Algorithms, 2023 Fall, Homework 2
(Due: Oct 7)

September 28, 2023

Readings: Please do the following if you have not done so.

1. You should know that sorting (n numbers) can be done in $O(n \log n)$ time. You also need to learn some $O(n^2)$ time sorting algorithms (insertion sort (CLRS Ch 2.1), bubble sort (see below)) by yourself.

You can discuss homeworks with your classmates. But you will have to write down the solution on your own. You need to acknowledge other students who help you in the homework. If you read some other source that is helpful, you should list all of them.

Problems marked with (@) can be skipped by those who have extensive experience in OI (you can show TA ANY certificate that verifies your OI experience).

Problem 1 :(@) CLRS problem 2-2 (pp. 40) Correctness of bubble sort.

Problem 2 :(@) Implement the greedy algorithm for the interval scheduling problem (in $O(n \log n)$ time).

Problem 3 :(@) Modify the topological sorting algorithm so that it can detect a directed cycle (if any). If the graph is a DAG, the algorithm outputs a topological order. Otherwise, it outputs a directed cycle.

Problem 4 :(@) KT book pp.109 problem 8

Problem 5 :(@) KT book pp.190 problem 4

Problem 6 :(@) KT book pp.190 problem 5

Problem 7 : (1) In the algorithm for finding all strongly connected components (SCCs), we perform the 2nd DFS in G^T according to the decreasing order of finishing times of the first DFS. Provide a counter example for the following algorithm: perform the 2nd DFS in G according to the increasing order of finishing times of the first DFS.

(2) Suppose we have two SCCs C and D in a directed graph G. D is reachable from C. Is the following claim true for DFS? we use $f[v]$ to denote the finishing time of v. $f[u] > f[v]$ for all $u \in C$ and $v \in D$. If yes, prove it. If no, provide a counter example. (note the subtle difference between the above claim and the claim we made in the class)

Problem 8 : Let $G(V, E)$ be a directed graph with every vertex v labelled a unique number $L(v) \in [n]$ ($n = |V|$). For each $v \in V$, let $R(v)$ be the set of vertices

reachable from v. Define $\min(v)$ to be the vertex in $R(v)$ whose label is the minimum. Give an $O(|V| + |E|)$-time algorithm to compute $\min(v)$ for all $v \in V$.

Problem 9 : We define an interval graph G as follows: Suppose we have a collection of intervals on the real line. Each node of G represent an interval and two nodes are connected by an edge if two corresponding interval overlap. We have shown that the maximum independent set problem (an independent set is a subset of nodes that are not connected by any edge) in interval graphs (the first interval scheduling problem) (if we know the interval representation) is polynomial time solvable (in fact $O(n \log n)$ time). In this problem, we consider the coloring problem in such graphs. In the coloring problem, we assign each vertex a color such that no edge has two endpoints colored by the same color. We would like to use as few colors as possible. Consider the following algorithm which runs in iterations. In each iteration, we find a maximum independent set, assign a new color to all vertices in this set, and delete all vertices in the set. Prove this algorithm finds the optimal coloring, or give a counter example to show the algorithm is not optimal. You can assume we already have an interval representation. (hint: a clique is complete graph, i.e., every pair of vertices is connected by an edge. Consider the largest clique in the graph. It is easy to see that the minimum number of colors you can use is at least the number of vertices in this clique.)

Bonus problems: Bonus problems are problems that may be (may be not) more challenging than other problems. You do not lose points if you do not answer bonus problems. But you will get extra points if you get correct answers for such problems.

Problem 1 : Consider the following interval packing problem: given a list of intervals $[a_i, b_i]$, $i = 1, \ldots, n$ and an integer k, find a maximum subset of intervals such that no points is contained in more than k of them.

1. Give a simple greedy algorithm for the interval packing problem. (Hint: the algorithm is a simple greedy algorithm. you do not get points using other methods). You need to provide a rigorous proof of correctness.