Lecture 11, Tuesday 5/15/01

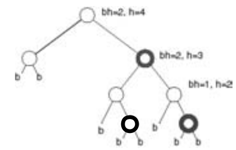
Red/Black trees

- One of the ways to approximately balance, i.e. depth O(lg n).
- Red/Black property:
  - » Node is either black (default) or red.
  - » Add NIL to leaves so "real nodes" have 2 children, color leaves black. ⇨
  - » x red children(x) are black.
  - » Every simple root-leaf path has same # of black nodes.
  - » Root is black (for convenience).
- One of the ways to approximately balance.
  - » We will show that above condition implies depth O(lg n).

95

Black height

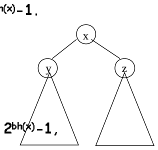
- bh(h) = # of black nodes ~~on x~~ leaf path, not counting x.



96

Balancing

- Observation: bh(root)≥h/2
- THM: h≤2lg(n+1)
  Proof:
  - » Claim: #nodes in subtree rooted at x ≥$2^{bh(x)}$-1.
  - » By induction on h(x). Base case is trivial.
  - » if y black, bh(y)=bh(x)-1
    otherwise, bh(y)=bh(x)
  - » Thus, bh(y) ≥bh(x)-1
  - » #nodes in x-tree at least ($2^{bh(x)-1}$-1)2+1= $2^{bh(x)}$-1, which proves the claim.
  - » But: bh(root) ≥ h/2, QED.



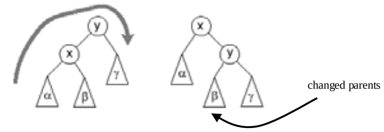
97

Maintaining the red/black trees

•Consider example where we can not recolor and have to rearrange.

•O(1) time

•B-tree property maint.

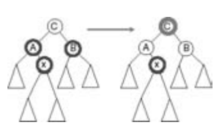


98

Insertion

- Insert into B-tree and color the new node red.
  - » B-tree properties satisfied.
  - » Only property that might break is if parent of new x is red.
- Case 1:
  both parent and its sibling are

  Problem was bumped-up to C.
  (no other possible problems
  besides between C and its
  parent)

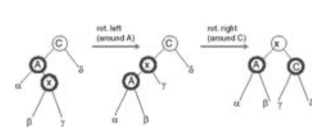


Notice that bh property not violated !

99

Insertion continued

- Case 2: parent red, parent's sibling black.



- Note that the problem disappeared.
- Need to consider symmetric cases.
- Deletion – read at home.

100