

1a. The path below is the path I would take. Since 35 is less than 45, we go left from the root. Next we go to the internal node, since 35 is greater than 19, we go right. We keep going right until we get near 65. The last red line we check the last block and see that 65 is less than 70, so we stop. We read 6 blocks in total.

Bookmarks Profiles Tab Window Help

re/Downloads/hw4-index-qe-fa22.pdf

1 / 2 | 100% +

(Fall 2022)

100 points, Due Sunday Nov 20 11:59 PM

1. [40 points] Consider the following B+ tree for the search key "age. Suppose the degree d of the tree = 2, that is, each node (except for root) must have at least two keys and at most 4 keys. Note that sibling nodes are nodes with the same parent.

a. [10 points] Describe the process of finding keys for the query condition "age \geq 35 and age \leq 65". How many blocks I/O's are needed for the process?

b. [15 points] Draw the B+ tree after inserting 14, 15, and 16 into the tree. Only need to show the final tree after all insertions.

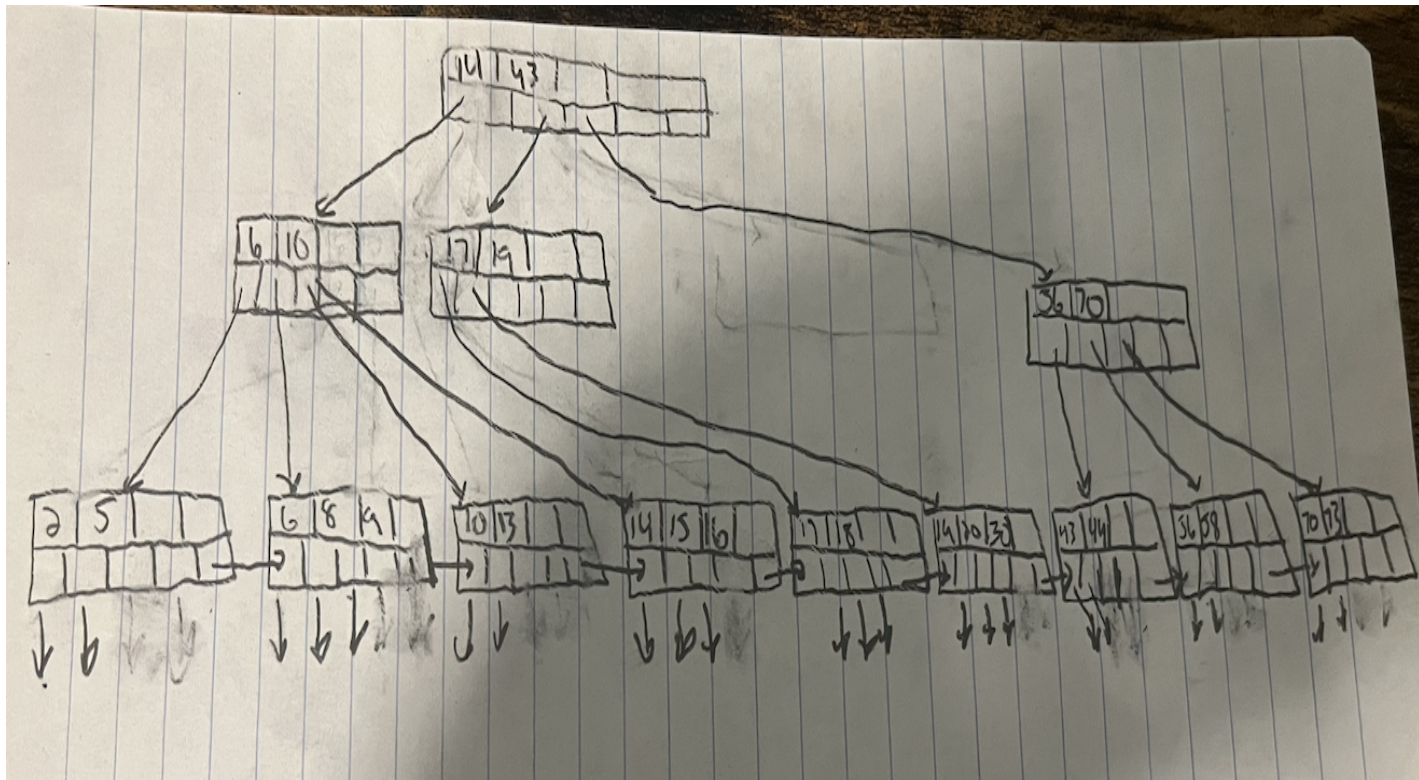
c. [15 points] Draw the tree after deleting 2 from the original tree.

2. [60 points] Consider natural-joining tables $R(a, b)$ and $S(a, c)$. Suppose we have the following scenario.

- R is a clustered relation with 5,000 blocks.
- S is a clustered relation with 20,000 blocks.
- 100 records available in join result for the join.

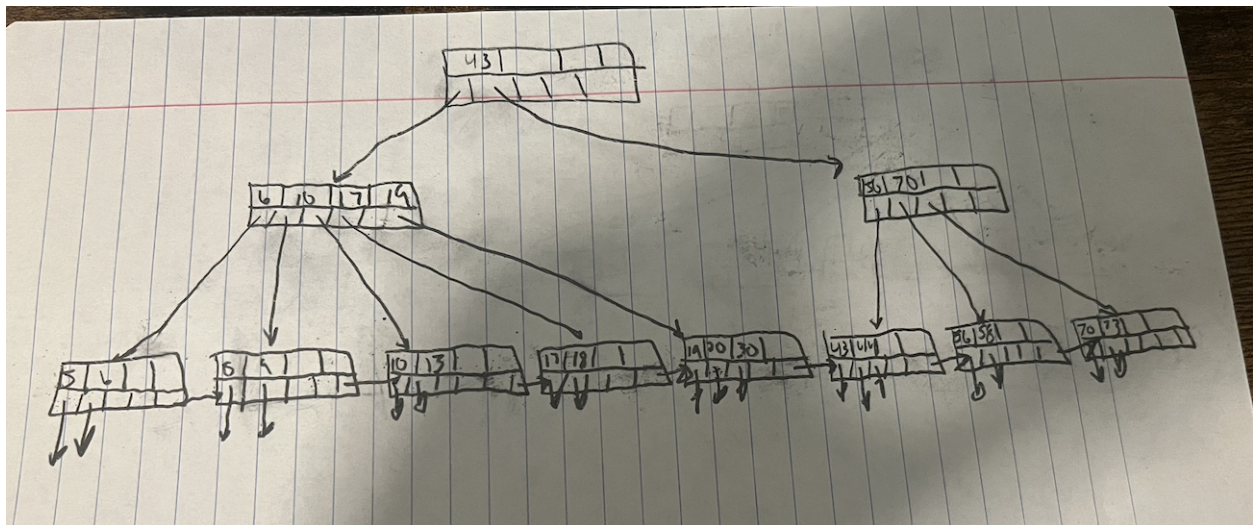
untitled15.py myslupload.py

1b.



In this diagram, we split both on the middle of 10,13,14,15,16 (which 14 is in the middle so it goes right) and then we split again 6,10,17,19 because there are too many pointers and we add 14 to the root node.

1c.



For this one, once we delete two there's only one 5 in that leaf node. We move the 6 to left most leaf node, so there is a minimum of two numbers and leave the 8 and 9.

2a.

$$\text{Cost of R} = B(R) + B(R)/(M-2) * B(S)$$

$$5000 + 5000/(102-2) * 20000 = \mathbf{1,005,000}$$

2b.

$$\text{Cost of S} = B(S) + B(S)/(M-2) * B(R)$$

$$20000 + 20000 (102-2) * 5000 = \mathbf{1,020,000}$$

2c.

Pass 1:

$$\text{Sort S} = 20,000/100 = 200 \text{ runs}$$

$$\text{Sort R} = 5,000/100 = 50 \text{ runs}$$

Cost of Reading/Writing:

$$2 B(R) + 2 B(S)$$

Pass 2:

However, since $B(S) = 20,000$, we need to run two times:

$$2 B(R) + 4 B(S)$$

Pass 3:

We then need to merge after, at the cost of $B(R) + B(S)$:

TOTAL COST:

$$3 B(R) + 5 B(S)$$

$$3 * 5000 + 5 * 20000 = \mathbf{115,000}$$

2d.

Pass 1:

Load and write the bucket

$$2 B(S) + 2 B(R)$$

Pass 2:

Joining

$$B(R) + B(S)$$

$$\text{Total Cost: } 3B(R) + 3B(S)$$

$$= 3 * 5000 + 3 * 20000 = \mathbf{75,000}$$

Therefore, partition,based-hashing is the most efficient because it runs the smallest amount of i/o blocks and therefore is the fastest.