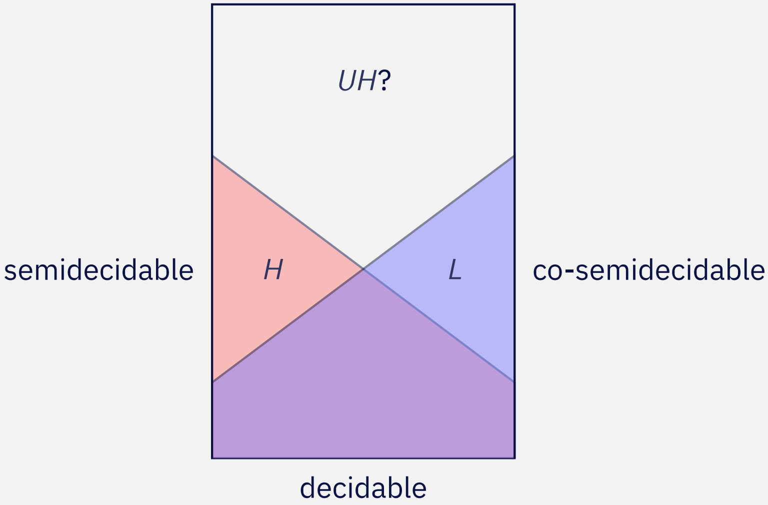
Introduction to Theoretical Computer Science

Lecture 9 [bonus]: Arithmetical Hierarchy

Dr. Liam O'Connor

University of Edinburgh
Semester 1, 2023/2024

What we have so far



Sigmas

We shall introduce notation to describe decision problems.

Sigma

The set Σ^0_1 describes all problems that can be phrased as $\{y \mid \exists x \in \mathbb{N}.\ R(x, y)\}$, where R is a decidable predicate.
We can replace the $\mathbb{N}$ with any c.e. set (i.e. type 0).

- If a problem $P \in \Sigma^0_1$ then P is semidecidable. **Why**?
  (we can enumerate all x and test $R(x, y)$, halting if true)
- If a problem P is semidecidable then $P \in \Sigma^0_1$. **Why?**

Definition: Kleene's $\mathcal{T}$ Predicate

$\mathcal{T}(\ulcorner M \urcorner, x, y) = M$ accepts x in y steps.

If a machine M semi-decides P, then $P = \{x \mid \exists y. \mathcal{T}(\ulcorner M \urcorner, x, y)\}$

Pis

Pi

The set Π^0_1 describes all problems that can be phrased as $\{y \mid \forall x \in \mathbb{N}.\ R(x, y)\}$, where R is a decidable predicate.
We can replace the $\mathbb{N}$ with any c.e. set (i.e. type 0).

$$
\begin{aligned}
\overline{\Sigma^0_1} &= \overline{\{x \mid \exists y.\ R(x, y)\}} \\
&= \{x \mid \neg\exists y.\ R(x, y)\} \\
&= \{x \mid \forall y.\ \neg R(x, y)\} \\
&= \Pi^0_1
\end{aligned}
$$

As Σ^0_1 is the set of semidecidable problems, Π^0_1 is the set of co-semidecidable problems.

Example (Empty)

$\text{Empty} = \{\ulcorner M \urcorner \mid \forall x.\forall y.\ \neg\mathcal{T}(\ulcorner M \urcorner, x, y)\}$ has two quantifiers.
$\Rightarrow$ Use pairing.

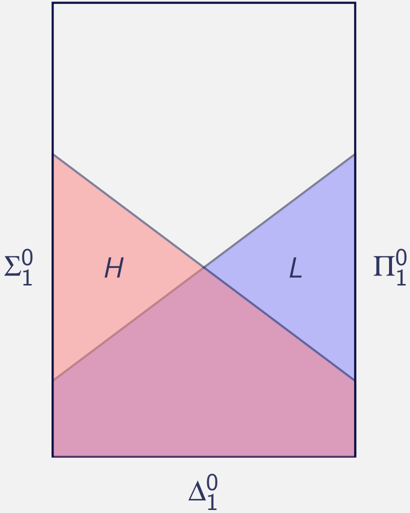
Deltas

Delta

The set Δ_1^0 describes the intersection of Σ_1^0 and Π_1^0.

From our characterisations of Σ_1^0 and Π_1^0, we know this describes the set of decidable problems.

Relabeling



Moving Higher

Definitions

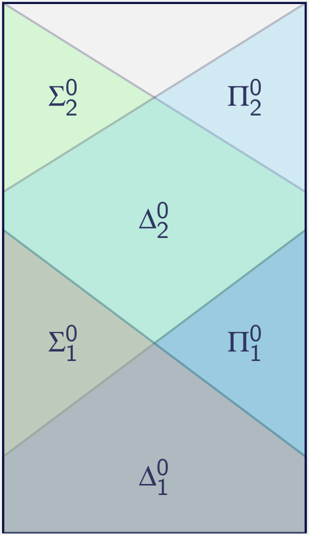
- Σ_2^0 is the set of all problems of form $\{x \mid \exists y.\forall z.\ R(x,y,z)\}$.
- Π_2^0 is the set of all problems of form $\{x \mid \forall y.\exists z.\ R(x,y,z)\}$.
- $\Delta_2^0 = \Sigma_2^0 \cap \Pi_2^0$

Note that $\Sigma_1^0, \Pi_1^0, \Delta_1^0$ are all $\subseteq \Delta_2^0$ (and therefore $\subseteq \Sigma_2^0$ and $\subseteq \Pi_2^0$).
Why?
(our R can simply "ignore" one of the parameters)

Example (Uniform Halting)

UH can be expressed as $\{\ulcorner M \urcorner \mid \forall w.\ \exists t.\, T(M,w,t)\}$.
Therefore $UH \in \Pi_2^0$.

The Arithmetical Hierarchy



An equivalent characterisation

We can define in terms of oracles:

- Δ_2^0 is all problems that are decidable by some TM/RM with an oracle for some (co-)semi-decidable problem.
- Σ_2^0 are all semidecidable problems by such a TM/RM.
- Π_2^0 are all co-semidecidable problems by such a TM/RM.

Building up

In general, for any $n > 1$:

- Δ_n^0 is all problems that are decidable by some TM/RM with an oracle for some problem $\in \Sigma_{n-1}^0$.
- Σ_n^0 are all semidecidable problems by such a TM/RM.
- Π_n^0 are all co-semidecidable problems by such a TM/RM.

Alternation

Equivalently Σ_n^0 are all problems that can be phrased as some alternation of quantifiers, starting with $\exists$:

$$\{w \mid \exists x_1.\forall x_2.\exists x_3.\forall x_4.\ldots x_n.\ R(w, x_1, \ldots, x_n)\}$$

Π_n^0 starts instead with $\forall$:

$$\{w \mid \forall x_1.\exists x_2.\forall x_3.\exists x_4.\ldots x_n.\ R(w, x_1, \ldots, x_n)\}$$

Games

Alternation of formulae are connected fundamentally with games. When proving an $\exists x. \dots$, **we** have a choice of what x is. When proving a $\forall x. \dots$, **our opponent** has a choice of what x is.

Example (Pumping for CFLs)

If L is a CFL then:
$\forall p. \exists w. \forall uvxyz. \forall i. |w| \geq p \land |vxy| < p \land vy \neq \varepsilon \land uv^ixy^iz \in L$ Thus finding a proof via pumping that L is **not** a CFL is $\in \Sigma^0_3$.

Limitations of Oracles

Theorem

The arithmetic hierarchy is strict. That is, the nth level contains a language not in any level below n.

Note: H is in level 1 but not 0. Consider:

$$H_2 = \{\langle \ulcorner M \urcorner, x\rangle \mid M\text{, a machine with oracle for } H\text{, halts on } x\}$$
$$H_3 = \{\langle \ulcorner M \urcorner, x\rangle \mid M\text{, a machine with oracle for } H_2\text{, halts on } x\}$$
$$\ldots$$
$$H_n = \{\langle \ulcorner M \urcorner, x\rangle \mid M\text{, a machine with oracle for } H_{n-1}\text{, halts on } x\}$$

Each of these H_k-oracle machines cannot decide H_k or higher. And, $H_k \in \Sigma_k^0$.

Conclusions

This conclues our study of computability theory. Next week, we'll start on complexity theory.

If there's time, I'll talk about some other interesting topics.

Ask me things!