CS 423: Theory of Algorithms
Spring, 1999
Tarjan

Problem Set 4
Due: Thursday, April 1, 1999
Half credit if turned on Tuesday, 6

Problem Set No. 4

1. (One-Way Streets) Let G be an undirected graph. We wish to assign a direction to every edge of G in such a way that, in the resulting directed graph, every vertex is reachable from every other. That is, the directed graph is strongly connected. Prove that this is possible if and only if G is connected and has no bridges (edges whose removal disconnects the graph). Describe a linear-time algorithm to test these conditions and to direct the edges to produce strong connectivity if these conditions are met.

2. (a) Given an unordered set of n integers, the problem is to find in this set an arithmetic sequence of maximum length. An arithmetic sequence is a sequence of numbers of the form $a, a+b, a+2b, a+3b, \ldots, a+kb$; its length is $k+1$. Describe an $\mathrm{O(n^2)}$-time algorithm to find such a sequence of maximum length.

   (b) **(extra credit)** Describe an algorithm for this problem that is asymptotically faster than $\mathrm{O(n^2)}$.

3. A directed graph is called *semiconnected* if, for any two vertices x and y, either x is reachable from y or y from x (or both). Describe a linear-time algorithm to test whether a given directed graph is semiconnected.

4. ($1\frac{1}{2}$-dimensional search trees) Consider data that consists of *pairs* of real numbers (x, y). We want to construct a kind of binary search tree that is *simultaneously* heap-ordered on x-values and symmetrically ordered on y-values. Thus a pair (x, y) of minimum x-value is stored in the root, all pairs in the left subtree of the root have smaller 2^{nd} coordinate than y, all pairs in the right subtree have larger 2^{nd} coordinate than y, and these properties hold recursively in each subtree. Such a structure is called a *treap*.

   (a) Show how to insert or delete a pair in a treap in time proportional to the depth of the tree.

   (b) Given a triple (a, b, c), show how to search in a treap for all pairs (x, y) such that $x \leq a$ and $b \leq y \leq c$ in time proportional to the depth of the tree plus the number of pairs that satisfy the constraints.

   (c) Treaps have the (serious) drawback that their depth can be linear in the number of nodes they contain. Show this by example.

   (d) **(extra credit)** Modify the treap data structure so that the maximum depth of an n-node tree is $\mathrm{O(\log n)}$, insertion or deletion of a pair takes $\mathrm{O}(\log n)$ time, and a range search as in (b) takes O(log n) time plus time proportional to the number of pairs that satisfy the query.

Hint: Store in each node (at most) two pairs: one with minumum x-value and another with y-value used to split on y between the left and right subtree. Use a red-black tree structure to maintain balance. To obtain an $O(\log n)$ time bound for insertions and deletions, use the fact that an ordinary insertion or deletion in a red-black tree takes $O(1)$ rotations.