choice is incompatible with intuitionistic set theory.

We take choice to mean that any family of inhabited sets admits a choice function.

Theorem 1.1.1 (Diaconescu)**.** The law of excluded middle can be intuitionistically deduced from the Axiom of Choice.

Proof. Let φ be a proposition. By the Axiom of Separation, the following are sets (i.e. we can construct a proof that they are sets):

$$A := \{x \in \{0,1\} : \varphi \vee (x = 0)\} \qquad B := \{x \in \{0,1\} : \varphi \vee (x = 1)\}.$$

As $0 \in A$ and $1 \in B$, we have that $\{A, B\}$ is a family of inhabited sets, thus admits a choice function $f : \{A, B\} \to A \cup B$ by the Axiom of Choice. This satisfies $f(A) \in A$ and $f(B) \in B$ by definition.

Thus we have

$$(f(A) = 0 \vee \varphi) \wedge (f(B) = 1 \vee \varphi)$$

and $f(A), f(B) \in \{0,1\}$. Now $f(A) \in \{0,1\}$ means that $(f(A) = 0) \vee (f(A) = 1)$ and similarly for $f(B)$.

We can have the following:

(1) We have a proof of $f(A) = 1$, so $\varphi \vee (1 = 0)$ has a proof, so we must have a proof of φ.

(2) We have a proof of $f(B) = 0$, which similarly gives a proof of φ.

(3) We have $f(A) = 0$ and $f(B) = 1$, in which case we can prove $\not\varphi$: given a proof of ϕ, we can prove that $A = B$ (by Extensionality), in which case $0 = f(A) = f(B) = 1$, a contradiction.

So we can always specify a proof of φ or a proof of φ or a proof of $\neg\varphi$. □

Why bother?

- Intuitionistic maths is more general: we assume less.

- Several ntions that are conflated in classical maths are genuinely different constructively.
- Intuitionistic proofs have a computable content that may be absent in classical proofs.
- Intuitionistic logic is the internal logic of an arbitrary topos.

Let's try to formalise the BHK interpretation of logic.

We will inductively define a provability relation by enforcing rules that implement the BHK interpretation.

Lecture 2 We will use the notation $\Gamma \vdash \varphi$ to mean that φ is a consequence of the formulae in the set Γ.

Rules for Intuitionistic Propositional Calculus (IPC)

(∧-I) $\frac{\Gamma \vdash A, \Gamma \vdash B}{\Gamma \vdash A \wedge B}$

(∨-I) $\frac{\Gamma \vdash A}{\Gamma \vdash A \vee B}$, $\frac{\Gamma \vdash B}{\Gamma \vdash A \vee B}$

(∧-E) $\frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A}$ and $\frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}$

(∨-E) $\frac{\Gamma, A \vdash C \quad \Gamma, B \vdash C \quad \Gamma \vdash A \vee B}{\Gamma \vdash C}$

(→-I) $\frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}$

(→-E) $\frac{\Gamma \vdash A \to B, \Gamma \vdash A}{\Gamma \vdash B}$

(⊥-E) $\frac{\Gamma \vdash \bot}{\Gamma \vdash A}$ for any A

(Ax) $\frac{}{\Gamma, A \vdash A}$ for any A

(Weak) $\frac{\Gamma \vdash B}{\Gamma, A \vdash B}$

(Contr) $\frac{\Gamma, A, A \vdash B}{\Gamma, A \vdash B}$

We obtain classical propositional logic (CPC) by adding either:

- $\frac{}{\Gamma \vdash A \vee \neg A}$
- $\frac{\Gamma, \neg A \vdash \bot}{\Gamma \vdash A}$ (reductio ad absurdum)

By

$$\dfrac{\begin{array}{cc}[A] & [B]\\ \vdots & \vdots \\ X & Y\end{array}}{C}(A,B)$$

we mean 'if we canprove X assuming A and we can prove Y assuming B, then we can infer C by "discharching / closing" the open assumptions A and B'.

In particular, the ($\to$-I)-rule can be written as

$$\dfrac{\begin{array}{c}\Gamma,[A]\\ \vdots \\ B\end{array}}{\Gamma\vdash A\to B}(A).$$

We obtain intiuitionistic first-order logic (IQC) by adding rules for quantification:

($\exists$-I) $\frac{\Gamma\vdash\varphi[x:=t]}{\Gamma\vdash\exists x.\varphi(x)}$, where t is a term.

($\exists$-E) $\frac{\Gamma\vdash\exists x.\varphi \quad \Gamma,\varphi\vdash\psi}{\Gamma\vdash\psi}$, if x is not free in Γ,ψ.

($\forall$-I) $\frac{\Gamma\vdash\varphi}{\Gamma\vdash\forall x.\varphi}$ if x is not free in Γ.

($\forall$-E) $\frac{\Gamma\vdash\forall x.\varphi(x)}{\Gamma\vdash\varphi[x:=t]}$, where t is a term.

Example 1.1.2. Let's give a natural deduction proof of $A\wedge B\to B\wedge A$.

$$\dfrac{\dfrac{\frac{[A\wedge B]}{A} \quad \frac{[A\wedge B]}{B}}{B\wedge A}}{A\wedge B\to B\wedge A}(A\wedge B).$$

Example 1.1.3. Let's prove the Hilbert-style axioms $\varphi\to(\psi\to\varphi)$ and $(\varphi\to(\psi\to\chi))\to((\varphi\to psi)\to(\varphi\to\chi))$.

$$\dfrac{\frac{[\varphi] \quad [\psi]}{\psi\to\varphi}(\psi)}{\varphi\to(\psi\to\varphi)}(\varphi)$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[\varphi\to(\psi\to\chi)] \quad [\varphi\to\psi] \quad [\varphi]}{\psi\to\chi \quad \psi}(\text{toE})}{\chi}(\text{toE})}{\varphi\to\chi}(\text{toI},\psi)}{(\varphi\to\psi)\to(\varphi\to\chi)}(\text{toI}, \varphi{\to}\psi)}{(\varphi\to(\psi\to\chi))\to((\varphi\to\psi)\to(\varphi\to\chi))}(\text{toI}, (\varphi{\to}(\psi{\to}\chi)))$$

If Γ is a set of propositions in the language and φ is a poroposition, we write $\Gamma\vdash_{\text{IPC}}\varphi$, $\Gamma\vdash_{\text{IQC}}\varphi$, $\Gamma\vdash_{\text{CPC}}\varphi$, $\Gamma\vdash_{\text{CQC}}\varphi$, if there is a proof of φ from Γ in the respective logic.

Lemma 1.1.4. If $\Gamma \vdash_{\text{IPC}} \varphi$, then $\Gamma, \psi \vdash_{\text{IPC}} \varphi$ for any proposition ψ. Moreover, if p is a primitive proposition and ψ is any proposition, then

$$\Gamma[p := \psi] \vdash_{\text{IPC}} \varphi[p := \psi].$$

Proof. Induction over the size of proofs. □

1.2 The simply typed λ-calculus

For now we assume given a set Π of *simple types* generated by a grammar

$$\Pi := U|\Pi \to \Pi,$$

Lecture 3 where U is a countable set of *type variables*, as well as an inifinite set V of variables.

Definition 1.2.1 (Simply typed lambda-term)**.** The set Λ_Π of simply typed λ-terms is defined by the grammar

$$\Lambda_\Pi := \underbrace{V}_{\text{variables}} \mid \underbrace{\lambda V : \Pi.\Lambda_\Pi}_{\lambda\text{-abstraction}} \mid \underbrace{\Lambda_\Pi \Lambda_\Pi}_{\lambda\text{-application}} .$$

A *context* is a set of pairs $\{x_1 : \tau_1, \dots, x_n : \tau_n\}$ where the x_i are (distinct) variables and each $\tau_i \in \Pi$. We write C for the set of all possible contexts. Given a context $\Gamma \in C$, we also write $\Gamma, x : \tau$ for the context $\Gamma \cup \{x : \tau\}$ (if x dous not appear in Γ).
The domain of Γ is the set of variables that occur in it, and the range $|\Gamma|$ is the set of types that it manifests.

Definition 1.2.2 (Typability relation)**.** We define the *typability relation* $\Vdash \subseteq C \times \Lambda_\Pi \times \Pi$ via:

(1) For every context Γ, and variable x not occurring in Γ, and type τ, we have $\Gamma, x : \tau \Vdash x : \tau$.

(2) Let Γ be a context, x a variable not occurring in Γ, and let $\sigma, \tau \in \Pi$ be types, and M be a λ-term. If $\Gamma, x : \sigma \Vdash M : \tau$, then $\Gamma \Vdash (\lambda x : \sigma.M) : (\sigma \to \tau)$.

(3) Let Γ be a context, $\sigma, \tau \in \Pi$ be types, and $M, N \in \Lambda_\Pi$ be terms. If $\Gamma \Vdash M : (\sigma \to \tau)$ and $\Gamma \Vdash N : \sigma$, then $\Gamma \Vdash (MN) : \tau$.

Notation. We will refer to the λ-calculus of Λ_Π with this typability relation as $\lambda(\to)$.

A variable x occurring in a λ-abstraction $\lambda \underline{x} : \sigma.M$ is *bound*, and it is *free* otherwise. We say that terms M and N are *α-equivalent* if they differ only in the names of the bound variables.

If M and N are λ-terms and x is a variable, then we define the *substitution of N for x in M* by:

- $x[x := N] = N$;

- $y[x := N] = y$ if $x \neq y$;
- $(PQ)[x := N] = P[x := N]Q[x := N]$ for λ-terms P, Q;
- $(\lambda y : \sigma.P)[x := N] = \lambda y : \sigma.(P[x := N])$, where $x \neq y$ and y is not free in N.

Definition 1.2.3 (beta-reduction)**.** The β-reduction relation is the smallest relation $\to_\beta$ on Λ_Π closed under the following rules:

- $(\lambda x : \sigma.P)Q \to_\beta P[x := Q]$,
- if $P \to_\beta P'$, then for all variables x and types $\sigma \in \Pi$, we have $\lambda x : \sigma.P \to_\beta \lambda x : \sigma.P'$,
- $P \to_\beta P'$ and z as a λ-term, then $PZ \to_\beta P'Z$ and $ZP \to_\beta ZP'$.

We also define β-equivalence $\equiv_\beta$ as the smallest equivalence relation containing $\to_\beta$.

Example 1.2.4 (Informal)**.** We have $(\lambda x : \mathbb{Z}.(\lambda y : \tau.x))Z \to_\beta (\lambda y : \tau.Z)$.

When we reduce $(\lambda x : \sigma.P)Q$, the term being reduced is called a β-redex, and the result is its β-contraction.

Lemma 1.2.5 (Free variables lemma)**.** Assuming that:

- $\Gamma \Vdash M : \sigma$

Then

(1) If $\Gamma \subseteq \Gamma'$, then $\Gamma' \Vdash M : \sigma$.

(2) The free variables of M occur in Γ.

(3) There is a context $\Gamma^* \subseteq \Gamma$ comprising exactly the free variables in M, with $\Gamma^* \Vdash M : \sigma$.

Proof. Exercise. □

Lecture 4

Lemma 1.2.6 (Generation Lemma)**.**

(1) For every variable x, context Γ, and type σ, if $\Gamma \Vdash x : \sigma$, then $x : \sigma \in \Gamma$;

(2) If $\Gamma \Vdash (MN) : \sigma$, then there is a type τ such that $\Gamma \Vdash M : \tau \to \sigma$ and $\Gamma \Vdash N : \tau$;

(3) If $\Gamma \Vdash (\lambda x.M) : \sigma$, then there are types τ and ρ such that $\Gamma, x : \tau \Vdash M : \rho$ and $\sigma = (\tau \to \rho)$.

Lemma 1.2.7 (Substitution Lemma)**.**

(1) If $\Gamma \Vdash M : \sigma$ and α is a type variable, then $\Gamma[\alpha := \tau] \Vdash M : \sigma[\alpha := \tau]$;

(2) If $\Gamma, x : \tau \Vdash M : \sigma$ and $\Gamma \Vdash M : \tau$, then $\Gamma \Vdash M[x := N] : \sigma$.

Proposition 1.2.8 (Subject reduction)**.** Assuming that:

- $\Gamma \Vdash M : \sigma$
- $M \to_\beta N$

Then $\Gamma \Vdash N : \sigma$.

Proof. By induction on the derivation of $M \to_\beta N$, using Lemma 1.2.6 and Lemma 1.2.7. □

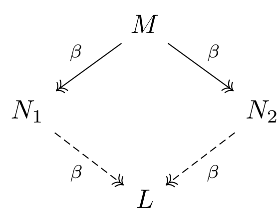
Notation. We will write $M \twoheadrightarrow_\beta N$ if M reduces to N after (potentially multiple) β-reductions.

Theorem 1.2.9 (Church-Rosser for lambda(->))**.** Assuming that:

- $\Gamma \Vdash M : \sigma$
- $M \twoheadrightarrow_\beta N_1$
- $M \twoheadrightarrow_\beta N_2$

Then there is a λ-term L such that $N_1 \twoheadrightarrow_\beta L$, $N_2 \twoheadrightarrow_\beta L$, and $\Gamma \Vdash L : \sigma$.

Pictorially:



Definition (β-normal form)**.** A λ-term M is in β-*normal form* if there is no term N such that $M \to_\beta N$.

Corollary 1.2.10 (Uniqueness of normal form)**.** If a simply typed λ-term admits a β-normal form, then it is unique.

Proposition 1.2.11 (Uniqueness of types)**.**

(1) If $\Gamma \Vdash M : \sigma$ and $\Gamma \Vdash M : \tau$, then $\sigma = \tau$.

(2) If $\Gamma \Vdash M : \sigma$, $\Gamma \Vdash N : \tau$, and $M \equiv_\beta N$, then $\sigma = \tau$.

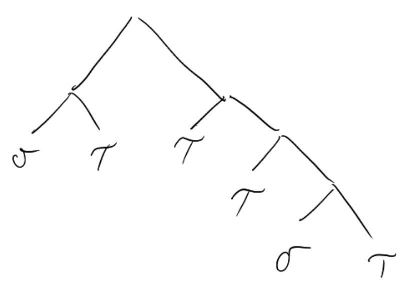
Proof.

(1) Induction.

(2) By the hypothesis and Church-Rosser for lambda(->), there is a term L which both M and N reduce to. By Lemma 1.2.7, we have $\Gamma \Vdash L : \sigma$ and $\Gamma \Vdash L : \tau$, so $\sigma = \tau$ by (1). □

Example 1.2.12. There is no way to assign a type to $\lambda x : x.x$. If x is of type τ, then in order to apply x to x, it has to be of type $\tau \to \sigma$ for some σ. But $\tau \neq \tau \to \sigma$.

$$(\sigma \to \tau) \to (\tau \to (\tau \to (\sigma \to \tau)))$$



Definition 1.2.13 (Height)**.** The *height* function is the recursively defined map $h : \Pi \to \mathbb{N}$ that maps a type variable to 0, and a function type $\sigma \to \tau$ to $1 + \max(h(\sigma), h(\tau))$.
We extend the height function from types to β-redexes by taking the height of its λ-abstraction.

Not.: $(\lambda x : \sigma.P^\tau)^{\sigma\to\tau} R^\sigma$.

Theorem 1.2.14 (Weak normalisation for lambda(->))**.** Assuming that:

- $\Gamma \Vdash M : \sigma$

Then there is a finite reduction path $M := M_0 \to_\beta M_1 \to_\beta M_2 \to_\beta \cdots \to_\beta M_n$, where M_n is in β-normal form.

Proof ("Taming the Hydra"). The idea is to apply induction on the complexity of M. Define a function $m : \Lambda_\Pi \to \mathbb{N} \times \mathbb{N}$ by

$$m(M) = \begin{cases} (0,0) & \text{if } M \text{ is in } \beta\text{-normal form} \\ (h(M), \text{redex}(M)) & \text{otherwise} \end{cases},$$

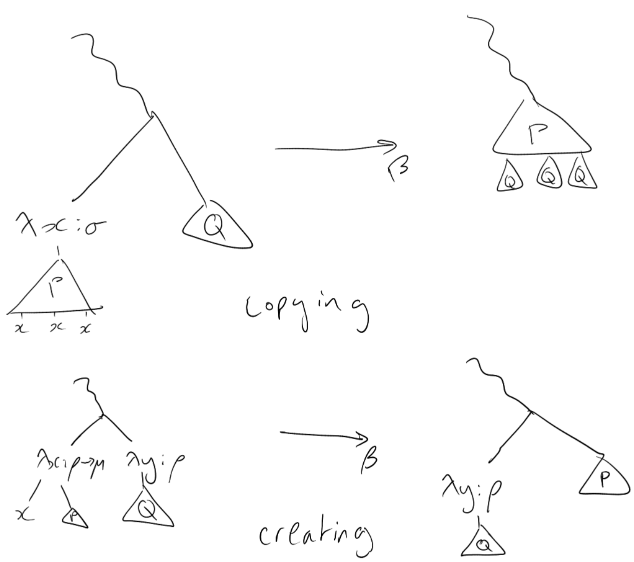
where $h(M)$ is the greatest height of a redex in M, and $\text{redex}(M)$ is the number of redexes in M of that height.

We will use induction over $\omega \times \omega$ to show that if M is typable, then it admits a reduction to β-normal form.

Problem: reductions can copy redexes or create new ones.

Strategy: always reduce the right most redex of maximum height.

We will argue that by following this strategy, any new redexes we generate have to be lower than the
Lecture 5 height of the redex we picked to reduce.



If $\Gamma \Vdash M : \sigma$ and M is already in β-normal form, then claim is trivially true. If M is not in β-normal form, let Δ be the rightmost redex of maximal height h.

By reducing Δ, we may introduce copies of existing redexes, or create new ones. Creation of new redexes of Δ has to happen in one of the following ways:

(1) If Δ is of the form $(\lambda x : (\rho \to \mu) \ldots x P^\rho \ldots)(\lambda y : \rho . Q^\mu)^{P \to \mu}$, then it reduces to $\ldots (\lambda y : \rho . Q^\mu)^{\rho \to \mu} P^\mu \ldots$, in which case there is a new redex of height $h(\rho \to \mu) < h$.

(2) We have $\Delta = (\lambda x : \tau.(\lambda y : \rho.R^\mu))P^\tau$ occuring in M in the scenario $\Delta^{\rho \to \mu} Q^\rho$. Say Δ reduces to $\lambda y : \rho.R_1^\mu$. Then we create a new redex of height $h(\rho \to \mu) < h(\tau \to (\rho \to \mu)) = h$.

(3) The last possibility is that $\Delta = (\lambda x : (\rho \to \mu).x)(\lambda y : \rho.P^\mu)$, and that it occurs in M as $\Delta^{\rho \to \mu} Q^\rho$. Reduction then gives the redex $(\lambda y : \rho.P^\mu)^{\rho \to \mu} Q^\rho$ of height $h(\rho \to \mu) < h$.

Nowe Δ itself is gone (lowering the count by 1), and we just showed that any newly created redexes have height $< h$.

If we have $\Delta = (\lambda x : \tau.P^\rho)Q^\tau$ and P contains multiple free occurrences of x, then all the redexes in Q are multiplied when performing β-reduction.

However, our choice of Δ ensures that the height of any such redex in Q has height $< h$, as they occur to the right of Δ in M. It is this always the case that $m(M') < m(M)$ (in the lexicographic order), so by the induction hypothesis, M' can be reduced to β-normal form (and thus so can M). □

Theorem 1.2.15 (Strong Normalisation for lambda(->))**.** Assuming that:

- $\Gamma \Vdash M : \sigma$

Then there is no infinite reduction sequence $M \to_\beta M_1 \to_\beta \cdots$.

Proof. See Example Sheet 1. □

1.3 The Curry-Howard Correspondence

Propositions-as-types: idea is to think of φ as the "type of its proofs".

The properties of the STλC match the rules of IPC rather precisely.

First we will show a correspondence between $\lambda(\to)$ and the implicational fragment IPC($\to$) of IPC that includes only the $\to$ connective, the axiom scheme, and the $(\to -I)$ and $(\to -E)$ rules. We will later extend this to the whole of IPC by introducing more complex types to $\lambda(\to)$.

Start with IPC($\to$) and build a STλC out of it whose set of type variables U is precisely the set of primtive propositions of the logic.

Lecture 6 Clearly, the set Π of types then matches the set of propositions in the logic.

Comment: $\lambda x : \sigma.(Mx) \to_\eta M$ if x is not free in M.

Proposition 1.3.1 (Curry-Howard for IPC(->))**.** Assuming that:

- Γ is a context for $\lambda(\to)$

- φ a proposition

Then

(1) If $\Gamma \Vdash M : \varphi$, then $|\Gamma| = \{\tau \in \Pi : (x : \tau) \in \Gamma \text{ for some } x\} \vdash_{\mathrm{IPC}(\to)} \varphi$

(2) If $\Gamma \vdash_{\mathrm{IPC}(\to)}$, thene there is a simply typed λ-term $M \in \lambda(\to)$ such that $\{(x_\psi : \psi) \mid \psi \in \Gamma\} \Vdash M : \varphi$.

Proof.

(1) We induct over the derivation of $\Gamma \Vdash M : \varphi$.

If x is a variable not occurring in Γ' and the derivation is of the form $\Gamma', x : \varphi \Vdash x : \varphi$, then we're supposed to prove that $|\Gamma', x : \varphi| \vdash \varphi$. But that follows from $\varphi \vdash \varphi$ as $|\Gamma', x : \varphi| = |\Gamma'| \cup \{\varphi\}$.

If the derivation has M of the form $\lambda x : \sigma.N$ and $\varphi = \sigma \to \tau$, then we must have $\Gamma, x : \sigma \Vdash N : \tau$. By the induction hypothesis, we have that $|\Gamma, x : \sigma| \vdash \tau$, i.e. $|\Gamma|, \sigma \vdash \tau$. But then $|\Gamma| \vdash \sigma \to \tau$ by ($\to$-I).

If the derivation has the form $\Gamma \Vdash (PQ) : \varphi$, then we must have $\Gamma \Vdash P : (\sigma \to \varphi)$ and $\Gamma \Vdash Q : \sigma$. By the induction hypothesis, we have that $|\Gamma| \vdash \sigma \to \varphi$ and $|\Gamma| \vdash \sigma$, so $|\Gamma| \vdash \varphi$ by ($\to$-E).

(2) Again, we induct over the derivation of $\Gamma \vdash \varphi$. Write $\Delta = \{(x_\psi : \psi) \mid \psi \in \Gamma\}$. Then we only have a few ways to construct a proof at a given stage. Say the derivation is of the form $\Gamma, \varphi \vdash \varphi$. If $\varphi \in \Gamma$, then clearly $\Delta \Vdash x_\varphi : \varphi$, and if $\varphi \notin \Gamma$ then $\Delta, x_\varphi : \varphi \Vdash x_\varphi : \varphi$.

Suppose the derivation is at a stage of the form

$$\frac{\Gamma \vdash \varphi \to \psi \quad \Gamma \vdash \varphi}{\Gamma \vdash \psi}.$$

Then by the induction hypothesis, there ar λ-terms M and N such that $\Delta \Vdash M : (\varphi \to \psi)$ and $\Delta \Vdash N : \varphi$, from which $\Delta \Vdash (MN) : \varphi$.

Finally, if the stage is given by

$$\frac{\Gamma, \varphi \vdash \psi}{\Gamma \vdash \varphi \to \psi},$$

then we have two subcases:

- If $\varphi \in \Gamma$, then the induction hypothesis gives $\Delta \Vdash M : \psi$ for some term M. By weakening, we have $\Delta, x : \varphi \Vdash M : \psi$, where x does not occur in Δ. But then $\Delta \Vdash (\lambda x : \varphi.M) : (\varphi \to \psi)$ as needed.
- If $\varphi \notin \Gamma$, then the induction hypothesis gives $\Delta, x_\varphi : \varphi \Vdash M : \psi$ for some M, thus $\Delta \Vdash (\lambda x_\varphi : \varphi.M) : (\varphi \to \psi)$ as needed. $\square$

Example 1.3.2. Let φ, ψ be primitive propositions. The λ-term

$$\lambda f : (\varphi \to \psi) \to \varphi . \lambda : \varphi \to \psi . \underset{\psi}{\overbrace{g(\underbrace{fg}_{\varphi})}}$$

has type $((\varphi \to \psi) \to \varphi) \to ((\varphi \to \psi) \to \psi)$, and therefore encodes a proof of that proposition in IPC($\to$).
$g : \varphi \to \psi,\ f : (\varphi \to \psi) \to \varphi$.

$$\begin{array}{cl}
g : [\varphi \to \psi] \quad f : [(\varphi \to \psi) \to \varphi] & \\
\hline
fg : \varphi \qquad g : [\varphi \to \psi] & (\text{toE}) \\
\hline
g(fg) : \psi & (\text{toE}) \\
\hline
\lambda g . g(fg) : (\varphi \to \psi) \to \psi & (\text{toI, } \varphi{\to}\psi) \\
\hline
\lambda f . \lambda g . g(fg) : ((\varphi \to \psi) \to \varphi) \to ((\varphi \to \psi) \to \psi) & (\text{toI, } (\varphi{\to}\psi) \to\varphi)
\end{array}$$

Definition 1.3.3 (Full STlambdaC)**.** The types of the full symply typed λ-calculus are generated by the following grammar:

$$\Pi := U \mid \Pi \to \Pi \mid \Pi \times \Pi \mid \Pi + \Pi \mid 0 \mid 1,$$

where U is a set of type variables (usually countable).
Its terms are given by Λ_Π given by:

$$\Lambda_\Pi := V | \lambda V : \Pi . \Lambda_\Pi \mid \Lambda_\Pi \Lambda_\Pi | \Pi_1(\Lambda_\Pi) | \Pi_2(\Lambda_\Pi) | \iota_1(\Lambda_\Pi) | \iota_2(\Lambda_\Pi) | \operatorname{case}(\Lambda_\Pi ; V . \Lambda_\Pi ; V . \Lambda_\Pi) | * | !_\Pi \Lambda_\Pi,$$

where V is an infinite set of variables, and $*$ is a constant.

Lecture 7

We have new typing rules:

- $\frac{\Gamma \Vdash M : \psi \times \varphi}{\Gamma \Vdash \pi_1(M) : \psi}$
- $\frac{\Gamma \Vdash M : \psi \times \varphi}{\Gamma \Vdash \pi_2(M) : \varphi}$
- $\frac{\Gamma \Vdash M : \psi}{\Gamma \Vdash \iota_1(M) : \psi + \varphi}$
- $\frac{\Gamma \Vdash N : \varphi}{\Gamma \Vdash \iota_2(N) : \psi + \varphi}$
- $\frac{\Gamma \Vdash M : \psi \quad \Gamma \Vdash N : \varphi}{\Gamma \Vdash \langle M, N \rangle : \varphi \times \psi}$
- $\frac{\Gamma \Vdash L : \psi + \varphi \quad \Gamma, x : \psi \Vdash M : \rho \quad \Gamma, y : \varphi \Vdash N : \rho}{\Gamma \Vdash \operatorname{case}(L ; x^\psi . M ; x^\varphi . N)}$
- $\frac{}{\Gamma \Vdash * : 1}$
- $\frac{\Gamma \Vdash M : 0}{\Gamma \Vdash !_\varphi M : \varphi}$ for each $\varphi \in \Pi$

They come with new reduction rules:

- **Projections:** $\pi_1\langle M, N\rangle \to_\beta M$ and $\pi_2\langle M, N\rangle \to_\beta N$
- **Pairs:** $\langle \pi_1 M, \pi_2 M\rangle \to_\eta M$
- **Definition by cases:** $\text{case}(\iota_1(M); xK; y.L) \to_\beta K[x := M]$ and $\text{case}(\iota_2(M); x.K; y.L) \to_\beta L[y := M]$
- **Unit:** If $\Gamma \Vdash M : 1$, then $M \to_\eta *$

When setting up Curry-Howard with these new types, we let:

- $0 \leftrightsquigarrow \bot$
- $\times \leftrightsquigarrow \wedge$
- $+ \leftrightsquigarrow \vee$
- $\to \leftrightsquigarrow \to$

Example 1.3.4. Consider the following proof of $(\varphi \wedge \chi) \to (\psi \to \varphi)$:

$$\dfrac{\dfrac{\dfrac{[\varphi\wedge\chi]}{\varphi} \qquad [\psi]}{\psi \to \varphi}\ ()}{(\varphi \wedge \chi) \to (\psi \to \varphi)}\ ()$$

We decorate this proof by turning the assumptions into variables and following the Curry-Howard correspondence:

$$\dfrac{\dfrac{\dfrac{[\varphi\wedge\chi] : p}{\varphi : \pi_1(p)} \qquad [\psi] : b}{\psi \to \varphi : \lambda b : \psi.\pi_1(p)}\ ()}{(\varphi \wedge \chi) \to (\psi \to \varphi)}\ ()$$

STλC	IPC
(primitive) types	(primitive) propositions
variable	hypothesis
STλ-term	proof
type constructor	logical connective
term inhabitation	provability
term reduction	proof normalisation

1.4 Semantics for IPC

Definition 1.4.1 (Lattice)**.** A *lattice* is a set L equipped with binary commutative and associative operations $\wedge$ and $\vee$ that satisfy the absorption laws:

$$a \vee (a \wedge b) = a; \qquad a \wedge (a \vee b) = a,$$

for all $a, b \in L$.
A lattice is:

- *Distributive* if $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$ for all $a, b, c \in L$.
- *Bounded* if there are elements $\bot, \top \in L$ such that $a \vee \bot = a$ and $a \wedge \top = a$.
- *Complemented* if it is bounded and for every $a \in L$ there is $a^* \in L$ such that $a \wedge a^* = \bot$ and $a \vee a^* = \top$.

A *Boolean algebra* is a complemented distributive lattice.

Note that $\wedge$ and $\vee$ are idempotent in any lattice. Moreover, we can define an ordering on L by setting $a \leq b$ if $a \wedge b = a$.

Example 1.4.2.

(1) For every set I, the power set $\mathcal{P}(I)$ with $\wedge := \cap$ and $\vee := \cup$ is the prototypical Boolean algebra. More generally, the clopen subsets of a topological space form a Boolean algebra. Interestingly: every Boolean algebra corresponds to a Boolean algebra constructed in this way.

(2) The set of finite and cofinite subsets of $\mathbb{Z}$ is a Boolean algebra.

(3) The set of Zariski-closed subsets of the affine variety $\mathbb{C}^n$ is a distributive lattice but not a Boolean algebra.

Lecture 8

Proposition 1.4.3. Assuming that:

- L is a bounded lattice
- $\leq$ is the order induced by the operations in L ($a \leq b$ if $a \wedge b = a$)

Then $\leq$ is a partial order with least element $\bot$, greatest element $\top$, and for any $a, b \in L$, we have $a \wedge b = \inf\{a, b\}$ and $a \wedge b = \sup\{a, b\}$. Conversely, every partial order with all finite infs and sups is a bounded lattice.

Proof. Exercise. □

Classically, we say that $\Gamma \models t$ if for every valuation $v : L \to \{0,1\}$ with $v(p) = 1$ for all $p \in \Gamma$ we have $v(t) = 1$.

We might want to replace $\{0,1\}$ with some other Boolean algebra to get a semantics for IPC, with an accompanying Completeness Theorem. But Boolean algebras believe in the Law of Excluded Middle!

Definition 1.4.4 (Heyting algebra)**.** A Heyting algebra is a bounded lattice equipped with a binary operation $\Rightarrow: H \times H \to H$ such that

$$a \wedge b \leq c \qquad \Longleftrightarrow \qquad a \leq (b \Rightarrow c)$$

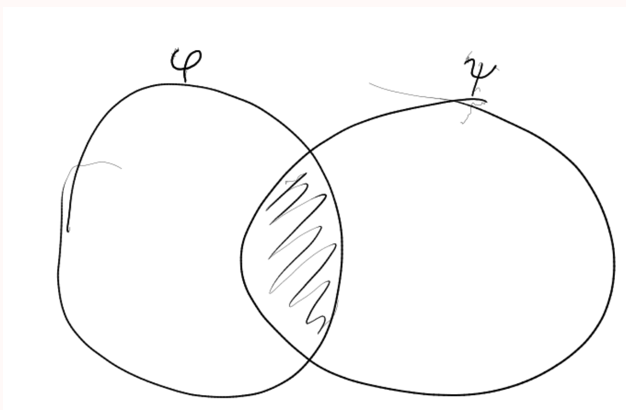
for all $a, b, c \in L$.
A morphism of Heyting algebras is a function that preserves all finite meets, finite joins, and $\Rightarrow$.

Example 1.4.5.

(1) Every Boolean algebra is a Heyting algebra: define $a \Rightarrow b := a^* \vee b$, where a^* is the complement of a. Note that we must have $a^* = (a \Rightarrow \bot)$.

(2) Every topology on a set X is a Heyting algebra, where

$$(U \Rightarrow V) := \mathrm{int}((X \setminus U) \cup V).$$



(3) A finite distributive lattice has to be a Heyting algebra (see Example Sheet 2).

Definition 1.4.6 (Valuation in Heyting algebras)**.** Let H be a Heyting algebra and L be a propositional language with a set P of primitive propositions. An *H-valuation* is a function $v : P \to H$, extended to the whole of L recursively by setting:

- $v(\bot) = \bot$,

- $v(A \wedge B) = v(A) \wedge v(B)$,
- $v(A \vee B) = v(A) \vee v(B)$,
- $v(A \to B) = v(A) \Rightarrow v(B)$.

A proposition A is *H-valid* if $v(A) = \top$ for all H-valuations v, and is an *H-consequence* of a (finite) set of propositions Γ if $v(\bigwedge \Gamma) \leq v(A)$ for all H-valuations v (written $\Gamma \models_H A$).

Lemma 1.4.7 (Soundness of Heyting semantics)**.** Assuming that:

- H is a Heyting algebra
- $v : L \to H$ is a valuation

Then $\Gamma \vdash_{\text{IPC}} A$ implies $\Gamma \models_H A$.

Proof. By induction over the structure of the proof $\Gamma \vdash A$.

(Ax) As $v((\bigwedge \Gamma) \wedge A) = v(\bigwedge) \wedge v(A) \leq v(A)$ for any Γ and A.

($\wedge$-I) $A = B \wedge C$ and we have derivations $\Gamma_1 \vdash B$, $\Gamma_2 \vdash C$, with $\Gamma_1, \Gamma_2 \subseteq \Gamma$. By the induction hypothesis, we have $v(\bigwedge \Gamma) \leq v(\bigwedge \Gamma_1) \cap v(\bigwedge \Gamma_2) \leq v(B) \wedge v(C) = v(B \wedge C) = v(A)$, i.e. $\Gamma \models_H A$.

($\to$-I) $A = B \to C$ and so we must have $\Gamma \cup \{B\} \vdash C$. By induction hypothesis, we have $v(\bigwedge \Gamma) \wedge v(B) = v(\bigwedge \gamma \wedge B) \leq v(C)$. By the definition of $\Rightarrow$, this implies $v(\bigwedge \Gamma) \leq [v(B) \Rightarrow v(C)] = v(B \to C) = v(A)$, i.e. $\Gamma \models_H A$.

($\vee$-I) $A = B \vee C$ and without loss of generality we have a derivation $\Gamma \vdash B$. By the induction hypothesis we have $v(\bigwedge \Gamma) \leq v(B)$, but $v(B \vee C) = v(B) \vee v(C)$, and hence $v(B) \leq v(B \vee C) = v(A)$.

Lecture 9

($\wedge$-E) By the induction hypothesis, we have $v(\bigwedge \Gamma) \leq v(B \wedge C) = v(B) \wedge v(C) \leq v(B), v(B)$.

($\to$-E) We know that $v(A \to B) = (v(A) \Rightarrow v(B))$. From $v(A \to B) \leq v(A) \Rightarrow v(B)$, we derive $v(A) \wedge v(A \to B) \leq v(B)$ by definition of $\Rightarrow$. So if $v(\bigwedge \Gamma) \leq v(A \to B)$ and $v(\bigwedge \Gamma) \leq v(A)$, then $v(\bigwedge \Gamma) \leq v(B)$, as needed.

($\vee$-E) By induction hypothesis: $v(A \vee \bigwedge \Gamma) \leq v(C)$, $v(B \vee \bigwedge \Gamma) \leq v(C)$ and $v(\bigwedge \Gamma) \leq v(A \vee B) = v(A) \vee v(B)$. This last fact means that $v(\bigwedge \Gamma) \wedge (v(A) \vee v(B)) = v(\bigwedge \Gamma)$. Now this is the same as $(v(\bigwedge \Gamma) \wedge v(A)) \vee (v(\bigwedge \Gamma) \wedge v(B))$ as Heyting algebras are distributive lattices (see Example Sheet 2), and this is $\leq v(C)$ by the first two inequalities of this paragraph.

($\bot$-E) If $v(\bigwedge \Gamma) \leq v(\bot) = \bot$, then $v(\bigwedge \Gamma) = \bot$, in which case $v(\bigwedge \Gamma) \leq v(A)$ for any A by minimality of $\bot$ in H. $\square$

Example 1.4.8. The Law of Excluded Middle is not intuitionistically valid. Let p be a primitive proposition and consider the Heyting algebra given by the topology $\{\emptyset, \{1\}, \{1,2\}\}$ on $\{1,2\}$. We can define a valuation v with $v(p) = \{1\}$, in which case $v(\neg p) = \neg\{1\} = \mathrm{int}(X \setminus \{1\}) = \emptyset$. So $v(p \vee \neg p) = \{1\} \vee \emptyset = \{1\} \neq \top$. Thus Soundness of Heyting semantics implies that $\nvdash_{\mathrm{IPC}} p \vee \neg p$.

Example 1.4.9. Peirce's Law $((p \to q) \to p) \to p$ is not intuitionistically valid.
Take the valuation on the usual topology of $\mathbb{R}^2$ that maps p to $\mathbb{R}^2 \setminus \{(0,0)\}$ and q to $\emptyset$.

Classical completeness: $\Gamma \vdash_{\mathrm{CPC}} A$ if and only if $\Gamma \models_2 A$.

Intuitionistic completeness: no single finite replacement for 2.

Definition (Lindenbaum-Tarski algebra)**.** Let Q be a logical doctrine (CPC, IPC, etc), L be a propositional language, and T be an L-theory. The *Lindenbaum-Tarski algebra* $F^Q(T)$ is built in the following way:

- The underlying set of $F^Q(T)$ is the set of equivalence classes $[\varphi]$ of propositions φ, where $\varphi \sim \psi$ when $T, \varphi \vdash_Q \psi$ and $T, \psi \vdash_Q \varphi$;
- If $\bowtie$ is a logical connective in the fragment Q, we set $[\varphi] \bowtie [\psi] := [\varphi \bowtie \psi]$ (should check well-defined: exercise).

We'll be interested in the case $Q = \mathrm{CPC}$, $Q = \mathrm{IPC}$, and $Q = \mathrm{IPC} \setminus \{\to\}$.

Proposition 1.4.10. The Lindenbaum-Tarski algebra of any theory in $\mathrm{IPC} \setminus \{\to\}$ is a distributive lattice.

Proof. Clearly, $\wedge$ and $\vee$ inherit associativity and commutativity, so in order for $F^{\mathrm{IPC}\setminus\{\to\}}(T)$ to be a lattice we need only to check the absorption laws:

$$[\varphi] \vee [\varphi \wedge \psi] = [\varphi] \qquad (\alpha)$$
$$[\varphi] \wedge [\varphi \vee \psi] = [\varphi] \qquad (\beta)$$

Equation (α) is true since $T, \varphi \vdash_{IPC\setminus\{\to\}} \varphi \vee (\varphi \wedge \psi)$ by ($\vee$-I), and also $T, \varphi \vee (\varphi \wedge \psi) \vdash_{\mathrm{IPC}\setminus\{\to\}} \varphi$ by ($\vee$-E). Equation (β) is similar.

Now, for distributivity: $T, \varphi \wedge (\psi \vee \chi) \vdash (\varphi \wedge \psi) \vee (\varphi \wedge \chi)$ by ($\wedge$-E) followed by ($\vee$-E):

$$\dfrac{\dfrac{\varphi \wedge (\psi \vee \chi)}{\varphi \qquad \psi \vee \chi}\ (\wedge\text{-E})}{(\varphi \wedge \psi) \vee (\varphi \wedge \chi)}\ (\vee\text{-E})$$

Conversely, $T, ((\varphi \wedge \psi) \vee (\varphi \wedge \chi)) \vdash \varphi \wedge (\psi \vee \chi)$ by ($\vee$-E) followed by ($\wedge$-I). ∎

Lemma 1.4.11. The Lindenbaum-Tarski algebra of any theory relative to IPC is a Heyting algebra.

Proof. We already saw that $F^{\mathrm{IPC}}(T)$ is a distributive lattice, so it remains to show that $[\varphi] \Rightarrow [\psi] := [\varphi \to \psi]$ gives a Heyting implication, and that $F^{\mathrm{IPC}}(T)$ is bounded.

Suppose that $[\varphi \wedge [\psi] \leq [\chi]$, i.e. $\tau, \varphi \wedge \psi \vdash_{\mathrm{IPC}} \chi$. We want to show that $[\varphi] \leq [\psi \to \chi]$, i.e. $\tau, \varphi \vdash (\psi \to \chi)$. But that is clear:

$$\dfrac{\dfrac{\dfrac{\varphi \qquad [\psi]}{\varphi \wedge \psi}}{\chi}\,(\text{hyp})}{\psi \to \chi}\,(\to\text{-I}, \psi)$$

Conversely, if $\tau, \varphi \vdash (\psi \to \chi)$, then we can prove $\tau, \varphi \wedge \psi \vdash \chi$:

$$\dfrac{\dfrac{\dfrac{\varphi \wedge \psi}{\varphi \qquad \psi}\,(\wedge\text{-E})}{\psi \to \chi \qquad \psi}\,(\text{hyp})}{\chi}\,(\to\text{-E})$$

So defining $[\varphi] \Rightarrow [\psi] := [\varphi \to \psi]$ provides a Heyting $\Rightarrow$.

The bottom element of $F^{\mathrm{IPC}}(T)$ is just $[\bot]$: if $[\varphi]$ is any element, then $T, \bot \vdash_{\mathrm{IPC}} \varphi$ by $\bot$-E.

The top element is $\top := [\bot \to \bot$: if φ is any proposition, then $[\varphi] \leq [\bot \to \bot]$ via

$$\dfrac{\dfrac{\varphi \qquad [\bot]}{\bot}\,(\bot\text{-E})}{\bot \to \bot}$$

□

Theorem 1.4.12 (Completeness of the Heyting semantics)**.** A proposition is provable in IPC if and only if it is H-valid for every Heyting algebra H.

Proof. One direction is easy: if $\vdash_{\mathrm{IPC}} \varphi$, then there is a derivation in IPC, thus $\top \leq v(\varphi)$ for any Heyting algebra H and valuation v, by Soundness of Heyting semantics. But then $v(\varphi) = \top$ and φ is H-valid.

For the other direction, consider the Lindenbaum-Tarski algebra $F(L)$ of the empty theory relative to IPC, which is a Heyting algebra by Lemma 1.4.11. We can define a valuation v by extending $P \to F(L)$, $p \mapsto [p]$ to all propositions.

As v is a valuation, it follows by induction (and the construction of $F(L)$) that $v(\varphi) = [\varphi]$ for all propositions.

Now φ is valid in every Heyting algebra, and so is valid in $F(L)$ in particular. So $v(\varphi) = \top = [\varphi]$, hence $\top \to \top \vdash_{\text{IPC}} \varphi$, hence $\vdash_{\text{IPC}} \varphi$. ☐

Given a poset S, we can construct sets $a\uparrow := \{s \in S : a \leq s\}$ called *principal up-sets*.

Recall that $U \subseteq S$ is a *terminal segment* if $a\uparrow \subseteq U$ for each $a \in U$.

Proposition 1.4.13. If S is a poset, then the set $T(S) = \{U \subseteq S : U \text{ is a terminal segment of } S\}$ can be made into a Heyting algebra.

Proof. Order the terminal segments by $\subseteq$. Meets and joins are $\cap$ and $\cup$, so we just need to define $\Rightarrow$. If $U, V \in T(S)$, define $(U \Rightarrow V) := \{s \in S : (s\uparrow) \cap U \subseteq V\}$.

If $U, V, W \in T(S)$, we have

$$W \subseteq (U \Rightarrow V) \qquad \iff \qquad (w\uparrow) \cap U \subseteq V \forall w \in W,$$

which happens if for every $w \in W$ and $u \in U$ we have $w \leq u \implies u \in V$. But W is a terminal segment, so this is the same as saying that $W \cap U \subseteq V$. ☐

Definition 1.4.14 (Kripke model)**.** Let P be a set of primitive propositions. A *Kripke model* is a tuple $(S, \leq, \Vdash)$ where $(S, \leq)$ is a poset (whose elements are called "worlds" or "states", and whose ordering is called the "accessibility relation") and $\Vdash \subseteq S \times P$ is a binary relation ("forcing") satisfying the persistence property: if $p \in P$ is such that $s \Vdash p$ and $s \leq s'$, then $s' \Vdash p$.

Lecture 11 Every valuation v on $T(S)$ induces a Kripke model by setting $s \Vdash p$ is $s \in v(p)$.

Definition 1.4.15 (Forcing relation)**.** Let $(S, \leq, \Vdash)$ be a Kripke model for a propositional language. We define the extended forcing relation inductively as follows:

- There is no $s \in S$ with $s \Vdash \bot$;
- $s \Vdash \varphi \wedge \psi$ if and only if $s \Vdash \varphi$ and $s \Vdash \psi$;
- $s \Vdash \varphi \vee \psi$ if and only if $s \Vdash \varphi$ or $s \Vdash \psi$;
- $s \Vdash (\varphi \to \psi)$ if and only if $s' \Vdash \varphi$ implies $s' \Vdash \psi$ for every $s' \geq s$.

It is easy to check that the persistence property extends to arbitrary propositions.

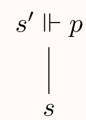
Moreover:

- $s \Vdash \neg\varphi$ if and only if $s' \nVdash \varphi$ for all $s' \geq s$.
- $s \Vdash \neg\neg\varphi$ if and only if for every $s' \geq s$, there exists $s'' \geq s'$ with $s'' \Vdash \varphi$.

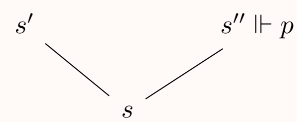
Notation. $S \Vdash \varphi$ for φ a proposition if all worlds in S force φ.

Example 1.4.16. Consider the following Kripke models:

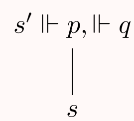
(1)



(2)



(3)



In (1), we have $s \nVdash \neg p$, since $s' \geq s$ and $s' \Vdash p$. We also know that $s \nVdash p$, thus $s \nVdash p \vee \neg p$.
It is also the case that $s \Vdash \neg\neg p$, yet $s \nVdash p$, so $s \nVdash (\neg\neg p \to p)$ either.
In (2), $s \nVdash \neg\neg p$ since $s' \geq s$ can't access a world that forces p. Also $s \nVdash \neg p$ either, as $s'' \geq s$ forces p. So $s \nVdash \neg\neg p \vee \neg p$.
In (3), $s \nVdash (p \to q) \to (\neg p \vee q)$. All worlds force $p \to q$, and $s \nVdash q$. So to check the claim we just need to verify that $s \nVdash \neg p$. But that is the case, as $s' \geq s$ and $s' \Vdash p$.

Definition 1.4.17 (Filter)**.** A *filter* F on a lattice L is a subset of L with the following properties:

- $F \neq \emptyset$
- F is a terminal segment of L (i.e., if $f \leq x$ and $f \in F$, then $x \in F$)
- F is closed under finite meets

Example 1.4.18.

(1) Given an element $j \in I$ of a set I, then the family F_j of all subsets of I containing j is a filter on $\mathcal{P}(I)$. Such a filter is called a *principal filter*.

(2) The family of all cofinite subsets of I is a filter on $\mathcal{P}(I)$, the Fréchet filter.

Exercise: a maximal proper filter (known as an *ultra filter*) is not principal if and only if it contains the Fréchet filter.

(3) The family of all subsets of $[0, 1]$ with Lebesgue measure 1 is a filter.

A filter is *proper* if $F \neq L$.

A filter F on a Heyting algebra is *prime* if it is proper and satisfies: whenever $(x \vee y) \in F$, we can conclude that $x \in F$ or $y \in F$.

If F is a proper filter and $x \notin F$, then there is a prime filter extending F that still doesn't contain x (by Zorn's Lemma).

Lemma 1.4.19. Assuming that:

- H a Heyting algebra
- v a H-valuation

Then there is a Kripke model $(S, \leq, \Vdash)$ such that $v \models_H \varphi$ if and only if $S \Vdash \varphi$, for every proposition φ.

Lecture 12

Proof (sketch). Let S be the set of all prime filters of H, ordered by inclusion. We write $F \Vdash p$ if and only if $v(p) \in F$ for primitive propositions p.

We prove by induction that $F \Vdash \varphi$ if and only if $v(\varphi) \in F$ for arbitrary propositions.

For the implication case, say that $F \Vdash (\psi \to \psi')$ and $v(\psi \to \psi') = [v(\psi) \Rightarrow v(\psi')] \notin F$. Let G' be the least filter containing F and $v(\psi)$. Then

$$G' = \{b : (\exists f \in F)(f \wedge v(\psi) \leq b)\}.$$

Note that $v(\psi') \notin G'$, or else $f \wedge v(\psi) \leq v(\psi')$ for some $f \in F$, whence $f \leq v(\psi \to \psi')$ and so $v(\psi \to \psi') \in F$ (as F is a terminal segment).

In particular, G' is proper. So let G be a prime filter extending G' that does not contain $v(\psi')$ (exists by Zorn's lemma).

By the induction hypothesis, $G \Vdash \psi$, and since $F \Vdash (\psi \to \psi')$ and G' (this G) contains F, we have that $G \Vdash \psi'$. But then $v(\psi') \in G$, contradiction.

This settles that $F \Vdash (\psi \to \psi')$ implies $v(\psi \to \psi') \in F$.

Conversely, say that $v(\psi \to \psi') \in F \subseteq G \Vdash \psi$. By the induction hypothesis, $v(\psi) \in G$, and so $v(\psi) \Rightarrow v(\psi) \in G$ (as $F \subseteq G$). But then $v(\psi') \geq v(\psi) \wedge (v(\psi) \Rightarrow v(\psi')) \in G$, as G is a filter.

So the induction hypothesis gives $G \Vdash \psi'$, as needed.

The cases for the other connectives are easy ($\vee$ needs primality). So $(S, \leq, \Vdash)$ is a Kripke model. Want to show that $v \models_H \varphi$ if and only if $S \Vdash \varphi$, for each φ.

Conversely, say $S \Vdash \varphi$, but $v \not\models_H \varphi$. Since $v(\varphi) \neq \top$, there must be a proper filter that does not contain it. We can extend it to a prime filter G that does not contain it, but then $G \nVdash \varphi$, contradiction. □

Theorem 1.4.20 (Completeness of the Kripke semantics)**.** Assuming that:

- φ a proposition

Then $\Gamma \vdash_{\text{IPC}} \varphi$ if and only if for all Kripke models $(S, \leq, \Vdash)$, the condition $S \Vdash \Gamma$ implies $S \Vdash \varphi$.

Proof. **Soundness:** indcution over the complexity of φ.

Adequacy: Say $\Gamma \nvdash_{\text{IPC}} \varphi$. Then $v \models_H \Gamma$ but $v \not\models_H \varphi$ for some Heyting algebra H and H-valuation v (Theorem 1.4.12). But then Lemma 1.4.19 applied to $H$and v provides a Kripke model $(S, \leq, \Vdash)$ such that $S \Vdash \Gamma$, but $S \nVdash \varphi$, contradicting the hypothesis on every Kripke model. □

1.5 Negative translations

Definition 1.5.1 (Double-negation translation)**.** We recursively define the $\neg\neg$-translation φ^N of a propositon φ in the following way:

- If p is a primitive proposition, then $p^N := \neg\neg p$;
- $(\varphi \wedge \psi)^N := \varphi^N \wedge \psi^N$
- $(\varphi \to \psi)^N := \varphi^N \to \psi^N$
- $(\varphi \vee \psi)^N := \neg(\neg\varphi^N \wedge \neg\psi^N)$
- $(\neg\varphi)^N := \neg\varphi^N$

Lemma 1.5.2. Assuming that:

- H a Heyting algebra

Then the map $\neg\neg : H \to H$ preserves $\wedge$ and $\Rightarrow$.

Proof. Example Sheet 2. □

Lemma 1.5.3 (Regularisation)**.** Assuming that:

- H a Heyting algebra

Then

(1) The subset $H_{\neg\neg} := \{x \in H : \neg\neg x = x\}$ is a Boolean algebra;

(2) For every Heyting homomorphism $g : H \to B$ into a Boolean algebra, there is a unique map of Boolean algebras $g_{\neg\neg} : H_{\neg\neg} \to B$ such that $g(x) = g_{\neg\neg}(\neg\neg x)$ for all $x \in H$.

Lecture 13

Proof.

(1) Give $H_{\neg\neg} := \{x \in H : \neg\neg x = x\}$ the inherited order, so that $\wedge$, $\Rightarrow$, $\perp$ and $\top$ (which are preserved by $\neg\neg$) remain the same. We just need to define disjunctions in $H_{\neg\neg}$ properly.

Define $a \vee_{\neg\neg} b := \neg\neg(a \vee b)$ in H. It is easy to show that this gives $\sup\{a, b\}$ in $H_{\neg\neg}$ (as $\neg\neg$ preserves order), so $H_{\neg\neg}$ is a Heyting algebra.

As every element of $H_{\neg\neg}$ is regular (i.e. $\neg\neg x = x$), it is a Boolean algebra (see Example Sheet 2).

(2) Given a Heyting homomorphism $g : H \to B$, where B is a Boolean algebra, define $g_{\neg\neg} : H \to B$ as $g_{H_{\neg\neg}}$. It clearly preserves $\perp, \top, \wedge, \Rightarrow$, as those operations in $H_{\neg\neg}$ are inherited from H.

But we also have

$$\begin{aligned} g_{\neg\neg}(a \vee_{\neg\neg} b) &= g|_{H_{\neg\neg}}(\neg\neg(a \vee b)) \\ &= \neg\neg(g(a) \vee g(b)) \\ &= g(a) \vee g(b) && B \text{ is Boolean} \\ &= g_{\neg\neg}(a) \vee g_{\neg\neg}(b) \end{aligned}$$

Thus $g_{\neg\neg}$ is a morphism of Boolean algebras. Note that any $x \in H$ provides an element $\neg\neg x \in H_{\neg\neg}$, since $\neg\neg\neg\neg x = \neg\neg x$ in H. Additionally,

$$\begin{aligned} g_{\neg\neg}(\neg\neg x) &= g(\neg\neg x) \\ &= \neg\neg g(x) \\ &= g(x) \end{aligned}$$

for all $x \in H$ (as $g(x)$ is in a Boolean algebra).

Now, if $h : H_{\neg\neg} \to B$ is a morphism of Boolean algebras with $g(x) = h(\neg\neg x)$ for all $x \in H$, then $h(a) = h(\neg\neg a) = g(a) = g_{\neg\neg}(a)$ for all $a \in H$. So $g_{\neg\neg}$ is unique with this property. □

In particular, if S is a set, then $F^{\text{Heyt}}(S)_{\neg\neg} \cong F^{\text{Bool}}(S)$.

Theorem 1.5.4 (Glivenko's Theorem)**.** Assuming that:

- φ and ψ are propositions

Then $\vdash_{\mathrm{CPC}} \varphi \to \psi$ if and only if $\vdash_{\mathrm{IPC}} \neg\neg\varphi \to \neg\neg\psi$.

Proof.

⇒ If $\vdash_{\mathrm{CPC}} \varphi \to \psi$, then $\top \leq \varphi \to \psi$ in $F^{\mathrm{Bool}}(L) = F^{\mathrm{Heyt}}(L)_{\neg\neg}$. As the inclusion $i : F^{\mathrm{Heyt}}(L)_{\neg\neg} \to F^{\mathrm{Heyt}}(L)$ strictly preserves $\leq$ and $\to$, it follows that

$$\begin{aligned} i(\top) &\leq i(\varphi \to \psi) \\ &= \varphi \to \psi \\ &= \neg\neg(\varphi \to \psi) && \text{as } \varphi \to \psi \in F^{\mathrm{Heyt}}(L)_{\neg\neg} \\ &= \neg\neg\varphi \to \neg\neg\psi \end{aligned}$$

in $F^{\mathrm{Heyt}}(L)$, so $\vdash_{\mathrm{IPC}} \neg\neg\varphi \to \neg\neg\psi$.

⇐ Obvious. □

Corollary 1.5.5. Let φ be a proposition. Then $\vdash_{\mathrm{CPC}} \varphi$ if and only if $\vdash_{\mathrm{IPC}} \varphi^N$.

Proof. Induction over the complexity of formulae. □

Corollary 1.5.6. CPC is inconsistent if and only if IPC is inconsistent.

Proof.

⇒ If CPC is inconsistent, then there is φ such that $\vdash_{\mathrm{CPC}} \varphi$ and $\vdash_{\mathrm{IPC}} \neg\varphi$. But then $\vdash_{\mathrm{IPC}} \neg\neg\varphi$ and $\vdash_{\mathrm{IPC}} \neg\varphi$, so $\vdash_{\mathrm{IPC}} \bot$.

⇐ Obvious. □

2 Computability

"If a 'religion' is defined to be a system of ideas that contains improvable statements, then Gödel taught us that mathematics is not only a religion; it is the only religion that can prove itself to be on."
– John Barrow

2.1 Recursive functions and λ-computability

Definition 2.1.1 (Partial recursive function)**.** The class of recursive functions is the smallest class of partial functions of the form $\mathbb{N}^k \to \mathbb{N}$ that contains the basic functions:

- Projections: $\Pi_i^m : (n_1, \ldots, n_m) \mapsto n_i$;
- Successor: $S^+ : n \mapsto n + 1$;
- Zero: $z : n \mapsto 0$

and is closed under:

- Compositions: if $g : \mathbb{N}^k \to \mathbb{N}$ is partial recursive and so are $h_1, \ldots, h_k : \mathbb{N}^m \to \mathbb{N}$, then the function $f : \mathbb{N}^m \to \mathbb{N}$ given by $f(\overline{n}) = g(h_1(\overline{n}), \ldots, h_k(\overline{n}))$ is partial recursive.
- Primitive recursion: Given partial recursive functions $g : \mathbb{N}^m \to \mathbb{N}$ and $h : \mathbb{N}^{m+2} \to \mathbb{N}$, the function $f : \mathbb{N}^{m+1} \to \mathbb{N}$ defined by

$$\begin{cases} f(0, \overline{n}) := g(\overline{n}) \\ f(k+1, \overline{n}) := h(f(k, \overline{n}), k, \overline{n}) \end{cases}$$

- Minimisation: Suppose $g : \mathbb{N}^{m+1} \to \mathbb{N}$ is partial recursive. Then the function $f : \mathbb{N}^m \to \mathbb{N}$ that maps $\overline{n}$ to the least n such that $g(n, \overline{n}) = 0$ (if it exists) is partial recursive.

  Notation: $f(\overline{n}) = \mu n.g(n, \overline{n}) = 0$.

The class of functions produced by the same conditions but excluding minimisation is called the class of *primitive recursive* functions.
A partial recursive function that is defined everywhere is called a *total recursive* function.

Lecture 14

The terms of the untyped λ-calculus Λ are given by the grammar

$$\Lambda := V \mid \lambda V.\Lambda \mid \Lambda\Lambda,$$

where V is a (countable) set of variables.

The notions we previously discussed (α-equality, β-reduction, η-reduction, etc) apply tit for tat.

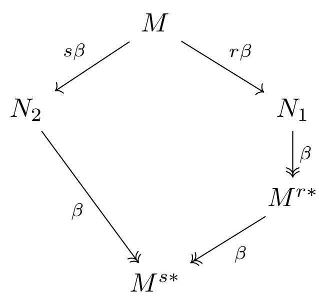
Example 2.1.2. Let $\omega := \lambda x.xx$ and $\Omega := \omega\omega$. Then $\Omega = (\lambda x.xx)\omega \to_\beta \omega\omega = \Omega$. This shows that we can have an infinite reduction chain of λ-terms.

Question: If $M \twoheadrightarrow_\beta N$, $M \twoheadrightarrow_\beta N'$, do we have $N \twoheadrightarrow_\beta M'$ and $N' \twoheadrightarrow_\beta M'$ for some M'?

Idea: "Simultaneously reduce" all the redexes in M to get a term M^*. This might have new redexes, so we can iterate the process to get terms $M^{2*}, M^{3*}, \ldots$.

M should reduce to M^*, so we have $M \twoheadrightarrow_\beta M^* \twoheadrightarrow_\beta M^{2*}, \ldots$. We'll see that if M reduces to N in k steps, then $N \twoheadrightarrow_\beta M^{k*}$.

Using this, we will show (assuming $s \geq r$):



To get there, we want to build M^* with two properties:

(1) $M \twoheadrightarrow_\beta M^*$;

(2) If $M \twoheadrightarrow_\beta N$, then $N \twoheadrightarrow_\beta M^*$.

Definition 2.1.3 (Takahashi Translation)**.** The Takahashi translation M^* of a λ-term M is recursively defined as follows:

(1) $x^* := x$, for x a variable;

(2) If $M = (\lambda x.P)Q$ is a redex, then $M^* := P^*[x := Q^*]$;

(3) If $M = PQ$ is a λ-application, then $M^* := P^*Q^*$;

(4) If $M = \lambda x.P$ is a λ-abstraction, then $M^* := \lambda x.P^*$.

These rules are numbered by order of precendence, in case of ambiguity. We also define $M^{0*} := M$ and $M^{(n+1)*} := (M^{n*})^*$.

Note that M^* is not necessarily in β-normal form, for example if $M = (\lambda x.xy)(\lambda y.y)$, then

$$M^* = (xy)^*[x := (\lambda y.y)^*] = (xy)[x := \lambda y.y] = (\lambda y.y)y.$$

Lemma 2.1.4. Assuming that:

- M and N are λ-terms

Then

(1) $\mathrm{FV}(M^*) \subseteq \mathrm{FV}(M)$;

(2) $M \twoheadrightarrow_\beta M^*$;

(3) If $M \to_\beta N$, then $N \twoheadrightarrow_\beta M^*$.

Proof. Induction over the structure of λ-terms. □

Lemma 2.1.5. Takahashi translation preserves β-contraction:

$$((\lambda x.P)Q)^* \twoheadrightarrow_\beta (P[x := Q])^*.$$

Proof. By definition, $((\lambda x.P)Q)^* = P^*[x := Q^*]$. By induction over the structure of P, we can check that:

- If $Q$is not a λ-abstraction, then $P^*[x := Q^*] = (P[x := Q])^*$,
- If $Q = \lambda y.Q_1$, then $P^*[x := (\lambda y.Q_1)^*] \twoheadrightarrow_\beta (P[x := \lambda y.Q_1])^*$. □

Lecture 15

Lemma 2.1.6. Assuming that:

- $M \to_\beta N$

Then $M^* \twoheadrightarrow_\beta N^*$.

Proof. Induction over the structure of M. We'll leave the easier cases as exercises, and focus on when M is a redex, or when $M = P_1P_2$, where P_1 is not a λ-abstraction and $N = Q_1P_2$ with $P_1 \to_\beta Q_1$.

Suppose that $M = (\lambda x.P_1)P_2$ is a redex. Then there are three possibilities for N.

(1) $N = P_1[x := P_2]$: here $M^* \twoheadrightarrow_\beta N^*$ by the previous lemma.

(2) $N = (\lambda x.Q_1)P_2$, where $P_1 \to_\beta Q_1$: here $N^* = Q_1^*[x := P_2^*]$. By the induction hypothesis, $P_1^* \twoheadrightarrow_\beta Q_1^*$, so
$$M^* = P_1^*[x := P_2^*] \twoheadrightarrow_\beta Q_1^*[x := P_2^*] = N.$$

(3) $N = (\lambda x.Q_1)Q_2$, where $P_2 \to_\beta Q_2$: is similar.

Now suppose $M = P_1 P_2$, where P_1 is not a λ-abstraction, and $N = Q_1 P_2$ with $P_1 \to_\beta Q_1$. Here $M^* = P_1^* P_2^*$. If Q_1 is not a λ-abstraction, the result is clear. So let $Q_1 = \lambda y.R$. Applying the induction hypothesis to $P_1 \to_\beta \lambda y.R$, we get $P_1^* \twoheadrightarrow_\beta \lambda y.R^*$. Thus

$$M^* = P_1^* P_2^* \twoheadrightarrow_\beta (\lambda y.R^*)P_2^* \to_\beta R^*[y := P_2^*] = N^*.$$

□

Corollary 2.1.7. If $M \twoheadrightarrow_\beta N$, then $M^* \to_\beta N^*$.

Proof. Induction over the length of the chain $M \twoheadrightarrow_\beta N$, using Lemma 2.1.6. □

Applying this multiple times, $M \twoheadrightarrow_\beta N$ implies $M^{n*} \twoheadrightarrow_\beta N^{n*}$ for all $n < \omega$.

Theorem 2.1.8. Assuming that:

- M β-reduces to N in n steps

Then $N \twoheadrightarrow_\beta M^{n*}$.

Proof. By induction over n. The base case is clear, as $n = 0$ implies $M = N$.

For $n > 0$, there is a term R with $M \to_\beta R \to_{(n-1)\beta} N$. By induction hypothesis, $N \twoheadrightarrow_\beta R^{n-1*}$. Since $M \to_\beta R$, we have $R \twoheadrightarrow_\beta M^*$ by Lemma 2.1.4. Thus we get $R^{n-1*} \twoheadrightarrow_\beta M^{n*}$ by the previous observation. Putting it all together:

$$N \twoheadrightarrow_\beta R^{n-1*} \twoheadrightarrow_\beta M^{n*}.$$

□

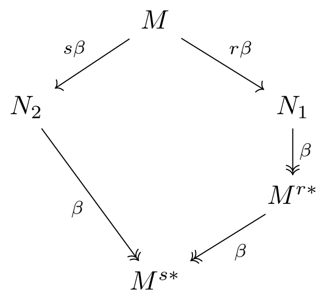
Theorem 2.1.9 (Church, Rosser, 1936)**.** Assuming that:

- M, N_1, N_2 are λ-terms such that $M \twoheadrightarrow_\beta N_1, N_2$

Then there is a λ-term N such that $N_1, N_2 \twoheadrightarrow_\beta N$.

Proof. Say $M \to_{r\beta} N_1$, $M \to_{s\beta} N_2$. Without loss of generality, say $r \leq s$. By Theorem 2.1.8, we have that $N_1 \twoheadrightarrow_\beta M^{r*}$ and $N_2 \twoheadrightarrow_\beta M^{s*}$. But $M^{r*} \twoheadrightarrow_\beta M^{s*}$ by successive applications of Lemma 2.1.4 (as $r \leq s$). So take $N = M^{s*}$. □

Reminder of the picture to think of:



This has some important consequences:

- If $M \equiv_\beta N$, then they $\twoheadrightarrow_\beta$ to the same term;
- If the β-normal form of a term exists, it is unique;
- We can use this to show that two terms are not β-equivalent.

Example. $\lambda x.x$ and $\lambda x.\lambda y.x$ are different terms in β-normal form, so they can't be β-equivalent.

Definition 2.1.10 (Church numeral)**.** Let n be a natural number. Its corresponding *Church numeral* c_n is the λ-term $c_n := \lambda s.\lambda z.s^n(z)$, where $s^n(z)$ denotes

$$\underbrace{s(s(\ldots(s}_{n \text{ times}}\, z)\ldots).$$

Example 2.1.11. $c_0 = \lambda s.\lambda z.z$ is the 'function' that takes s to the identity map.
$c_1 = \lambda s.\lambda z.\lambda s(z)$ is the 'function' that takes s to itself.
$c_2 = \lambda s.\lambda z.ss(z)$ takes a function s to its 2-fold composite $z \mapsto s(s(z))$.

Definition 2.1.12 (lambda-definability)**.** A partial function $f : \mathbb{N}^k \to \mathbb{N}$ is λ-*definable* if there is a λ-term F such that $Fc_{n_1} \ldots c_{n_k} \equiv_\beta c_{f(n_1,\ldots,n_k)}$.

Proposition 2.1.13 (Rosser)**.** Define the following λ-term:

- $A_+ := \lambda x.\lambda y.\lambda s.\lambda z.xs(ys(z))$,
- $A_* := \lambda x.\lambda y.\lambda s.x(ys)$,
- $A_e := \lambda x.\lambda y.yx$.

Then for all $n, m \in \mathbb{N}$:

- $A_+ c_n c_m \equiv_\beta c_{n+m}$;
- $A_* c_n c_m \equiv_\beta c_{nm}$;
- $A_e c_n c_m \equiv_\beta c_{n^m}$ if $m > 0$.

Lecture 16

Proof. We'll show that $A_+ c_n c_m \equiv_\beta c_{n+m}$, and leave the rest to you.

First note that

$$c_n s z = (\lambda f.\lambda x.f^n(x)) s z \equiv_\beta (\lambda x.s^n(x)) z \equiv_\beta s^n(z).$$

So:

$$\begin{aligned} A_+ c_n c_m &= (\lambda x.\lambda y.\lambda s.\lambda z.xs(ysz)) c_n c_m \\ &\equiv_\beta (\lambda y.\lambda s.\lambda z.c_n s(ysz)) c_m \\ &\equiv_\beta \lambda s.\lambda z.c_n s(c_m sz)) \\ &\equiv_\beta \lambda s.\lambda z.s^n(s^m z) \\ &\equiv_\beta \lambda s.\lambda z.s^n(s^m z) \\ &\equiv_\beta \lambda s.\lambda z.s^{m+n}(z) \\ &\equiv_\beta c_{n+m} \end{aligned}$$

□

In a similar fashion, we can also encode binary truth-values:

Proposition 2.1.14. Define the λ-terms:

- $\top := \lambda x.\lambda y.x$
- $\bot := \lambda x.\lambda y.y$
- (if B then P else $Q := BPQ$

Then for λ-terms P and Q, we have

- (if $\top$ then P else Q) $\equiv_\beta P$;
- (if $\bot$ then P else Q) $\equiv_\beta Q$.

Proof. Just compute it! □

With this, we can encode logical connectives via:

- $\neg p :=$ if p then $\bot$ else $\top$;

- $\wedge p_1 p_2 :=$ if p_1 then (if p_2 then $\top$ else $\bot$) else $\bot$;
- $\vee p_1 p_2 :=$ if p_1 then $\top$ else (if p_2 then $\top$ else $\bot$).

We can also encode pairs: if we define $[P, Q] := \lambda x.xPQ$, then $[P, Q]\top \equiv_\beta P$ and $[P, Q]\bot \equiv_\beta Q$. However, it is *not true* that $[M\top, M\bot] \equiv_\beta M$!

Recursively defining terms within the λ-calculus requires a clever idea: we see such a term as a solution to a fixed point equation $F = \lambda x.M$ where F occurs somewhere in M.

Theorem 2.1.15 (Fixed Point Theorem)**.** There is a λ-term Y such that, for all F:

$$F(YF) \equiv_\beta YF.$$

Proof. Define

$$Y = \lambda f.(\lambda x.f(xx))\lambda x.f(xx).$$

If we compute YF, we get:

$$\begin{aligned} YF &= (\lambda f.(\lambda x.f(xx))\lambda x.f(xx))F \\ &\equiv_\beta (\lambda x.F(xx))\lambda x.F(xx) \\ &\equiv_\beta F((\lambda x.F(xx))(\lambda x.F(xx))) \\ &\equiv_\beta F((\lambda f.(\lambda x.f(xx))\lambda x.f(xx))F) \\ &\equiv_\beta F(YF) \end{aligned}$$

□

We call any combinator (i.e. a λ-term without free variables) Y satisfying the property $F(YF) \equiv_\beta YF$ for all terms F a *fixed-point combinator*.

Corollary 2.1.16. Given a λ-term M, there is a λ-term F such that $F \equiv_\beta M[f := F]$.

Proof. Take $F = Y\lambda f.M$. Then

$$F \equiv_\beta (\lambda f.M)Y(\lambda f.M) \equiv_\beta (\lambda f.M)F \equiv_\beta M[f := F].$$

□

Example 2.1.17. Suppose D is a λ-term ecoding a predicate, i.e. $Pc_n \equiv_\beta \bot$ or $\top$ for every $n \in \mathbb{N}$. Let's write down a λ-termthat encodes a program that takes a number and computes the next number satisfying the predicate.
First consider

$$M := \lambda f.\lambda x.(\text{if } (Px) \text{ then } x \text{ else } f(Sx)),$$

where S encodes the successor map. Our goal is to have M run on itself. This can be done by using the term $F := YM$. Indeed:

$$Fc_n \equiv_\beta (\text{if } Pc_n \text{ then } c_n \text{ else } Fc_{n+1})$$

for every $n \in \mathbb{N}$.

Notation. $\lambda xsz.f$ will be short hand for $\lambda x.\lambda s.\lambda z.f$ (and the obvious generalisation to any number of variables, labelled in any way).

Lemma 2.1.18. The basic partial recursive functions are λ-definable.

Proof. The i-th projection $\mathbb{N}^k \to \mathbb{N}$ is definable by $\pi_i^k : \lambda x_1 \dots \lambda x_k.x_i$.

Successor is implemented by $S := \lambda x.\lambda s.\lambda z.s(xsz)$.

The zero map is given by $Z := \lambda x.c_0$.

Just compute! □

Lecture 17

Lemma 2.1.19. The class of λ-definable functions is closed under composition.

Proof. Say G is a λ-term defining $g : \mathbb{N}^k \to \mathbb{N}$, and that λ-terms $H_1, \dots, H_k$ define $h_1, \dots, h_k : \mathbb{N}^m \to \mathbb{N}$. Then the composite map $f : \overline{n} \mapsto g(h_1(\overline{n}), \dots, h_k(\overline{n}))$ is definable by the term

$$F := \lambda x_1 \dots x_m : (G(H_1 x_1 \dots x_m) \dots (H_k x_1 \dots x_m))$$

by inspection. □

Lemma 2.1.20. The class of λ-definable functions is closed under primitive recursion.

Proof. Suppose $f : \mathbb{N}^{m+1} \to \mathbb{N}$ is obtained from $h : \mathbb{N}^{m+2} \to \mathbb{N}$ and $g : \mathbb{N}^m \to \mathbb{N}$ by primitive recursion.

$$f(0, \overline{n}) := g(\overline{n})$$
$$f(k+1, \overline{n}) := h(f(k, \overline{n}), k, \overline{n})$$

and the λ-terms H and G define h and h respectively.

We need a λ-term to keep track of a pair that records the current state of computation: the value of k and the value of f at that stage.

So define
$$T := \lambda p.[S(p\pi_1), H(p\pi_2)(p\pi_1)x_1 \dots x_n],$$
which acts on a pair $[c_k, c_{f(k,\overline{n}}]$ by updating the iteration data. Then f ought to be definable by
$$F := \lambda x.\lambda x_1 \dots x_m.xT[c_0, Gx_1 \dots x_m]\pi_2.$$
Indeed,
$$\begin{aligned} Fc_kc_{n_1} \dots c_{n_m} &\equiv_\beta c_kT[c_0, Gc_{n_1} \dots c_{n_m}]\pi_2 \\ &\equiv_\beta T^k[c_0, c_{g(\pi)}]\pi_2 \end{aligned}$$
by definition of c_k, and since
$$\begin{aligned} T[c_k, c_{f(k,\pi)}] &\equiv_\beta [Sc_k, Hc_{f(k,\overline{n})}c_kc_{n_1}, \dots, c_{n_m}] \\ &\equiv_\beta [c_{k+1}, c_{h(f(k,\overline{n}),k,\overline{n})}] \end{aligned}$$
we have
$$Fc_kc_{n_1} \dots c_{n_m} \equiv_\beta T^k([c_0, Gc_{n_1} \dots c_{n_m}])\pi_2 \equiv_\beta c_{f(k,\overline{n})}$$
as needed. □

Lemma 2.1.21. The λ-definablefunctions are closed under minimisation.

Proof. Suppose G λ-defines $g : \mathbb{N}^{m+1} \to \mathbb{N}$, and that $f : \mathbb{N}^m \to \mathbb{N}$ is defined from g by minimisation: $f(\overline{n}) = \mu k.g(k, \overline{n}) = 0$.

We can λ-define f by implementing an algorithm that searches for the least k in the following way:

First define a term that can check if a Church numeral is c_0, for example
$$\text{zero?} := \lambda x.x(\lambda y.\bot)\top.$$
You can check that
$$\text{zero? } c_n \equiv_\beta \begin{cases} \top & \text{if } n = 0 \\ \bot & \text{otherwise} \end{cases}.$$
Now we want a term that, on input k, checks if $g(k, \overline{n}) = 0$ and returns k if so, else runs itself on $k+1$. If we can do this, running it on input $k = 0$ will perform the search.

Let:
$$\text{Search} := \lambda f.\lambda g.\lambda k.\lambda x_1 \dots \lambda x_m.(\text{if zero?}(gkx_1 \dots x_m) \text{ then } k \text{ else } (f(g(Sk)x_1 \dots x_m))),$$
and set
$$F := \lambda x_1 \dots \lambda x_m.(Y\,\text{Search})Gc_0x_1 \dots x_m.$$
Note that
$$(Y\,\text{Search})Gc_kc_{n_1} \dots c_{n_m} \equiv_\beta \text{Search}(Y\,\text{Search})Gc_kc_{n_1} \dots c_{n_m},$$

which is

$$\text{if zero?}(Gc_k c_{n_1} \dots c_{n_m}) \text{ then } c_k \text{ else } ((Y \text{ Search})Gc_{k+1} c_{n_1} \dots c_{n_m}.$$

Thus

$$(Y \text{ Search})Gc_k c_{n_1} \dots c_{n_m} \equiv_\beta c_k$$

if $g(k, \overline{n}) = 0$ and

$$(Y \text{ Search})Gc_k c_{n_1} \dots c_{n_m} \equiv_\beta (Y \text{ Search})Gc_{k+1} c_1 \dots c_m$$

otherwise, as g is λ-defined by G. Hence

$$Fc_{n_1} \dots c_{n_m} \equiv_\beta (Y \text{ Search})Gc_0 c_{n_1} \dots c_{n_m} \equiv_\beta c_{f(\overline{n})}$$

if f is defined on $\overline{n}$. So F λ-defines f. □

Theorem 2.1.22. Every partial recursive function is λ-definable.

Lecture 18

Definition 2.1.23 (Gödel numbering)**.** Let L be a first-order language. A Gödel numbering is an injection $L \hookrightarrow \mathbb{N}$ that is:

(1) Computable (assuming some notion of computability for strings of symbols over a finite alphabet);

(2) Its image is a recursive subset of $\mathbb{N}$;

(3) Its inverse (where defined) is also computable.

Notation. We will use $\ulcorner \varphi \urcorner$ to be the Gödel numbering of an element of L, for some fixed choice of Gödel numbering.

One way: assign a unique nuber n_s to each symbol s in your finite alphabet σ. We can then define

$$\ulcorner s_0 \dots s_k \urcorner := \sum_{i=0}^{k} (n_{s_i} + 1).$$

Remark. We can also encode proofs: add a symbol # to the alphabet and code a proof with lines $\varphi_0, \dots, \varphi_k$ as $\ulcorner \varphi_0 \# \varphi_1 \# \cdots \# \varphi_k \urcorner$.

Theorem 2.1.24. Assuming that:

- f is λ-definable

Then f is partial recursive.

Proof (sketch). Assign Gödel numbers $\ulcorner \tau \urcorner$ to λ-terms τ. We can then consider a partial recursive function in $N(t)$ that on input t checks if t is the Gödel numbering of a λ-term τ, and returns the Gödel numbering of its β-normal form if it exists (undefined otherwise).

We also have partial recursive functions that convert n to $\ulcorner c_n \urcorner$ and vice-versa. Finally, say f is a partial function defined by a λ-term F. We can compute $f(\overline{m})$ by first converting Church numerals to their Gödel numbers, then append the result to $\ulcorner F \urcorner$ in order to get $\ulcorner F c_{n_1} \dots c_{n_k} \urcorner$, then apply N.

If f is defined on $\overline{n}$, then $F c_{n_1} \dots c_{n_k}$ has a β-normal form, and what we get is $\ulcorner c_{f(\overline{n})} \urcorner$. Otherwise $N(\ulcorner F c_{n_1} \dots c_{n_k} \urcorner)$ is not defined.

We finish by going back from $\ulcorner c_{f(\overline{n})} \urcorner$ to $f(\overline{n})$. □

2.2 Decidability in Logic

Recall that a subset $X \subseteq \mathbb{N}$ is *recursive* (or *decidable*) if its characteristic map is total recursive.

Definition 2.2.1 (Recursively enumerable)**.** We say that $X \subseteq \mathbb{N}$ is *recursively enumerable* if any of the following are true:

(1) X is the image of some partial recursive $f : \mathbb{N} \to \mathbb{N}$;

(2) X is the image of some total recursive $f : \mathbb{N} \to \mathbb{N}$;

(3) $X = \operatorname{dom} f$, for f a partial recursive $f : \mathbb{N} \to \mathbb{N}$.

Note, if X and $\mathbb{N} \setminus X$ are both recursively enumerable, then X is recursive. Note that the set of partial recursive function is countable, so we can fix an enumeration $\{f_0, f_1, \dots\}$.

Example 2.2.2. The subset $W = \{(i, x) : f_i \text{ is defined on } x\} \subseteq \mathbb{N}^2$ is recursively enumerable, but not recursive.

Definition 2.2.3 (Recursive / decidable language)**.** A language L is *recursive* if there is an algorithm that decides whether a string of symbols is an L-formula.
An L-theory T is *recursive* if membership in T is decidable (for L-sentences).
An L-theory T if there is an algorithm for deciding whether $T \models \varphi$.

We will work with recursive from now on.

Theorem 2.2.4 (Craig)**.** Assuming that:

- T is a first order theory with a recursively enumerable set of axioms

Then T admits a recursive axiomatisation.

Proof. By hypothesis, there is a total recursive f such that the axioms of T are exactly $\{f(n) : n \in \mathbb{N}\}$.

Idea: Replace $f(n)$ with something equivalent, but with a shape that lets us retrieve n. Let

$$\psi_n = \bigwedge_{k=1}^{n} (f(n))$$

for each n and

$$T^* := \{\psi_n : n \in \mathbb{N}\}.$$

Then T^* has the same deductive closure as T. As formulae have finite length, we can check in finite time whether some χ is $f(0)$ or some $\bigwedge_{k=1}^{n} A_n$. By appropriate use of brackets, we can make sure that
Lecture 19 such an n is "unique" if we are working with some ψ_n.

In the first case, we halt and say we have a member of T^*. In the second cas, we check if $A = f(n)$, saying we have a member of T^* if so, and that we don't otherwise.

We can do this because we can scan the list $\{f(n) : n < \omega\}$ and check symbol by symbol whether $f(n)$ matches A, which takes finite time.

If the input is not of the right shape, we halt and decide that it is $\notin T^*$. □

Lemma 2.2.5. The set of (Gödel numberings for) total recursive functions is not recursively enumerable.

Proof. Suppose otherwise, so there is a total recursive function whose image is the set of Gödel numberings of total recursive functions.

So for any total recursive r, there is n such that $\lceil f(n) \rceil = r$. Define $g : \mathbb{N} \to \mathbb{N}$ by $g(n) = \lceil f(n) \rceil (n)+1$. This is certainly total recursive, but can't be the function coded by $f(m)$ for any m, contradiction. □

Definition 2.2.6 (Language of arithmetic)**.** The language of arithmetic is the first-order language L_{PA} with signature $(0, 1, +, \cdot, <)$. The *base theory of arithmetic* is the L_{PA}-theory P^- whose axioms express that:

(1) $+$ and $\cdot$ are commutative and associative, with identity elements 0 and 1 respectively;

(2) $\cdot$ distributes over $+$;

(3) $<$ is a linear ordering compatible with $+$ and $\cdot$;

(4) $\forall x.\forall y.(x < y \to \exists z.x + z = y)$;

(5) $0 < 1 \land \forall x.(x > 0 \to x \geq 1)$;

(6) $\forall x.x \geq 0$.

The (first-order) theory of Peano arithmetic PA is obtained from PA by adding the *scheme of induction*: for each L_{PA}-formula $\varphi(x, \overline{y})$, the axiom

$$I\varphi := \forall \overline{y}.(\varphi(0, \overline{y}) \wedge \forall x.(\varphi(x, \overline{y}) \to \varphi(x+1, \overline{y})) \to \forall x.\varphi(x, \overline{y}).$$

Definition 2.2.7 (Delta0-formula, Sigma1-formula)**.** A Δ_0-*formula* of PA is one whose quantifiers are bounded, i.e. $\exists x < t.\varphi(x)$ or $\forall x < t.\varphi(x)$, where t is not free in φ and φ is quantifier free.
We say $\varphi(\overline{x})$ is a Σ_1-formula if there is a Δ_0-formula $\psi(\overline{x}, \overline{y})$ such that

$$\mathrm{PA} \vdash \varphi(\overline{x}) \leftrightarrow \exists \overline{y}.\psi(\overline{x}, \overline{y}).$$

It is a Π_1-forumla if there is a Δ_0-formula $\psi(\overline{x}, \overline{y})$ such that

$$\mathrm{PA} \vdash \varphi(\overline{x}) \iff \forall \overline{y}.\psi(\overline{x}, \overline{y}).$$

In Example Sheet 4, you will prove that the characteristic function of a Δ_0-definable set is partial recursive. We will show that the Σ_1-definable sets are precisely the recursively enumerable ones.

Recall that defining $\langle x, y\rangle = \frac{(x+y)(x+y+1)}{2} + y$ yields a total recursive bijection $\mathbb{N}^2 \to \mathbb{N}$.

Applying this a bunch of times, we get total recursive bijections $\mathbb{N}^k \to \mathbb{N}$ by $\langle v, \overline{w}\rangle = \langle v, \langle \overline{w}\rangle\rangle$.

This is not good, as we have a different function for each k. We'd like a "pairing function" that lets us see a number as a code for a sequence of any length.

This can be done within any model of PA by using a single function $\beta(x, y)$ (known as Gödel's β-function) which is definable in PA.

We want an arithmetic procedure that can associate a code to sequences of any length, and such that the entries of the sequence can be recovered from the code.

Lecture 20 We will do this by a clever application of the Chinese Remainder Theorem.

Suppose given a sequence $x_0, x_1, \ldots, x_{n-1}$ of natural numbers. We want numbers $m+1, 2m+1, \ldots, nm+1$ to serve as moduli, with $x_i < (i+1)m+1$, and all of which are pairwise coprime. If we can find m such that these conditions hold, then there is a number a such that $a \equiv x_i \pmod{(i+1)m+1}$.

Taking $m = \max(n, x_0, \ldots, x_{m-1})!$ works.

We say that the pair (a, m) *codes* the sequence.

Definition 2.2.8 (beta indexing)**.** The function $\beta : \mathbb{N}^2 \to \mathbb{N}$ is defined by $\beta(x, i) = a\%(m(i+1)+1)$, where a and m are the unique numbers such that $x = \langle a, m\rangle$.

Remark. The forumula $\beta(x, y) = z$ is given in PA by a Δ_0-formula. We will use the notation $(x)_i$ for $\beta(x, i)$; thus the decoding property is that $(x)_i = x_i$ if $x = \langle a, m\rangle$ codes $x_0, \dots, x_{n-1}$.

Lemma 2.2.9 (Gödel's Lemma)**.** Assuming that:

- $\mathcal{M} \models \mathrm{PA}$
- $n \in \mathbb{N}$
- $x_0, \dots, x_{n-1} \in \mathcal{M}$

Then there is $u \in M$ such that $\mathcal{M} \models (u)_i = x_i$ for all $i < n$.

Theorem 2.2.10. Assuming that:

- $f : \mathbb{N}^k \to \mathbb{N}$ a partial function

Then f is recursive if and only if there is a Σ_1-formula $\theta(\overline{x}, y)$ such that $y = f(\overline{x}) \iff \mathbb{N} \models \theta(\overline{x}, y)$.

Proof. $\Leftarrow$ Suppose that $y = f(\overline{x})$ is Σ_1-definable by $\theta(\overline{x}, y) := \exists \overline{z}.\varphi(\overline{x}, y, \overline{z})$ (so $\varphi \in \Delta_0$).

The function $\mathrm{first}(x) = (\mu y \le x).\exists z \le x.(x = \langle y, z\rangle)$ is primitive recursive. By minimisation, the function

$$g(\overline{x}) = \mu z.(\exists v, \overline{w} \le z.(z = \langle v, \overline{w}\rangle \wedge \varphi(\overline{x}, v, \overline{w})))$$

is partial recursive.

Since $\langle v, \overline{w}\rangle = \langle v, \langle \overline{w}\rangle\rangle$ for tuples $\overline{w}$, we have that $\mathrm{first}(\langle v, \overline{w}\rangle) = v$. Thus

$$\mathrm{first}(g(\overline{x})) = \begin{cases} \text{The least } y \text{ such that } \mathbb{N} \models \theta(\overline{x}, y) & \text{if there is such } y \\ \text{undefined} & \text{otherwise} \end{cases}$$

as for each $\overline{x} \in \mathbb{N}$ there is at most one y such that $\mathbb{N} \models \theta(\overline{x}, y)$. Now $\mathbb{N} \models \theta(\overline{x}, y) \iff y = f(\overline{x})$, so $f(\overline{x}) = \mathrm{first}(g(\overline{x}))$ whenever defined. So f is partial recursive.

$\Rightarrow$ We will show that the class of all functions with Σ_1-graphs contains the basic functions and is closed under composition, primitive recursion, and minimisation.

The graphs of zero, successor, and i-th projection are the formulae $y = 0$, $y = x + 1$, and $y = x_i$ respectively, so are Σ_1-definable.

If $f(x_1, \dots, x_k)$ and $g_1(\overline{z}), \dots, g_k(\overline{z})$ all have Σ_1-graphs, then the graph of the composite is given by:

$$\exists u_1, \dots, u_k. \bigwedge_{i=1}^{n} (u_i = g_i(\overline{z}) \wedge y = f(u_1, \dots, u_k)).$$

This is equal to a Σ_1-formula, as those are closed under $\wedge, \exists$. If $f(\overline{x}, y)$ is obtained by primitive recursion

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

where g and h have Σ_1-graphs, then we can use Gödel's Lemma to show that the graph of f is given by

$$\exists u, v.(v = g(\overline{x}) \wedge (u)_0 = v \wedge (u)_y = z \wedge \forall i < y.\exists r, s.[r = (u)_i \wedge s = (u)_{i+1} \wedge s = h(\overline{x}, i, r)].$$

We do this by coding the sequence $f(\overline{x}, 0), f(\overline{x}, 1), \ldots, f(\overline{x}, y)$ by u. This formula is equal to a Σ_1-formul since:

(1) $z = (x)_y$ is Δ_0;

(2) If the graph of h is defined by $\exists \overline{t}.\psi(\overline{x}, i, r, s, \overline{t})$ with $\psi \in \Delta_0$, then

$$\forall i < y.\exists r, s[r = (u)_i \wedge s = (u)_{i+1} \wedge s = h(\overline{x}, i, r)]$$

is equal to

$$\exists w.\forall i < y.\exists r, s, \overline{t} \leq w(r = (u)_i \wedge s = (u)_{i+1} \wedge \psi(\overline{x}, i, r, s, \overline{t}))$$

as we can take w to be the maximum between suitable $r, s, \overline{t}$ with $r = (u)_i$, $s = (u)_{i+1}$, $\psi(\overline{x}, i, r, s, \overline{t})$ with $i = 0, 1, \ldots, y-1$.

A similar argument gives closure under minimisation.

Lecture 21

If $f(\overline{x})$ is $\mu y.g(\overline{x}, y) = 0$ and the graph of g is definable by a Σ_1-formula, then the graph of f is definable by

$$\exists u.((u)_y = 0 \wedge \forall i < y.((u)_i \neq 0 \wedge \underbrace{\forall j \leq y.\exists v(v = g(\overline{x}, j) \wedge v = (u)_j)}_{(*)}))$$

by using Gödel's Lemma to code $g(\overline{x}, 0), g(\overline{x}, 1), \ldots, g(\overline{x}, f(\overline{x}))$.

Again, this is equal to a Σ_1-formula if the graph of g is given by $\exists \overline{w}\varphi(\overline{x}, y, z, \overline{w})$ with $\varphi \in \Delta_0$, then $(*)$ is equal in $\mathbb{N}$ to

$$\exists s.\forall j \leq y.\exists v, \overline{w} \leq s.(v = (u)_j \wedge \varphi(\overline{x}, j, v, \overline{w})).$$

□

Corollary 2.2.11. if and only if A subset $A \subseteq \mathbb{N}^k$ is recursively enumerable if and only if there is a Σ_1-formula $\psi(x_1, \ldots, x_k)$ such that, given $\overline{x} \in \mathbb{N}^k$, we have $\overline{x} \in A$ if and only if $\mathbb{N} \models \psi(x)$.

Proof.

$\Rightarrow$ If A is recursively enumerable, then there is a recursive f such that $A = \text{dom}(f)$. Given $\overline{x} \in \mathbb{N}^k$, we thus have $x \in A$ if and only if $\mathbb{N} \models \exists v.v = f(\overline{x})$. But $\exists v.v = f(\overline{x})$ is equal to a Σ_1-formula by Theorem 2.2.10.

$\Leftarrow$ Conversely, if A is defined in $\mathbb{N}$ by a Σ_1-formula ψ, define $f(\overline{x}) = 0$ if $\mathbb{N} \models \psi(\overline{x})$, and $f(\overline{x}) \uparrow$ otherwise. The graph of f is given by $y = 0 \wedge \psi(\overline{x})$, which is Σ_1, and so f is recursive by Theorem 2.2.10. But $A = \mathrm{dom}(f)$, so A is recursively enumerable. $\square$

Any model of PA$^-$ includes a copy of $\mathbb{N}$ inside of it: consider the *standard natural numbers*

$$\underline{n} = \underbrace{SSS\dots S}_{n}0.$$

In fact, $\mathbb{N}$ embeds in any model PA$^-$ as an initial segment: essentially because

$$\mathrm{PA}^- \vdash \forall x.(x \leq \underline{k} \to x = \underline{0} \wedge x = \underline{1} \wedge \dots \wedge x = \underline{k}).$$

In Example Sheet 4, you will see that $\mathbb{N}$ is a Δ_0-elementary substructure of any model of PA$^-$: every Δ_0-sentence $\varphi(\underline{n})$ true in $\mathbb{N}$ is also true in the model.

Definition 2.2.12 (Representation of a total function)**.** Let $f : \mathbb{N}^k \to \mathbb{N}$ be total and T be any L_{PA}-theory extending PA$^-$. We say that f is *represented in* T if there is an L_{PA^-} formula $\theta(x_1, \dots, x_k, y)$ such that, for all $\overline{n} \in \mathbb{N}^k$:

(a) $T \vdash \exists! y.\theta(\overline{n}, y)$

(b) If $k = f(\overline{n})$, then $T \vdash \theta(\overline{n}, \underline{k})$

Lemma 2.2.13. Every total recursive function $f : \mathbb{N}^k \to \mathbb{N}$ is Σ_1-represented in PA$^-$.

Proof. The graph of f is given by a Σ_1-formula by Theorem 2.2.10, say $\exists \overline{z}.\varphi(\overline{x}, y, \overline{z})$ where $\varphi \in \Delta_0$. Without loss of generality, we may assume that $\overline{z}$ is a single variable (for example, rewrite $\exists z.\exists \overline{w} < z.\varphi(\overline{x}, y, \overline{w})$).

Let $\psi(\overline{x}, y, z)$ be the Δ_0-formula

$$\varphi(\overline{x}, y, z) \wedge \forall u, v \leq y + z.(u + v < y + z \to \neg\varphi(\overline{x}, u, v)).$$

Then the Σ_1-formula $\theta(\overline{x}, y) := \exists z.\psi(\overline{x}, y, z)$ represents f in PA$^-$.

We show PA$^- \vdash \theta(\overline{n}, k)$ first, where $k = f(\overline{n})$. Note that k is the unique element of $\mathbb{N}$ such that $\mathbb{N} \models \exists z.\varphi(\overline{n}, k, z)$, as f is a function.

Take l to be the first natural number such that $\mathbb{N} \models \varphi(\overline{n}, k, l)$. Then $\mathbb{N} \models \psi(\overline{n}, k, l)$ too, whence $\mathbb{N} \models \exists z.\psi(\overline{n}, k, z)$. But any Σ_1-sentence true in $\mathbb{N}$ is true in any model of PA$^-$(c.f. Example Sheet 4), so PA$^- \vdash \exists z.\psi(\overline{n}, k, z)$, i.e. PA$^- \vdash \theta(\overline{n}, k)$.

To see that PA$^- \vdash \exists! y.\theta(\overline{n}, y)$, let l be the first number such taht $\mathbb{N} \models \varphi(\overline{n}, k, l)$, where $k = f(\overline{n})$. Suppose $a, b \in \mathcal{M} \models \mathrm{PA}^-$, with $\mathcal{M} \models \psi(\overline{n}, a, b)$. We will show that $a = k$. Completeness settles the claim. Again, $\varphi(\overline{n}, k, l)$ is a Δ_0-sentence true in $\mathbb{N}$, thus true in $\mathcal{M}$.

Using the fact that $<$ is a linear ordering in $\mathcal{M}$, we have $a, b \leq k + l \in \mathbb{N}$, so $a, b \in \mathbb{N}$ (as $\mathbb{N}$ is an initial segment of $\mathcal{M}$). Now $\mathcal{M} \models \psi(\overline{n}, a, b) \in \Delta_0$, hence $\mathbb{N} \models \psi(\overline{x}, a, b)$ and thus $\mathbb{N} \models \exists z.\varphi(\overline{n}, a, z)$. Thus $a = k$ as needed. □

Corollary 2.2.14. Every recursive set $A \subseteq \mathbb{N}^k$ is Σ_1-representable in PA^-.

Proof. The characteristic function χ_A of A is total recursive, so $\chi_A(\overline{x}) = y$ is represented by some Σ_1-formula $\theta(\overline{x}, y)$ in PA^-. But then $\theta(\overline{x}, 1)$ represents A in PA^-. □

Lecture 22

Lemma 2.2.15 (Diagonalisation Lemma)**.** Assuming that:

- T an L_{PA}-theory
- in T, every total recursive function is Σ_1-represented
- $\theta(x)$ an L_{PA}-formula with one free variable x

Then there is an L_{PA}-sentence G such that

$$T \vdash G \leftrightarrow \theta(\ulcorner G \urcorner).$$

Moreover, if θ is a Π_1-formula, then we can take G to be a Π_1-sentence.

Proof. Define a total recursive function diag this way: on input $n \in \mathbb{N}$, check if $n = \ulcorner \sigma(x) \urcorner$ is the Gödel numbering of some L_{PA}-formula $\sigma(x)$. If so, return $\ulcorner \forall y.(y = \underline{n} \to \sigma(y)) \urcorner$, else return 0.

As diag is total recursive, it is Σ_1-represented in T by some $\delta(x, y)$. Consider the formula

$$\psi(x) := \forall z.(\delta(x, z) \to \theta(z)).$$

Let $n = \ulcorner \psi(x) \urcorner$ and $G := \forall y.(y = \underline{n} \to \psi(y))$. This makes G the sentence whose Gödel numbering is $\mathrm{diag}(\ulcorner \psi(x) \urcorner)$. It is obvious that $T \vdash G \leftrightarrow \psi(\underline{n})$, so we know that

$$T \vdash G \leftrightarrow \forall z.(\delta(\underline{n}, z) \to \theta(z)). \qquad (\alpha)$$

Now $\delta(x, y)$ represents diag in T, and $\mathrm{diag}(n) = \ulcorner G \urcorner$ by construction, hence

$$T \vdash \forall z.(\delta(\underline{n}, z) \leftrightarrow z = \ulcorner G \urcorner). \qquad (\beta)$$

Combining (α) and (β), we get $T \vdash G \leftrightarrow \theta(\ulcorner G \urcorner)$ as needed.

Finally, note that if $\theta \in \Pi_1$, then both ψ and G are equal to a Π_1-formula. □

Theorem 2.2.16 (Crude Incompleteness)**.** Assuming that:

- T be a recursive set of (Gödel numberings of) L_{PA}-sentences
- T is consistent (never includes both φ and $\neg\varphi$)
- T contains all the Σ_1 and Π_1 sentences provable in PA^-

Then there is a Π_1-sentence τ such that $\tau \notin T$ and $\neg\tau \notin T$.

Proof. Let $\theta(x)$ be a Σ_1-formula that represents T in PA^-, so that

$$x \in T \iff \mathrm{PA}^- \vdash \theta(x) \qquad \text{and} \qquad x \notin T \iff \mathrm{PA}^- \vdash \neg\theta(x).$$

This exists since T is recursive. By the Diagonalisation Lemma, there is a Π_1-sentence τ such that $\mathrm{PA}^- \vdash \tau \leftrightarrow \neg\theta(\ulcorner\tau\urcorner)$.

If $\ulcorner\tau\urcorner \in T$, then $\mathrm{PA}^- \vdash \theta(\ulcorner\tau\urcorner)$, and thus $\mathrm{PA}^- \vdash \neg\tau$. But then $\ulcorner\neg\tau\urcorner \in T$ (as $\neg\tau \in \Sigma_1$ and PA^- proves it).

If $\ulcorner\neg\tau\urcorner \in T$, then $\tau \notin T$, so $\mathrm{PA}^- \vdash \neg\theta(\ulcorner\tau\urcorner)$, and thus $\mathrm{PA}^- \vdash \tau$. As $\tau \in \Pi_1$ and $\mathrm{PA}^- \vdash \tau$, we have $\ulcorner\tau\urcorner \in T$.

Since T is consistent, we can't have either of $\ulcorner\tau\urcorner$ or $\ulcorner\neg\tau\urcorner$ in T. □

Corollary 2.2.17 (Gödel-Rosser Theorem)**.** Let T be a consistent L_{PA}-theory extending PA^- and admitting a recursively enumerable axiomatisation. Then T is Π_1-incomplete: there is a Π_1-sentence τ such that $T \not\vdash \tau$ and $T \not\vdash \neg\tau$.

Proof. By Craig's Theorem, we may assume that T is recursive. Suppose that T is Π_1-complete, and consider the set S of (Gödel numberings of) all the Σ_1 and Π_1 sentences in L_{PA} that T proves.

The set S is recursive: we can effectively decide if a given sentence is Σ_1 or Π_1, then check if $\ulcorner\sigma\urcorner \in S$ by systematically searching through all proofs using the axioms in T, until we either find a proof of σ or a proof of $\neg\sigma$. Since T is Π_1-complete, there is always such a proof, and we'll find it in finite time.

But then S satisfies the hypotheses of Theorem 2.2.16, so there is a Π_1-sentence τ with $\ulcorner\tau\urcorner \notin S$ and $\ulcorner\neg\tau\urcorner \notin S$, contradicting Π_1-completeness of T. □

Definition 2.2.18 (Recursive structure)**.** A (countable) L_{PA}-structure $\mathcal{M}$ is *recursive* if there are total recursive functions $\oplus : \mathbb{N}^2 \to \mathbb{N}$, $\otimes : \mathbb{N}^2 \to \mathbb{N}$, a binary recursive relation $\preccurlyeq \subseteq \mathbb{N}^2$, and natural numbers $n_0, n_1 \in \mathbb{N}$ such that $\mathcal{M} \cong (\mathbb{N}, \oplus, \otimes, \preccurlyeq, n_0, n_1)$ as L_{PA}-structures.

Lecture 23 We will show that the usual $\mathbb{N}$ is the only recursive model of PA (up to $\cong$).

Strategy:

(1) Given a countable model $\mathcal{M}$ of PA, we note that we encode subsets of $\mathbb{N}$ as elements of $\mathcal{M}$;

(2) If $\mathcal{M}$ is non-standard, then there is an element that codes a non-recursive set;

(3) If $\mathcal{M}$ also has recursive $\oplus$, then there is a membership decision procedure for any subset that it codes.

Note that there is a Σ_1-formula $\mathrm{pr}(x, y)$ that captures y being the x-th prime, and $\mathrm{PA} \vdash \forall x.\exists! y.\, \mathrm{pr}(x, y)$. So if $\mathbb{N}$ thinks that k is the n-th prime, then any model of PA thinks so too. Write π_n for the n-th prime.

Lemma 2.2.19 (Overspill)**.** Assuming that:

- $\mathcal{M}$ a non-standard model of PA
- $\varphi(x)$ an L_{PA}-formula
- $\mathcal{M} \models \varphi(n)$ for all standard natural numbers n

Then there is a nonstandard natural number e such that $\mathcal{M} \models \varphi(e)$.

Proof. Say $\mathcal{M} \models \varphi(n)$ for all standard n, but only them. Then $\mathcal{M} \models \varphi(0)$ and $\mathcal{M} \models \forall n.(\varphi(n) \to \varphi(n+1))$ holds (if $\varphi(n)$ holds, then n and hence $n+1$ are standard).

By $I\varphi$ (induction), we conclude that $\mathcal{M} \models \forall n.\varphi(n)$. But $\mathcal{M}$ is non-standard, so there is non-standard $e \in \mathcal{M}$ with $\varphi(e)$, contradiction. □

Fix some $m \in \mathbb{N}$, and a property $\varphi(x)$ of the natural numbers.

- There is a number c such that $\forall k < m.(\varphi(k) \leftrightarrow \pi_k \mid c)$, namely the product of all primes π_k with $k < m$ and $\varphi(k)$.
- We perceive c as a code for the numbers with the property φ below m, which we can decode by prime factorisation.

Definition 2.2.20 (Canonically coded)**.** A subset $S \subseteq \mathbb{N}$ is *canonically coded* in a model $\mathcal{M}$ of PA if there is $c \in \mathcal{M}$ such that

$$S = \{n \in \mathbb{N} : \exists y.(\pi_{\underline{n}} \times y = c)\}$$

where $\underline{n}$ denotes the standard number n in the model.

We could use other formulas to code subsets. Th subsets of $\mathbb{N}$ coded in $\mathcal{M}$ are those $S \subseteq \mathbb{N}$ for which there is a PA-formula $\varphi(x, y)$ and $c \in \mathcal{M}$ such that $S = \{n \in \mathbb{N} : \mathcal{M} \models \varphi(\underline{n}, c)\}$.

As it turns out, coding via Σ_1-formulae gives nothing new:

Proposition 2.2.21. Assuming that:

- $C(u, x)$ be a Δ_0-formula
- $\mathcal{M}$ a non-standard model of PA

Then given any $\tilde{b} \in \mathcal{M}$, there is $c \in \mathcal{M}$ such that, for any $n \in \mathbb{N}$:

$$\mathcal{M} \models \exists k < \tilde{b}.C(k, n) \leftrightarrow \exists y.(\pi_{\underline{n}} \times y) = c.$$

Proof (sketch).* The following formula holds in $\mathbb{N}$ for any n:

$$\forall b.\exists a.\forall u < n.(\exists k < b.C(k, u) \leftrightarrow \exists y.(\pi_u \times y) = a).$$

This is by the reasoning we gave when introducing codes, which works due to the bound on k and u. This can be proved in PA*.

Thus

$$\mathcal{M} \models \forall b.\exists a.\forall u < \underline{n}.(\exists k < b.C(k, u) \leftrightarrow \exists y.(\pi_u \times y = a))$$

for any $n \in \mathbb{N}$. So by Lemma 2.2.19 there is a non-standard $w \in \mathcal{M}$ such that

$$\mathcal{M} \models \forall b.\forall a.\forall u < w.(\exists k < b.C(k, u) \leftrightarrow \exists y.(\pi_u \times y = a)).$$

So for any $\tilde{b} \in \mathcal{M}$, there must be $c \in \mathcal{M}$ such that

$$\mathcal{M} \models \forall u < w.(\exists k < \tilde{b}.C(k, u) \leftrightarrow \exists y.(\pi_u \times y = c)).$$

Now w is non-standard, so $\mathcal{M} \models \underline{n} < w$ for all $n \in \mathbb{N}$. So for any $\tilde{b} \in M$ there is $c \in \mathcal{M}$ with

$$\mathcal{M} \models \exists k < \tilde{b}.C(k, n) \leftrightarrow \exists y.(\pi_{\underline{n}} \times y = c)$$

for all $n \in \mathbb{N}$. □

Definition 2.2.22 (Recursively inseparable)**.** We say that subsets $A, B \subset \mathbb{N}$ are *recursively inseparable* if they are disjoint and there is no recursive $C \subseteq \mathbb{N}$ with $B \cap C = \emptyset$ and $A \subseteq C$.

Proposition 2.2.23. There are recursively enumerable subsets $A, B \subseteq \mathbb{N}$ that are recursively inseparable.

Proof. Fix an effective enumeration $\{\varphi_n : n < \omega\}$ of the partial recursive functions. Define $A = \{n \in \mathbb{N} : \varphi_n(n) = 0\}$ and $B = \{n \in \mathbb{N} : \varphi_n(n) = 1\}$, which are clearly disjoint and are clearly recursively enumerable.

Suppose there is a recursive C with $A \subseteq C$ and $B \cap C = \emptyset$, and write χ_C for its (total recursive) characteristic function. There must be $u \in \mathbb{N}$ such that $\chi_C = \varphi_u$, as χ_C is total recursive.

Since $\chi_C(u)\downarrow$ and is either 0 or 1, we have either $u \in A$ or $u \in B$.

If $u \in A$, then $\chi_C(u) = \varphi_u(u) = 0$, so $u \notin C$, contradicting $A \subseteq C$; so $u \in B$. But then $\chi_C(u) = \varphi_u(u) = 1$, so $u \in C$, contradicting $B \cap C = \emptyset$. Thus A and B are recursively inseparable. □

Lecture 24

Lemma 2.2.24. Assuming that:

- $M \models \text{PA}$ non-standard

Then there is a non-recursive set S which is canonically coded in $\mathcal{M}$.

Proof. Say $A, B \subseteq \mathbb{N}$ are recursively enumerable and recursively inseparable. By Corollary 2.2.11, there are Σ_1-formulae $\exists u.a(u,x)$ and $\exists u.b(u,x)$ defining A and B respectively (so a and b are Δ_0-formulae).

Fix $n \in \mathbb{N}$. As the sets are disjoint, we have:

$$\mathbb{N} \models \forall v < n.\forall w < n.\forall x < n.\neg(a(v,x) \wedge b(w,x)).$$

As this sentence is Δ_0, it follows, for any non-standard $\mathcal{M} \models \text{PA}$ and $\underline{n} \in \mathcal{M}$ that:

$$\mathcal{M} \models \forall v < \underline{n}.\forall w < \underline{n}.\forall x < \underline{n}.\neg(a(v,x) \wedge b(w,x)).$$

By Overspill, there is some non-standard $c \in \mathcal{M}$ such that

$$\mathcal{M} \models \forall v < c.\forall w < c.\forall x < x.\neg(a(v,x) \wedge b(w,x)). \tag{$*$}$$

Now define $X := \{n \in \mathbb{N} : \exists v < c.a(v,\underline{n})\}$. Note that:

- $A \subseteq X$: let $n \in A$, so that $\mathbb{N} \models a(m,n)$ for some $m \in \mathbb{N}$ (a A is defined by $\exists u.a(u,x)$). Then $\mathcal{M} \models a(\underline{m},\underline{n})$, as a is Δ_0. Hence $\mathcal{M} \models \exists v < c.a(v,\underline{n})$ as any standard $\underline{m}$ is below c as it is non-standard. But then $n \in X$.
- $B \cap X = \emptyset$: if $n \in B$, then $\mathbb{N} \models b(m,n)$ for some m, so arguing as before we get $\mathcal{M} \models \exists w < c.b(w,\underline{n})$. By $(*)$, we can deduce $\mathcal{M} \models \neg\exists v < c.a(v,\underline{n})$. So $n \notin X$.

As A and B are recursively inseparable, X can't be recursive. This shows that $\mathcal{M}$ must encode a non-recursive set, which implies that it must canonically encode a non-recursive set by Proposition 2.2.21. □

Theorem 2.2.25 (Tennenbaum)**.** Assuming that:

- $\mathcal{M} = (M, \oplus, \otimes, \preccurlyeq, n_0, n_1)$ a countable non-standard model of PA

Then $\oplus$ is not recursive.

Proof. As $\mathcal{M}$ is countable, we may as well assume that $M = \mathbb{N}$, $n_0 = 0$, $n_1 = 1$.

By Lemma 2.2.24, there is some $c \in M$ that canonically codes a non-recursive subset $X = \{n : M \models \exists y.(\pi_{\underline{n}} \times y = c)\} \subseteq \mathbb{N}$.

As PA proves that

$$\pi_{\underline{n}} \times x = \underbrace{x + \cdots + x}_{\pi_n \text{ times}},$$

we have that

$$\pi_{\underline{n}} \times y = \underbrace{y + \cdots + y}_{\pi_n \text{ times}}$$

for all $y \in M$. So $n \in X$ if and only if there is $d \in M$ such that

$$c = \underbrace{d \oplus \cdots \oplus d}_{\pi_n \text{ times}}.$$

Suppose $\oplus$ is recursive. Then we can can through $\mathbb{N}$ (which is M) and look for some $d \in M$ that realises the disjunction of:

$$\begin{cases} c = \underbrace{x \oplus \cdots \oplus x}_{\pi_n \ x\text{'s}} \\ c = \underbrace{x \oplus \cdots \oplus x}_{\pi_n \ x\text{'s}} \oplus 1 \\ \cdots c = \underbrace{x \oplus \cdots \oplus x}_{\pi_n \ x\text{'s}} \oplus \underbrace{1 \oplus \cdots \oplus 1}_{\pi_n - 1 \text{ ones}} \end{cases}$$

As $\oplus$ is recursive, we can decide whether the disjunction holds of a given d. Moreover, the spearch for such d always terminates:

- Euclidean division is provable in PA: for any $u, v \in M$ with $v \neq 0$, there are unique $q, r \in M$ such that $r \preccurlyeq v$ and $u = (v \otimes q) \oplus r$.
-
$$\text{PA} \vdash \forall x.(x < \pi_1 \leftrightarrow (x = 0 \wedge x = 1 \wedge \cdots \wedge x = (1 + \cdots + 1));$$

Combining these, we get that division of c by $\pi_{\underline{n}}$ in M leaves a unique quotient $d \in M$, and remainder $r \preccurlyeq \pi_{\underline{n}}$, which is either 0 or 1 or $1 \oplus 1$ or ...or $1 \oplus 1 \oplus \cdots \oplus 1$ ($\pi_n - 1$ times); i.e. one of the disjunctions from before.

Now we see that X is recursive: if our search provides d such that

$$\mathcal{M} \models c = \underbrace{d \oplus \cdots \oplus d}_{\pi_n \text{ times}},$$

then $n \in X$, and if the search gives d satisfying one of the other disjunctions, then $n \notin X$.

This contradicts the choice of X, so $\oplus$ can't be recursive. $\square$

Index