LLRBs & Hashing

Exam Prep 08

Announcements

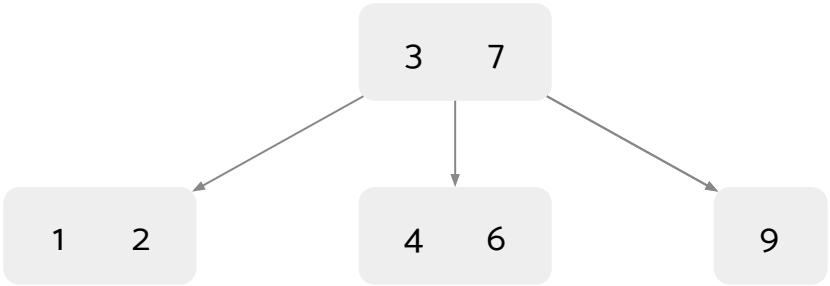
Week 8

- TBD

Content Review

B-Trees

B-Trees (also referred to as 2-3 Trees) are trees that serve a similar function to binary trees while ensuring a bushy structure.

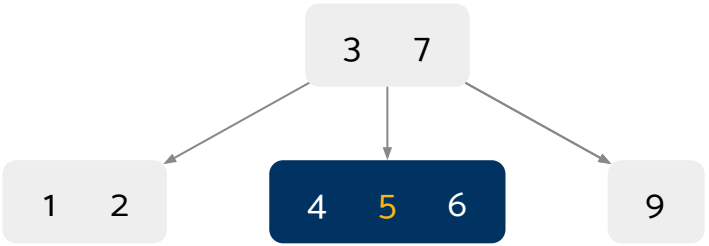


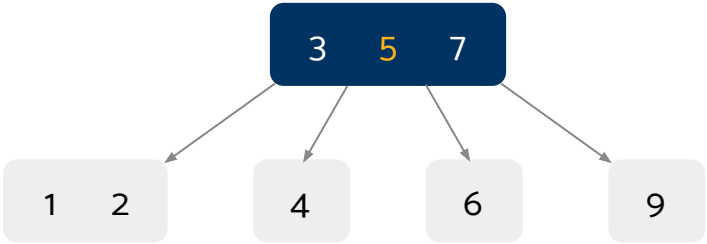
Each node can have up to 2 items and 3 children (there are variations where these values are higher known as 2-3-4 trees).

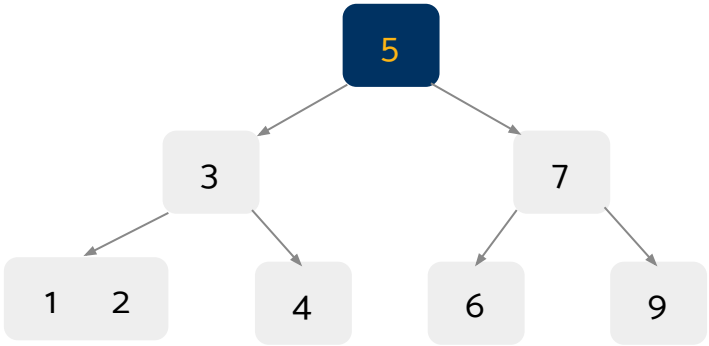
All leaves are the same distance from the root, which makes getting take Θ(log N) time.

B-Trees

When adding to a B-Tree, you first start by adding to a leaf node, and then pushing the excess items up the tree until it follows the rules from the last slide.

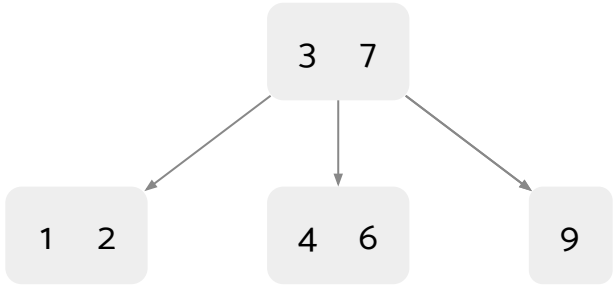


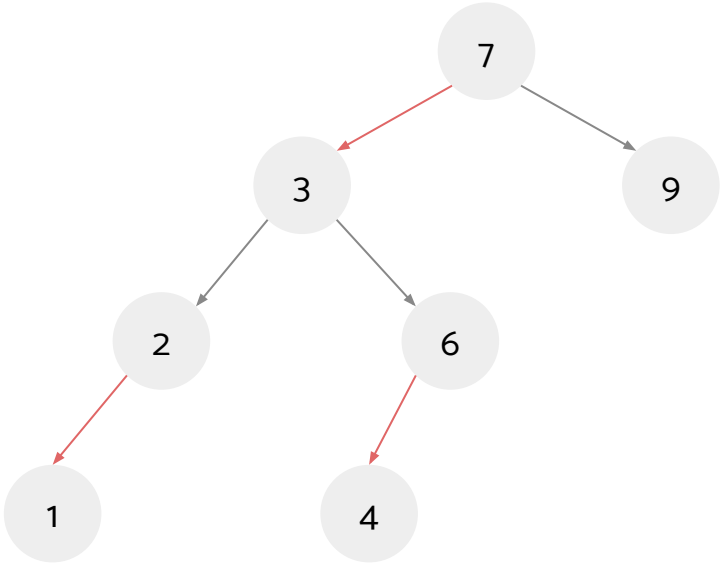




Left Leaning Red Black Trees

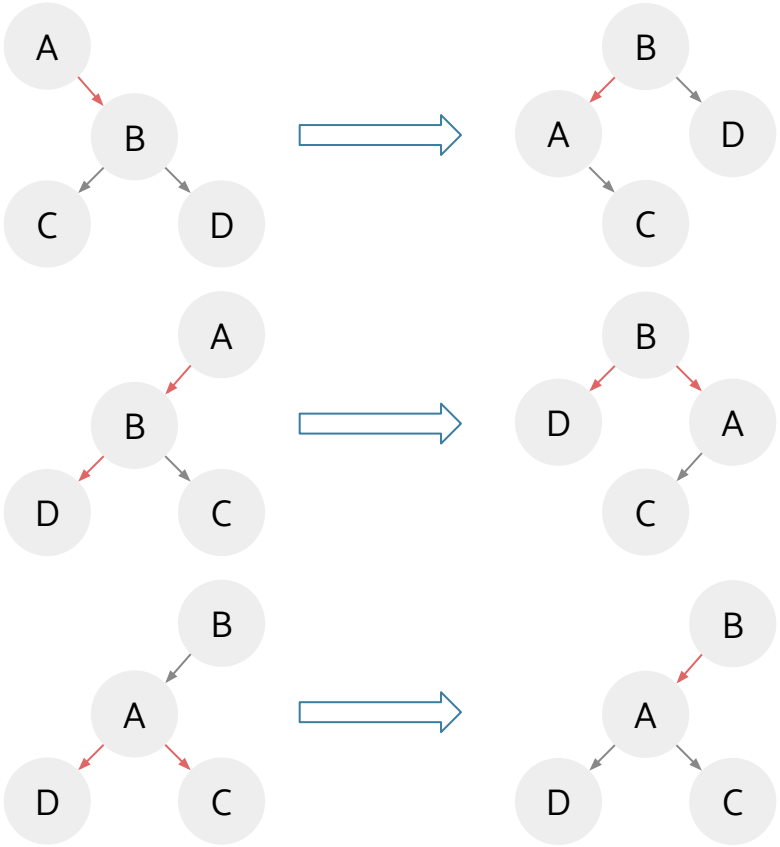
LLRBs are a representation of B-trees that we use because it is easier to work with in code. In an LLRB, each multi-node in a 2-3 tree is represented using a red connection on the left side.





LLRB Balancing Operations

`rotateLeft(A);`



`rotateRight(A);`

`colorFlip(A);`

Hashing

Hash functions are functions that represent an object using an integer. We use them to figure out which bucket of our hashset the item should go in.

Once we have a hash for our object we use mod to find out which bucket it goes into.

In each bucket, we deal with having lots of items by chaining the items and using .equals to find what we are looking for.

It is important that your `.equals()` function matches the result of comparing hashcodes - if two items are equal, they must also have the same hashcode

Worksheet

1 LLRB Insertions

2 LLRB Maximization

3 Hashing Gone Crazy

```
ECHashMap<TA, Integer> map = new ECHashMap<>();
TA sohum = new TA(“Sohum”, 10);
TA vivant = new TA(“Vivant”, 20);
map.put(sohum, 1);
map.put(vivant, 2);

vivant.charisma += 2;
map.put(vivant, 3);

sohum.name = “Vohum”;
map.put(vivant, 4);

sohum.charisma += 2;
map.put(sohum, 5);

sohum.name = “Sohum”;
TA shubha = new TA(“Shubha”, 24);
map.put(shubha, 6);
```

4 Buggy Hash

```
class Timezone {
     String timeZone;
     boolean dayLight;
     String location;
     ...
     public int currentTime() {
          // returns current time in that time zone
     }
     public int hashCode() {
          return currentTime();
     }
     public boolean equals(Object o) {
          Timezone tz = (Timezone) o;
          return tz.timeZone.equals(timeZones);
     }
}
```

4 Buggy Hash

```
class Course {
     int courseCode;
     int yearOffered;
     String[] staff;
     ...
     public int hashCode() {
          return yearOffered + courseCode;
     }
     public boolean equals(Object o) {
          Course c = (Course) o;
          return c.courseCode == courseCode;
     }
}
```