Lecture 3. Time/Space Hierarchy Theorems

Now we are ready to discuss Time (resp., Space) Hierarchy Theorem, one of the central theorems of computational complexity theory. That is, the following two theorems. Below we use the "small ω" notation, which is defined as follows: for any functions $f, g : \mathcal{Z}^+ \to \mathcal{Z}^+$, we say $g(n) = \omega(f(n))$ if we have

$$\forall c > 0, \exists n_c, \forall n > n_c \; [\; g(n) > cf(n) \;].$$

Theorem 3.1 (Time Hierarchy Theorem)
For any functions t_1, t_2: $\mathcal{Z}^+ \to \mathcal{Z}^+$, if $t_2(\ell) = \omega(t_1(\ell) \log t_1(\ell))$, then we have

$$\mathrm{TIME}(\, t_1(\ell)\,) \subsetneq \mathrm{TIME}(\, t_2(\ell)\,).$$

Remark: Precisely speaking, we require t_1 to be "time constructible." Intuitively, a time constructible function is a natural function for a time bound. We omit explaining this notion; see [1:Def. 9.8].

Theorem 3.2 (Space Hierarchy Theorem)
For any functions s_1, s_2: $\mathcal{Z}^+ \to \mathcal{Z}^+$, if $s_2(\ell) = \omega(s_1(\ell))$, then we have

$$\mathrm{SPACE}(\, s_1(\ell)\,) \subsetneq \mathrm{SPACE}(\, s_2(\ell)\,).$$

Remark: Precisely speaking, we require s_1 to be "space constructible." Intuitively, a space constructible function is a natural function for a space bound. We omit explaining this notion; see [1:Def. 9.1].

In the following, we prove Theorem 3.1 for specific timve bound functions for t_1 and t_2; $t_1(\ell) = \ell^2$ and $t_2(\ell) = \ell^5$.

3.1 A universal Turing machine

As a key tool for proving We first introduce the notion of "universal Turing machine." Intuitively, a universal Turing machine is an "interpreter" of Turing machines. Thus, for defining a universal Turing machine, we need to fix our programming language for Turing machines. Here we consider very basic one; it is nothing but the binary encoding of a given Turing machine.

Consider any Turing machine $\mathtt{M} = (Q, \Sigma, \Gamma, \delta, \mathsf{q}_0, \mathsf{q}_{\text{accept}}, \mathsf{q}_{reject})$. We can simply use nonnegative integers (encoded by the binary representation) for members of Q and Γ; thus, we only need to specify $q = |Q|$ and $g = |\Gamma|$. We may assume that $\Sigma = \{0, 1\}$, q_0 is the first state, i.e., state 0, and $\mathsf{q}_{\text{accept}}$ and q_{reject} are the last two states respectively. Since δ is a function from $(\, Q - \{\mathsf{q}_{\text{accept}}, \mathsf{q}_{\text{reject}}\}\,) \times \Gamma$ to $(\, \Gamma - \{\llcorner\!\lrcorner\}\,) \times \{\mathrm{L}, \mathrm{R}\}$, we can encode it by $\{0, 1\}^*$ as illustrated in Figure 3.1. Let $\overline{\delta}$ denote this binary encoding of δ. In summary, $\mathtt{M}$ is encoded as $\langle q, g, \overline{\delta} \rangle$ in $\{0, 1\}^*$, which is referred as $\overline{\mathtt{M}}$ in the following.

The notion of "universal Turing machine" is defined as follows.

Definition 3.1 A *universal Turing machine* is a Turing machine $\mathtt{M}_{\rm eval}$ that takes $\langle\overline{\mathtt{M}}, x\rangle$, $x \in \{0,1\}^*$ as an input and simulate the execution of $\mathtt{M}$ on x.
Remark: If $\mathtt{M}(x)$ does not terminate, then so does $\mathtt{M}_{\rm eval}$ on $\langle\overline{\mathtt{M}}, x\rangle$.

Theorem 3.1 There exists a universal Turing machine $\mathtt{M}_{\rm eval}$ such that for any Turing machine $\mathtt{M}$ and for any $x \in \{0,1\}^*$, it simulates $\mathtt{M}(x)$ with the following efficiency for a constant $c_{\mathtt{M}}$ determined by $\mathtt{M}$:

$$\mathrm{time}_{\mathtt{M}_{\rm eval}}(\langle\overline{\mathtt{M}}, x\rangle) \ \leq \ c_{\mathtt{M}}\mathrm{time}_{\mathtt{M}}(x).$$

<u>A time bounded universal Turing machine</u>

For proving Time Hierarchy Theorem, we need an Turing machine interpreter that can stop the computation when the number of moves exceeds a given time bound.

Theorem 3.2 There exists a Turing machine $\mathtt{M}_{\rm eval\_in\_time}$ such that for any Turing machine $\mathtt{M}$, for any $x \in \{0,1\}^*$, and for any $t \geq 1$, (i) $\mathtt{M}_{\rm eval\_in\_time}(\langle\overline{\mathtt{M}}, 0^t, x\rangle)$ simulates $\mathtt{M}(x)$ up to t moves (and rejects the input if $\mathtt{M}(x)$ does not terminate in t moves), and (ii) it has the following efficiency for a constant $c_{\mathtt{M}}$ determined by $\mathtt{M}$:

$$\mathrm{time}_{\mathtt{M}_{\rm eval\_in\_time}}(\langle\overline{\mathtt{M}}, 0^t, x\rangle) \ \leq \ c_{\mathtt{M}} \min\{\, \mathrm{time}_{\mathtt{M}}(x), t\,\} \log t.$$

3.2 A proof by the diagonalization

Clearly, we have $\mathrm{TIME}(\ell^2) \subseteq \mathrm{TIME}(\ell^5)$. For proving $\mathrm{TIME}(\ell^5) - \mathrm{TIME}(\ell^2) \neq \emptyset$, it is necessary and sufficient to show that there exists a problem in $\mathrm{TIME}(\ell^5)$ that is not in $\mathrm{TIME}(\ell^2)$. For defining a problem $L \notin \mathrm{TIME}(\ell^2)$, we use the diagonalization technique. We begin by proving the following fact by the diagonalization technique.

Fact There is a real number in $[0,1)$ that is not a rational number.

The key point for proving this fact is that we can "enumerate" all rational numbers under a certain linear ordering. Note that each Turing machines is given a binary string as its code; hence, we can enumerate Turing machines based on the order of binary strings. Let $\overline{\mathtt{M}}_1, \overline{\mathtt{M}}_2, \ldots$ be Turing machines indexed under this order. For enumerating $O(\ell^2)$-time Turing machines, we use a pair $(\overline{\mathtt{M}}, c)$ to denote a Turing machine that terminates in $c\ell^2$ moves for any input of length ℓ. More specifically, we enumerate them under the following ordering:

$$(\overline{\mathtt{M}}_1, 1), (\overline{\mathtt{M}}_1, 2), (\overline{\mathtt{M}}_2, 2), (\overline{\mathtt{M}}_1, 3), (\overline{\mathtt{M}}_2, 3), (\overline{\mathtt{M}}_3, 3), \ldots$$

We then define an infinite binary string for an index $(\overline{\mathtt{M}}_i, c_j)$ by the results of executing $\mathtt{M}_i$ on 0^ℓ for $c_j\ell^2$ moves (which is 0 if $\mathtt{M}_i$ does not halt within $c_j\ell^2$ moves) for all $\ell \geq 1$. By using this enumeration, we can define a problem L_2 that is not in $\mathrm{TIME}(\ell^2)$ by the diagonalization.

Unfortunately, it is not so easy to show that $L_2 \in \mathrm{TIME}(\ell^5)$ (though this may be true). We define a problem that is much simpler for showing also that it is $O(\ell^5)$-time solvable.

Here we define a problem by a set of 'yes' instances, i.e., binary strings that should be answered `1`. The following problem L_{25} is our target problem:

$$L_{25} = \{ \langle \overline{\mathtt{M}}, c, 0^n \rangle \mid c, n \geq 1, \text{ and } \mathtt{M}_{\mathrm{eval\_in\_time}}(\langle \overline{\mathtt{M}}, 0^{c\ell^2}, \langle \overline{\mathtt{M}}, c, 0^n \rangle \rangle) \neq 1, \text{where } \ell = |\langle \overline{\mathtt{M}}, c, 0^n \rangle| \}.$$

Then we can prove the following lemmas from which the theorem follows.

Lemma 3.3 $L_{25} \in \mathrm{TIME}(\ell^5)$.
Remark: In fact, the lemma is provable by using a larger bound such as $c_M\{\min\{\,\mathrm{time}_{\mathtt{M}}(x), t\,\}^2$ at Theorem 3.2.

Lemma 3.4 $L_{25} \notin \mathrm{TIME}(\ell^2)$.