Splay tree amortized analysis

14 September 2020 20:05

Let T be a splay tree with 'n' keys.

Let v be a node of T.

Size $n(v)$ → number of nodes in the subtree rooted at 'v'.

∴ Size of an internal node is one more than the sum of the sizes of its two children.

Rank $r(v)$ of a node defined as

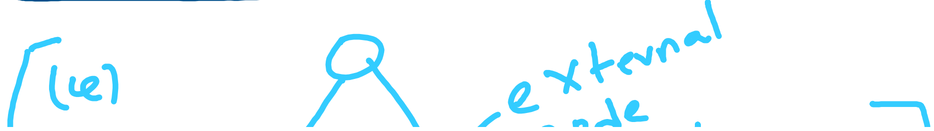
$$r(v) = \boxed{\log_2 n(v)}$$

Root T has maximum size $2n+1$

(left – n , right – n , +1)

and max rank $\log(2n+1)$

– Each external node has size 1 & rank 0.



We define the potential function $\phi_i = \sum_{v \in T_i} r_i(v)$

$r_1 v$
$r_2 v$
$r_3 v$

Now the amortized time Complexity A_i of splay step 'i' is defined as

after, before

$$A_i = t_i + \underbrace{\phi_i - \phi_{i-1}}$$

2

1

Where

t_i - actual cost of splay operation

ϕ_i - Value of potential function before splay

ϕ_{i-1} - Value of potential function after splay

Since the ranks of participating nodes u, Par(u), grand par(u) alone changes only for these nodes $\underbrace{\boxed{\phi_i - \phi_{i-1}}}$ is computed.

Also cost of splay rotation

(i.e) single rotation - 1

Double rotation - 2

Results

1) The amortized Complexity A_i of splay tree, if step i initiates

(i) Zig-Zig or Zag-Zag, then

$A_i < 3 \cdot (r_i(x) - r_{i-1}(x))$

(ii) Zig-Zag or Zag-Zig then

$A_i < 2 \cdot (r_i(x) - r_{i-1}(x))$

(iii) Zig or Zag then

$A_i < 1 + (r_i(x) - r_{i-1}(x))$

2) Amortized cost of search, insertion or deletion in a splay tree is $O(\log n)$ where 'n' is the size of the splay tree.

3) Let a sequence of 'm' operations on a splay tree, each a search, insertion or deletion starting

from an empty splay tree.
Let n_i be the number of keys after operation i and n be the total number of insertions.
The total running time of sequence of operations is

$$O\left(m + \sum_{i=1}^{m} \log n_i\right)$$

which is $\boxed{O(m \log n)}$

$\log n^m$

$\Rightarrow m \log n$