CSOR W4231.002 – Spring, 2018

## Homework 6

Out: Monday, Apr 16, 2018
Due: 8pm, Monday, April 30, 2018

*For all algorithms you design, you should always describe them clearly in English, give pseudocode, prove correctness and give the best upper bound that you can for the running time.*

*If you give a reduction, you should state the input instances to both problems, clearly describe the reduction transformation and argue that it requires polynomial time, and prove equivalence of the original and the derived instances.*

*Collaboration is limited to discussion of ideas only. You should write up the solutions entirely on your own. You should list your collaborators on your write-up. If you do not type your solutions, be sure that your hand-writing is legible, your scan is high-quality and your name is clearly written on your homework.*

*I encourage you to work on both recommended exercises.*

**As usual, you may not use any external resources such as the internet or other text-books to solve this homework. You should adhere to the department's academic honesty policy (see the course website, too).**

1. (20 points) A large store has $m$ customers and $n$ products and maintains an $m \times n$ matrix $A$ such that $A_{ij} = 1$ if customer $i$ has purchased product $j$; otherwise, $A_{ij} = 0$.

   Two customers are called *orthogonal* if they did not purchase any products in common. Your task is to help the store determine a maximum subset of orthogonal customers.

   Give an efficient algorithm for this problem or state its decision version and prove it is $\mathcal{NP}$-complete.

2. (20 points) Suppose you had a polynomial-time algorithm that, on input a graph, answers **yes** if and only if the graph has a Hamiltonian cycle.

   Show how, on input a graph $G = (V, E)$, you can return in polynomial time

   - a Hamiltonian cycle in $G$, if one exists,
   - **no**, if $G$ does not have a Hamiltonian cycle.

3. (20 points) There is a set of ground elements $E = \{e_1, e_2, \ldots, e_n\}$ and a collection of $m$ subsets $S_1, S_2, \ldots, S_m$ of the ground elements (that is, $S_i \subseteq E$ for $1 \leq i \leq m$).

   The goal is to select a minimum cardinality set $A$ of ground elements such that $A$ contains at least one element from each subset $S_i$.

   State the decision version of this problem and prove that it is $\mathcal{NP}$-complete.

4. (24 points) A paper mill manufactures rolls of paper of standard width 3 meters. Customers want to buy paper rolls of shorter width, and the mill has to cut such rolls from the 3m rolls. For example, one 3 m roll can be cut into 2 rolls of width 93cm and one roll of width 108cm; the remaining 6cm goes to waste. The mill receives an order of

   - 97 rolls of width 135cm
   - 610 rolls of width 108cm
   - 395 rolls of width 93cm
   - 211 rolls of width 42cm

   Form a linear program to compute the smallest number of 3m rolls that have to be cut to satisfy this order, and explain how they should be cut.

5. (36 points) Formulate linear or integer programs for the following optimization problems. (Full-credit will be given to LP solutions, when they are possible.)

   (a) `Min-cost flow` (see also hw5, problem 5): Given a flow network with capacities $c_e$ and costs $a_e$ on every edge $e$, and supplies $s_i$ on every vertex $i$, find a feasible flow $f : E \to R_+$—that is, a flow satisfying edge capacity constraints and node supplies—that minimizes the total cost of the flow.

   (b) `The assignment problem`: There are $n$ persons and $n$ objects that have to be matched on a one-to-one basis. There's a given set $A$ of ordered pairs $(i, j)$, where a pair $(i, j)$ indicates that person $i$ can be matched with object $j$. For every pair $(i, j) \in A$, there's a value $a_{ij}$ for matching person $i$ with object $j$. Our goal is to assign persons to objects so as to maximize the total value of the assignment.

   (c) `Uncapacitated facility location`: There is a set $F$ of $m$ facilities and a set $D$ of $n$ clients. For each facility $i \in F$ and each client $j \in D$, there is a cost $c_{ij}$ of assigning client $j$ to facility $i$. Further, there is a one-time cost $f_i$ associated with opening and operating facility $i$. Find a subset $F'$ of facilities to open that minimizes the total cost of (i) operating the facilities in $F'$ and (ii) assigning every client $j$ to one of the facilities in $F'$.

**RECOMMENDED exercises: do NOT return, they will not be graded!**

1. (a) A *clique* in an undirected graph $G = (V, E)$ is a subset $S$ of vertices such that *all* possible edges between the vertices in $S$ appear in $E$. Computing the maximum clique in a network (or the number of cliques of at least a certain size) is useful in analyzing social networks, where cliques corresponds to groups of people who all know each other.

   State the decision version of the above maximization problem and show that it is $\mathcal{NP}$-complete. *Hint: reduction from* `Independent Set`*.*

   (b) We say that $G$ is a *subgraph* of $H$ if, by deleting certain vertices and edges of $H$ we obtain a graph that is, up to renaming of the vertices, identical to $G$.

   The following problem has applications, e.g., in pattern discovery in databases and in analyzing the structure of social networks.

   `Subgraph Isomorphism:` Given two undirected graphs $G$ and $H$, determine whether $G$ is a subgraph of $H$ and if so, return the corresponding mapping of vertices in $G$ to vertices in $H$.

   Show that `Subgraph Isomorphism` is $\mathcal{NP}$-complete.

   (c) Similarly, consider the following problem.

   `Dense Subgraph:` Given a graph $G$ and two integers $a$ and $b$, find a set of $a$ vertices of $G$ such that there are at least $b$ edges between them.

   Show that `Dense Subgraph` is $\mathcal{NP}$-complete.

2. Give an $O(n^2 2^n)$ dynamic programming algorithm to solve an instance of TSP with $n$ cities (that is, compute the cost of the optimal tour and output the actual optimal tour). What are the space requirements of your algorithm?

   *Hint: Let $V = \{1, \ldots, n\}$ be the set of cities. Consider progressively larger subsets of cities; for every subset $S$ of cities including city 1 and at least one other city, compute the shortest path that starts at city 1, visits all the cities in $S$ and ends at city $j$, for every $j \in S$.*