

## Complexity Theory

Easter 2014

### Suggested Exercises 2

1. Given a graph  $G = (V, E)$ , a set  $U \subseteq V$  of vertices is called a *vertex cover* of  $G$  if, for each edge  $(u, v) \in E$ , either  $u \in U$  or  $v \in U$ . That is, each edge has at least one end point in  $U$ . The decision problem **V-COVER** is defined as:

given a graph  $G = (V, E)$ , and an integer  $K$ , does  $G$  contain a vertex cover with  $K$  or *fewer* elements?

- (a) Show a polynomial time reduction from **IND** to **V-COVER**.
- (b) Use (a) to argue that **V-COVER** is **NP**-complete.

2. The problem of four dimensional matching, **4DM**, is defined analogously with **3DM**:

Given four sets,  $W, X, Y$  and  $Z$ , each with  $n$  elements, and a set of quadruples  $M \subseteq W \times X \times Y \times Z$ , is there a subset  $M' \subseteq M$ , such that each element of  $W, X, Y$  and  $Z$  appears in exactly one triple in  $M'$ .

Show that **4DM** is **NP**-complete.

3. Given a graph  $G = (V, E)$ , a *source vertex*  $s \in V$  and a *target vertex*  $t \in V$ , a *Hamiltonian Path* from  $s$  to  $t$  in  $G$  is a path that begins at  $s$ , ends at  $t$  and visits every vertex in  $V$  exactly once. We define the decision problem **HamPath** as:

given a graph  $G = (V, E)$  and vertices  $s, t \in V$ , does  $G$  contain a Hamiltonian path from  $s$  to  $t$ ?

- (a) Give a polynomial time reduction from the Hamiltonian cycle problem to **HamPath**.
- (b) Give a polynomial time reduction from **HamPath** to the problem of determining whether a graph has a Hamiltonian cycle.

*Hint:* consider adding a vertex to the graph.

4. We know from the Cook-Levin theorem that every problem in **NP** is reducible to **SAT**. Sometimes it is easy to give an explicit reduction. In this exercise you are asked to give such explicit reductions for two graph problems: **3-Col** and **HAM**. That is,

- (a) describe how to obtain, for any graph  $G = (V, E)$ , a Boolean expression  $\phi_G$  so that  $\phi_G$  is satisfiable if, and only if,  $G$  is 3-colourable; and
  - (b) describe how to obtain, for any graph  $G = (V, E)$ , a Boolean expression  $\phi_G$  so that  $\phi_G$  is satisfiable if, and only if,  $G$  contains a Hamiltonian cycle.
5. We use  $x;0^n$  to denote the string that is obtained by concatenating the string  $x$  with a separator  $;$  followed by  $n$  occurrences of 0. If  $[M]$  represents the string encoding of a *non-deterministic* Turing machine  $M$ , show that the following language is NP-complete:

$$\{[M];x;0^n \mid M \text{ accepts } x \text{ within } n \text{ steps}\}.$$

*Hint:* rather than attempting a reduction from a particular NP-complete problem, it is easier to show this from first principles, i.e. construct a reduction for any NDTM  $M$ , and polynomial bound  $p$ .

6. **0-1 Integer Linear Programming.** An instance of a *linear programming* problem consists of a set  $X = \{x_1, \dots, x_n\}$  variables and a set of constraints, each of the form

$$\sum_{1 \leq i \leq n} c_i x_i \leq b,$$

where each  $c_i$  and  $b$  is an integer.

The 0-1 Integer Linear Programming Feasibility problem is, to determine, given such a linear programming problem, whether there is an assignment of values from the set  $\{0, 1\}$  to the variables in  $X$  so that substituting these values in the constraints leads to all constraints being simultaneously satisfied.

Prove that this problem is NP-complete.

7. **Self-Reducibility.** *Self-reducibility* refers to the property of some problems in  $L \in \text{NP}$ , where the problem of finding a *witness* for the membership of an input  $x$  in  $L$  can be reduced to the decision problem for  $L$ . This question asks you to give such arguments in two specific instances.
- (a) Show that, given an oracle (i.e. a black box) for deciding whether a given graph  $G = (V, E)$  is Hamiltonian, there is a polynomial-time algorithm that, on input  $G$ , outputs a Hamiltonian cycle in  $G$  if one exists.
  - (b) Show that, given an oracle for deciding whether a given graph  $G$  is 3-colourable, there is a polynomial-time algorithm that, on input  $G$ , produces a valid 3-colouring of  $G$  if one exists.