

CSCI 570 Homework #6 Solution

Prof. Ming-Deh Huang

TAs: Iftikhar Burhanuddin, Chansook Lim

11/12/03

$P=NP$ when $N=1$. Anonymous Stanford student's graffiti in Donald Knuth's Concrete Mathematics.

General recipe to prove that a problem represented by language L is in NP-complete based on Lemma 34.8

1. Prove $L \in NP$
2. Select a known NP-complete language L'
3. $L' \leq_P L$, that is, construct a algorithm f which transforms every instance of L' to an instance $f(x)$ of L such that $x \in L' \Leftrightarrow f(x) \in L$.

34.2-3

Suppose HAM-CYCLE $\in P$. We will use HAM-CYCLE as a subroutine to develop an algorithm for enumerating vertices of a Hamiltonian Cycle. For each edge $e \in E(G)$, remove e from the graph and use HAM-CYCLE to find if there still is a Hamiltonian Cycle in the graph. If HAM-CYCLE returns *No*, put back e in the graph and move to the next edge, else, simply move to the next edge keeping e out. At the end we will have a set of $n - 1$ edges which will constitute a Hamiltonian Cycle. Using these edges we can easily enumerate the vertices in the cycle.

34.2-8

A boolean formula is not a TAUTOLOGY if there exists an assignment of 1 and 0 to the input variables such that the boolean formula evaluates to 0. The assignment to the input variables can be used as a certificate to verify that a boolean formula is not a TAUTOLOGY. Hence, TAUTOLOGY \in co-NP.

34.4-4

We first show that L is complete for NP iff \bar{L} is complete for co-NP. Suppose $L' \in NP$, then $L' \leq_P L$. This implies, there exists a polynomial-time computable function f such that $x \in L'$ iff $f(x) \in L$. Hence, $x \in \bar{L}'$ iff $x \notin L'$ iff $f(x) \notin L$ iff $f(x) \in \bar{L}$. Therefore, $\bar{L}' \leq_P \bar{L}$, which implies that \bar{L} is complete for co-NP.

Now, a boolean formula is in UNSATISFIABILITY if there exists an assignment of 1 and 0 to the input variables such that the boolean formula evaluates to 0. As shown in the previous question a boolean formula is in UNSATISFIABILITY iff it is not in TAUTOLOGY. Hence, we are through if we show that SATISFIABILITY reduces to UNSATISFIABILITY.

- Clearly UNSATISFIABILITY $\in NP$.
- SATISFIABILITY \leq_P UNSATISFIABILITY, this can be done by putting transforming a boolean formula by putting a NOT before the formula.

34.5-1

- Observe that given a mapping of the subgraph of G_2 to G_1 , we can easily verify whether the mapping is isomorphic or not, in polynomial time. Hence, subgraph-isomorphism problem is in NP.
- CLIQUE \leq_p SUBGRAPH-ISOMORPHISM, that is we reduce the Clique problem to Subgraph-isomorphism problem to show NP-completeness. Let G_2 be the graph we are interested in finding a clique of size k . Now we construct G_1 with k vertices and edges between each pair of vertices. It is fairly straightforward to see that G_1 is isomorphic to a subgraph of G_2 if and only if G_2 has a clique of size k .

34.5-2

- Given an x , one can easily verify in polynomial time whether $Ax \leq b$, hence, the 0-1 integer-programming problem is in NP.
- 3-CNF-SAT \leq_P 0-1 INTEGER-PROGRAMMING, that is we reduce the 3-CNF-SAT problem to the 0-1 integer-programming problem to show NP-completeness. Let the 3-CNF-SAT problem have n variables. A variable x in integer-programming corresponds to a $(0, 1)$ -integer variable x , the negation of x corresponds to $(1 - x)$. Now each clause is translated it into a row of A . So for example, the clause x or (not y) or z is translated into $x + (1 - y) + z \geq 1$ which further translates into $-x + y - z \leq 0$.

34.5-5

- Clearly a solution to an instance of the set-partition problem can be verified in polynomial time, hence it is in NP.

- SUBSET-SUM \leq_P SET-PARTITION, that is we reduce the subset-sum problem to the set-partition problem. Let (S, t) be an instance of subset-sum where S is a set of numbers and t a target subset-sum. Let m be the sum of numbers in S . Choose a number M which is much larger than m . Add the two numbers, $M - t, M - (m - t)$ to S to obtain S' . Then S has a subset of sum t iff S' is a yes instance of set-partition problem. This is because $M - t$ and $M - (m - t)$ can't be on the same side of the partition.

34.5-6

- Clearly the Hamiltonian path problem is in NP as a solution to an instance can be verified in polynomial time.
- HAM-CYCLE \leq_P HAM-PATH, that is we will reduce the Hamiltonian Cycle problem to the Hamiltonian Path Problem. Choose any vertex u in G and duplicate it, i.e. add another vertex u' and for each edge (u, v) add the edge (u', v) . Also add two more vertices t and t' and the edges (t, u) and (t', u') . It is fairly easy to see that the new graph has a Hamiltonian Path if and only if G has a Hamiltonian Cycle.

34-1

(a)

A possible decision problem for the independent-set problem: "Is there an independent set of size k in G ?"

- Verification of a solution to an instance of the INDEPENDENT-SET problem can be done in polynomial time. Hence is in NP.
- CLIQUE \leq_P INDEPENDENT-SET, that is we show that the clique problem can be reduced to the independent-set problem. We define $\overline{G} = (V, \overline{E})$ as the graph with the same vertices as V but the edge $(u, v) \in \overline{E}$ iff $(u, v) \notin E$. Now, $V' \subset V$ forms a clique in G iff V' is an independent-set in \overline{G} .

(b)

Starting from $k = n$ down to 1, give the query to the blackbox, "Is there an independent set of size k in G ?". Stop whenever the answer is "yes". This gives us the size of the maximum independent-set.

Assume that the maximum independent-set size is k for $G = (V, E)$. For every vertex v do the following: Let G' be the graph obtained by removing v from V and removing all edges incident on v from E . Observe that an

independent set of G which does not contain v remains an independent set of G' and an independent set of G' is also an independent set of G . Hence, just check if the maximum independent set size is less than k . If it is less than k , then v is part of the maximum independent set, undo the change in the graph and continue. At the end of the process, we are left with the maximum independent set.