Qualifying Examination

Theoretical Computer Science

Wednesday, March 13, 2013

Part I: Algorithms

Instructions:

1. This is a closed book exam.
2. The exam has four problems worth 25 points each. Read all the problems carefully to see the order in which you want to tackle them. You have all day (9am–5pm) to solve the problems.
3. Write clearly and concisely. You may appeal to some standard algorithms/facts from text books unless the problem explicitly asks for a proof of that fact or the details of that algorithm.
4. If you cannot solve a problem, to get partial credit write down your main idea/approach in a clear and concise way. For example you can obtain a solution assuming a clearly stated lemma that you believe should be true but cannot prove during the exam. However, please do not write down a laundry list of half baked ideas just to get partial credit.

May the force be with you.

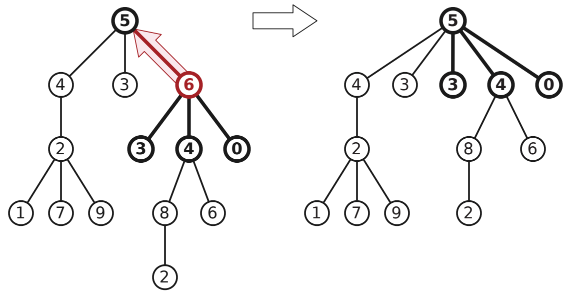
Problem 1:

Consider a finite metric space defined over n points; that is, a complete undirected graph $G = (V, E)$ with weights on the edges complying the triangle inequality. We are interested in designing a subgraph that would be resistant to attacks of removing some of its vertices, and would still have certain desired properties even such attacks.

(A) Prove that there is always a subgraph $H = (V, E')$ of G of weight $O(w(\mathsf{MST})f(n, k))$, such that H remains connected even if we delete any set of k of its vertices. Here $w(\mathsf{MST})$ is the weight of the minimum spanning tree of G. Your purpose in this question is to find the best possible $f(n, k)$. Naturally, the smaller $f(n, k)$ is, the better your solution is. Here, $f(n, k) = O(k^2)$ is possible.

(B) You are given a black box B, such that given any weighted complete graph Z (which complies with the triangle inequality), it outputs a subgraph $H \subseteq Z$, of weight $O(w(\mathsf{MST}(Z))$, such that the shortest path distance in H is at most twice longer than the shortest path distance in Z for any pair of vertices (such a graph is a 2-*spanner*). You can assume B works in polynomial time.
Given a finite metric space G as above, prove that one can compute in polynomial time (using the black box B) a subgraph of total weight $O(w(\mathsf{MST})g(n, k))$. As before, H needs to be a 2-spanner even if we delete any k of its vertices, for any pair of the remaining vertices.
Again, you need to find a scheme with $g(n, k)$ being as small as possible as a function of k and n. Doing $g(n, k) = k^{O(1)} \log^{O(1)} n$ is possible here.
Hint: Randomized construction algorithm might be easier to analyze here.

Problem 2: Let T be a rooted tree whose vertices are labeled with integers. To *contract* the edge from any node v to its parent p, we make p the new parent of all children of v and then delete v. A *minor* of T is any labeled rooted tree obtained by contracting one or more edges of T.



Contracting an edge in a rooted tree

A rooted tree is *heap-ordered* if each node has a smaller label than its children. Describe an efficient algorithm to find the *largest heap-ordered minor* of a given tree T.

Problem 3: Let $G = (V, E)$ be a directed graph and let s, t be distinct nodes. Each edge $e \in E$ has a non-negative integer cost $c(e)$ and a non-negative integer capacity $u(e)$. In the min-cost s-t flow problem the goal is to find an s-t-maximum flow $f : E \to \mathbb{R}_+$ of minimum total cost where the cost of the flow is $\sum_{e \in E} c(e)f(e)$.

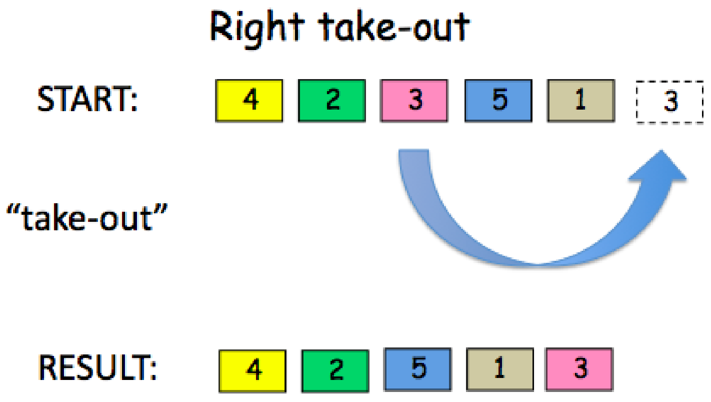
(A) Prove that a max-flow f is a min-cost flow if and only if there is no negative length cycle in the residual graph G_f.

(B) Using the previous part prove that there is always an integer valued max-flow of minimum cost when the capacities are integer valued.

Problem 4:

A cigar-box juggler typically holds a row of three or more cigar-boxes between his/her hands, and manipulates them by moving boxes around. The two end boxes are grasped by the juggler's hands, and the other boxes are supported by squeezing the two end boxes together. For example, the following picture shows a cigar-box juggler holding three cigar-boxes.

A typical move is a "take-out" in which the juggler grabs a box from somewhere in the middle of a line of boxes, and puts it at either the right end (right-handed-take-out) or left end (left-handed take-out). The following picture shows the results of a right-handed-take-out performed on box number 3.



A left-handed-take-out would result with the sequence of boxes 3-4-2-5-1.

The *cigar box sorting problem* is the following: Given an arbitrary permutation of boxes, using only take-outs, sort the boxes into increasing order.

(A) Suppose only right-handed take-outs are allowed. Give the best upper and lower bounds on the number of (right-handed) take-outs needed in the worst case to sort n cigar-boxes.

(B) Suppose both right-handed and left-handed take-outs are allowed. Give the best upper and lower bounds on the number of take-outs needed in the worst case to sort n cigar-boxes.