COMP3011: Design and Analysis of Algorithms Due: 1 Dec 2024

Assignment 3

The Hong Kong Polytechnic University

Problem 1 (20 points)

Our TA is analyzing the background of the students by maintaining a two-dimensional array A. In array A, the rows correspond to its students and the columns correspond to the subjects provided by the university. The entry $A[i, j]$ specifies whether student i has registered subject j. The following array gives an example.

	Data Structure	Programming	Discrete Math	Algorithm
Allen	1	0	1	0
Bo	0	0	1	0
Chen	0	1	0	1

Let us say that a subset S of the students is *diverse* if no two of the students in S have ever registered the same subject (i.e., for each subject, at most one of the students in S has ever registered it).

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 students that is diverse? Prove the problem is NP-complete.

Problem 2 (20 points)

Recall the *k-coloring* problem. Given a graph $G = (V, E)$ and an integer k, the task is to determine if the graph can be colored using at most k colors such that no two adjacent vertices are given the same color. Construct a polynomial-time reduction from the 3-coloring problem to the 4-coloring problem.

Problem 3 (20 points)

Consider the following maximization version of the 3-Dimensional Matching Problem. Given disjoint sets X, Y, and Z, and given a set $T \subseteq X \times Y \times Z$ of ordered triples, a subset $M \subseteq T$ is a 3-dimensional matching if each element of $X \cup Y \cup Z$ is contained in at most one of these triples. The Maximum 3-Dimensional Matching Problem is to find a 3-dimensional matching M of maximum size. (The size of the matching, as usual, is the number of triples it contains. You may assume $|X| = |Y| = |Z|$ if you want.) Unfortunately, this optimization problem is NP-hard (you don't have to prove this). Give a polynomial-time algorithm that finds a 3-dimensional matching of size at least 1/3 times the maximum possible size. Justify the correctness.

Problem 4 (40 points)

Consider the following *Bin Packing problem*. Suppose that we are given a set of n objects, where the size s_i of the i-th object satisfies $0 < s_i < 1$. We wish to pack all the objects into the minimum number of unit-size bins. Each bin can hold any subset of the objects whose total size does not exceed 1.

(1) Prove the decision version of the Bin Packing problem (i.e., deciding whether we can pack all objects in K bins) is NP-complete.

(2) Design a polynomial time algorithm with constant approximation ratio. Justify its correctness.