CS601:	**Homework 10**	Due in main office before 4:30 pm, Tuesday, April 28, 2009

Problems:

1. [20 pts.] Let an **alternating graph** $G = (V, E, A)$ be a directed, acyclic graph whose vertices are labeled universal or existential. $A \subseteq V$ is the set of universal vertices. Alternating graphs have a different notion of accessibility. Let APATH(x,y) be the smallest relation on vertices of G such that the following three conditions hold:

   - APATH(x,x)
   - If x is existential and for some edge $\langle x, z \rangle$ APATH(z,y) holds, then APATH(x,y).
   - If x is universal, there is at least one edge leaving x, and for all edges $\langle x, z \rangle$ APATH(z,y) holds, then APATH(x,y).

   Let $\text{REACH}_a = \{\langle G, s, t \rangle \mid APATH_G(s, t)\}$. Prove that REACH_a is P-complete. [I'd prefer that you did this using the theorem that P $=$ ASPACE$[\log n]$ analogously to how I proved in class that REACH is complete for NL.]

2. [20 pts.] Show that the Monotone Circuit Value Problem (MCVP) is P-Complete by reducing REACH_a to MCVP.

3. [25 pts.] NP can be described as the set of decision problems, S, such that membership in S admits short – i.e., polynomial-length – proofs of membership. For example, SAT admits short proofs of membership: given a boolean formula, φ, if φ is satisfiable then I can prove it by presenting a satisfying assignment of length $O(|\varphi|)$. Similarly, CLIQUE admits short proofs of membership: given an input to CLIQUE, G, k, if G has a k-clique then I can prove it by presenting k vertices that form a complete subgraph of G. (It is also interesting to compare this characterization of NP with r.e. – the set of decision problems, S, such that membership in S admits proofs of membership, with no size limit.)

   Prove that the following problem is NP-complete. [I suggest that you reduce 3-SAT to this problem.]

$$\text{FO-THM-SHORT-PROOFS} = \{\varphi, \#^r \mid \vdash \varphi, \text{ and the proof has total length at most } r\}$$

4. [35 pts.] This problem concerns tilings in which we have a fixed set of k tile types and we have as many tiles of each tile type as we want. We have to fill a square grid from these tiles, always placing a tile of type t_1 in the upper-left corner. The shapes of the tiles determine which tiles can fit next to each other. This is given formally via the horizontal and vertical relations, below.

   Formally, in the tiling problem, TILING, we are given a set of tile types, $T = \{t_1, \ldots, t_k\}$, two relations (the tiling rules) $H, V \subseteq T \times T$, and an integer n. A valid tiling is a map $f : \{1, \ldots, n\}^2 \to T$ such that

   - $f(1, 1) = t_1$,
   - for all $1 \le i \le n$ and all $1 \le j < n$, $(f(i, j), f(i, j + 1)) \in H$,
   - for all $1 \le i < n$ and all $1 \le j \le n$, $(f(i, j), f(i + 1, j)) \in V$.

   (a) Prove that TILING is NEXP complete.

   (b) Prove that UNARY-TILING, where n is given in unary instead of binary, is NP complete.

   (c) Prove that ALL-TILING, the problem of deciding whether a valid tiling exists for all $n \ge 1$, is undecidable.

   [Hint: Encode the computation of a non-deterministic Turing machine as a tiling. I suggest that you do the encoding similarly as in the proof of Fagin's theorem, except that the bit δ should be part of the cell containing the state, i.e., $\langle q, \delta, a \rangle$ rather than $\langle q, a \rangle$ with δ somewhere else. To do this, you must describe how to construct your tiling rules so that the first row of a valid tiling correctly encodes the input, that every new row follows via a valid transition from the previous row, and that the last row corresponds to an accepting configuration. Depending on how you do this encoding, you might have to add padding tiles at the bottom of the tiling to make sure that all remaining rows can be filled.]