COMS 4236 Homework 2

Mengqi Zong < *mz2326@columbia.edu* >

February 28, 2012

Problem 1

a.

- $L_1 \cup L_2$
  Suppose L_1 can be recognized by nondeterministic Turing machine M_1, and L_2 can be recognized by nondeterministic Turing machine M_2. Then what we need to do here is simply simulating M_1 and M_2 at the same time by another NTM M_3 and then check their outputs to generate the final output.

  We only accept the input when M_1 and M_2 accept the input. If not, we reject the input. Note both simulations takes $\theta(f(n))$, then the time for M_3 to simulate M_1 and M_2 takes $\theta(f(n))$. And the time for comparing the two results then generating the final result takes constant time. So the total time is still $\theta(f(n))$. And $L_1 \cup L_2$ is in NTIME(f(n)).

- $L_1 \cap L_2$
  Similar to $L_1 \cap L_2$, the only difference is that we accept the input if not both M_1 and M_2 reject the input. If not, we reject the input. Obviously, the running time is still $\theta(f(n))$. And $L_1 \cap L_2$ is in NTIME(f(n))

- $L_1 \cdot L_2$
  First, we guess the length of x. Note that if we know the length of x, then we will know the length of y.

  Second, try to recognize the first length(x) cells of the input. If x is in L_1, then continue. If not, reject. If y is in L_2, then we accept the whole input. If not, we reject. In total, we just run the total time of M_1 and M_2 plus a constant. In sum, the total running time is $\theta(f(n))$. And $L_1 \cdot L_2$ is in NTIME(f(n)).

b. Basically, The Kleene star of a language L can be treated as a concatenation of L:

$$\prod_{i=1}^{m} x_i, \quad x_i \in L$$

where m is the number of strings of language L in the input.

First, we guess how many strings of L are there in the input. We do the guessing by the following procedure: guess the length of the first string, then guess the length of the second string, and so on untill the total length reaches n.

Second, we begin to recognize each of thoes strings. If all of them are in L, then we accept the whole input.

Note that the first part takes at most $\theta(n)$, the second part also takes $\theta(n)$. As a result, the total running time of L^* is $\theta(n)$, given that $L \in NTIME(n)$. So, $L^* \in NTIME(n)$.

Question 2

a. First, we can get that $TIME(2^n) \subseteq TIME(2^{2n})$ since a TM of time complexity 2^{2n} can always simulate any TM of time complexity 2^n. Second, we can see that $2^n = o(2^{2n}) = o((2^n)^2)$. From time hierarchy theorem, we can know that there is a language in $TIME(2^{2n}) - TIME(2^n)$. At last, we get $TIME(2^n) \subset TIME(2^{2n})$.

b. From the theorems that we learned on the relations between complete classes, we know that $NTIME(f(n)) \subseteq SPACE(f(n))$. In our case, we get $NTIME(n^2) \subseteq SPACE(n^2)$.

Similar to the proof of part a, we can get $SPACE(n^2) \subseteq SPACE(n^3)$. And by the space hierarchy theorem, we know that there is a language in $SPACE(n^3) - SPACE(n^2)$. As last, we get $SPCE(n^2) \subset SPACE(n^3)$, then $NTIME(n^2) \subset SPACE(n^3)$.

Question 3

a. $\{n^k : k > 0\}$

$\{n^k : k > 0\}$ are left closed.

$$\begin{aligned} p(n^k) &= \theta((n^k)^c) \\ &= \theta(n^{ck}) \end{aligned}$$

where c is a constant. In this case, for every constant c, we can always find

$$g(n) = n^{ck}$$

such that

$$\begin{aligned} p(n^k) &= \theta(n^k)^c \\ &\geq c' \cdot n^{ck} \\ &= O(g(n)) \end{aligned}$$

where c' is a constant. As we can see, $\{n^k : k > 0\}$ are closed under left composition.

b. $\{k \cdot n : k > 0\}$

$\{k \cdot n : k > 0\}$ are left closed.

$$\begin{aligned} p(k \cdot n) &= \theta((k \cdot n)^c) \\ &= \theta(n^c) \end{aligned}$$

where c is a constant. In this case, for every constant c, we can always find

$$g(n) = n^c$$

such that

$$\begin{aligned} p(k \cdot n) &= \theta(n^c) \\ &\geq c' \cdot n^c \\ &= O(g(n)) \end{aligned}$$

where c' is a constant. As we can see, $\{k \cdot n : k > 0\}$ are closed under left composition.

f. $\{\log n\}$

$\{\log n\}$ are not left closed. For example, we can never find a $g(n) \in \{\log n\}$ such that $(\log n)^{0.2} = O(g(n))$.

$\{\log n\}$ are right closed. We have

$$\begin{aligned} \log p(n) &= \log \theta(n^c) \\ &= c \log \theta(n) \end{aligned}$$

where c is a constant. In this case, let $g(n) = \log n$. Then for any constant c, we have

$$\begin{aligned} \log p(n) &= \log \theta(n^c) \\ &= c \log \theta(n) \\ &\geq c \log c' \cdot n \\ &= c \log c' + c \log n \\ &= O(g(n)) \end{aligned}$$

where c' is a constant. As we can see, $\{\log n\}$ are closed under right composition.

Question 4

We first prove that R and $\phi(R)$ has the same time complexity both with respect to deterministic time and nondeterministic time.

- For deterministic time, on one hand, we can reduce R to $\phi(R)$ using a TM with time complexity $\theta(n)$. All we need to do is to replace every letter a_i with the length k string $\phi(a_i)$. And the reduction takes $\theta(kn)$ time. Since k is a constant, then the time complexity is $\theta(n)$. On the other hand, we can reduce $\phi(R)$ to R using a TM with time complexity $\theta(n)$. All we need to do is replace every length k string $\phi(a_i)$ with a_i. And the reduction takes $\theta(n)$, which is $\theta(n)$. So R and $\phi(R)$ have the same time complexity with respect to deterministic time.

- As to nondeterministic time, since deterministic Turing machine is a special case of nondeterministic Turing machine, we can do the same reduction as we did on deterministic Turing machines. So, R and $\phi(R)$ have the same time complexity with respect to nondeterministic time.

Now, we will construct a NTM n with time complexity $\theta(f(n))$ that diagonalizes over all k-tape TMs with time complexity $\theta(f(n))$, where f(n) is polynomial.

- The input alphabet $A = \{0, 1\}$ (As we showed above, we can encode any language into $\{0, 1\}$ without changing the time complexity both with respect to deterministic time and nondeterministic time).
- As we already know that P $\subset$ NP, so any language L in P can be decided in NP time. If M is a k-tape $\theta(f(n))$ bounded TM M, then for almost all of strings $x \in A^*$, the NTM N on input $code(M)\#x$ does the opposite of what M does on the same input.
- Since P $\neq$ NP, then for L that is in NTIME(f(n)), if M can't get the output in $\theta(f(n))$, simply reject.
- $\Rightarrow L(N) \neq L(M)$ for all such M.

Question 5

a. We will try to prove that L and $pad(L, n^2)$ have the same time complexity with respect to deterministic time.

We can reduce L to $pad(L, n^2)$ in $\theta(n^2)$. For any string $w \in L$, we copy it to the other tape and count its length $|w|$. If $|w| \geq n^2$, then we do nothing. If $|w| < n^2$, then we pad the w with $n^2 - |w|$ #'s. This reduction takes $\theta(n^2)$ time.

This means that if $pad(L, n^2)$ is in $TIME(n)$, then we can decide L in $\theta(n^2)$. For any string w in L, we first reduce it into w' in $pad(L, n^2)$ using $\theta(|w|^2)$. Then we decide w' in $\theta(w')$. Since $w' \leq |w|^2$, we can decide w in $\theta(|w|^2)$. That is, L is in TIME(n^2).

We can reduce $pad(L, n^2)$ to L in $\theta(n)$. For any string in $pad(L, n^2)$, all we need to do is copy it to another tape, if we meet a #, then finish. If there's no #, then we just copy the whole string. This reduction takes $\theta(n)$ time.

This means that If L is in $TIME(n^2)$, then we can decide $pad(L, n^2)$ in $\theta(n)$. For any string w in $pad(L, n^2)$, we first reduce it into w' in L using $\theta(|w|)$. Then we decides w' using $\theta(|w'|^2)$. Since $w' \leq \sqrt{w}$, for a given string w in $pad(L, n^2)$, we can decide it in $\theta(|w|)$. That is, $pad(L, n^2)$ is in $TIME(n)$.

As a result, L and $pad(L, n^2)$ has the same time complexity with respect to deterministic time.

b. Since deterministic Turing machine is a special case of nondeterministic Turing machine, we can do the same thing here as we did in part a. The only difference is to replace "TIME" with "NTIME".

c. As we already know, for any TM with time complexity n^2, there is a NTM with time complexity n^2 that can simulate it. This is due to the fact that $TIME(f(n)) \subset NTIME(n^2)$.

Now, we will prove that any language L in $NTIME(n^2)$ can be decided using a TM with time complexity $\theta(n^2)$.

For any string w in L, we first reduce it to w' in $pad(L, n^2)$ using a NTM with time complexity $\theta(|w|^2)$. From part b, we know that we can decide the w' with a NTM with time complexity $\theta(|w'|)$. And since $TIME(n) = NTIME(n)$, we can also decide w' using a TM with time complexity $\theta(|w'|)$. Due to $w' \leq w^2$, we can decide w in $TIME(|w|^2)$. That is, for any language L in $NTIME(n^2)$, we can decide it using a TM with time complexity $\theta(n^2)$. That is, $L \in TIME(n^2)$. Then, we get $NTIME(f(n)) \subset TIME(n^2)$.

In sum, we get $TIME(f(n)) = NTIME(n^2)$.