CSI 4105 Design and Analysis of Algorithms II — Winter 2012
Computer Science — University of Ottawa

Homework Assignment #1 (100 points, weight 6.67%)
Due: Friday Feb 10, at 11:30 p.m. (in lecture)

---

1. (20 points) Let LongestPath denote the decision problem that asks, given a graph $G = (V, E)$, two vertices $s, t \in V$ and an integer k, whether G has a simple path between s and t with at least k edges.

   Consider the optimization problem FindLongestPath that given a graph $G = (V, E)$ we need to find the longest simple path between s and t in G.

   Prove that if LongestPath $\in \mathcal{P}$, then the problem of listing the longest simple path between given vertices s and t, FindLongestPath, can be solved in polynomial time.

   Hint: Design an algorithm for FindLongestPath *that performs a polynomial number of steps plus makes a polynomial number of calls to the algorithm that solves* FindLongestPath. *Justify that your algorithm does the promised job (i.e. is correct and runs in polytime).*

2. (20 marks) For each of the following statements about problems and class of problems discussed in lecture, specify whether the statement is "true" or "false"; **briefly justify your answer**. You may use any known results.

   (a) For all problems $X \in \mathcal{NP}$, we have $X \leq_P$ SubsetSum.

   (b) Given the current knowledge on the theory of NP-completeness, it is still conceivable that HamCycle $\in P$ and IndependentSet $\notin P$.

   (c) If $\mathcal{P} = \mathcal{NP}$ then every problem in $\mathcal{NP}$ is NP-complete.

   (d) If LongestPath $\in \mathcal{P}$ then IndependentSet $\in \mathcal{P}$.

   (e) (Requires extra research) Suppose that P $\neq$ NP. Then for any problem $X \in \mathcal{NP}$ then, either $X \in P$ or X is NP-complete.

3. (30 points) Textbook chapter 8, exercise 4:
   Suppose you are consulting for a group that manages a high-performance real-time system in which asynchronous processes make use of shared resources. Thus the system has a set of n *processes* and a set of m *resources*. At any given point in time, each process specifies a set of resources that it requests to use. Each resource might be requested by many processes at once; but it can only be used by a single process at a time. Your job is to allocate resources to processes that request them. If a process has allocated all the resources it requests, then it is *active*; otherwise it is *blocked*. You want to perform the allocation problem so that as many processes as possible are active. Thus we phrase the Resource Reservation Problem as follows: Given a set of processes and resources, the set of requested resources for each process, and a number k, is it possible to allocate resources to processes so that at least k processes will be active?

Consider the following list of problems, for each problem either give a polynomial time algorithm or prove that the problem is NP-complete:

(a) The general RESOURCE RESERVATION PROBLEM defined above.

(b) The special case of the problem when $k = 2$.

(c) The special case of the problem when each resource is request by at most 2 processes.

4. (30 points) Textbook chapter 8, exercise 18: You've been asked to help some organizational theorists analyze data on group decision-making. In particular, they've been looking at a dataset that consists of decisions made by a particular governmental policy committee, and they are trying to decide whether it's possible to identify a small set of influential members of the committee.

   Here is how the committee works. It has a set $M = \{m_1, \ldots, m_n\}$ of n members, and over the past year it has voted on t different issues. On each issue, each member can vote either "yes", "no", or "abstain"; the overall effect is that the committee presents an affirmative decision on the issue if the number of "yes" votes is strictly greater that the number of "no" votes (the "abstain" votes don't count for either side), and it delivers a negative decision otherwise.

   Now we have a big table consisting of the vote cast by each committee member on each issue, and we'd like to consider the following definition. We say that a subset of members $M' \subseteq M$ is *decisive* if, had we looked just at the votes cast by the members of M', the committee's decision on *every* issue would have been the same. (In other words, the overall outcome of the voting among the members of M' is the same on every issue as the outcome of the voting by the entire committee.) Such a subset can be viewed as a kind of "inner circle" that reflects the behaviour of the committee as a whole.

   Here is the question: Given the votes cast by each member on each issue, and given a parameter k, we want to know whether there is a decisive subset consisting of at most k members. We'll call this an instance of the DECISIVE SUBSET Problem.

   Example: Suppose we have four committee members and three issues. We're looking for a decisive set of size at most $k = 2$, and the voting went as follows:

Issue#	m_1	m_2	m_3	m_4
Issue 1	yes	yes	abstain	no
Issue 2	abstain	no	no	abstain
Issue 3	yes	abstain	yes	yes

   Then the answer to this instance is "yes", since members m_1 and m_3 constitute a decisive subset.

   Prove that DECISIVE SUBSET is NP-complete.

   Hint: Remember to justify why the problem is in $\mathcal{NP}$, and then prove that an NP-complete problem X of your choice polynomial-time reduces to DECISIVE SUBSET *($X \leq_P$ DECISIVE SUBSET).*