Computability

Turing	1936	Undecidability of Halting Problem, universal TM
Gödel	1931	Incompleteness of first-order arithmetic
Church	1936	undecidability of first-order predicate logic

Textbook: Christos H. Papadimitriou, Computational Complexity, 1995

Computable Functions

Idea: Functions $\mathcal{Z} \to \mathcal{Z}$ that can be computed by simple programs

Fix: $\bot \notin \mathcal{Z}$, $X_0 \in Loc$

$\mathcal{Z}_\bot \stackrel{def}{=} \mathcal{Z} \cup \{\bot\}$ $\qquad \boxed{\mathcal{Z} \to \mathcal{Z}_\bot \;\simeq\; \mathcal{Z} \to \mathcal{Z}}$

$\sigma_x \stackrel{def}{=} (\lambda X \in Loc.\, 0)[X_0 := x]$ for $x \in \mathcal{Z}$

$\mathcal{F} \in PS \to \mathcal{Z} \to \mathcal{Z}_\bot$

- PS is set of all simple programs
- will consider only simple programs

$$\mathcal{F}_p\, x = \begin{cases} \sigma X_0 & \text{if } (G_{x,\sigma}) \in \mathcal{R}_p \\ \bot & \text{otherwise} \end{cases}$$

$CF \stackrel{def}{=} \{ \mathcal{F}_p \mid p \in PS \}$ computable functions

$TCF \stackrel{def}{=} CF \cap (\mathcal{Z} \to \mathcal{Z})$ total computable functions

Executive Summary

- $CF \subseteq \mathcal{Z} \to \mathcal{Z}_\bot$ is universal since $\mathbb{N}$ is a universal data structure
- Formulas and programs can be represented as numbers
- Can write programs whose inputs are programs
- Program properties concerning $\mathcal{R}_p$ cannot be decided by programs. For instance: Does p terminate for σ ?

$\mathbb{N}$ is a Universal Data Structure

Representation for X: $\#_X \in X \to \mathbb{N}$ injective

$\mathcal{Z}$: $\#_{\mathcal{Z}}\, x = \text{if } x \geq 0 \text{ then } 2x \text{ else } 1-2x$

$\mathbb{N} \times \mathbb{N}$: $\#(x,y) = 2^x \cdot 3^y$ (uniqueness of prime factorization)

$\mathbb{N}^*$: $\#(x_1, \ldots, x_n) = 2^n \cdot 3^{x_1} \cdot 5^{x_2} \cdot \ldots \cdot p_{n+1}^{x_n}$ ($(n+1)$-th prime number)

$Tree\,\mathcal{Z} = \mathcal{Z} \times (Tree\,\mathcal{Z})^*$: above plus recursion (trees over $\mathcal{Z}$)

programs and formulas can be represented as trees over $\mathcal{Z}$

Universal Program

$$\exists u \in PS\ \forall p \in PS\ \forall x \in \mathbb{Z}: \quad \mathcal{F} p x = \mathcal{F} u (\#(\#p, \#x))$$

- u is an interpreter for simple programs
- one program suffices for all computable functions

$$\exists u \in PS\ \forall p \in PS\ \forall x \in \mathbb{Z}:$$
$$\mathcal{F} p x = \bot \iff \forall h \in \mathbb{N}: \mathcal{F} u (\#(\#p, \#x, h)) = 0$$
$$\wedge\ \forall h \in \mathbb{N}\ \forall z \in \mathbb{Z}: \mathcal{F} u (\#(\#p, \#x, h)) = z + 1 \Rightarrow$$
$$\mathcal{F} p x = \text{if } z > 0 \text{ then } z-1 \text{ else } z$$

max number of execution steps

controlled universal program

Turing's Law

$$\vdash \neg \exists x \forall y.\ \overline{f x y} \leftrightarrow f y y$$

see 9-7

$f \in PS \to PS \to \mathbb{B}$
$f p q = (p(\#q) = \bot)$

$$\neg \exists p \in PS\ \forall q \in PS: \quad p(\#q) \neq \bot \iff q(\#q) = \bot$$

Often referred to as halting problem
But the use of $\bot$ is not essential (use 0 instead)

$$\lambda x \in \mathbb{Z}.\ \text{if } \exists p \in PS: x = \#p \wedge p(\#p) = 0 \text{ then } 1 \text{ else } 0$$

not computable

Decidability

decidable ≙ char. function computable

$A \subseteq \mathbb{Z}$ decidable: $(\lambda z \in \mathbb{Z}.\ z \in A) \in CF$

$A \subseteq \mathbb{Z}$ semi-decidable: $\exists f \in CF\ \forall z \in \mathbb{Z}: z \in A \iff f z \neq \bot$

$$\{\#p \mid p \in PS \wedge \mathcal{F} p (\#p) \neq \bot\}$$
is semi-decidable but not decidable

Proof. Undecidability follows with Turing's Law
Semi-decidability follows with universal program:
if $x_0 \in \#PS$ then $x_0 := \#(x_0, x_0)$; u else $\bot$ is semi-decider □

$$A \subseteq \mathbb{Z}: \quad A \text{ decidable} \iff A \text{ and } \mathbb{Z}-A \text{ semi-decidable}$$

Proof

"⇒" easy since $\bot$ easy to produce

"⇐" run semi-deciders for A and $\mathbb{Z}-A$ for a certain number of steps that is doubled until one of them terminates. Made possible by controlled universal program.

Reductions

$A \subseteq \mathbb{Z}$ reduces to $B \subseteq \mathbb{Z}$: $\exists f \in TCF\ \forall z \in \mathbb{Z}:\ z \in A \Leftrightarrow fz \in B$ (reduction)

A at most as difficult as B

Reducibility is reflexive and transitive

If A reduces to B, then:

1) B decidable $\Rightarrow$ A decidable
2) B semi-decidable $\Rightarrow$ A semi-decidable
3) A undecidable $\Rightarrow$ B undecidable
4) A not semi-decidable $\Rightarrow$ B not semi-decidable

Reductions are a tool for propagating (un)decidability results

$\{\#p \mid p \in PS \wedge \mathcal{F}p0 = \bot\}$ not semi-decidable

Proof. It suffices to show that

$\{\#p \mid p \in PS \wedge \mathcal{F}p(\#p) = \bot\}$ reduces to $\{\#p \mid p \in PS \wedge \mathcal{F}p0 = \bot\}$.

For all $p \in PS$: $\mathcal{F}p(\#p) = \bot \Leftrightarrow \mathcal{F}(\mathcal{V}_0 := \#p;\ p)0 = \bot$

Hence $\lambda z \in \mathbb{Z}.\ \text{if } z \in \#PS \text{ then } \#(\mathcal{V}_0 := z;\ \#^{-1}z) \text{ else } -1$

is a reduction as required. □

Problems

Problem: $(\mathcal{V}, \#, M)$ (base set, representation) where $M \subseteq \mathcal{V}$ and $\# \in \mathcal{V} \to \mathbb{N}$ injective

Example: $(PS, \#, \{p \in PS \mid \mathcal{F}p0 = \bot\})$

$\#M \overset{def}{=} \{\#x \mid x \in M\}$

$(\mathcal{V}, \#, M)$ decidable: $\#M$ decidable

$(\mathcal{V}, \#, M)$ semi-decidable: $\#M$ semi-decidable

$(\mathcal{V}, \#, M)$ reduces to $(\mathcal{V}', \#', M')$: $\#M$ reduces to $\#M'$

Reductions: Problem Level → Code Level

Let $(\mathcal{V}, \#, M)$ and $(\mathcal{V}', \#', M')$ problems

$g \in \mathcal{V} \to \mathcal{V}'$ such that $\forall x \in \mathcal{V}:\ x \in M \Leftrightarrow gx \in M'$

$f = \lambda z \in \mathbb{Z}.\ \text{if } z \in \#\mathcal{V} \text{ then } \#'(g(\#^{-1}z)) \text{ else } -1$

Then $f \in \mathbb{Z} \to \mathbb{Z}$ and $\forall z \in \mathbb{Z}:\ z \in \#M \Leftrightarrow fz \in \#M'$

and (f computable $\Rightarrow$ $(\mathcal{V}, \#, M)$ reduces to $(\mathcal{V}', \#', M')$)

g: reduction at problem level (intuitive notion of computability)

f: reduction at code level (formal notion of computability)

Rule of thumb: g intuitively computable $\Rightarrow$ f computable

$T \overset{\text{def}}{=} \{p \in PS \mid \forall z \in \mathbb{Z}: \mathcal{F}p\,z \neq \bot\}$

Neither T nor $PS-T$ is semi-decidable

Proof. Reduction $\{p \in PS \mid \mathcal{F}p\,0 = \bot\}$ to $PS-T$:

$\mathcal{F}p\,0 = \bot \iff \exists z \in \mathbb{Z}: \mathcal{F}(X_0 := 0; p)\,z = \bot \iff p \notin T$

Reduction $\{p \in PS \mid \mathcal{F}p\,0 = \bot\}$ to T:

$\mathcal{F}p\,0 = \bot \iff \forall z \in \mathbb{Z}: \mathcal{F}u\,(\#(\#p, 0, |z|)) = 0$

(u: controlled universal program)

$\iff \forall z \in \mathbb{Z}: \mathcal{F}(X_0 := \#(\#p, 0, |X_0|); u;$
if $X_0 = 0$ then skip
else while true do skip$)\,z \neq \bot$ □

Rice's Theorem 1953

Semantic program properties are undecidable

$\forall F \subseteq CF: \emptyset \subset F \subset CF \Rightarrow \{p \in PS \mid \mathcal{F}p \in F\}$ undecidable

follows from

If $(\lambda x \in \mathbb{Z}.\bot) \in F \subset CF$,
then $\{p \in PS \mid \mathcal{F}p \in F\}$ not semi-decidable

Proof. Let $F \subset CF$ such that $(\lambda x \in \mathbb{Z}.\bot) \in F$.

It suffices to show that $\{p \mid \mathcal{F}p(\#p) = \bot\}$ reduces to $\{p \mid \mathcal{F}p \in F\}$.

Let $q \in PS$ such that $\mathcal{F}q \notin F$. Then for all $p \in PS$:

$\mathcal{F}p(\#p) = \bot$

$\iff$ (if $\mathcal{F}p(\#p) = \bot$ then $\lambda x.\bot$ else $\mathcal{F}q$) $\in F$

$\iff \mathcal{F}(X_1 := X_0;\ X_0 := \#(\#p, \#p);\ u;\ X_0 := X_1;\ q) \in F$

(X_1: location not used otherwise; u: universal program) □

Undecidability Result for S

∃A finite and first-order: $\{e \mid A \vdash e$ and e first-order$\}$ undecidable

Proof idea: implement interpreter for simple programs

$0 : \mathbb{N}$; $s : \mathbb{N} \to \mathbb{N}$;

$+, \cdot : \mathbb{N} \to \mathbb{N} \to \mathbb{N}$ and $\leqslant : \mathbb{N} \to \mathbb{N} \to \mathbb{B}$ can be defined by equations (functional programming)

other operations as well

First such result obtained by Church 1937 (undecidability of first-order predicate logic)

First-order Arithmetic

signature

$\mathbb{Z}$ $\quad 1, -1 : \mathbb{Z}$; $\quad +, \cdot : \mathbb{Z} \to \mathbb{Z} \to \mathbb{Z}$, $\quad \leq : \mathbb{Z} \to \mathbb{Z} \to \mathbb{B}$

$\mathbb{B}$ $\quad \neg : \mathbb{B} \to \mathbb{B}$; $\quad \wedge, \vee : \mathbb{B} \to \mathbb{B} \to \mathbb{B}$; $\quad \forall, \exists : (\mathbb{Z} \to \mathbb{B}) \to \mathbb{B}$

FOA: set of all terms t such that:

- $\forall t = \mathbb{B}$
- abstractions appear only as arguments of $\forall, \exists$
- all free variables have type $\mathbb{Z}$

Gödel 1931 $\quad \boxed{\{\varphi \in FOA \mid \mathbb{Z} \models \varphi\} \text{ not semi-decidable}}$

Gödel's Incompleteness Theorem

Complete deduction for S doesn't exist

$\exists$ finite set A of equations (Peano axioms)
such that $\forall \varphi \in FOA: \quad \mathbb{Z} \models \varphi \iff A \vdash \varphi = \top$

Requirement for deduction systems:

A semi-decidable $\Rightarrow$ $\{e \mid A \vdash e\}$ semi-decidable

Presburger 1929

$\boxed{\{\varphi \in FOA \mid \mathbb{Z} \models \Lambda,\ \Lambda \text{ doesn't contain } \cdot\} \text{ is decidable}}$

Proof uses technique of quantifier elimination

Cook's Theorem (1978)

Relative Completeness
Expressiveness

Assume: variables of type $\mathbb{Z}$ are locations
Then every $t \in FOA$ describes a set $\mathcal{D}t$ of program states

$\boxed{\forall p \in PR \ \forall \varphi \in FOA \ \exists t \in FOA: \quad \mathcal{D}t = [p](\mathcal{D}\varphi)}$

There is an algorithm that computes t from p and φ
Proof technical but not difficult

Gödel follows from Cook

Suffices to show: $\{p \mid \mathcal{F}p\,0 = \bot\}$ reducible to $\{\sigma \in \mathrm{FOA} \mid \mathcal{U} \models \sigma\}$

$\mathcal{F}p\,0 = \bot$

$\iff \sigma_0 \in [p]\emptyset$

$\iff \sigma_0 \in \mathcal{D}(s_p)$ Cook

$\iff \mathcal{U} \models \sigma_0 \to \Delta_p$ suffices to consider locations in p

Yuri Matiyasevich 1970 (Hilbert's 10th Problem)

$a \in AE = -^{\frown} \mid \neg \mid x \mid a+a \mid a \cdot a$

$\{a \in AE \mid \exists \sigma : \mathcal{D}a\sigma = 0\}$ not decidable

Gödel's Incompleteness Theorem is a consequence:

$\Rightarrow \{a \in AE \mid \forall \sigma : \mathcal{D}a\sigma \neq 0\}$ not semi-decidable

$\Rightarrow \{a \in AE \mid \mathcal{U} \models \neg(a = 0)\}$ not semi-decidable

$\Rightarrow \{\sigma \in \mathrm{FOA} \mid \mathcal{U} \models \sigma\}$ not semi-decidable