TEL AVIV UNIVERSITY
Department of Computer Science
0368.4281 – Advanced topics in data structures
Spring Semester, 2002/2003

Homework 3, April 18, 2003

Due on Monday May 12. Don't forget to keep a copy of the homework.

1. Consider the dynamic graph connectivity problem **on a forest** where we allow only two operations:
a) *connected*(v, w) that returns "yes" if and only if v and w are in the same connected component, and
b) *delete*(v, w) that deletes the edge (v, w) from the forest.
Assume we perform m such operations on a forest with n nodes. Describe an efficient (in an amortized sense) algorithm for the problem. (Note that from the material we learned in class easily follows a solution that runs in $O(m \log n)$ time, but better solutions exist in particular if $m >> n$.)

2. 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.

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.

3. Consider the Jordan sorting algorithm presented in class. Lets call a pair in the lower tree or the upper tree together with all its children a *family*.
a) Prove the following observation:
Suppose we have already traversed the points $x_1, \ldots, x_i$. Let F be a family so far constructed containing the pair $\{x_{i-1}, x_i\}$. Let $x_j \neq x_{i-1}$ be the point among the points in pairs of F such that x_i and x_j are adjacent (assume that there is such a point), i.e., no point in a pair of F lies

between x_i and x_j. Let R be the region bounded by the parts of the curve connecting the pairs in F together with the segments of the x-axis interconnecting them. Let x_k, $k > i$, be the point where the curve next enters R. Then either x_k lies between x_i and x_j or the parent pair in F is $\{-\infty, \infty\}$ and x_i and x_k bracket all other finite points in pairs of F. (Use the fact that the curve is a Jordan curve)

b) Show how to use the observation above to simplify the data structures we used to implement the algorithm.

4. Consider the move to root heuristic on a tree containing the integers 1 to n in symmetric order (i.e. for every node v all items in the left subtree are smaller than the item at v, and the item at v is smaller than all items at the right subtree). Recall that in the move to root heuristic, after you access an item you perform rotations bottom up on the edge between the accessed item to its parent until the item is at the root. Assume $i < j$.
a) Prove that i is an ancestor of j if the most recent request for i has occured since the most recent request for any of $i+1, \ldots, j$. Similarly, j is an ancestor of i if the most recent request for j has occured since the most recent one for any of $i, \ldots, j-1$.
b) Prove that any tree T containing these items in symmetric order can be produced from any other tree containing these items in symmetric order by a sequence of at most n requests under the move to root heuristics.
c) For any m sufficiently large show a sequence of accesses such that serving it with the move to root heuristics takes $\Omega(n)$ time on average per access. (The initial tree is arbitrary.)