CS 473: Algorithms, Fall 2009
HBS 12

Problem 1. Show that the following problems are in **NP**. That is, give a certificate and a certifier that checks the certificate. *Note that the size of the certificate must be polynomial in the input size, and the running time of the certifier must be polynomial in the size of the certificate.*

(a) NETWORK FLOW: Given a network G with source s and sink t, and an integer k, does G have an s,t-flow of value at least k?

(b) BOX DEPTH: Given a set of n axis-aligned rectangles in the plane and an integer k, is there a subset of at least k rectangles that contain a common point?

Problem 2. Consider the following problem, called ISBIPARTITE: Given a graph G, is G bipartite?

(a) Describe a polynomial-time reduction from ISBIPARTITE to 2SAT and prove that it is correct.

The reduction maps an instance G of ISBIPARTITE *to an instance I of* 2SAT. *To show that the reduction is correct, you need to prove that G is bipartite if and only if I is satisfiable.*

(b) Conclude that there is a polynomial-time algorithm for ISBIPARTITE.

Problem 3. In the CLIQUE problem, we are given a graph G and an integer k, and the goal is to decide whether G has a *clique* of size at least k[1]. The CLIQUE problem is **NP**-complete. The CLIQUE3 problem is a special case of the CLIQUE problem in which the input graph G has maximum degree at most 3.

(a) Describe a polynomial-time reduction from CLIQUE3 to CLIQUE.

(b) Give a polynomial-time algorithm for CLIQUE3.

Why don't these two results together with the fact that CLIQUE is **NP**-complete imply that $\mathbf{P} = \mathbf{NP}$?

(Slightly harder. You can skip it during the hbs.) Recall the BOX DEPTH problem defined in **Problem 1**.

(a) Describe a polynomial-time reduction from BOX DEPTH to CLIQUE.

(b) Give a polynomial-time algorithm for BOX DEPTH.

Why don't these two results together with the fact that CLIQUE is **NP**-complete imply that $\mathbf{P} = \mathbf{NP}$?

[1] That is, does G have a subgraph H with at least k nodes such that H is a complete graph?

Problem 4. A boolean formula is in disjunctive normal form (DNF) if it is a disjunctions (OR) of several clauses, each of which is the conjunction (AND) of several literals, each of which is either a variable or its negation. For example,

$$(a \wedge b \wedge c) \vee (\bar{a} \wedge b) \vee (\bar{c} \wedge x)$$

Give a polynomial-time algorithm that decides whether a DNF formula is satisfiable. Why doesn't this imply that $\mathbf{P} = \mathbf{NP}$?

Problem 5. *(Harder. You can skip it during the hbs.)* In the NODE DISJOINT PATHS problem, we are given an undirected graph G, k vertices $s_1, s_2, \ldots, s_k$ (the sources), and k vertices $t_1, t_2, \ldots, t_k$ (the destinations). The goal is to decide whether G has k node-disjoint paths (that is, paths which have no nodes in common) such that the i-th path goes from s_i to t_i. Show that the NODE DISJOINT PATHS problem is **NP**-complete.

Here is a sequence of progressively stronger hints.

(a) Reduce from 3SAT.

(b) For a 3SAT formula with m clauses and n variables, use $k = m + n$ sources and destinations. Introduce one source/destination pair (s_x, t_x) for each variable x, and one source/destination pair (s_c, t_c) for each clause c.

(c) For each 3SAT clause, introduce 6 new intermediate vertices, one for each literal occurring in that clause and one for its complement.

(d) Notice that if the path from s_c to t_c goes through some intermediate vertex representing, say, an occurrence of variable x, then no other path can go through that vertex. What vertex would you like the other path to be forced to go through instead?