CS 225

#22: kd-Trees and BTrees Intro

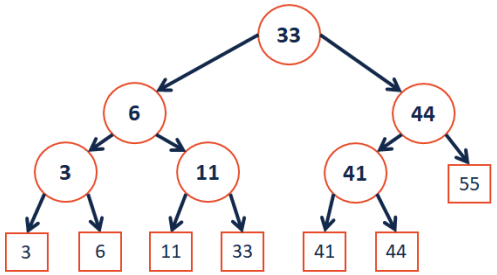
October 17, 2018 · *Wade Fagen-Ulmschneider*

Range-based Searches:

Q: Consider points in 1D: p = {p_1, p_2, …, p_n}.

…what points fall in [11, 42]?

Tree Construction:

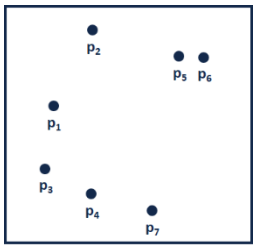


Range-based Searches:

Running Time:

Extending to k-dimensions:

Consider points in 2D: **p = {p_1, p_2, …, p_n}:**

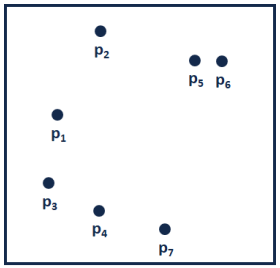


…what points are inside a range (rectangle)?

…what is the nearest point to a query point **q**?

kd-Tree Motivation:

First, let's try and divide our space up:

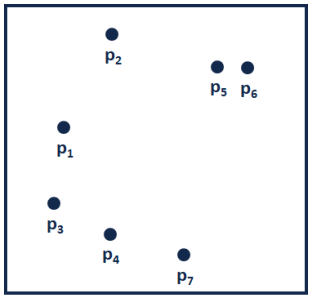


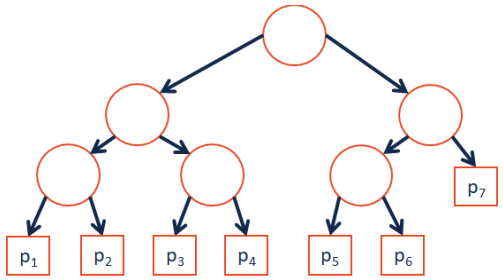
---

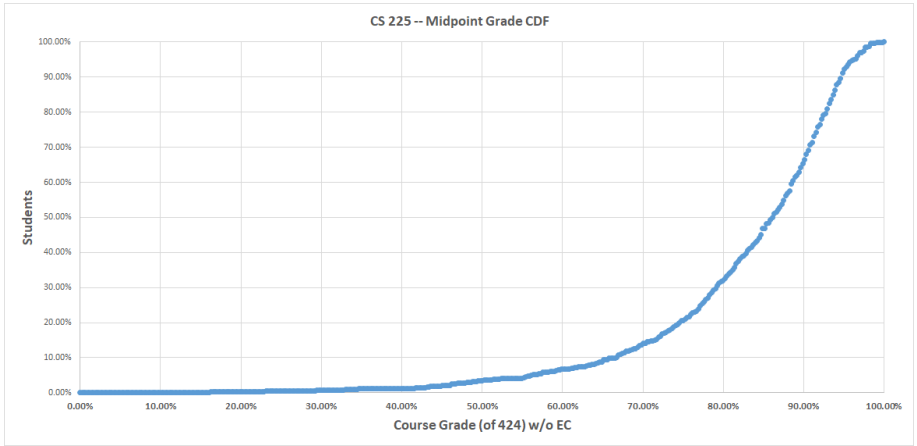
kd-Tree Construction:

How many dimensions exist in our input space?

How do we want to "order" our dimensions?







Motivation
Can we always fit our data in main memory?

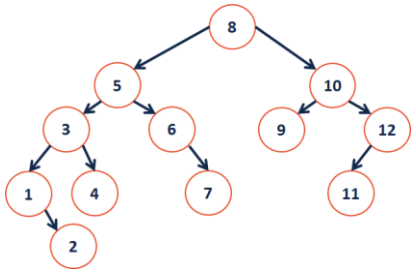
Where else do we keep our data?

-

-

vs. CPU: 3 GHz == 3m ops / _____ * ____ cores

AVL Operations on Disk:



How deep do AVL trees get?

BTree Motivations
Knowing that we have long seek times for data, we want to build a data structure with two (related) properties:

1.

2.

BTree$_m$

-3	8	23	25	31	42	43	55

m=9

Goal: Build a tree that uses _____ /node!
...optimize the algorithm for your platform!

A **BTree of order m** is an m-way tree where:

1. All keys within a node are ordered.
2. All leaves contain no more than **m-1** nodes.

BTree Insert, using m=5

...when a BTree node reaches **m** keys:

CS 225 – Things To Be Doing:
1. Programming Exam B starts next Thursday (Oct 25th)
2. MP4 extra credit ongoing (final deadline Monday, Oct. 17th)
3. lab_avl released this week; course feedback in lab this week!
4. Daily POTDs are ongoing!