CS 225
Data Structures

Oct. 23 – BTree
B-Trees

Q: Can we always fit our data in main memory?

Q: Where else can we keep our data?

However, big-O assumes uniform time for all operations.
Vast Differences in Time

A 3GHz CPU performs 3m operations in _______.

Old Argument: “Disk Storage is Slow”
- Bleeding-edge storage is pretty fast:
  NVMe (M.2, PCIe 3.0 x4):
- Large Disks (10 TB+) still have slow throughout:

New Argument: “The Cloud is Slow!”
AVLs on Disk
Real Application

Imagine storing driving records for everyone in the US:

How many records?

How much data in total?

How deep is the AVL tree?
Exams

Exam 7 (Theory Exam)
- Live right now!

Exam 8 (Programming Exam)
- Review Assignments: MP4, lab_avl
- Topics: AVL trees, iterators
Share Your #cs225animation

On Facebook/Twitter/Instagram:

#cs225animation

...I’ll search this tag every few days and like/heart your work!

On Piazza:

See pinned post: “MP4: Animation Sharing”
BTree Motivations

Knowing that we have large seek times for data, we want to:
**Goal:** Minimize the number of reads!

Build a tree that uses ______________________ / node
- [1 network packet]
- [1 disk block]
A BTrees of order $m$ is an $m$-way tree:
- All keys within a node are ordered
- All leaves contain hold no more than $m-1$ nodes.

$m=5$
BTree Insertion

When a BTree node reaches $m$ keys:
BTree Recursive Insert

```
-3  8
```

```
23  42
```

```
25  31
```

```
43  55
```

m=3
BTree Recursive Insert

\[ m = 3 \]

-3  8  25  31  43  55
BTree Visualization/Tool

https://www.cs.usfca.edu/~galles/visualization/BTree.html
Btree Properties

A **B**t**rees** of order **m** is an m-way tree:
- All keys within a node are ordered
- All leaves contain hold no more than **m-1** nodes.
- All internal nodes have exactly **one more key than children**
- Root nodes can be a leaf or have **[2, m]** children.
- All non-root, internal nodes have **[ceil(m/2), m]** children.
- All leaves are on the same level
BTree Search
bool Btree::_exists(BTreeNode & node, const K & key) {
    unsigned i;
    for (i = 0; i < node.keys_ct_ && key < node.keys_[i]; i++) {
    }
    if (i < node.keys_ct_ && key == node.keys_[i]) {
        return true;
    }
    if (node.isLeaf() ) {
        return false;
    } else {
        BTreeNode nextChild = node._fetchChild(i);
        return _exists(nextChild, key);
    }
}
BTree Analysis

The height of the BTree determines maximum number of ______ possible in search data.

...and the height of the structure is: ____________.

Therefore: The number of seeks is no more than ________.

...suppose we want to prove this!
BTree Analysis

In our AVL Analysis, we saw finding an upper bound on the height (given $n$) is the same as finding a lower bound on the nodes (given $h$).

We want to find a relationship for BTrees between the number of keys ($n$) and the height ($h$).
CS 225 – Things To Be Doing

Exam 7 (theory) starts Monday!
Review Document: On Piazza
Review Session: 7pm, 1404 SC
More Info: https://courses.engr.illinois.edu/cs225/fa2017/exams/

MP4: Due Monday
Due: Monday, Oct. 23 at 11:59pm

Lab: lab_avl
Due Sunday, Oct. 22 at 11:59pm

POTD
Every Monday-Friday – Worth +1 Extra Credit /problem (up to +40 total)