TEL AVIV UNIVERSITY
Department of Computer Science
0368.4281 – Advanced topics in algorithms
Spring Semester, 2013/2014

Homework 3, May 27, 2014

Due on June 10. Please submit a pdf electronically.

1. Show how to augment the distance oracle of Thorup and Zwick so that given two vertices u and v we can also return a path of length at most $(2k-1)\delta(u,v)$ between u and v. The space requirement of the data structure should stay the same and the query time should be the same as for finding the distance plus $O(1)$ per edge reported on the path.

2. Consider the distance oracles of Thorup and Zwick presented in class. Given a pair of vertices u and v let i be the least index such that $p_i(v) \in B(u)$. (Recall that $B(u)$ is the bunch of u and $p_i(v)$ is the closest vertex to v among the centers of level i denoted by A_i in class.) Prove that $\delta(u, p_i(v)) + \delta(v, p_i(v)) \leq (4i + 1)\delta(u, v)$.

3. Let P be a pattern of length m and let T be a text of length n. Describe an algorithm that finds for each position i of T the longest substring of P that matches the text starting at position i. In other words, for each i find the maximum k for which there exists an ℓ such that $T[i+j] = P[\ell+j]$ for $j = 0, 1, \ldots, k-1$. Prove correctness of your algorithm and analyze its running time and space requirements.

4. A *cartesian tree* is a binary tree defined for a sequence of distinct integers $i_1, \ldots, i_n$ recursively as follows. The tree of an empty sequence is empty. Otherwise, let i_j be the smallest integer in the sequence. The root of the tree is a node containing i_j. The left child of the root is the root of a cartesian tree defined for the subsequence containing the elements preceding i_j in the sequence and the right child of the root is the root of a cartesian tree defined for the subsequence of elements following i_j in the sequence.

a) Find an algorithm to construct a cartesian tree for a given sequence, prove its correctness and analyze its running time.

b) In class we showed a solution to the range minima problem in an array (preprocess an array such that you can find the minimum in a query interval fast) in $O(1)$ query time and $O(n)$ space. The solution was specific to instances in which the difference between consecutive numbers in the array was ± 1. Use a cartesian tree to give a general solution to this range minima problem that works for any array. Analyze your data structure.

5. Given a string s, $|s| = n$, the suffix array, SA, of s, is a permutation of $\{1, 2, \ldots, n\}$ such that $SA[j] = i$ if and only if the suffix of s starting with the character i, $(i = 1, \ldots, n)$ is the j^{th} when we order the suffixes lexicographically. We add a special character \$ to each suffix which is smaller than any other character so that the lexicographic order of the suffixes is well defined.

(a) Given a permutation π of $1, 2, \ldots, n$, is there always a string s of length n such that π is the suffix array of s ? Prove your answer.
(b) Below are three suffix arrays. For each of these suffix arrays find a string s of length n, over the smallest possible alphabet Σ, such that the correponding array is a suffix array of s. Prove that there is no string s' over a smaller alphabet such that the suffix array is a suffix array of s'.

1)

n	n-1	$\cdots$	2	1

2)

1	2	$\cdots$	n-1	n

3) Assume n is even:

n	1	n-1	2	n-3	3	$\cdots$	$n/2+1$	$n/2$