Weekly exercises IN3020&4020 (2024)
Week 6: Query Compilation and Optimisation

1 Review

1. What are the 4 main steps that a query processor must perform (as outlined in the lectures), and what is their input and output?
2. What is a logical query plan?
3. What is the ideal goal of query optimisation?
4. What is the pragmatic goal of query optimisation, and what is the key ingredients for this approach?
5. What is our cost function and what is its main properties?

2 Parsing

Consider this grammar for a subset of SQL:

```
<Query> ::= SELECT <SelList> FROM <FromList> WHERE <Condition>
<SelList> ::= <Attribute>, <SelList>
<SelList> ::= <Attribute>
<FromList> ::= <Relation>, <FromList>
<FromList> ::= <Relation>
<Condition> ::= <Condition> AND <Condition>
<Condition> ::= <Attribute> IN ( <Query> )
<Condition> ::= <Attribute> = <Attribute>
<Condition> ::= <Attribute> LIKE <Pattern>
```

1. Extend (or change) the rules from `<Query>` to include the following:
   (a) Sort the output using the `ORDER BY` a list of attributes
   (b) A query with no `WHERE`-clause
2. Using the subset of SQL grammar above, construct parse trees for the following queries over relations $R[a, b]$ and $S[b, c]$:

(a)
```
SELECT a,c FROM R,S WHERE R.b = S.b;
```

(b)
```
SELECT a FROM R WHERE a IN
    (SELECT a FROM R, S WHERE R.b = S.b);
```

3 Compilation

Consider the two schemas $R[A, B, C]$, $S[A, D, F]$. Compile the following SQL queries into initial query plans (i.e., Relational Algebra expressions):

1. `SELECT A FROM R`
2. `SELECT DISTINCT A FROM R`
3. `SELECT * FROM R NATURAL JOIN S`
4. `SELECT * FROM R, S`
5. `SELECT * FROM R INNER JOIN S ON R.B = S.D`
6.* `SELECT * FROM R LEFT OUTER JOIN S ON R.B = S.D`
7. `SELECT SUM(B) FROM R GROUP BY A`
8. `SELECT R.C FROM R WHERE R.A IN (SELECT S.A FROM S WHERE S.D > 0)`
9. `SELECT R.C FROM R WHERE NOT EXISTS (SELECT S.A FROM S WHERE R.A = S.A)`

4 Optimization

4.1

When it is possible to push a selection to both arguments of a binary operator, we need to decide whether or not to do so. How would the existence of indexes on one of the arguments affect our choice? Consider for instance an expression $\sigma_C(R \cap S)$, where there is an index on S.

4.2

Consider the statistics for the four relations W, X, Y and Z in Table 1, additionally assuming that each attribute takes 5 bytes. Estimate the size (in bytes) of the relations that are the results of the following expressions:

1. $W \bowtie X \bowtie Y \bowtie Z$
2. $\sigma_{A=10}(W)$

Table 1: Statistics for W, X, Y and Z

$W\,[A, B]$	$X\,[B, C]$	$Y\,[C, D]$	$Z\,[D, E]$
$Tups(W) = 100$	$Tups(X) = 200$	$Tups(Y) = 300$	$Tups(Z) = 400$
$Vals(W.A) = 20$	$Vals(X.B) = 50$	$Vals(Y.C) = 50$	$Vals(Z.D) = 40$
$Vals(W.B) = 60$	$Vals(X.C) = 100$	$Vals(Y.D) = 50$	$Vals(Z.E) = 100$

3. $W \times Y$
4. $\sigma_{D>10}(Z)$
5. $\sigma_{A=1\,\&\,B=2}(W)$
6. $\sigma_{C=20}(Y) \bowtie Z$
7. $\sigma_{C=20}(Y \bowtie Z)$

4.3

Observe that the last two expressions in Question 4.2 are equivalent—that is, can be seen as two different logical query plans of the same SQL query. Compute the costs of these two plans (under our definition of cost); compare the relationship between the costs with the relationship between the sizes of these relations, which you computed in Question 4.2. Justify our definition of the cost function on the base of this comparison.

4.4

How would you estimate the size of a semijoin, assuming, for simplicity, that it is applied to set relations?

4.5 (*)

Consider relations $R\,[A, B]$ and $S\,[B, C]$ with $Vals(R.B) = Vals(S.B) = 20$ and (combined) histograms for the values of B as in Table 2. Estimate the number of tuples in the join $R \bowtie S$ using an assumption that the values of B are evenly distributed in the interval (in the histogram).

Table 2: Frequency of most common values

Values:	1	2	3	4	5	6...20
$R.B$:	5	6	4	6	0	30
$S.B$:	10	8	5	0	7	30