COS 423
Spring, 2009

Disjoint Set Union

Problem: Maintain a collection of disjoint sets.

Two operations: *find* the set containing a given element;
unite two sets into one (destructively).

Approach: "Canonical element" method: for each set, the algorithm maintains a canonical element (arbitrary but unique), holding any desired information about the set.

Two low-level operations:

find (x): given element x, return the canonical element of the set containing x;

link(x, y): given canonical elements x and y, destructively unite the sets containing them, and make x or y the canonical element of the new set. (Do nothing if $x = y$.)

Then *unite* can be implemented as follows:

unite (x, y) : unite the sets containing (arbitrary) elements x and y (if they differ)

unite (x, y) = link (*find* (x), *find* (y))

Tree-based implementation: the elements of each set form a tree, with each node pointing to its parent, via a "p" pointer. Each tree root points to itself.

Assume n singleton sets initially ($p(x) = x$ for every x initially); m total finds, interspersed with links; $m \geq n$.

To perform *find*, follow parent pointers to tree root. To perform compression after a find, make every node on the find path point directly to the root.

Linking by rank (rank is maximum length, in *edges*, of an uncompressed path from a descendant)

$r(x) = 0$ for every x initially.

To link x and y, make the smaller-ranked root point to the larger; in case of a tie, increase the rank of the new root by one.

Question: What is the total time for m finds interspersed with links?

Answer: $O(m\alpha(n))$, where

$A_0(x) = x + 1$ for $x \geq 1$

$A_{k+1}(x) = A_k^{x+1}(x)$ for $x \geq 1$ $(A_k^0(x) = x, A_k^{i+1}(x) = A_k(A_k^i(x)))$

$\alpha(n) =$ the smallest k such that $A_k(1) \geq n$

From these definitions, $A_1(x) = 2x+1, A_2(x) > 2^x \; A_3(x) > 2^{2^{\cdot^{\cdot^{2}}}} \}x+1$
and $\alpha(n)$ grows *very* slowly.

Exercise: Prove that $A_k(x)$ is an increasing function of both k and x.

To prove the $O(m\alpha(n))$ bound we use an amortized analysis.

Observe that the rank of a node x starts at 0, can increase but not decrease while x is a tree root, and remains constant once x is a nonroot. Observe also that $r(p(x)) > r(x)$. Once x has a parent, $r(x)$ is constant, but $r(p(x))$ can increase (but not decrease), either because $p(x)$ changes due to a compression or $r(p(x))$ changes due to a link. The maximum node rank is at most $n-1$. (Why?) (Actually, it is at most lgn, but we won't use this.)

We will define a potential function that assigns a non-negative integer potential of at most $\alpha(n)r(x)$ to each node x; the total potential is the sum of all the node potentials.

Any tree root x has potential $\alpha(n)r(x)$. (Thus the total initial potential is 0.) Let x be a nonroot with $r(x) \geq 1$. Define the *level* of x, denoted by $k(x)$, to be the largest k for which $r(p(x)) \geq A_k(r(x))$.

We have $A_0(r(x)) = r(x) + 1 \leq r(p(x))$ and $A_{\alpha(n)}(r(x)) \geq A_{\alpha(n)}(1) \geq n > n-1 \geq r(p(x))$. Thus $k(x)$ is well-defined and $0 \leq k(x) < \alpha(n)$. Furthermore, since $r(p(x))$ can never decrease, $k(x)$ can never decrease, only increase.

Define the *index* of x, denoted by $i(x)$, to be the largest i for which $r(p(x)) \geq A^i_{k(x)}(r(x))$.

We have $A^i_{k(x)}(r(x)) = A_{k(x)}(r(x)) \leq r(p(x))$ by the definition of $k(x)$, and $A^{r(x)+1}_{k(x)}(r(x)) = A_{k(x)+1}(r(x)) > r(p(x))$, by the definitions of A_k and $k(x)$. Thus $i(x)$ is well-defined and $1 \leq i(x) \leq r(x)$. Also, since $r(p(x))$ can never decrease, $i(x)$ cannot decrease unless $k(x)$ increases: while $k(x)$ remains constant, $i(x)$ can only increase or stay the same.

Now we are ready to define the potential of a node x.

$\phi(x) = \alpha(n)r(x)$ if x is a root or $r(x) = 0$

$\phi(x) = (\alpha(n) - k(x))r(x) - i(x)$ if x is a nonroot and $r(x) > 0$

We define the total potential Φ to be the sum over all nodes x of $\phi(x)$.

Let us show that $0 \le \phi(x) \le \alpha(n)r(x)$ for every node *x*. This is obvious if *x* is a root or $r(x) = 0$. Suppose *x* is a nonroot and $r(x) > 0$. Since $k(x) \le \alpha(n) - 1$ and $i(x) \le r(x)$, $\phi(x) \ge r(x) - i(x) \ge 0.$ Since $k(x) \ge 0$ and $i(x) \ge 1$, $\phi(x) \le \alpha(n)r(x) - 1.$

What remains is to show that the amortized cost of a link or find is $O(\alpha(n))$. First consider a link, say *link* (*x, y*). Without loss of generality suppose the link makes *y* the new root. The actual cost of the link is (order of) one. The potential of any node other than *y* can only decrease. (Exercise: show this.). The potential of *y* stays the same or increases by $\alpha(n)$, since $r(y)$ stays the same or increases by one. Thus the increase of Φ due to the link is at most $\alpha(n)$, and the amortized cost of the link is at most $\alpha(n) + 1$.

Consider a find with compression. The actual cost of the find is (order of) the number of nodes on the find path. No node can have its potential increase as a result of the find. (*Exercise:* prove this.) We shall show that if ℓ is the number of nodes on the find path, at least max $\{0, \ell - (\alpha(n) + 2)\}$ of these nodes have their potential decrease (by at least one) as a result of the compression. This implies that the amortized cost of the find is at most $\alpha(n) + 2$.

Specifically, let *x* be a node on the find path such that $r(x) > 0$ and *x* is followed on the find path by another nonroot node *y* such that $k(y) = k(x)$. All but at most $\alpha(n) + 2$ nodes on the find path satisfy this constraint; those that do not are the first node on the path (if it has rank zero), the last node on the path (the root), and the last node on the path of level *k*, for each possible *k* in the range $0 \le k < \alpha(n)$.

Let $k = k(x) = k(y)$. Before the compression, $r(p(x)) \ge A_k^{i(x)}(r(x)), r(p(y)) \ge A_k(r(y))$, and $r(y) \ge r(p(x))$. These inequalities imply $r(p(y)) \ge A_k(r(y)) \ge A_k(r(p(x)))$ $\ge A_k(A_k^{i(x)}(r(x))) = A_k^{i(x)+1}(r(x))$, which means that the compression causes $i(x)$ to increase or $k(x)$ to increase, in either case decreasing $\phi(x)$ by at least 1. (Exercise:prove this.)

Suppose now that instead of using path compression and linking by rank, we use path compression and naïve linking, in which we link *x* and *y* by making *x* the parent of *y*. The amortized analysis that gives $\alpha(n)$ per operation breaks down for two reasons. The first is that ranks (hence levels and indexes) are undefined. We can fix this by defining ranks as follows: an initial singleton node has rank 0; when a link of *x* and *y* is performed, making *x* the parent of *y*, we replace the rank of *x* by $\max\{r(x),\ r(y)+1\}$. Then many of the needed properties of ranks hold. Specifically, the rank of a node starts at 0, can increase but not decrease while *x* is a root, and remains constant once *x* is a non-root. Also, $r(p(x)) > r(x)$, and once *x* has a parent $r(x)$ remains fixed but $r(p(x))$ can only increase. Even with this definition of rank, the analysis breaks down, because the rank of a root can increase by up to $n-1$ during a link, and the amortized time for a link is no longer $\mathrm{O}(\alpha(n))$.

We can obtain an $\mathrm{O}(\log n)$ amortized time bound per operation in this case by changing the potential function, however. We define the *log-level* of a non-root node *x* to be $g(x) = \lg\lfloor r(p(x)) - r(x)\rfloor$. Then $0 \le g(x) \le \lg n.$ We define the potential of a node *x* to be 0 if it is a root and $\lg n - g(x)$ if it is a non-root. Then the initial potential is zero, and the potential is always non-negative, so the total time of an arbitrary sequence of operations is at most the sum of their amortized times.

Consider an operation $link(x, y).$ The actual time is O(1). The potential of every node except *x* either stays the same or decreases. The potential of *x* can increase by at most $\lg n$ (from 0 to $\lg n$). Thus the amortized time of a link is $\mathrm{O}(\lg n).$

Consider a find with compression. Let ℓ be the number of nodes on the find path. No node can have its potential increase as a result of the find. We shall show that at least $\max\{0, \ell - \lg n - 2\}$ nodes on the find path have their potential decrease by at least one as a result of the compression. This implies that the amortized time of the find is at most $\lg n + 2.$

Let *x* be a node on the find path such that *x* is followed on the find path by another non-root node *y* such that $g(y) = g(x).$ All but at most $\lg n + 2$ nodes on the find path (the last for each possible log-level and the root) satisfy this property. Let $g = g(x) = g(y).$ Before the compression, $r(p(x)) - r(x) \ge 2^g$ and $r(p(y)) - r(y) \ge 2^g.$ After the compression, the new parent of *x* has rank at least that of the old parent of *y*, which means that $r(p'(x)) - r(x) \ge 2^g + 2^g = 2^{g+1},$ where p' denotes the new parent, and the compression causes the log-level of *x* to increase by at least one, and hence its potential to decrease by at least one.