1. Reading: D. Kozen Automata and Computability, lecture 33

2. The main message of this lecture:

One of the main methods of establishing undecidability is reduction. It is about time to give its systematic account.

When proving undecidability for a given CFG $G$ whether or not $L(G) = \Sigma^*$ we have described an algorithmic procedure $R$ that for each TM $M$ and its input $x$ builds a CFG $G = R(M,x)$ such that $L(G) = \sim V L C O M P S(M,x)$. Having done that we reduced the halting set

$$ HP = \{ M \# x \mid M \text{ halts on } x \}$$

to the set of context free grammars

$$ T = \{ G \mid G \text{ is a CFG and } G \text{ accepts all strings in its alphabet} \}. $$

Indeed, we have established that

$$ M \# x \in HP \iff R(M,x) \in T, $$

and concluded that any decision algorithm for $T$ would immediately yield a decision procedure for $HP$: given $M,x$ build a CFG $R(M,x)$ and check $R(M,x) \in T$. Since there $HP$ is not recursive (undecidable) so is $T$.

Another example of reduction was given by the Gödel’s incompleteness theorem stating that the set of all true sentences of arithmetic $Th(N)$ is not r.e. There given $M,x$ we built an arithmetical sentence $\gamma = R(M,x)$ such that

$$ M \text{ does not halt on } x \iff \gamma \in Th(N). $$

Again, we performed a reduction $R$ of $\sim HP$ to the desired set $Th(N)$:

$$ M \# x \in \sim HP \iff R(M,x) \in Th(N) $$

and concluded that $Th(N)$ is not r.e., since otherwise we would have a positive test for $\sim HP$: transform $M,x$ into an arithmetical sentence $R(M,x)$ and check $R(M,x) \in Th(N)$.

There is a general definition of reducibility behind those examples.

**Definition 36.1.** Given sets $A \in \Sigma^*$ and $B \in \Delta^*$, a reduction of $A$ to $B$ is a total computable function $\sigma : \Sigma^* \rightarrow \Delta^*$ such that for all $x \in \Sigma^*$,

$$ x \in A \iff \sigma(x) \in B. $$

Notation: $A \leq_m B$.

**Theorem 36.2.** If $A \leq_m B$ and $B$ is recursive (r.e.), then so is $A$.

**Proof** is a straightforward repetition of the above reasoning.
Corollary 36.3. If $A \leq_m B$ and $A$ is not recursive (not r.e.), then so is $B$.

Example 36.4. The set $FIN = \{ M \mid L(M) \text{ is finite} \}$ is not r.e. We establish that by reducing $\sim HP$ to $FIN$, i.e. by showing that $\sim HP \leq_m FIN$. We have to describe a computable procedure that given $M, x$ produces a TM $M'$ such that $M$ does not halt on $x$ iff $L(M')$ is finite (note that both $M$ and $x$ should be hard-wired in the finite control for $M'$). $M'$ works as follows: given input $y$ $M'$ erases $y$ and writes $x$ on the tape, runs $M$ on $x$, accepts if $M$ halts on $x$. Obviously, if $M$ does not halt on $x$, then $L(M') = \emptyset$, otherwise $L(M') = \Sigma^*$. Therefore,

$$M \text{ does not halt on } x \iff L(M') \text{ is finite.}$$

Example 36.5. The complement of $FIN$ is also not r.e. Now we have to build another reduction $R$ that given $M, x$ produces $M''$ (with both $M$ and $x$ hard-wired in) such that $M$ does not halt on $x$ iff $L(M'')$ is infinite. Given input $y$ the machine $M''$ simulates $|y|$ steps of $M$ on $x$, accepts if $M$ has not halted within that time, otherwise rejects. Let $M$ halts on $x$ after $n$ of steps. Then $M''$ rejects on all inputs $y$ longer than $n - 1$. In this case $L(M'') = \{ y \in \Sigma^* \mid |y| < n \}$ and therefore is finite. If $M$ does not halt on $x$, then $M''$ accepts on all inputs and therefore $L(M'') = \Sigma^*$. We have established that

$$M \not\in x \in \sim HP \iff R(M \not\in x) \in \sim FIN.$$ 

Therefore, $\sim FIN$ is not r.e.

Example 36.6. Every r.e. set is $m$-reducible to the halting problem (i.e. $A \leq_m HP$ for any r.e. set $A$). This fact can be interpreted as saying that the halting problem is the most difficult semidecidable problem. Proof: let $A$ be any r.e. set. Define a computable function $f(x, y)$ by

$$f(x, y) = \begin{cases} 1 & \text{if } x \in A \\ \text{undefined} & \text{if } x \notin A \end{cases}$$

The parameter theorem for the universal function $U$ gives a total computable function $\varphi$ such that $f(x, y) \equiv U(\varphi(x), y)$. It is clear from the definition of $f$ above that

$$x \in A \iff f(x, 1) \text{ is defined} \iff U(\varphi(x), 1) \text{ is defined} \iff M_{\varphi(x)}(1) \in HP.$$