TEL AVIV UNIVERSITY
Department of Computer Science
0368.4281 – Advanced topics in data structures
Spring Semester, 2000/2001

Homework 1, March 21, 2001

Heaps

1. Define an alternative implementation for the `delete(x,h)` operation on binomial heaps. In the heap after the delete operation each item (except the deleted one) should be in the same node that contained it before the delete. The delete should run in $O(\log n)$ time.

2. For any positive integer n, give a sequence of Fibonacci heaps operations that creates a Fibonacci heap consisting of just one tree that is a linear chain of n nodes (make your sequence as short as you can).

3. Consider an implementation of Fibonacci heaps without cascading cuts (all other details are as shown in class, the only difference is that delete and decrease-key just cut the subtree and do not continue with cascading cuts). For any n show a sequence of n operations that takes $\Omega(f(n))$ time for as large $f(n)$ as you can.

4. Describe a heap data structure that supports find-min in $O(1)$ time, insert and meld in $o(\log(n))$ time (recall that little "o" means asymptotically faster than $\log n$), and as fast Delete-min as you can. All time bounds are on the worst case. You'd get extra credit if you do meld and insert in $O(1)$ time and delete-min in $O(\log n)$.

Redundant counters

5. Augment the redundant binary counters shown in class to support both increment and decrement in $O(1)$ worst case time. Describe the algorithm that decrements the counter and establish its correctness.