6.045J/18.400J: Automata, Computability and Complexity Nancy Lynch, Nati Srebro

Homework 9

Due: May 5, 2004 *Susan Hohenberger*

Readings: Sipser, Section 7.5. Also (optionally) see Garey and Johnson's book, "Computers and Intractability: a Guide to NP-Completeness".

Problem 1: (Sipser 7.16) Let G represent an undirected graph and let

$$SPATH = \{\langle G, a, b, k\rangle |\ G \text{ contains a simple path of length at } \textbf{most } k \text{ from } a \text{ to } b\}$$

and

$$LPATH = \{\langle G, a, b, k\rangle |\ G \text{ contains a simple path of length at } \textbf{least } k \text{ from } a \text{ to } b\}$$

1. Show that SPATH $\in$ P.
2. Show that LPATH is NP-complete. You may assume the NP-completeness of UHAMPATH, the Hamiltonian path problem for undirected graphs.

Problem 2: (Sipser 7.22) Let ϕ be a 3cnf-formula. An **$\neq$-assignment** to the variables of ϕ is one where each clause contains two literals with unequal truth values. In other words an $\neq$-assigment satisfies ϕ without assigning three true literals in any clause.

1. Show that the negation of any $\neq$-assignment to ϕ is also an $\neq$-assignment.
2. Let $\neq$-SAT be the collection of 3cnf-formulas that have an $\neq$-assignment. Show that we obtain a polynomial time reduction from 3SAT to $\neq$-SAT by replacing each clause c_i

$$(y_1 \vee y_2 \vee y_3)$$

   by the two clauses

$$(y_1 \vee y_2 \vee z_i) \text{ and } (\overline{z_i} \vee y_3 \vee b)$$

   where z_i is the new variable for each clause c_i and b is a single additional new variable.
3. Conclude that $\neq$-SAT is NP-complete.

Problem 3: (Sipser 7.23) A **cut** in an undirected graph is a separation of the vertices V into two disjoint subsets S and T. The size of a cut is the number of edges that have one endpoint in S and the other in T. Let

$$MAXCUT = \{\langle G, k\rangle |\ G \text{ has a cut of size } k \text{ or more}\}.$$

Show that MAXCUT is NP-complete. You may assume the result of Problem 2 above.
(Hint: Show that $\neq$-SAT $\leq_P$ MAXCUT. The variable gadget for variable x is a collection of $3k$ nodes labeled with x and another $3k$ nodes labeled with $\overline{x}$, where k is the number of clauses. All nodes labeled x are connected with all nodes labeled $\overline{x}$. The clause gadget is a triangle of three edges connecting three nodes labeled with the literals appearing in the clique. Do not use the same node in more than one clause gadget. Prove that this reduction works.)