**Lecture 6**

**Linear Hashing**

Files: primary data + overflow data + *split pointer* (sp)

Advantage: does not require auxiliary storage for a directory

Disadvantage: requires overflow pages (splits don’t occur on full pages)

**Splitting**

Splitting algorithm:

**A screenshot of text

Description automatically generated**

Two approaches to triggering a split:

* Split every time a tuple is inserted into full block
* Split when load factor reaches threshold (every *k* inserts)

Systematic splitting like this:

* Eventually reduces length of every overflow chain
* Helps to maintain short average overflow chain length

**Insertion Cost**

If no split required, cost same as for standard hashing:

*Costinsert*= Best: *1r + 1w*  Worst: *(1+max(OV)r + 2w*

If split occurs, incur *Costinsert* plus cost of splitting:

* Read block sp (plus all of its overflow blocks)
* Write block sp (and its new overflow blocks)
* Write block sp+2d (and its new overflow blocks)

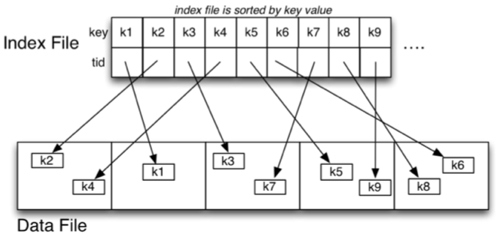
On average, Costsplit = (1+Ov)r + (2+Ov)w

**Hash Files in PostgreSQL**



A picture containing table, bottle, indoor

Description automatically generated

**Indexing**

****

**Indexes**

A 1-d index is based on the value of a single attribute *A*

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

A close up of a sign

Description automatically generated

**Dense Primary Index**

Data file unsorted; one index entry for each tuple

**Sparse Primary Index**

Data file sorted; one index entry for each page

**Selection/Insertion/Deletion with Primary Index**

*Costone, prim = log2i + 1 + Ov*

*Costinsert,prim  =  (log2i)r + i/2.(1r+1w) + (1+Ov)r + (1+δ)w*

If we delete index entries by marking ...

* *Costdelete,prim  =  (log2 i + 1 + Ov)r + 2w*

If we delete index entry by index file reorganisation ...

* *Costdelete,prim  =  (log2 i + 1 + Ov)r + i/2.(1r+1w) + 1w*

**Clustering Index**

Data file sorted; one index entry for each key value

Insertions are expensive: rearrange index file and data file

Deletions relatively cheap (similar to primary index)

**Secondary Index**

Generally, dense index on non-unique attribute *As*

* data file is not ordered on attribute *As*
* index file is ordered on attribute *As*

Problem: multiple tuples with same value for *As*.

A solution:

* dense index (**Ix2**) containing just **TupleId**'s
* sparse index (**Ix1**) on dense index containing *(key,offset)*pairs

Each *offset* references an entry in **Ix2**

A close up of a map

Description automatically generated

*Costpmr  =  Costrange  =  (log2i + aq2 + bq.(1 + Ov))*   
*Costrange  =  (log2i + aq1 + aq2 + bq.(1 + Ov))*

**Select with Multi-level Index**

Read *d* index blocks and *1+Ov* data blocks.

Thus, *Costone,mli  =  (d + 1 + Ov)r*

**B-Trees**

B-trees are MSTs with the properties:

* they are updated so as to remain balanced
* each node has at least *(n-1)/2* entries in it
* each tree node occupies an entire disk page

B-tree insertion and deletion methods

* are moderately complicated to describe
* can be implemented very efficiently

Advantages of B-trees over general MSTs

* better storage utilisation (around 2/3 full)
* better “worst case” performance (shallower)

**B-Tree Depth**

load *Li = 0.69 × ci*   and   depth of tree *~ ceil( logLi r )*

Example: *ci=128,    Li=88*

A close up of a white background

Description automatically generated

**B-Tree Insertion Cost**

Insertion cost = *CosttreeSearch + CosttreeInsert + CostdataInsert*

Best case: write one page (most of time)

*Costinsert = Dr + 1w + 1r + 1w*

Common case: *3* node writes (rearrange 2 leaves + parent)

*Costinsert**= Dr + 3w + 1r + 1w*

Worst case: *2D-1* node writes (propagate to root)

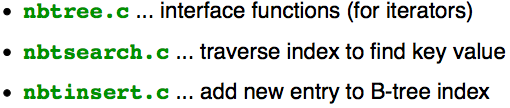
*Costinsert  =  Dr + (2D-1)w + 1r + 1w*

**Selection with B-Trees**

*Costone  =  (D + 1)r*

*Costrange  =  (D + bi + bq)r*

**B-Tree in PostgreSQL**



**N-d Selection via Heaps**

*Costpmr  =  Costspace  =  b*

**N-d Queries and Indexes**

Cost = *Costindex + bqix*  (some pages do not contain answers, *bqix > bq)*

**Bitmap Indexes**

A screenshot of a cell phone

Description automatically generated