Assignment 1

Problem 1. *In the following we will prove that any one-tape TM deciding the language* PAL *where* $\mathsf{PAL} = \{w \in \{0,1\}^* : w \text{ is a palindrome }\}$ *requires time* $\Omega(n^2)$.

The proof will take place over several steps. First we define the notion of a **crossing sequence** $S_w(i)$ for input w denoting the behavior of the tape head as it crosses between tape cells i and $i+1$. Such a sequence has the form $(q_{k_1}, D_1), (q_{k_2}, D_2), ...$ where q_{k_i} is a state and $D_i \in \{R, L\}$.

1. Show that for any two strings $x = x_1x_2$ and $y = y_1y_2$ such that $|x_1| = |y_1| = i$ and $S_x(i-1) = S_y(i-1)$, if a TM M accepts x and y, the it also accepts the strings x_1y_2 and x_2y_1.

   Proof. Consider the execution of the TM M on the input x_1y_2. It runs just as it would on input x until the first element of $S_x(i-1)$. Since $S_x(i-1) = S_y(i-1)$, if M crosses over to the cell of y_2, M runs just as it would on input y. Every time the tape head crosses between cells i and $i+1$, since the crossing sequences are identical, it is as if M was running on input x or y. Eventually M will ending in the terminating state of x or y so M will accept. The same is true for input y_1x_2. □

2. Let M be any one-tape TM deciding PAL and x, y be two distinct strings of length n each. If $X = x0^nx^R$ and $Y = y0^ny^R$ where x^R is the reverse of x. Show that for every $n < i < 2n$, it must be the case that $S_X(i) \neq S_Y(i)$.

   Proof. Suppose for a contradiction that for some $i \in (n, 2n)$, $S_X(i) \neq S_Y(i)$. Since X is a palindrome, X is accepted by M. Let $X = x_1x_2$ and $Y = y_1y_2$ where $|x_1| = |y_1| = i$. By the previous part we know that x_1y_2 is also accepted by M. However, since x and y is distinct, x_1y_2 is not a palindrome. This contradicts the fact that M decides PAL. □

3. Show that there exists a constant $\epsilon > 0$ such that for every $n < i < 2n$, the number of strings of the form $x0^nx^R$ with $|x| = n$ that have $|S_X(i)| < \epsilon n$ is less than $2^{n/2}$.

   Proof. To choose ϵ we first need a handle on the moving parts of $S_X(i)$. Let there be k states in M and let $p = \log k$. q represents the number of bits used to represent each state. The maximum number of bits needed to express a crossing sequences of length less than ϵn is $(p+1)$. Thus the total number of such crossing sequences is $2^{\epsilon(p+1)n}$ (each bit can be 0 or 1). By choosing $\epsilon = \frac{1}{3(p+1)}$, for sufficiently large n, $2^{\epsilon(p+1)n} < 2^{n/2}$. □

4. Argue that there will always exists a string $X = x0^nx^R$ for large enough n such that $|S_X(i)| \geq \epsilon n$ for every $n < i < 2n$.

   Proof. From the previous part we know that the number of palindromes X which have $|S_X(i)| < \epsilon n$ is less than $2^{n/2}$. □

Problem 2. *Show that the language* $L = \{0\}$ *is complete for* P*, under polytime reduction.*

Proof. L is certainly in P. Consider any language $L' \in \mathsf{P}$. Then there exists a TM M which solves L' in polynomial time. For your reduction simply simulate the input x to L' on M. If M accepts, output 0; otherwise if M rejects, output 1. Thus $w \in L'$ if and only if the the reduction is in L. □

Problem 3. *Show that if* $\mathsf{P} = \mathsf{NP}$*, then* $\mathsf{EXP} = \mathsf{NEXP}$.

Proof. The trick here is to pad liberally. Since $\mathsf{EXP} \subseteq \mathsf{NEXP}$, to show equality all we need to do is show $\mathsf{NEXP} \subseteq \mathsf{EXP}$. To this end, choose a language $L \in \mathsf{NEXP}$ which can be decided in time 2^{n^c}. Let the $L' \in \mathsf{NP}$ be the language $L_p = \{x01^{2^{n^c}} : x \in L\}$. On input x of L with $|x| = n$ construct the padded input $x_p = x01^{2^{n^c}}$ in exponential time. Since $\mathsf{P} = \mathsf{NP}$, there exists a DTM M_p which decides L_p. M_p executes in polynomial time on x_p. By unpadding and interpreting the result, we get an exponential algorithm for L. Thus $\mathsf{NEXP} \subseteq \mathsf{EXP}$. □

Problem 4. *Show that* $\mathsf{LINSPACE} \neq \mathsf{P}$ *where* $\mathsf{LINSPACE} = \mathsf{Space}(n)$.

Proof. By contradiction of space hierarchy. In particular we will show that if $\mathsf{LINSPACE} = \mathsf{P}$ then $\mathsf{PSPACE} = \mathsf{P}$. Consider a language $L \in \mathsf{PSPACE}$. Then L has a TM M which uses at most space $O(n^c)$ on input x of size n. Again by padding, we obtain the language $L_p = \{x01^{|x|^c} : x \in L\}$ which can be decided in linear space and thus polynomial time. Thus $L \in \mathsf{P}$ by the following reduction: construct $x_p = x01^{|x|^c}$ in polynomial time. Run x_p on the polytime algorithm for L_p to get a result. Since $\mathsf{P} = \mathsf{PSPACE}$, $\mathsf{PSPACE} = \mathsf{LINSPACE}$ contradicting space hierarchy. □

Problem 5. *Let t be the alloted time. Can we use a k-tape TM in the proof of* $\mathsf{Time}(t) \subsetneq \mathsf{Size}(t^2)$*? How can you get a simulating circuit of size $O(t^2)$?*

Proof. Yes! Recall the proof of $\mathsf{Time}(t) \subsetneq \mathsf{Size}(t^2)$. We had a table were each row consisted of the tape configuration at each time step. By piping together three consecutive cells on the ith row we could determine the state of a cell on the $i+1$th row. This was the structure we used to build our circuit. We can do something similar with k-tapes. Observe that turning the k-tape TM into a one tape TM is too slow, the conversion would take $O(n^2)$ thus the number of configurations would balloon to $O(n^4)$.
Let us instead change our k-tape TM into a t^2 circuit in one go. Just as before we lay out our configuration, k times larger since the configuration of each tape must be recorded. For a cell in the $i+1$th row, we still three consecutive cells from the row i, but now we also need the cells of all other tapes in row i as well. Even though this looks bad, we only need to collected all cells from each tape once to be piped to the cells on the other tapes. The size of this circuit is same as before. □

Problem 6. *An **oblivious** TM has a fixed set of tape-head movements for all inputs of the same size. We work towards constructing an oblivious TM M' which decides the same language as TM M with only a* log *factor slowdown.*

1. Show that every language $L \in \mathsf{Time}(t(n))$, for a time-constructible $t(n)$, can be decided by an oblivious TM in time $O(t^2(n))$.

   Proof. Introduce a new "hatted" version of each symbol; this will allow you to figure out which cell you were just scanning. Let $L \in \mathsf{Time}(t(n))$ and M be a $t(n)$-time TM which decides L. Mark off $t(n)$ cells to the left and right of the input. This will be the effective work space of M. Our oblivious TM M' will work as follows: for each step of M sweep left for $t(n)$ steps then right for $t(n)$ steps, noting the position of the "hatted" symbol. After registering the state of the tape make another sweep left for $t(n)$ steps and left for $t(n)$ steps to make the necessary changes. □

2. Show that every language L, as above, can be decided by an oblivious TM in time $O(t(n) \cdot \log t(n))$.

Proof. It idea is to use the construction of an efficient universal TM shown in lecture 2. There are quite a few details, so I recommend reading the original. □

Problem 7. *Show that there is a universal* nondeterministic *TM U when given a description of a NTM $\langle M \rangle$ which runs input x in time t, U runs x on M in time $c_M \cdot t$ for some constant c_M depending on M.*

Proof. The provided solution shows that any k-tape NTM M with running time t can be simulated by a 2-tape universal NTM U in time $O(t)$.
U works by treating tape one as the work tape and tape two as the guess tape. Start with the input x on tape one. Randomly guess a sequence $S_1 = (a_1^1, ..., a_k^1, q_i, r_j)$ where a_l^1 is the currently scanned symbol on tape l, q_i is a guessed state number, and r_j is the guessed transition rule. U checks to make sure that, starting from state q_i, it is possible to use r_i to get state where the tape cells contain $a_1^1, ..., a_k^1$. After running all t steps of M, U checks that the final state is an excepting one. □