CS 473: Fundamental Algorithms, Fall 2011

Homework 4 (due Monday, 23:55:00, October 3, 2011)

Collaboration Policy & submission guidelines: See homework 1.
Each student individually have to also do **quiz 4** online.

Version: **1.2**

1. (30 PTS.) Saying goodbye to Bellman-Ford.

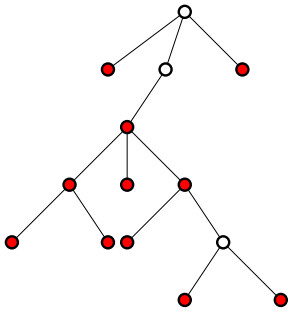
   You are given a directed graph G with (possibly negative) weights on the edges, and a source vertex s.

   (A) (20 PTS.) Show how to modify Bellman-Ford so that it outputs a negative cycle it had found in the graph reachable from the source s. **Prove** that your algorithm indeed outputs a negative cycle in the graph.

   (B) (10 PTS.) Describe an algorithm that computes for all the vertices in the given graph their distance from s.
   Notice, that your algorithm needs to correctly handle vertices in G whose distance from s is $-\infty$ (there is a walk from s to such a vertex that includes a negative cycle).
   For full credit, the running time of your algorithm must be $O(mn)$. (You do not have to use (A) to do this part.)

2. (40 PTS.) Packing trees.

   You are given a rooted undirected tree T with n vertices (let r denote the root). (The tree is not necessarily a binary tree.) Every vertex v of T has a positive weight $w(v)$ associated with it. A path is ***monotone*** if the distance of its vertices from the root strictly increases.
   Describe an algorithm, as fast as possible, that picks a subset of the vertices of the graph, such that no monotone path of length 5 contains more than three vertices in the set picked. Furthermore, the total weight of the computed set of vertices should be largest among all such sets. The figure on the right shows one legal solution picking 11 out of the 14 vertices of this tree (it is of course not necessarily the heaviest solution possible).

3. (30 PTS.) Some data-structure magic.

   (A) (10 PTS.) Consider a data-structure for implementing balanced binary tree (say AVL-tree[1]). We would like to modify it such that it supports inserting pairs (x, y). Here x is the key, and y is some associated value. Describe how to modify such a data-structure, such that it allows insertions, deletions and queries. The queries supported are:

   (I) Given x', return the pair (x, y) stored in the tree with maximum value of x, such that $x \leq x'$,

[1] If you do not know what AVL tree is – you need to learn it on your own – this is part of the prerequisites for the course.

(II) Given a query interval $[\alpha, \beta]$ it returns the numbers of pairs (x, y) stored in the data-structure such that $x \in [\alpha, \beta]$. Formally, it returns $\sum_{(x,y)\in\mathcal{D}, x\in[\alpha,\beta]} 1$, where $\mathcal{D}$ is the set of pairs currently stored in the data-structure.

(III) Given a query interval $[\alpha, \beta]$ it returns $\sum_{(x,y)\in\mathcal{D}, x\in[\alpha,\beta]} y$.

(IV) Given a query interval $[\alpha, \beta]$ it returns $\max_{(x,y)\in\mathcal{D}, x\in[\alpha,\beta]} y$.

All these operations should take $O(\log n)$ time.

Hint: For the modification, figure out what you need to store in each node of the tree, and describe how to maintain this information for rotations in this tree (and other situations where updates happen). As for the queries, consider the search paths in the tree from the root to the values α and β. (Try to keep your solution short – describe only the necessary modifications.)

(B) (10 PTS.) You are given a sequence $x_1, x_2, \ldots, x_n$ of n numbers. You had already computed pairs $(x_1, y_1), \ldots, (x_i, y_i)$, where y_k is the length of the longest increasing subsequence in the given sequence that ends in x_k (and includes it). Furthermore, you have stored all these pairs in the data-structure $\mathcal{D}$ of (A). Describe how to compute y_{i+1} in logarithmic time given $\mathcal{D}$ and using (A).
As a concrete example, if the sequence is $17, 1, 7, 5, 6, 4, 2, 8, 3$ then we will have:

$x_1 = 17$	$x_2 = 1$	$x_3 = 7$	$x_4 = 5$	$x_5 = 6$	$x_6 = 4$	$x_7 = 2$	$x_8 = 8$	$x_9 = 3$
$y_1 = 1$	$y_2 = 1$	$y_3 = 2$	$y_4 = 2$	$y_5 = 3$	$y_6 = 2$	$y_7 = 2$	$y_8 = 4$	$y_9 =???$

(C) (10 PTS.) Given a sequence $x_1, \ldots, x_n$, describe how to compute the longest increasing subsequence in this sequence in $O(n \log n)$ time, using (A) and (B).