Page No.
Date

Rohit Kudache
1BM18CS083

LAB 5 - 2-3 Tree - ADS

Insertion

* Insert new leaf in appropriate place.
* Repeat until all nonleaf nodes have 2 or 3 children.
  - If there is a node with 4 children,

do → split the parent into two parent node, with 2 children each

  - If split the root

then do → add a new root

* Adjust search values along insertion path

Delete

* Delete x from 2-3 tree
  - let P - parent(x)
  - If x is root, remove x
  - else if P has 3 children
    then remove x
  - else (P has 2 children)
    - if P is the root
      then remove x and P
    - else (P not the root)

cases:

*let S be the other child of P
let L be the left sibling of P
let R be the right sibling of P
(L or R may not exist)

Pawit

if l has 3 children, replace x by one of l's children.

else if r has 3 children, replace x by one of r's children.

else

remove x

combine 'p' with l (or r)

make 's' a children l (or r)

rename p to x

recursively remove x

Insert Algorithm

insert (TwoThreeNode N, Comparable value)

locate leaf in which to put value (as prior)

if the leaf is a 2-node,

make it a 3-node

insert value as either small key or large key

if the leaf is a 3-node

insert value (temporarily) so node has 3 keys

small key, mid key, large key

split the node

split (TwoThreeNode N)

If N is the root

make mid key into a 2-node (as root (new))

make small key, large key into 2-nodes (children)

redistribute children

else N has a parent P → move midd key to P

make small key, large key into 2 nodes

redistribute children if necessary

Rewil

Delete

One node n (which need to be deleted) is found

case 1 : n has 3 node (children)

Remove the child with value k, the fix n.leftMax (small key), n.middleMax (mid key) n's ancestor's small key and mid key field if necessary.

case 2 : n has only 2 children

if n is the root of the tree, then remove the node containing k. Replace the root node the other child.

if n has a left or right sibling with 3node then
- remove the node containing k
- "Steal" one of the sibling's children.
- fix small key and mid key, and small key and mid key field n's Sibling and ancestor as needed.

if n's sibling(s) have only 2 children, then
- remove node containing k
- make n's remaining child a child of n's sibling.
- Small key and mid key fields of n's sibling as needed.
- remove n as child of it's parent.

```
insert (a, r) {
if (r consists of a single leaf l labeled b)
    create a new root r'
    create a new leaf v labeled a
    make l and v children of r' in proper order
    update L and M for r'
else
    set f to SEARCH (a, r)
    create a new leaf l labeled a
    if f has 2 children
        insert l into proper position
        update L and M.
    else
        create a transitory 4-node tree at f
        ADDCHILD (f)
}

ADDCHILD (v) {
    create new interior node v'
    move 2 rightmost children of v to v'
    if (v has no parent) // that is, if v is the root
        make new root r'
        make v the left child and v' the right child
        update L and M
    else
        let f be the parent of v
        make v' the child of f immediately to the
right of v
        if f now has 4 children.
            ADDCHILD (f)
        else
            update L and M
}
```


Recent

Page No.
Date

```
Delete (){
Let f be the parent of the node just deleted
while (f is an illegal interior node){
    if (f has no parent)
        make the single child of f the new node
        delete f
        set of to the root
    else
        let g be parent of f
        if (one of f's siblings is a 3-node)
        move one child from 3-node into f
        update K1 and K2 every where
        else
        give f's remaining child to one of f's siblings
        delete f
        update K1 and K2 every where
        set f to g
    }
}.
```

05

Rakesh.