1 Honors Theory: Solution for HW 4

1.1 QUESTION 1

Show that $DEXPT$ has complete language. ONLY SKETCH THIS PROOF (you must not use more than one page).

SOLUTION: Let $U = \{U_i\}$ a universal TM for the characteristic class $DEXPT|\Sigma$ where $\Sigma = \{0, 1, \#\}$. Assume U_i accepts in time $|i|^{"}|x|$. Consider the language L_0 comprising all string of the form

$$i\#x$$

where $i, x \in \{0,1\}^*$ and U_i accepts x. To show L_0 is complete, we need to show two claims:
(a) CLAIM $L_0 \in DEXPT$. Here is a simple exponential time algorithm to recognize L_0. On input w, we check if w has the form $i\#x$, and if so, we simulate U_i on x. This takes time

$$O(|i|^{|x|}) = O(2^{|w|}.$$

NOTE: many students got simple but crucial equation wrong! Thus $L_0 \in DEXPT$.
(b) CLAIM L_0 is $DEXPT$-hard under log-space many-one reducibility. This is easy.

1.2 QUESTION 2

Consider the many-one reducibility using transformations computed in deterministic log space and linear time. Notice that this only a subclass of the transformations denoted **Llin** in chapter 4. Show the following:
(A) For each i, show that if L is in $D\text{-}TIME\text{-}SPACE(n^i, \log n)$ and L' is reducible to L, then L' is in $D\text{-}TIME\text{-}SPACE(n^{i+1}, \log n$
(B) Show that $NTIME(n^2)$ has a complete language under this reducibility. HINT: The proof of Theorem 3 in Chapter 5 may be useful in thinking about this problem.
(C) $NTIME(n^2)$ is different from $DLOG$.

SOLUTION:
(A) This is simply the technique to show that the class of log-space transformations is closed under composition.
(B) Let $U = \{U_i\}$ be a universal acceptor obtained from the U^{RE} by clocking each U_i to run in time $(|i|n)^2$. We assume that U has 2 work tapes. We claim that U is an efficient universal acceptor for the characteristic class $NTIME(n^2)$. Clearly each $L(U_i) \in NTIME(n^2)$. Conversely, we claim that for each $L \in NTIME(n^2)$, there is a $c > 0$ such that L is accepted in time cn^2 by U_i for infinitely many i's. This is because there is an tape reduction lemma (Chapter 2) showing that each $L' \in NTIME(n^2)$ is accepted in time cn^2 ($c > 0$) by some 2-tape Turing machine. Then there is some i such that $L(U_i) = L'$ in time $c'n^2$ (for some $c' \geq c > 0$). But there are infinitely many j's such that $|j| \geq c'$, $L(U_i) = L(U_j)$ and U_i behaves exactly like U_j (in particular, accepts in time $c'n^2$).

Finally, define the language

$$L_0 := \{x\#i : U_i \text{ accepts } x\}$$

where $x, i \in \{0,1\}^*$. It is easy to see that $L_0 \in NTIME(n^2)$ because, on input w, we chech that w has the form $x\#i$ and then simulates U_i on x. This takes time $O(|ix|^2) = O(|w|^2)$. It is also easy to see that L_0 is $NTIME(n^2)$-hard.
(C) Show $NTIME(n^2) \neq DLOG$. We need a hierarchy theorem of this form: $D\text{-}TIME\text{-}SPACE(n^i, \log n)$ is properly contained in $D\text{-}TIME\text{-}SPACE(n^{i+1}, \log n)$. This is a straight forward extension of the proof of the deterministic time hierarchy theorem. (But in your solutions, I expected you to at least sketch the arguments, but certainly not take it for granted.) This implies $DLOG$ is the limit of the following Now, $DLOG$ can be written $\cup_{i \geq 1} DTIME(n^i, \log n)$. Let L_0 be complete for $NTIME(n^2)$. If $NTIME(n^2) = DLOG$, then $L_0 \in D\text{-}TIME\text{-}SPACE(n^i, \log n)$ for some $i \geq 1$. By part (A), every language in $DLOG$ is contained in $D\text{-}TIME\text{-}SPACE(n^{i+1}, \log n)$. THis contradicts the hierarchy theorem just stated.

1.3 QUESTION 3

A *alternating finite automata* (afa) is an alternating machine with no work tapes. Furthermore, its input tape is one-way and the machine halts upon reading the first blank symbol after the input.
(A) Show that we can assume that the afa moves its input head on reading each input symbol, and that it halts upon reading the first blank symbol after the input.
(B) Show that the language accepted by an afa is regular language.

SOLUTION:
(A) Basically, the first part asks you so show how to eliminate those moves in which the input head does not move. This may appear tricky because if you look at the computation tree, there may be infinite paths in which the input

head does not move. You may know that a similar problem arise in converting non-deterministic finite automata to deterministic ones.

Although we normally think of a alternating machine as having two kinds of basis functions only (AND and OR), it is clear that to merge several (possibly infinitely many) moves than one move, we need to allow more general Boolean functions which may have arity greater than 2. In fact, I will allow you to attach a *generalized Boolean function* of the form

$$f : \{0, 1, \perp\}^k \to \{0, 1, \perp\}$$

to any state q. The arity of this state is k.

The key idea is to define a generalized Boolean function $f_{q,a}(x_1, \ldots, x_k)$ to each state q and each input symbol a: this function is the least fixed point solution for a finite state diagram corresponding to the ϵ-moves, and whose arguments $x_1, \ldots, x_k$ corresponds to the non-ϵ-moves out of this finite state diagram. For each assignment of values $a_i \in \{0, 1, \perp\}$ to x_i ($i = 0, 1, \ldots, k$), we can define the least fixed point value for the states of this diagram. Then $f_{q,a}(a_0, \ldots, a_k)$ is just the value of the starting state q in this diagram. NOTE: this least fixed point solution is similar to what we do in space-bounded choice computations.

Another technical problem is this: our definition of generalized afa insists that we attach a function to each state q. We can overcome this restriction as follows: if M causes a non-epsilon transition to state q and the previous symbol read is a, we will enter state designated $[q, a]$. The function attached to $[q, a]$ is $f_{q,a}$.

(B)