E0 225 : Homework 1

Deadline : 29th August, 2019, 2 pm

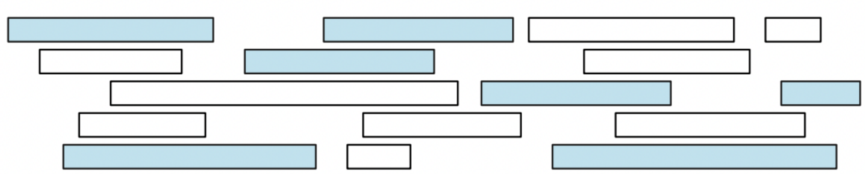
Instructions

- All problems carry equal weight.
- You need to attempt the homework problems on your own (no peer discussions are allowed). You are also forbidden from consulting the internet and other external sources.
- Academic dishonesty/plagiarism will be dealt with severe punishment.
- Late submissions are accepted only with prior approval or medical certificate.

1. *Stable Matching* (Erickson: Ch 4 Problem 11).
   Describe and analyze an efficient algorithm to determine whether a given set of male and female preferences admits a *unique* stable matching.
   (*Hint.* In the Deferred Acceptance algorithm, proposers get their *best* valid partners and proposees get their *worst* valid partners.)

2. *Greedy: Interval Cover* (Erickson: Ch 4 Problem 2).
   Let X be a set of n intervals on the real line. We say that a subset of intervals $Y \subseteq X$ *covers* X if the union of all intervals in Y is equal to the union of all intervals in X. The *size* of a cover is just the number of intervals.

   Describe and analyze an efficient algorithm to compute the smallest cover of X. Assume that your input consists of two arrays $L[1 \ldots n]$ and $R[1 \ldots n]$, representing the left and right endpoints of the intervals in X. You should also provide the proof of correctness for your proposed algorithm.



A set of intervals, with a cover (shaded) of size 7.

3. *Greedy: Scheduling unit-time tasks with deadlines.*
   Let $S = \{1, 2, \ldots, n\}$ be a finite set of n unit-time tasks, i.e., jobs that require exactly one unit of time to complete. The tasks should be scheduled on a single processor, which means that the ith task in the schedule begins at time $i-1$ and completes at time i. Each task i is supposed to finish by its deadline $1 \leq d_i \leq n$ and a penalty $w_i \geq 0$ is incurred if task i is not finished by time d_i. No penalty is incurred if a task finishes by its deadline. The goal is to find a schedule for S that minimizes the total penalty incurred by missed deadlines.

Consider the following algorithm. Let all n time slots be initially empty, where time slot i is the unit-length slot of time that finishes at time i. We consider the tasks in order of monotonically decreasing penalty. When considering task j, if there exists a time slot at or before j's deadline d_j that is still empty, assign task j to the latest such slot. If there is no such slot, assign task j to the latest of the as yet unfilled slots.

Prove that this algorithm always gives an optimal answer.

4. *MST: Feedback Edge Set.*
A feedback edge set of an undirected graph G is a subset F of the edges such that every cycle in G contains at least one edge in F. In other words, removing every edge in F makes the graph G acyclic.

   Describe and analyze an efficient algorithm to compute the minimum-weight feedback edge set of a given edge-weighted graph.

5. *MST: Second Smallest Spanning Tree.*
Assume that we are given a graph G where all the edge-weights are distinct. The second smallest spanning tree of a given graph G is the spanning tree of G with the smallest total weight except for the minimum spanning tree (MST).

   - If all edge-weights are distinct, is the second smallest spanning tree also unique? Either prove or give a counterexample.
   - Let T be the MST of G. Prove that G contains edges $e \in T$ and $e' \notin T$ such that $T - e + e'$ is a second smallest spanning tree.
   - Describe and analyze an efficient algorithm to find a second smallest spanning tree of G.

Recommended practice problems: (not for submission)
Stable Matching:
– Kleinberg-Tardos: Problems 1, 2, 3, 4, 5 from Chapter 1;
– Erickson Chapter 4 [1]: Problems 9, 10, 11, 12.
Greedy Algorithms:
– CLRS (3rd ed): Problems 16.1-4, 16.2-1, 16.2-3, 16.2-4.
– Erickson Chapter 4: Problems 1, 2, 3, 4, 5.
Minimum Spanning Trees:
– Erickson Chapter 7 [2]: Problems 1-9.
– CLRS (3rd ed): Exercises 23.1, 23.2.

[1] `https://jeffe.cs.illinois.edu/teaching/algorithms/book/04-greedy.pdf`
[2] `https://jeffe.cs.illinois.edu/teaching/algorithms/book/07-mst.pdf`