9. Disjoint set

2018年11月13日 17:09

Keep collection of disjoint sets $s_1, \dots, s_k$, each set has a representative $x_i \in s_i$
operations

$makeset(x)$ creates $\{x\}$, x does not belong to any other sets, or returns an error.
$findset(x)$ give the rep of a set that contains x, returns an error if x is not in any set
$union(x, y)$ if x, y in the same set, do nothing. in the different set, then remove $s_i, s_j, (x \in s_i, y \in s_j)$ and replace them with $s_i \cup s_j$
$link(x, y)$ same as union, but given precondition x, y must be different representatives.
Then, $union \coloneqq link(findset(x), findset(y))$

Define $T(m, n) \coloneqq max$ execute any set of m operations, n of which are $makeset$, starting from empty.
Can assume only $find, link, makeset$ will be performed.

Theorem there are at most $n - 1$ $link$ operations.
Since there are only n sets and $link$ decrease by 1, no new sets can be added other than $makeset$ (add 1).
At any time $1 \le \#\text{sets} = \#makeset - \#link$

Implementations
Example $\{1, \mathbf{3}, 7, 10\}, \{2, 4, 5, \mathbf{8}, 6\}$
The trees can be arbitrary, except the rep must be the root.
Each elements has

$value$
$parent$: $x.p$ pointer to its parent if x is not the root, $x.parent = x$ if x is the root

Consider $findset$ on a value x, we can use a direct access table or hash table, which assume to take expected constant time

$makeset(x)$: $O(1)$ create a new singleton tree of element x
$findset(x)$: $O(height)$

    if $x == x.p$
        return x
    else
        return $findset(x.p)$

$link(x, y)$: $O(1)$ take one root's parent pointer to the other

Issue height is not clear, may resulting to be a linked list. Hence $T(m, n) = \Theta(mn)$

Weighted union (by rank/height)
idea: augment each node with additional field rank = height, and always attract shorter tree to the taller one
new implementation

$makeset(x)$: $O(1)$
    $x.p = x$
    $x.rank = 0$

$link(x, y)$
    if $x.rank > y.rank$
        $y.p = x$
    else if $x.rank < y.rank$

$x.p = y$

else

$y.p = x$

$y.rank = y.rank + 1$

claim1 $x.rank = height(x)$

claim2 $if\ x.rep \in S_i. |S_i| > x.rank$

proof Induction on #links

Only way to create set is $makeset$, $|x| = 1 \geq 2^0$

Assume before $link$, $|s_i| \geq 2^{x.rank}$

Suppose $x.rank \neq y.rank$, then their rank does not change and the union increases size

Suppose $x.rank = y.rank$, then $\left|s_y'\right| = \left|s_x + s_y\right| \geq 2^{x.rank} + 2^{y.rank} = 2^{y.rank+1}$

claim3 $x.rank \leq floor(\log_2 n)$ is a corollary of claim2

Therefore, the amortized $T(m,n) = O(m \log n)$

Path compression

idea: when $findset$, after finding root, trace back and change the future tracing up tp directly to the root

new implementation

$findset(x)$

if $x.p == x$

return x

$y = findset(x.p)$

$x.p = y$

$return\ y$

claim1 $x.rank \geq height(x)$

lemma There are at most $\frac{n}{2^r}$ nodes of rank r

proof When $x.rank$ becomes r, mark the decendants of x

The only way a rank can change is $link(x,y), x.rank == y.rank$. Therefore, at least $2^r = |s_x|$nodes are marked

Every node is marked at most once since $x.rank$ won't decrease, and its child is only marked when $x.rank$ increases to r

Total number of times are mark nodes is at most n