Database Systems Query Costing

by Chris Chamberlain, current student - no guarantee of correctness

October 8, 2020

1 With an index

Clustered index, matching one or more predicates.

$$\text{Cost}(\text{B+ tree index}) = (\text{NPages}(I) + \text{NPages}(R)) \cdot \Pi_{i=1..n} RF_i$$
$$\text{Cost}(\text{Hash index}) = \text{NPages}(R) \cdot \Pi_{i=1..n} RF_i \cdot 2.2$$

Non-clustered index, matching one or more predicates.

$$\text{Cost}(\text{B+ tree index}) = (\text{NPages}(I) + \text{NTuples}(R)) \cdot \Pi_{i=1..n} RF_i$$
$$\text{Cost}(\text{Hash index}) = \text{NTuples}(R) \cdot \Pi_{i=1..n} RF_i \cdot 2.2$$

Only selecting a single tuple (index selection by primary key)

$$\text{Cost}(\text{B+ tree}) = \text{Height}(I) + 1$$

$$\begin{aligned}\text{Cost}(\text{Hash Probe}) &= \text{ProbeCost}(I) + 1 \\ &\approx 2.2\end{aligned}$$

Note: the +1 in these two formulas is for accessing the actual data page.

2 Without using an index

Sequential heap scan (no index, unsorted):

$$\text{Cost} = \text{NPages}(R)$$

Note: heap scan is always an option and sometimes will be faster than using an index. Don't assume an index will be faster just because it exists.

Binary search (no index, sorted by relevant column):

$$\text{Cost} = \underbrace{\log_2(\text{NPages}(R))}_{\text{Binary search cost to find first page}} + \overbrace{RF \cdot \text{NPages}(R)}^{\text{Read all pages matching predicate}}$$

3 Multi-relation plans / joins

1. Select order of relations (prefer "left-deep-joins", allows for "on-the-fly" reading and pipelining). Get rid of plans with cross-products
2. For each join, select join algorithm (hash join, sort merge)
3. For each input relation, select access method

3.1 Join algorithms

R is the outer relation, S is the inner relation.

$$\begin{aligned}
\text{Cost(Tuple-oriented NLJ)} &= \text{NPages}(R) + \text{NTuples}(R) \cdot \text{NPages}(S) \\
\text{Cost(Page-oriented NLJ)} &= \text{NPages}(R) + \text{NPages}(R) \cdot \text{NPages}(S) \\
\text{Cost(Block-oriented NLJ)} &= \text{NPages}(R) + \left\lceil \frac{\text{NPages}(R)}{B-2} \right\rceil \cdot \text{NPages}(S) \\
&\quad \text{where } B \text{ is the number of available memory pages}
\end{aligned}$$

$$\begin{aligned}
\text{Cost(Hash Join)} &= \underbrace{2 \cdot (\text{NPages}(R) + \text{NPages}(S))}_{\text{Partioning phase (read-write)}} + \underbrace{(\text{NPages}(R) + \text{NPages}(S))}_{\text{Matching phase (read)}} \\
&= 3 \cdot (\text{NPages}(R) + \text{NPages}(S))
\end{aligned}$$

$$\begin{aligned}
\text{Cost(Sort-Merge Join)} = &\underbrace{2 \cdot \text{NPages}(R) \cdot \text{NumPasses(R)}}_{\text{Sort outer}} \\
&+ \underbrace{2 \cdot \text{NPages}(S) \cdot \text{NumPasses(S)}}_{\text{Sort inner}} \\
&+ \underbrace{\text{NPages}(R) + \text{NPages}(S)}_{\text{Merge inputs}}
\end{aligned}$$

3.2 Pipelining

Todo.

4 Cost of projections

When performing a projection, you need to remove duplicates (`SELECT DISTINCT` in MySQL). There are two main methods:

Sort-based projection: (1) read table, keeping only projected attrs, (2) write pages with projected attrs to disk, (3) sort pages with external sort, (4) read sorted projected pages to discard adjacent duplicates.

$$\begin{aligned}
\text{Cost(Sort-based projection)} = &\underbrace{\text{NPages}(R)}_{\text{ReadTable}} + \underbrace{\text{NPages}(R) \cdot PF}_{\text{WriteProjectedPages}} + \\
&\underbrace{2 \cdot \text{NumPasses} \cdot \text{NPages}(R) \cdot PF}_{\text{SortProjectedPages}} + \underbrace{\text{NPages}(R) \cdot PF}_{\text{ReadProjectedPages}}
\end{aligned}$$

Hash-based projection: (1) read table, keeping only projected attrs, (2) write pages with projected attrs into corresponding buckets, (3) read buckets one-by-one, create another hash-table and discard duplicates within a bucket.

$$\text{Cost(Hash-based projection)} = \underbrace{\text{NPages}(R)}_{\text{ReadTable}} + \underbrace{\text{NPages}(R) \cdot PF}_{\text{WriteProjectedPages}} + \underbrace{\text{NPages}(R) \cdot PF}_{\text{ReadProjectedPages}}$$

5 Appendix

These formulas should be all mostly all intuitive, but they also underpin most of the formulas on the previous two pages.

5.1 Reduction factors

1. $col = value$

$$RF = \frac{1}{\text{NKeys}(col)}$$

2. $col > value$

$$RF = \frac{\text{High}(col) - value}{\text{High}(col) - \text{Low}(col)}$$

3. $col < value$

$$RF = (val - \text{Low}(col))/(\text{High}(col) - \text{Low}(col))$$

4. $col\_a = col\_b$ (join on a single column)

$$RF = \frac{1}{\max(\text{NKeys}(col\_a), \text{NKeys}(col\_b))}$$

5. No information, use a magic number $RF = \frac{1}{10}$

5.2 Result size after selection

Single table: $\text{ResultSize} = \text{NTuples}(R) \cdot \Pi_{i=1..n} RF_i$

Joins: $\text{ResultSize} = \Pi_{j=1..k}\text{NTuples}(R_j) \cdot \Pi_{i=1..n} RF_i$

If there are no predicates, reduction factors are ignored (i.e.: they are all 1)