Fall 2019 B503 Homework 5

Due date: Dec. 1, Sunday, 11:59PM Lecturer: Qin Zhang

Your Name:_____ Your University ID:_____

Instruction: Please submit your homework solution **before due date, via Canvas**. Homework solution must be typesetted in PDF format via LaTeX (preferred) or Word. Please add references to ALL the resources you have used for completing the assignment. You are allowed to discuss the assignment with other students, and if you do so, please list their names in the submission.

Due: Dec. 1, Sunday, 11:59PM

Total points: 60

Late Policy: No extensions or late homeworks will be granted, unless a request is made to the course instructor before due date and written documents are provided to support the reason for late submission.

Problem 1 (10 points). A DNF boolean formula is expressed as an OR of clauses:

$$clause_1 \vee clause_2 \vee \cdots \vee clause_n.$$

In the case of 3-DNF problem, each clause is the AND of 3 literals. As an example, we have the following problem instance:

$$(x_1 \wedge x_2 \wedge x_3) \vee (x_1 \wedge x_3 \wedge \neg x_4) \vee (x_1 \wedge x_2 \wedge x_4).$$

Is *3-DNF SAT* NP-complete, or in P? Prove your answer.

Problem 2 (10 points). Consider the following problem: given a graph $G = (V, E)$ with n vertices and m edges, does G contain a *vertex cover* of size 5 ($m > 5$)?

Is this problem NP-complete, or in P? Prove your answer.

Problem 3 (10 points). A store trying to analyze the behavior of its customers will often maintain a two-dimensional array A, where the rows correspond to its customers and the columns correspond to the products it sells. The entry $A[i, j]$ specifies the quantity of product j that has been purchased by customer i.

Here's a tiny example of such an array A.

One thing that a store might want to do with this data is the following. Let us say that a subset S of the customers is *diverse* if no two of the of the customers in S have ever bought the same product (i.e., for each product, at most one of the customers in S has ever

	liquid detergent	beer	diapers	cat litter
Raj	0	6	0	3
Alanis	2	3	0	0
Chelsea	0	0	0	7

bought it). A diverse set of customers can be useful, for example, as a target pool for market research.

We can now define the *Diverse Subset* Problem as follows: Given an $m \times n$ array A as defined above, and a number $k \leq m$, is there a subset of at least k of customers that is diverse?

Show that *Diverse Subset* is NP-complete.

Problem 4 (10 points). The following is a version of the *Independent Set* Problem. You are given a graph $G = (V, E)$ and an integer k. For this problem, we will call a set $I \subset V$ *strongly independent* if, for any two nodes $v, u \in I$, the edge (v, u) does not belong to E, and there is also no path of two edges from u to v, that is, there is no node w such that both $(u, w) \in E$ and $(w, v) \in E$. The *Strongly Independent Set* Problem is to decide whether G has a strongly independent set of size at least k.

Prove that the *Strongly Independent Set* Problem is NP-complete.

Problem 5 (10 points). We've seen the Interval Scheduling Problem in greedy algorithm's Chapter. Here we consider a computationally much harder version of it that we'll call *Multiple Interval Scheduling*. As before, you have a processor that is available to run jobs over some period of time (e.g., 9 A.M. to 5 P.M).

People submit jobs to run on the processor; the processor can only work on one job at any single point in time. Jobs in this model, however, are more complicated than we've seen in the past: each job requires a set of intervals of time during which it needs to use the processor. Thus, for example, a single job could require the processor from 10 A.M. to 11 A.M., and again from 2 P.M. to 3 P.M.. If you accept this job, it ties up your processor during those two hours, but you could still accept jobs that need any other time periods (including the hours from 11 A.M. to 2 A.M.).

Now you're given a set of n jobs, each specified by a set of time intervals, and you want to answer the following question: For a given number k, is it possible to accept at least k of the jobs so that no two of the accepted jobs have any overlap in time?

Show that *Multiple Interval Scheduling* is NP-complete.

Problem 6 (10 points). In this problem you need to show that finding a Hamiltonian cycle is not much harder than deciding if a graph contains a Hamiltonian cycle or not.

Formally, suppose you are given a polynomial-time black-box algorithm that decides whether a given graph G contains a Hamiltonian cycle or not. Design a polynomial time algorithm that, given a graph G which is promised to contain a Hamiltonian cycle, outputs a Hamiltonian cycle of G.