CS 466/666, Spring 2007
Timothy Chan

Assignment 2 (due June 6 Wednesday noon)

Please read `http://www.student.cs.uwaterloo.ca/~cs466/policies.html` first for general instructions.

1. [*7 marks*] Suppose we change Fredman and Tarjan's invariant to allow at most 2 (instead of 1) children to be cut from a non-root node. Prove that any tree of size n that can be formed by the algorithm still has degree bounded by $O(\log n)$. [Note: no pseudocode is required. Hint: a recurrence satisfying $f_i \geq 2f_{i-c}$ has solution $f_i = \Omega(2^{i/c})\ldots$]

2. [*16 marks*] Give a data structure to support the following operations on an undirected graph G consisting of a disjoint collection of paths:

   - delete(e): delete the edge e from G.
   - query(u, w): return yes iff the two vertices u and w lie on the same path.

   Initially, G is a single path through n vertices.

   Your solution should achieve $O(1)$ *worst-case* time for query, and $O(\log n)$ amortized time for delete, under the assumption that the number of deletes in the sequence is at least $\Theta(n)$.

   [Hint: deletion of an edge breaks a path into two subpaths. Can you keep the running time bounded by the length of the *smaller* subpath? For the amortized analysis, use an argument similar to the one from class for the weighted union heuristic (don't use potentials).]

3. [*15 marks*] Give a data structure to support the following operations on a set S of intervals in one dimension:

   - insert(a, b): given two integers $a, b \in \{1, 2, \ldots, U\}$, insert the interval $[a, b]$ to S.
   - query(x): given integer $x \in \{1, 2, \ldots, U\}$, return yes iff x lies in the union of the intervals in S.

   (For example, if S contains the intervals $[1, 4]$, $[6, 10]$, $[8, 13]$, the union of S is $[1, 4] \cup [6, 13]$.)

   Your solution should have $O(\alpha(U))$ amortized insert and query time under the assumption that the number n of inserts in the sequence is at least $\Theta(U)$.

   [Hint: as you might guess, you should use a union-find data structure as a subroutine... Over a sequence of n inserts, how many calls to union does your algorithm make?]

4. [*12 marks*] In a popular form of logic puzzles, you land on an imaginary island where inhabitants are classified into three types: "knights", who always tell the truth; "knaves", who always lie; and "spies", who sometimes lie and sometimes tell the truth.

   Suppose there are n inhabitants, where 90% are known to be decent folks, i.e., knights. The remaining 10% are bad, i.e., knaves or spies. You want to know who the good/bad guys are, i.e., you want to determine the types of all n inhabitants. You are allowed to ask only questions of the form, "is person A a knight/knave/spy?", to another person B. (All the inhabitants know each other.) Obviously, if you can find a person who you know is a knight, the problem is solved after asking n additional questions.

(a) [*2 marks*] Give a (very) efficient Monte-Carlo algorithm that finds a knight. State the probability of error. [Hint: this is supposed to be easy!]

(b) [*10 marks*] Give a Las-Vegas algorithm that finds a knight by asking $O(n)$ expected number of questions. Analyze the constant factor in the big-Oh and make it smaller than 0.5. [Hint: use (a). How can you confirm whether a specific person is a knight by asking $O(n)$ questions?]

[Note: there is also a deterministic algorithm that requires $O(n)$ questions, but it's more complicated and has a larger constant factor.]