Purdue University Hambrusch
CS 381 Spring 2015

Assignment 5

Due: Tuesday, March 31, 2015 (hand in before class)

1) (15 pts) The Activity Selection problem selects k of n tasks so that the k tasks can be scheduled on one resource without conflict and k is as large as possible. The algorithm seen in class is greedy and considers tasks by increasing finishing time. Assume there are two resources. The goal is to select the largest number of tasks, assign each task to either resource 1 or resource 2, and have no conflicts on each resource.

An ECE student who has not taken CS 381 looks at the activity selection algorithm in the textbook and suggests the following solution to the two resource problem: Run the one resource activity selection algorithm on the n tasks. It selects k_1 tasks to schedule on resource 1. Then, run the activity selection algorithm on the remaining $n - k_1$ tasks. The algorithm selects k_2 tasks. Schedule these k_2 tasks on resource 2.

A CS student who has not taken 381 looks at the activity selection algorithm and suggests an algorithm that generalizes the earliest-finish-time-first approach: One step of the algorithm selects the next task not yet assigned to a resource or discarded that has the smallest finishing time among all remaining tasks. If the task can be assigned to resource 1, assign it to resource 1. If the task cannot be assigned to resource 1, but can be assigned to resource 2, assign it to resource 2. Otherwise, discard the task. The next step follows (until all tasks have been considered).

1. Does the algorithm suggested by the ECE student maximize the total number of tasks selected? If so, give a proof. If not, give a counter example.

2. Does the algorithm suggested by the CS student maximize the total number of tasks selected? If so, give a proof. If not, give a counter example.

3. State and explain the running time for each algorithm when the n tasks are given sorted by finishing time.

2) (10 pts.) A conference facility has scheduled n events into k conference rooms, $1 \leq k \leq n$. Each event is represented by a triple (s_i, f_i, c_i), where s_i is the start time, f_i is the finish time and c_i is the assigned conference room for event i. You can assume that all entries are integers and the conference rooms are numbered from 1 to k. The n triples are given in arbitrary order.

The administration of the conference facility needs to run security training sessions for all scheduled events. A training session takes one tenth of a time unit and a security session running during the time interval t to $t + 1$ covers all participants in conference rooms at that time. The goal is to minimize the number of security training sessions and to have every event be able to participate in one training session. Note that if an event could participate in more than one training session, the event organizers decide.

Describe and analyze an efficient algorithm for solving this problem. The output should be the unit time slots the training sessions are to be scheduled. Argue the correctness of your algorithm (i.e., that is determines the minimum number of security sessions covering all events).

3) (15 pts.) Design a data structure D to manage transactions consisting of pairs (date, charge). Every date has only one charge. The following operations are to be supported by the data structure D:

- insert(d, c): insert a transaction of date d with a charge of c (you can assume date d is not present in D)
- report(d): return the charge associated with date d (you can assume date d is present in D)
- delete(d): delete the transaction of date d (you can assume date d is present in D)
- update(d, c'): update the charge associated with date d to c' (it can increase or decrease; you can assume date d is present in D)
- max_charge(d_1, d_2): return the highest charge made between (and including) dates d_1 and d_2. If there is more than one charge of the same size, return the most recent one; d_1 and d_2 do not need to be dates in D.
- sum_charge(d_1, d_2): return the sum of the charges made between (and including) dates d_1 and d_2; d_1 and d_2 do not need to be dates in D.

Use red-black trees to support each operation in $O(\log n)$ time when D contains n entries. State clearly what entries are used as the keys and what augmented entries are maintained and how. Describe and analyze each operation.

You do not need to elaborate on operations corresponding to queries covered in the lecture except for any changes needed. For those not covered in class and especially the ones using augmented entries, describe clearly how they are executed and give a complete analysis.

4) (10 pts.) $G = (V, E)$ is an undirected, weighted, connected graph having n edges and n vertices. It is represented in the form of an adjacency list (see 22.2 for details). Graph G contains exactly one cycle (see B.5 for details on properties of trees).

Describe an $O(n)$ time algorithm identifying an edge of maximum cost in this cycle. If more than one edge has maximum weight, all edges having this weight should be reported.