CS367 Lecture 21

Tuesday 22 July 2014

Announcements/Reminders:

- HW6 assigned

Last class:

- Associative Arrays/Maps
- Binary Search Trees
    - `delete` method (finish)
    - Complexities
- Balanced Search Trees (intro)

Today:

- Balanced Search Trees
    - Red-Black Trees

Balanced Search Trees

Goal

How?

Rotations

Types

Red-Black Trees (RBTs)

Idea

Properties:

- Every node is “colored” red or black
- Root property
- Red property
- Black property
- All leaves are null and black

Example:

Operations: `print, lookup,` **`insert, delete`**

Black-height of a node

Definition:

Immediate consequences of RBT properties:

Inserting into a red-black tree

Insert a value V into a red-black tree T WHILE _____

Trivial case: T is empty.

General case:

- Travel down and insert V as a leaf (like in a regular BST)
- Color the new node red
- What else?

Which property might be violated?

RBT Insertion (cont'd): V's parent P is red

Case 1: P's sibling S is black. In this case, **rotate**:

Case 2: P's sibling S is red. In this case, **recolor**:

Cascading Fixes

RBT insertion example

Starting with an empty RBT, insert 7, 14, 18, 23, 1, 11, 20, 29, 25, 27 (same values as before)

Complexity of RBT operations

`print:`

`lookup:`

`insert:`

`delete:`