CSC2401 Introduction to Computational Complexity
Student: **Jia Sheng**, jsheng@dgp.toronto.edu
Student #: 992810115
Instructor: Allan Borodin

Answer to Assignment 3

1 ANSWER TO PROBLEM 1

Proof

The claim is wrong, because otherwise, let $L \in \Sigma_{k+1}^P - \Sigma_k^P$ (L exists from the assumption of $\Sigma_k^P \subset \Sigma_{k+1}^P$), $\exists B \in \Sigma_k^P$, $L \in NP^B$ and $\exists C \in \Sigma_{k-1}^P$, $B \in NP^C \implies L \in NP^C \implies L \in \Sigma_k^P$, which is contradictory to our definition of L. So the claim in the problem is not correct. ■

2 ANSWER TO PROBLEM 2: sub-problem (1)

Proof

First we note that the following decision problem is in NP:

P1 Given a Boolean function $f_n : \{0,1\}^n \rightarrow \{0,1\}$ in the form of its truth table (length 2^n), and a natural number s_n, is f_n computable by a Boolean circuit of size at most s_n?

$P1 \in NP$ because checking whether a guessed circuit of size s_n computes a given Boolean function on n-variables takes only polynomial time wrt the length of the input (truth table, $O(2^n)$).

Next, for a given n, we can see that the search problem of the maximum circuit complexity, $MaxC(n)$, falls in NP^{NP}. We can find $MaxC(n)$ by asking a series of questions like this:

"Does there exist a string $t_1, ..., t_{2^n}$ representing the truth table of a boolean function over n-variables that requires circuit size at least s".

Given the fact $P1 \in NP$, for any given s, the above question is in NP^{NP}. Since $MaxC(n) \leq 2^n/n * (1+\epsilon)$, let s start from $s = \lceil 2^n/n * (1+\epsilon) \rceil$, we try s, $s-1$, $s-2$, ..., until we get the positive answer. The first value of $s = s^*$ that gives the positive answer will be $MaxC(n)$.

Then we shall find the boolean function f_n^* that has circuit complexity s^*. It equals to find its truth table. We begin from the empty truth table $T = \emptyset$, and append 0 to T if the answer to the question is true: "can the string $T0$ be extended to a truth table of a boolean function that requires circuit complexity at least s^*?". If the answer is false, we append 1 to T. Continue the process for 2^n steps, we can specify completely the truth table of a boolean function with circuit complexity $s^* = MacC(n)$. The overall running time of this process takes $2^{O(n)}$ time with an NP^{NP} oracle.

Now we can define $L = \cup_{n>0} L_n$, s.t. L_n is the set of strings in $(0,1)^n$ accepted by f_n^*. Clearly $L \in E^{\Sigma_2^P} \subseteq E^{PH}$, and $\forall n L_n$ requires circuit size $MaxC(n)$.

With this fact, we can prove the other two problems

(a) $ESPACE$ contains a language L such that $\forall n L_n$ requires $MaxC(n)$ circuit size. It holds because every step in the procedure described above only requires to write down the $2^{O(n)}$ values in the truth table to be checked, thus $L_n \in NSPACE(2^{O(n)})$. And since we have $NSPACE(f(n)) \subseteq DSPACE(f^2(n))$, L_n is also in $DSPACE(2^{O(n)})$. So the L defined above is contained in $ESPACE = \cup_{c>0} DSPACE(2^{cn})$.

(b) We prove the contrapositive: if $P = NP$, then $\exists L \in E$, s.t. $\forall n L_n$ requires circuit size $MaxC(n)$. Since $P = NP$, we have $PH = P \Rightarrow E^{PH} = E^P = E$. Using the fact we have proved, there is a language $L \in E$ that requires maximum circuit complexity for all L_n, given $P = NP$. ■

3 ANSWER TO PROBLEM 3

Proof

First we show that for any language $L \in ESPACE$, L is linear space transformable to $\overline{TSRE}$ for some Σ. More formally, if $L \in \{0,1\}^*$ and is accepted by a space $S(n) = 2^n$ bounded NTM M, $L \leq_l \overline{TSRE} = \{R | R$ is a squegular expression and $L(R) \neq \Sigma^*\}$ for some Σ. Here $\leq_l$ stands for linear space transformation.

Let Q stands for states in M, T for tape symbols, $b \in T$ for blank symbol. Let $\# \notin T \cup (Q \times T)$. The encoding of accepting computation of M on input $x \in \{0,1\}^*$ is as following:

$$\#c_1\#c_2\#...\#c_k\#$$

where c_i is a configuration of $M(x)$, each c_i is of length 2^n, c_1 is the initial configuration, $c_i \vdash c_{i+1}$ and c_k contains the accepting state q_a.

Now let $\Sigma = \{\#\} \cup T \cup (Q \times T)$. Given $x = x_1...x_n$, we will construct a squegular expression E of length cn for some c, such that:

If $x \notin L$, then $L(E) = \Sigma^*$, and
If $x \in L$, then $L(E) \neq \Sigma^*$.

$L(E)$ will have the above property if $L(E)$ contains the following sets of strings:

1. Strings which don't begin with: $\#(q_0, x_1)x_2...x_n b^{2^n - n}\#$.
2. Strings which don't contain the accepting state q_a.
3. strings which aren't of the form $\#c_1\#c_2\#...\#c_k\#$ with $c_i \vdash c_{i+1}$.

The details of each expressions are omitted here, which are similar to those in the proof of $\overline{TRE}$ complete of $PSPACE$. What needs to mention is that the "squaring" operation ensures that the length of E is bounded by cn, i.e. the transformation is linear space bounded, because the "squaring" operation exponentially saves space than concatenation operation otherwise would.

Thus we have known that every language in $ESPACE$ can be reduced to $\overline{TSRE}$ bounded by linear space. Now we only need to show $\overline{TSRE}$ itself is in $ESPACE$.

Let L be a language accepted by a $S_2(n) = 2^n$ machine, but not by a $S_1(n)$ machine where $\lim_{n\to\infty} \frac{S_1(n)}{S_2(n)} = 0$. From Space Hierarchy Theorem, L exists. Let Σ be that $L \leq_l \overline{TSRE}$ wrt Σ. If the $\overline{TSRE}$ is accepted by some space $S(n)$ NTM, then L is accepted by some $S(cn) + cn$ machine for some constant c. So we have

$$\lim_{n\to\infty} \frac{S(cn) + cn}{2^n} > 0$$

It implies that $S(n) > d^n$ for some constant $d > 1$ and infinitely many n. So we have $\overline{TSRE} \in ESPACE$.

So far we have finished the proof of $\overline{TSRE}$ is complete for $ESPACE$. ■