Due: Monday, April 2, beginning of lecture

1. By definition, **EXPTIME** $= \sum_k \mathrm{DTIME}(2^{n^k})$.

   Prove that **DSPACE**(2^n) is not equal to **EXPTIME**.

   Hint: See problem 9.15 in the text, and its solution.

   Solution:
   Assume to the contrary that

   $$\mathbf{EXPTIME} = \mathbf{DSPACE}(2^n) \tag{1}$$

   We will derive a contradiction.

   (See the definition of $pad(A, f)$ in problem 9.13 in the text.)

   By the Space Hierarchy Theorem (Theorem 9.3) there is a language A in **DSPACE**(2^{n^2}) but not in **DSPACE**(2^n).

   But then $pad(A, n^2) \in$ **DSPACE**(2^n), since an input of length $m = n^2$ to $pad(A, n^2)$ consists of a string w of length n followed by a string of #'s, and a TM can determine whether w is in A in space $O(2^{n^2}) = O(2^m)$. Therefore $pad(A, n^2)$ is in **DSPACE**(2^n), so by our assumption (1), $pad(A, n^2)$ is in **EXPTIME**. Thus $pad(A, n^2)$ is in **DTIME**(2^{n^k}) for some k.

   But then A is in **DTIME**$(2^{n^{2k}})$, because an input w to A of length n can be padded to length $m = n^2$ and accepted in time $O(2^{m^k}) = O(2^{(n^2)^k}) = O(2^{n^{2k}})$. Hence A is in **EXPTIME** so by our assumption (1), A is in **DSPACE**(2^n). But this contradicts our assumption on A.

2. A directed graph is *doubly connected* if every two vertices are connected by a directed path in each direction. Let

   $$DCG = \{\langle G\rangle \mid G \text{ is a doubly connected graph}\}$$

   Prove that DCG is **NL**-complete. (You may use the fact that PATH is **NL**-complete.)

   Solution:
   First we show that DCG $\in$ **NL**. Given a directed graph $G = (V, E)$ a nondeterministic TM enumerates all pairs (u, v) of nodes in V and for each pair, guesses a path from u to v. If the guess fails, then the machine rejects. If all guesses succeed, then the machine accepts. Thus there is an accepting computation iff G is doubly connected. The space required is $O(\log n)$, since the TM just needs to remember the current pair u, v, and the current node on the the guessed path. (We may code each node v_i by a binary number i, so this requires $O(\log n)$ space, where n is the number of nodes in G.)

   Now we show that PATH $\leq_L$ **DCG**. This will show that **DCG** is NL-complete, since PATH is NL complete.

Given an input $\langle G, s, t\rangle$ to PATH we must compute (in log space) an input $\langle G'\rangle$ to **DCG** such that G has a path from s to t iff G' is doubly connected. Let $G = (V, E)$. Then $G' = (V', E')$ where $V' = V$ and

$$E' = E \cup \{(v, s) \mid v \in V, v \neq s\} \cup \{(t, v) \mid v \in V, v \neq t\}$$

It is easy to see that G' can be computed in log space.

CORRECTNESS $\Longrightarrow$: If there is a path in G from s to t then for any pair u, v of nodes in V there is a path in G' from u to v as follows: use the edge (u, s), followed by the path from s to t, followed by the edge (t, v).

CORRECTNESS $\Longleftarrow$: If G' is doubly connected, then there is a path from s to t in G'. Consider a shortest such path. This shortest path cannot use any of the new edges (v, s) since this would bring the path back to the start. It cannot use any of the new edges (t, u) until it reaches t, and then there is no use for such an edge. Hence this shortest path connects s and t in G.

3. Consider the problem $FixedLengthPath$

   $FixedLengthPath$
   <u>Instance</u>
   $\langle G, s, t, d\rangle$, where G is an undirected graph, s and t are nodes in G, and d is a positive integer.

   <u>Question:</u> Is there a path from s to t of length d, and no shorter such path?

   (a) Show that $FixedLengthPath \in$ **NL**.

   Solution:
   To see that an **NL** machine can verify that G has a path from s to t of length at most d, the machine M starts at s, and then guesses successive nodes in the path, ending at t. M has a counter which counts how many nodes are in the path, and checks that this number is $d + 1$ (so the path has d edges).

   This test verifies that the distance from s to t is at most d. To verify that the distance is at least d, we use the fact that **NL** $=$ **coNL**, and test that G does **not** have a path of length $d - 1$ or less from s to t.

   (b) Show that $FixedLengthPath$ is **NL**-complete.

   Hint: Show that $PATH \leq_L FixedLengthPath$. Start with a directed graph G. Construct an undirected graph G' by making n copies of G. Each edge in G' should go from copy i to copy $i + 1$.

   Solution:
   Given $\langle G, s, t\rangle$, where G is a directed graph with nodes s, t, we must construct in $O(\log n)$ space $\langle G', s', t', d\rangle$, where G' is an undirected graph, and G has a path from s to t iff the distance from s' to t' in G' is exactly d.

   Let $G = (V, E)$, and set $n = |V|$. Let $d = n - 1$. We construct $G' = (V', E')$, where

   $$V' = \{(u, i) \mid u \in V \text{ and } 0 \leq i \leq d\}$$

Let

$$E' = \{(u,i),(v,i+1) \mid (u,v) \in E, 0 \leq i < d\} \cup \{(u,i),(u,i+1) \mid u \in V, 0 \leq i < d\}$$

Let $s' = (s, 0)$, $t' = (t, d)$ It is not hard to see that $\langle G', s', t', d\rangle$ can be constructed in $O(\log n)$ space.

CORRECTNESS PROOF: If there is a path from s to t in G, then there is such a path of length at most $n-1$. Hence there is a path from s' to t' in G' of length $d = n-1$; namely follow the path through the n copies of G in G'. If the path is shorter than $n-1$, then use the edges of the form $(t,i),(t,i+1)$ to reach the goal $(t, n-1)$.

Note that there is no path of length shorter than $n-1$ from s' to t', because any such path must go through all the copies of G in G'.

Conversely, if there is a path from s' to t' in G' of length $d = n-1$, then the successive nodes on this path must be in successive layers in G'. The only alterative is that some step in the path would go from level $i+1$ to level i, but then the path would have to have length more than d. Hence the path can be used to find a directed path from s to t in G.