

# Computer Science 511

## Fall 2008

### Exam 2

Wednesday, November 12

This closed-book, closed-notes two-hour test consists of 6 questions. The number of points for each problem is indicated on the next page.

- Read all questions carefully before starting.
- Work on the problems that seem easiest first.
- Attempt to solve all problems.
- Show your work, but also remember that we prefer concise answers.
- Write all your answers clearly on the space provided in the exam paper. If you need additional paper, please ask us.
- If you do not understand a problem, please ask us for clarification.
- Clearly state any simplifying assumptions you make in solving a problem.
- When asked to describe algorithm, you are expected to argue its correctness and analyze its running time.

Name: \_\_\_\_\_

# Score

1	2	3	4	5	6	Total
16	16	18	16	16	18	100

# 1 NP-Completeness (16 points)

As we saw in class, the following problem is NP-complete.

HAMILTONIAN CYCLE

**Input:** Undirected graph  $G = (V, E)$ .

**Question:** Does  $G$  have a cycle that visits each vertex exactly once?

Consider HAMILTONIAN CYCLE restricted to graphs in which every vertex has degree at most 2. Call this problem HAMILTONIAN CYCLE-2.

(a) (4 points) Prove that HAMILTONIAN CYCLE-2 is in NP.

- (b) (6 points) What is wrong with the following proof of NP-completeness for HAMILTONIAN CYCLE-2?

We know that the HAMILTONIAN CYCLE problem in general graphs is NP-complete, so it is enough to present a reduction from HAMILTONIAN CYCLE-2 to HAMILTONIAN CYCLE. Given a graph  $G$  with vertices of degree at most 2, the reduction leaves the graph unchanged: clearly the output of the reduction is a possible input for the HAMILTONIAN CYCLE problem. Furthermore, the answer to both problems is identical. This proves the correctness of the reduction and, therefore, the NP-completeness of HAMILTONIAN CYCLE-2.

(c) (6 points) Show that HAMILTONIAN CYCLE-2 can be solved in polynomial time.

## 2 Decision, Search, and Optimization (16 points)

As you recall, the *subset sum* problem is defined as follows.

SUBSET SUM

**Input:** A set of integers  $S = \{w_1, w_2, \dots, w_n\}$  and an integer  $W$ .

**Question:** Does there exist a subset of  $S$  that adds up to exactly  $W$ ?

- (a) (8 points) Suppose you have a procedure *solves* that solves SUBSET SUM in polynomial time. That is, this procedure takes an instance  $\langle S, W \rangle$  of SUBSET SUM and returns “yes” if there is a subset of  $S$  that adds up to  $W$ , and returns “no” otherwise. Show that you can use this procedure to develop a polynomial-time algorithm that returns a subset of  $S$  that adds up to  $W$ , if such a subset exists, or reports that no such subset exists otherwise.

(b) (8 points) Consider the following optimization version of SUBSET SUM:

MAX SUBSET SUM

**Input:** A set of positive integers  $S = \{w_1, w_2, \dots, w_n\}$  and a positive integer  $W$ .

**Goal:** Find a subset of  $S$  whose sum is as large as possible, without exceeding  $W$ .

Show that MAX SUBSET SUM is solvable in polynomial time if and only if SUBSET SUM is.

### 3 Proving NP-Completeness by Generalization (18 points)

For each of the problems below, prove that it is NP-complete by (i) arguing why it is in NP and (ii) stating which NP-complete problem it generalizes and how.

- (a) (6 points) MIN UNSAT: Given a CNF formula  $\varphi$  and an integer  $g \geq 0$ , determine whether there exists a truth assignment where at most  $g$  clauses of  $\varphi$  are not satisfied.

- (b) (6 points) SPARSE SUBGRAPH: Given a graph  $G$  and two integers  $a \geq 0$  and  $b \geq 0$ , determine if there is a subset of at least  $a$  vertices of  $G$  such that there are at most  $b$  edges between them.

- (c) (6 points) SUBGRAPH ISOMORPHISM: Given as input two undirected graphs  $G$  and  $H$ , determine whether  $G$  is a subgraph of  $H$  (that is, whether by deleting certain vertices and edges of  $H$  we obtain a graph that is, up to renaming of vertices, identical to  $G$ ).

## 4 Degree-Constrained Spanning Trees (16 points)

A *spanning tree* of a connected, undirected graph  $G = (V, E)$  is a tree  $T$  whose vertex set is  $V$  and whose edge set is a subset of  $E$  (note that, by definition,  $T$  must be connected). Now consider the following problem.

$k$ -SPANNING TREE

**Input:** An undirected graph  $G = (V, E)$ .

**Output:** A spanning tree of  $G$  in which each node has degree at most  $k$ , if such a tree exists.

Note that we place no restriction on the degrees of the nodes in  $G$ .

(a) (4 points) Show that, for  $k \geq 2$ ,  $k$ -SPANNING TREE is in NP.

(b) (12 points) Show that, for  $k \geq 2$ ,  $k$ -SPANNING TREE is NP-complete. (*Hint*: What happens when  $k = 2$ ?)

## 5 The Knapsack Problem (16 points)

Consider the following problem.

KNAPSACK

**Input:** A collection of  $n$  items, where item  $i$  has an integer weight  $w_i$  and an integer value  $v_i$ , along with two integers,  $W$  and  $V$ .

**Question:** Does there exist a subset of the items whose total weight is at most  $W$  and whose total value is at least  $V$ ?

(a) (4 points) Prove that KNAPSACK  $\in$  NP.

(b) (12 points) Show that KNAPSACK is NP-complete. (*Hint:* Use reduction from SUBSET SUM.)

## 6 Partitioning into Communities (18 points)

An *interaction matrix* for a set  $P$  of  $n$  people is a zero-one matrix  $C = [c_{ij}]$ , where

$$c_{ij} = \begin{cases} 1 & \text{if person } i \text{ is known to interact with person } j \text{ on a social basis,} \\ 0 & \text{otherwise.} \end{cases}$$

Note that  $C$  is symmetric; i.e.,  $c_{ij} = c_{ji}$ . Assume that  $c_{ii} = 1$  for  $i = 1, \dots, n$ .

Let us define a *community* to be a subset  $A$  of  $P$  such that every two people in  $A$  interact. Consider the following problem

PARTITIONING INTO COMMUNITIES

**Input:** An interaction matrix  $C$  for a set  $P$  of  $n$  people and an integer  $K$ .

**Question:** Can  $P$  be partitioned into at most  $K$  disjoint communities?

(a) (4 points) Show that PARTITIONING INTO COMMUNITIES is in NP.

(b) (14 points) Show that PARTITIONING INTO COMMUNITIES is NP-complete. (*Hint:* Consider the case where  $K = 3$ .)