Disjoint Sets

In mathematics, two sets are said to be disjoint sets if they have no element in common i.e. intersection is the empty set. Let for two sets A and B are disjoint if and only if $A \cap B = \phi$.

Implementation

Different optimizations Heuristics

- Weighted-union heuristic
  Suppose instead that each list also includes the length of the list and that we always append the shorter list onto the longer while union operation of sets. With this simple weighted-union heuristic, a single UNION operation can still take $\Omega(n)$ time if both sets have $\Omega(n)$ members. With a sequence of m MAKE-SET, UNION, and FIND-SET operations, n of which are MAKE-SET operations (or number of elements equals n), takes $O(m + n \lg n)$.

- Union by rank and path compression heuristic
  The first heuristic *union by rank*, is similar to the weighted-union heuristic. The obvious approach would be to make the root of the tree with fewer nodes point to the root of the tree with more nodes. Rather than explicitly keeping track of the size of the subtree rooted at each node(like in weighted-union), we shall for each node maintain a rank, which is an upper bound on the height of the node. In union by rank, we make the root with smaller rank point to the root with larger rank during a UNION operation.

  The second heuristic path compression, we use it during FIND-SET operations to make each node on the find path point directly to the root. Path compression does not change any ranks.

Disjoint operation API

1. makeSet
2. union
3. findSet

When we use both union by rank and path compression, the worst-case running time is $O(m\ \alpha(n))$, where $\alpha(n)$ is a very slowly growing function for m disjoint-set operations on n elements. In any conceivable application of a disjoint-set data structure, $\alpha(n) \leq 4$; thus, we can view the running time as linear in m in all practical situations.

Applications using Disjoint sets

- Represents network connectivity (or find connected components of a graph).
- Image Processing.
- In game algorithms.
- Kruskal's minimum spanning tree.
- Find cycle in undirected graphs.

References

- Disjoint Set Wiki
- YouTube tutorial