



WEEK 3

EXAMPLE OF TIME COMPLEXITY CALCULATION FOR A B+ TREE SEARCH

CS3319

STUDENT OBJECTIVES

- Upon completion of this video, you should be able to:
 - Calculate the worst time for a search on a B+ tree structure for a given example.

QUESTION: Find the worst case search time to find a record if you use a multilevel index for the following scenario:

- $r = 100,000$ records stored on a disk with block size $B = 2048$ bytes.
- Records are fixed size of $R = 500$ bytes.
- Blocking Factor = $2048 / 500 = \underline{4}$
- Number of blocks needed = $\underline{100,000/4 \rightarrow 25,000}$
- Ordering key field is $V = 10$ bytes, a block pointer $P = 7$ bytes, thus size of the primary record is $\underline{17}$ bytes per record
- Blocking Factor for index = $2048 / 17 = 120$ indices per block
- # of blocks needed for index = $100000 / 120 = \underline{834}$
- # of blocks needed for level 1 = # of blocks needed for index = $\underline{834}$
- # of blocks needed for level 2 = level 1 / 120 = $\underline{834/120 \rightarrow 7 \text{ blocks}}$
- # of blocks need for level 3 = level 2 / 120 = $\underline{7/120 \rightarrow 1 \text{ block}}$
- Search is 3 levels + 1 level to get to the data block = 4 Block accesses

QUESTION: Find the worst case search time for a multilevel index for the following scenario

Block 1

Record 1 → 500 bytes
Record 2 → 500 bytes
Record 3 → 500 bytes
Record 4 → 500 bytes

Block 2

Record 5 → 500 bytes
Record 6 → 500 bytes
Record 7 → 500 bytes
Record 8 → 500 bytes

Block 3

Record 9 → 500 bytes
Record 10 → 500 bytes
Record 11 → 500 bytes
Record 12 → 500 bytes

Block 25,000

Record 99997 → 500 bytes
Record 99998 → 500 bytes
Record 99999 → 500 bytes
Record 10000 → 500 bytes

Index Block 1

Key for Record 1	Pointer to Block 1 →
Key for Record 2	Pointer to Block 1 →
Key for Record 3	Pointer to Block 1 →
Key for Record 4	Pointer to Block 1 →
Key for Record 5	Pointer to Block 2 →
...	...
Key for Record 120	Pointer to Block 80 →

17 bytes long

17 * 120 = 2040 bytes

Index Block 2

Key for Record 121	Pointer to Block 31 →
Key for Record 122	Pointer to Block 31 →
Key for Record 123	Pointer to Block 31 →
Key for Record 124	Pointer to Block 31 →
Key for Record 125	Pointer to Block 32 →
...	...
Key for Record 240	Pointer to Block 60 →

834 * 17 = 14178 bytes

Index Block 834

Key for Record 99961	Pointer to Block 24991 →
Key for Record 99962	Pointer to Block 24991 →
Key for Record 99963	Pointer to Block 24991 →
Key for Record 99964	Pointer to Block 24991 →
Key for Record 99965	Pointer to Block 24992 →
...	...
Key for Record 100000	Pointer to Block 25000 →

100,000 / 4 = 25,000

Key for Record 1	Pointer to Block 1
Key for Record 120	Pointer to Block 2
Key for Record 240	Pointer to Block 3
...	...
Key for Record 14280	Pointer to Block 120

Block 1 – Level 2

Key for Record 14400	Pointer to Block 121
Key for Record 14520	Pointer to Block 122
Key for Record 14640	Pointer to Block 123
...	...
Key for Record 28800	Pointer to Block 240

Block 2 – Level 2

Key for Record 86400	Pointer to Block 720
Key for Record 86520	Pointer to Block 721
Key for Record 86640	Pointer to Block 722

Block 7 – Level 2

Key for Record 1	Pointer to Block 1 (Block 1 – level 2)
Key for Record 14400	Pointer to Block 121 (Block 2 – level 2)
Key for Record 28800	Pointer to Block 241 (Block 3 – level 2)
Key for Record 43200	Pointer to Block 361 (Block 4 – level 2)
Key for Record 57600	Pointer to Block 481 (Block 5 – level 2)
Key for Record 72000	Pointer to Block 601 (Block 6 – level 2)
Key for Record 86400	Pointer to Block 721 (Block 7 – level 2)
... LEFT OVER SPACE IN BLOCK 1	

Block 1 – ROOT

