CSCI 21062- Advanced Database Management Systems

Cost functions for SELECT

Cost functions for the selection algorithms S1 to S8 given in the slides – in terms of number of block transfers between memory and disk. The cost of Si is referred to as C_{Si} block access.

S1 – Linear search (brute force) approach

Must search all file blocks
$C_{S1a} = b$
For an equality condition on the key
$C_{S1b} = b/2$ - if the record is found
$C_{S1b} = b$ – if the record is not found

S2 Binary search

$C_{S2} = \log_2 b + \lceil (s/bfr) \rceil - 1$ file block
$= \log_2 b$ - if equality condition on key, because s = 1

S3 - Using a primary index(S3a) or hash key(S3b) to retrieve a single record

$C_{S3a} = x + 1$ – for primary index x – number of levels

$C_{S3b} = 1$ for static hashing or linear hashing

$= 2$ for extendible hashing

S4 Using an ordering index to retrieve multiple records

$C_{S4} = x + b/2$ – for any comparison condition on the key field. This is a rough estimate

S5 Using a clustering index to retrieve multiple records

Given an equality condition, s records will satisfy the condition where s is the selection cardinality of the indexing attribute. So $\lceil (s/bfr) \rceil$ file blocks will be accessed

$C_{S5} = x + \lceil (s/bfr) \rceil$

S6 Using a secondary (B^+-tree) index

For equality s records will satisfy the condition. s - selection cardinality of the indexing attribute.

Records may reside in different blocks

$C_{S6a} = x + s$ for non key
$= x + 1$ for key
$C_{S6b} = x + (b_{I1}/2) + (r/2)$ - for comparison condition (half the records are assumed to stratify the condition)

S7 Conjunctive selection

Use either S1 or one of the methods S2 to S6, In the later case, use one condition to retrieve and then check in the memory buffer whether the retrieve records stratify the other conditions in the conjunctive

S8 Conjunctive selection using a composite index

Same as S3a, S5 or S6a, depending on the type of index.

Cost function for JOIN

We need to have an estimate for the size of the file (number of tuples) that results after the join operation. This is kept as a ratio of the size of the resulting join file to the size of the cartesian product, if both are applied to the same file, it is called the join selectivity (js)

js = $|(R \bowtie_c S)|/|RXS)|$ = $|(R \bowtie_c S)|/|R|*|S|$ where $|R|$ - number of tuples in R
js = 1 if there is no join condition c (the join is same as the cartesian product)
js = 0 if no tuples satisfy the join condition c
In general $0 \leq js \leq 1$

For a join where the condition c is equality, we get the following two special cases:

1. If A is the key of R, then $|(R \bowtie_c S)| < |S|$, so $js \leq (1/|R|)$ (where c= R.A = S.B)
2. If B is the key of S, then $|(R \bowtie_c S)| < |R|$, so $js \leq (1/|S|)$ (where c= R.A = S.B)

Given the sizes of the two input files
The size of the resulting file $|(R \bowtie_c S)| = js*|R|*|S|$

If the join operation is of the form $(R \bowtie_{A=B} S)$

J1 Nested Loop Join

Suppose R is in the outer loop assuming three memory buffers

$C_{j1} = b_R + (b_R*b_S) + ((js*|R|*|S|)/bfr_{RS})$ bfr_{RS} – blocking factor of the resulting file

The last part of the formula is the cost of writing the resulting file to disk

J2 Single Loop Join

If the index exists for the join attribute B of S with index level x_B, we can retrieve each record s in R and then use the index to retrieve all the matching records t from S that satisfy t[B] = s[A]. The cost depends on the type of index.

For a secondary index where s_B is the selection cardinality for the join attribute B of S

$$C_{J2a} = b_R + |R|*(x_B + s_B) + ((js*|R|*|S|)/bfr_{RS})$$

For a clustering index where s_B is the selection cardinality for the join attribute B of S

$$C_{J2b} = b_R + |R|*(x_B + s_B/bfr_B) + ((js*|R|*|S|)/bfr_{RS})$$

For a primary index

$$C_{J2c} = b_R + |R|*(x_B + 1) + ((js*|R|*|S|)/bfr_{RS})$$

If a hash index exists for one of the two attributes say B of S

$$C_{J2d} = b_R +(|R|*h) + ((js*|R|*|S|)/bfr_{RS})$$

Where $h \geq 1$ is the average number of block accesses to retrieve a record, given its hash key value

J3 sort merge join

If the files are already sorted on the join attributes
$C_{J3a} = b_R + b_S + ((js*|R|*|S|)/bfr_{RS})$

If we must sort the files, then cost of sorting must be added.