CS3230: Design and Analysis of Algorithms (Fall 2014)

Tutorial Set #9

[For discussion during Week 11]

S-Problems are due (outside Prof. Leong's office): Friday, 24-Oct, before noon.

OUT: 20-Oct-2014 **Tutorials:** Tue & Wed, 28, 29 Oct 2014

IMPORTANT: Read "Remarks about Homework".

Submit solutions to S-Problem(s) by deadline given above.
Prepare your answers to all the D-Problems in every tutorial set.

When preparing to present your answers,
- Think of a CLEAR EXPLANATION
- Illustrate with a good worked example;
- Describe the main ideas,
- Can you sketch why the solution works;
- Give analysis of running time, if appropriate
- Can you think of other (perhaps simpler) solutions?

Helpful Hints Series: Understand definitions clearly.

Routine Practice Problems *-- do not turn these in -- but make sure you know how to do them.*

R1. Sorting by increasing sizes of intervals works in greedy algorithm for Interval Scheduling. True or False?

R2. Dijkstra's algorithm works well with negative edge weights? True of False?

R3. Dijkstra's algorithm can be used to produce shortest paths between all pairs of vertices? True of False?

R4. What is the data structure used by Dijkstra's algorithm for efficient implementation?

S-Problems: (To do and submit by due date given in page 1)
Solve this S-problem(s) and submit for grading.

IMPORTANT: Write your NAME, Matric No, Tutorial Group in your Answer Sheet.

S1. [Greedy gas filling]

Mr X is traveling by car on an expressway. Suppose there are several gas (petrol) stations on the way: at distances $0 = d_0 \leq d_1 \leq d_2 \leq \cdots \leq d_n$ from the starting point d_0. Mr X's car, when full, can travel a distance $D \geq max_i\{d_{i+1} - d_i\}$. Mr X wants to minimize the number of stops he makes to fill gas.

- Give idea of a greedy algorithm that returns the minimum number of stops for Mr X.
- Give pseudocode of your algorithm which takes as input $(d_1, d_2, \ldots, d_n, D)$ and outputs the minimum number of stops for Mr. X .
- Give a proof that your algorithm is correct.
- What is the running time of your algorithm?

D-Problems: Solve these D-problems and prepare to discuss them in tutorial class. You may be called upon to present your solution *or your best attempt at a solution*. Your solution presentation does NOT need to be fully correct, given your best attempt. The TA will help clarify and correct any issues or errors.

D1. [Interval partitioning]

You are given n time intervals (s_1, t_1) , (s_2, t_2) , …, (s_n, t_n) . You need to partition these intervals into minimum number of subsets such that any two intervals in the same subset are non-overlapping.

- Give idea of a greedy algorithm to output minimum number of such subsets.
- Give pseudocode of your algorithm.
- Give a proof that your algorithm is correct.
- What is the running time of your algorithm?

D2. [Lateness minimization]

You are given n tasks where task i, requires (contiguous) time t_i and has a deadline d_i . You need to schedule these tasks on a single server (only one task is active at a time) such that the maximum lateness

$$max_i\{0, f_i - d_i\}$$

is minimized, where f_i is the finishing time of task i in your schedule.

- Give idea of a greedy algorithm to output an optimal schedule with minimum maximum lateness.
- Give pseudocode of your algorithm.
- Give a proof that your algorithm is correct.
- What is the running time of your algorithm?

D3. [Greedy coin change]

You have coins of denominations $1, 5, 10, 25, 100$ dollars. You are supposed to find the minimum number of coins that add to n dollars.

- Give idea of a greedy algorithm that takes as input n and outputs the minimum number of coins that add to n.
- Give pseudocode of your algorithm.
- Give a proof that your algorithm is correct.
- What is the running time of your algorithm?

Advanced Problems – Try these for challenge and fun. There is no deadline for A-problems. *Turn in your attempts DIRECTLY to Prof. Leong. Do not combine it with your HW solutions.*)

A1. [Greedy coin change]

Prove that the greedy strategy for coin-change is optimal for denominations $d_1 > d_2 > \ldots > d_n = 1$ (for every amount A) if d_i divides d_{i-1} for i = 2, 3, …, n.