Summary on Lecture 19, May 18th, 2015

Turing Machines: Busy Beaver Problem.

Let us consider all possible binary Turing Machines which have n states $\{0, 1 \ldots, n-1\}$, and n is a halting state. Here we assume that the language is $\Sigma = \{0, 1\}$ and that a Turing Machine always halts when it starts at the blank tape

$$0\overset{\downarrow}{0}00000.$$

We denote by $\mathbf{Turing}_n$ the set of binary Turing Machines with n states halts when it starts at the blank tape. Then such a machine has $2n$ instructions of the type aDs, where $a \in \{0, 1\}$, $D \in \{R, L\}$, and $s \in \{0, 1, \ldots, n-1, n\}$ (here we include the halting state n). The number of choices for any particular instructions is $4(n+1)$. Since there are $2n$ possible instructions, we obtain:

$$|\mathbf{Turing}_n| = (4(n+1))^{2n}.$$

Not all of them halt, but clearly there are many binary Turing Machines which which will halt.[1]

We denote by $\mathbf{Turing}_n^h$ the set of all binary Turing Machines from $\mathbf{Turing}_n$ which halt. Clearly[2]

$$|\mathbf{Turing}_n^h| < |\mathbf{Turing}_n|.$$

Now, for each machine $M \in \mathbf{Turing}_n^h$, we denote by $b(M)$ the number of steps before it will halt. Then we take a maximum:

$$\beta(n) = \max_{M \in \mathbf{Turing}_n^h} b(M).$$

We obtain a function $\beta : \mathbf{Z}_+ \to \mathbf{Z}_+$, where $n \mapsto \beta(n)$.

Lemma 1. *The function $\beta : \mathbf{Z}_+ \to \mathbf{Z}_+$ is increasing.*

Proof. We should show that $\beta(n+1) > \beta(n)$. Indeed, let $M \in \mathbf{Turing}_n^h$ be a Turing Machine such that $\beta(n) = b(M)$, i.e., M halts in $\beta(n)$ steps. We use M to construct a Turing Machine $M' \in \mathbf{Turing}_{n+1}^h$ by adding one more line of new instructions:

	0	1
n	1L(n+1)	1L(n+1)

Here $(n+1)$ means the halting state. Clearly $b(M') > \beta(n)$. It means that $\beta(n+1) > \beta(n)$. □

Busy Beaver Problem: *Is it possible to compile a computing program which will give the value of $\beta(n)$ for every positive integer n?*

Theorem. *There is no algorithm which will compute the value of $\beta(n)$ for every positive integer n.*

What do we mean here? We do not mean that we cannot compute $\beta(n)$ for any particular n. What we really mean that **there is no one computational procedure which will produce $\beta(n)$ for every positive integer n.**

Proof. We assume that there exists an algorithm which computes $\beta(n)$ for every positive integer n. Then there exists a Turing Machine M_β which computes the the value of $\beta(n)$ for every positive integer n, i.e. M_β performs the operation:

$$0\underbrace{\overset{\downarrow}{1}1 \;\ldots\; 11}_{n}0 \;\mapsto\; 0\underbrace{\overset{\downarrow}{1}1 \;\ldots\; 11}_{\beta(n)}0$$

We assume that M_β has k states, i.e. $M_\beta \in \mathbf{Turing}_k^h$. We would like to use the Turing Machines M_2 from Example 2 which computes the function $f_2(n) = n+1$ and the Turing Machine M_5 from Example 5 which computes the function $f_5(n) = 2n$. By construction, $M_2 \in \mathbf{Turing}_2^h$ and $M_5 \in \mathbf{Turing}_9^h$.

[1] Prove that for any n there are binary Turing Machines which halt.

[2] Prove that for any n there are binary Turing Machines which do not halt.

Now we construct the Turing Machine $S_i = M_2 M_5^i M_\beta$ for each positive integer $i \geq 1$. The Turing Machine S_i performs the following operations:

$$0\,\underbrace{\overset{\downarrow}{1}1\ \ldots\ 11}_{n}\,0 \ \overset{M_2}{\mapsto}\ 0\,\underbrace{\overset{\downarrow}{1}1\ \ldots\ 11}_{n+1}\,0 \ \overset{M_5^i}{\mapsto}\ 0\,\underbrace{\overset{\downarrow}{1}1\ \ldots\ 11}_{2^i(n+1)}\,0 \overset{M_\beta}{\mapsto}\ 0\,\underbrace{\overset{\downarrow}{1}1\ \ldots\ 11}_{\beta(2^i(n+1))}\,0$$

The Turing Machine S_i will halt after at least $\beta(2^i)$ steps. Indeed, if starts with a blank tape, it need to put $\beta(2^i)$ 1's, and it will take at least $\beta(2^i)$ steps.

On the other hand, S_i has $2+9i+k$ states, and we obtain that

$$\beta(2^i) \leq \beta(2+9i+k)$$

for every i. However, for given k, there exists i such that $2^i > 2+9i+k$.[3] Let i_0 be such that $2^{i_0} > 2+9i_0+k$, then $\beta(2^{i_0}) > \beta(2+9i_0+k)$ by Lemma 1. We obtain a contradiction. Thus the Turing Machine M_β does not exist. □

[3] Use calculus to prove this.