TEL AVIV UNIVERSITY
Department of Computer Science
0368.4281 – Advanced topics in data structures
Fall Semester, 2006/2007

Homework 4, Dec. 31, 2006

Due on Monday Jan 14. Don't forget to keep a copy of the homework.

1. We perform a sequence of m operations on an (initially empty) heterogenous 2-4 finger search trees (i.e. a finger search tree without level links). Out of these m operations k are concatenations where the ith concatenation concatenates two trees such that the smaller among them contains n_i elements. The other $m-k$ operations are find, insert, delete, and split. The j-th operation among these operations is performed on an element at distance d_j from the beginning of a list containing n_j elements.
a) Prove that using the implementations of the operations as described in class one can perform the sequence in

$$O(\sum_{i=1}^{k} \log(n_i) + \sum_{j=1}^{m-k} \log(\min\{d_j, n_j - d_j\}))$$

time. (You can base your answer on previous homework just state exactly what you use!)

b) Improve the bound in the previous question to

$$O(k + \sum_{j=1}^{m-k} \log(\min\{d_j, n_j - d_j\}))$$

time.

2. Describe an implementation of fully persistent heaps that supports the operations *find-min*, *insert*, and *delete-min*. Make your implementation as efficient as you can. What are the time and space complexities of your data structure ?

3. Use persistent search trees to solve the following problem. Given a set S of n points in the plane, preprocess them to build a data structures which given a triple x_1, x_2, y_1 allows to report efficiently all the points $(x, y) \in S$ such that $x_1 \leq x \leq x_2$ and $y \geq y_1$. Analyze the space, preprocessing time, and query time of your data structure.

4. Analyze the Jordan sorting algorithm given in class assuming that the data structure used to represent the list of children of a node in the upper and lower trees is a doubly linked circular list.

5. Let P be a triangulated simple polygon.
(a) Describe an algorithm that given P and a vertex x of P, finds the segment on every edge of P which is visible from x. That is, for an edge $e \in P$ the required segment $s \subseteq e$ consists of all points

$y \in e$ such that the straight line connecting y and x lies entirely inside P. (It is not hard to prove that this set of points indeed is a segment.)
(b) Describe an algorithm that given P and an edge e of P, finds the segment on every edge of P which is visible from some point of e. That is, for an edge $e' \in P$ the required segment $s \subseteq e'$ consists of all points $y \in e'$ for which there exists a point $x \in e$, such that the straight line connecting y and x lies entirely inside P.

(Hint: For both problems there is a fairly straightforward linear time algorithm based on the shortest path algorithm given in class. There may be other methods as well. Give whatever algorithm you find even if it is not linear and analyse it.)