DS Class 11

September 13, 2022

NBA: Number of block accesses

B: Block size

N: Number of rows

l: row length

l_{PI}: length of primary index. Similarly, Secondary Index (SI), CI and SINk.

`l_k`: length of key

l_A: length of attribute

l_blockptr: Length of block pointer

Blocking Factor (bfr): for row and bfr for index

n_r / n_R: Number of blocks

Consider example: Unordered File

```
bfr = floor(B / l)
```

```
n_R = ceil(N / bfr)
```

$\sigma_{K=val}(R)$ - on average $n/2$ - best 1 - worst n

$\sigma_{A=i}(R) = \lceil \frac{c(A)_i}{bfr} \rceil$ - $c(A)_i$: number of rows where $A = i$ - worstcase `n` - average `n` - best 1 (only one block is accessed that contains all $c(A)_i$ rows).

Ordered File

Worst case $log(n_R)$: for $\sigma_{k=val}(R)$. - Average case here is almost same as the worst case. - Best case: 1.

For A attribute (assume ordered on A),

$log(n_R) + \lceil c(A)_i / bfr \rceil$

Primary index

row of PI `K | blkptr`.

$N_{PI} = n_R$: Number of rows in the PI.

$l_{PI} = l_K + l_{blkptr}$

$bfr_{PI} = \lfloor (B/l_{PI}) \rfloor$

$n_{PI} = \lceil n_R / bfr \rceil$

NBA = log (n_PI) + 1 (one for accessing the block from the block pointer in the index)>

Clustering Index

$l_c = l_A + l_{blkptr}$

$n_c = \lceil f_a / bfr_c \rceil$ where f_a is the number of distinct values in A.

$NBA = log(n_c) + \lceil c(A)_i / bfr \rceil$

(**Assumption**: every new value starts in a new block in CI)

If we do $\sigma_{K=val}(R)$. clustering index won't optimize it and it will still remain as an unordered file case.

Unordered File

Secondary Index on Key Attribute

$l_{SK} = l_k + l_{rptr}$

bfr same as (`B/l` here $l = l_{SK}$).

Reason for using record pointer (so that I don't have to do linear search in the block. because that is very inefficient).

$n_{SK} = \lceil n / bfr \rceil$

$NBA_{SK} = \lceil log(n_{SK}) \rceil + 1$

Secondary index on Non-key Attribute

$l_{SINK} = l_A + l_{blkptr}$

$bfr_{SINK} = \lfloor B / l_{SINK} \rfloor$

$n_{SINK} = \lceil f_a / bfr_{SINK} \rceil$.

$NBA = \lceil log(n_{SINK}) \rceil + \lceil \frac{c(A)_i}{\lfloor B / l_{recordptr} \rfloor} + c(A)_i$.

The second term corresponds to the indirection blocks - Meaning we take fraction of max correct rows by blocking factor for the indirection blocks - This gives the number of blocks that will contain the correct rows.

Then add max correct rows (in the worst case all the row pointers correspond to different blocks).

If $c(A)_i \sim n_R$. - In the worst case each attribute is present in a different block - Then the SI is a waste and no better than sequential search. Example: Gender attribute in a table.

B Tree (on key)

- p block pointers

$p * l_{blkptr} + (p - 1) * (l_k + l_{recpointer}) \leq B$

Calculate p from this.