Principles of Database Management Systems

6: Query Compilation and Optimization
Pekka Kilpeläinen
(partially based on Stanford CS245 slide originals by Hector Garcia-Molina, Jeff Ullman and Jennifer Widom)

Overview

- We have studied recently:
  - Algorithms for selections and joins, and their costs (in terms of disk I/O)
- Next: A closer look at query compilation
  - Parsing
  - Algebraic optimization of logical query plans
  - Estimating sizes of intermediate results
    * Focus still on selections and joins
- Remember the overall process of query execution:

Step 1: Parsing

- Check syntactic correctness of the query, and transform it into a parse tree
- Based on the formal grammar for SQL
- Inner nodes nonterminal symbols (syntactic categories for things like <Query>, <RelName>, or <Condition>)
- Leaves terminal symbols: names of relations or attributes, keywords (SELECT, FROM, ...), operators (+, AND, OR, LIKE, ...), operands (10, %1960, ...)
- Also check semantic correctness: Relations and attributes exist, operands compatible with operators, ...

Example: SQL query

Consider querying a movie database with relations

StarsIn(title, year, starName)
MovieStar(name, address, gender, birthdate)

SELECT title
FROM StarsIn, MovieStar
WHERE starName = name AND birthdate LIKE '%1960';

(Find the titles for movies with stars born in 1960)

Example: Parse Tree
Step 2: Parse Tree $\rightarrow$ Logical Query Plan

- Basic strategy:

```
SELECT A, B, C
FROM R1, R2
WHERE Cond;
```

becomes

\[ \pi_{A,B,C}[\sigma_{\text{Cond}}(R1 \times R2)] \]

---

Step 3: Improving the L.Q.P

- Transform the logical query plan into an equivalent form expected to lead to better performance
  - Based on laws of relational algebra
  - Normal to produce a single optimized form, which acts as input for the generation of physical query plans

Example: Improved Logical Query Plan

```
\Pi_{\text{title}}
\sigma_{\text{star\_name}=\text{name} \text{ and birthdate LIKE } \%1960\%}
StarsIn \rightarrow MovieStar
```

---

Step 4: Estimate Result Sizes

- Cost estimates of database algorithms depend on sizes of input relations
  $\rightarrow$ Need to estimate sizes of intermediate results
  - Estimates based on statistics about relations, gathered periodically or incrementally

Example: Estimate Result Sizes

```
Need expected size
StarsIn \rightarrow MovieStar
\sigma_{\text{birthdate LIKE } \%1960\%}
```

Steps 5, 6, ...

- Generate and compare query plans
  - generate alternate query plans $P_1, ..., P_k$ by selecting algorithms and execution orders for relational operations
  - Estimate the cost of each plan
  - Choose the plan $P_i$ estimated to be "best"
- Execute plan $P_i$ and return its result

Example: One Physical Plan

```
  Hash join
  +----------------+
  |                |
  | SEQ scan       |
  +----------------+

  Parameters: join order, memory size, project attributes, ...
```

```
  Hash join
  +----------------+
  |                |
  | Index scan     |
  +----------------+

  Parameters: Select Condition, ...
```

Next: Closer look at ...

- Transformation rules
- Estimating result sizes

Relational algebra optimization

- Transformation rules (preserve equivalence)
- What are good transformations?

Rules: Natural joins

- $R \bowtie S = S \bowtie R$ (commutative)
- $(R \bowtie S) + T = R \bowtie (S + T)$ (associative)
- Carry attribute names in results, so order is not important
  - Can evaluate in any order

Rules: Cross products & union similarly (both associative & commutative):

- $(R \times S) \times T = R \times (S \times T)$
- $R \times S = S \times R$

- $(R \cup (S \cup T) = (R \cup S) \cup T$
- $R \cup S = S \cup R$
Rules: Selects

1. $\sigma_{p_1 \land p_2}(R) = \sigma_{p_1} \sigma_{p_2}(R)$
2. $\sigma_{p_1 \lor p_2}(R) = [\sigma_{p_1}(R)] \cup [\sigma_{p_2}(R)]$

1. Especially useful (applied left-to-right):
   Allows compound select conditions to be split and moved to suitable positions
   (See next)

Rules: Products to Joins

Definition of Natural Join:

$R \bowtie S = \pi_L [\sigma_C (R \times S)]$

Condition $C$ equates attributes common to $R$ and $S$, and $\pi_L$ projects one copy of them out

Applied right-to-left
- definition of general join applied similarly

Rules: $\sigma + \bowtie$ combined

Let $p = \text{predicate with only R attrs}$
$q = \text{predicate with only S attrs}$
$m = \text{predicate with attrs common to R,S}$

$\sigma_p (R \bowtie S) = [\sigma_p(R)] \bowtie S$
$\sigma_q (R \bowtie S) = R \bowtie [\sigma_q(S)]$

More rules can be derived...

Derived rules for $\sigma + \bowtie$

$\sigma_{p \land q} (R \bowtie S) = [\sigma_p(R)] \bowtie [\sigma_q(S)]$
$\sigma_{p \lor q \land m} (R \bowtie S) = [\pi_L (\sigma_p R \bowtie (\sigma_q S))]$
$\sigma_{p \lor q} (R \bowtie S) = [\pi_L (\sigma_p R \bowtie (\sigma_q S))] \cup [R \bowtie (\sigma_q S)]$

Derivation for first one;
Others for homework:

$\sigma_{p \land q} (R \bowtie S) = [\sigma_p(R \bowtie S)]$
$\sigma_p [\sigma_q (R \bowtie S)] = [\sigma_p(R)] \bowtie [\sigma_q(S)]$

Rules for $\sigma, \pi$ combined with $X$

Similar...

E.g., $\sigma_p (R \times S) = ?$
Some “good” transformations:

- $\sigma_{p1 \land p2} (R) \rightarrow \sigma_{p1} [\sigma_{p2} (R)]$
- $\sigma_{p} (R \bowtie S) \rightarrow [\sigma_{p} (R)] \bowtie S$
- $R \bowtie S \rightarrow S \bowtie R$

- No transformation is always good
- Usually good: early selections

* Estimating cost of query plan

1. Estimating size of intermediate results
2. Estimating # of I/Os
   (considered last week)

* Estimating result size

- Maintain statistics for relation R
  - $T(R)$: # tuples in R
  - $L(R)$: Length of rows, # of bytes in each R tuple
  - $B(R)$: # of blocks to hold all R tuples
  - $V(R, A)$: # distinct values in R for attribute A

Example

| R | | | | | |
|---|---|---|---|---|
|   | cat | cat | dog | bat |
| A | 1 | 1 | 1 | 1 |
| B | 20 | 20 | 30 | 50 |
| C | 10 | 20 | 30 | 50 |
| D | a  | b  | a  | d  |

$T(R) = 6$  $L(R) = 100$
$V(R, A) = 3$  $V(R, C) = 5$
$V(R, B) = 1$  $V(R, D) = 4$

Size estimates for product $W = R1 \times R2$

$T(W) = T(R1) \times T(R2)$
$L(W) = L(R1) + L(R2)$
Estimates for selection $W = \sigma_{a = b}(R)$

$L(W) = L(R)$

$T(W) = \ ?$

$= \text{AVG number of tuples that satisfy an equality condition on } R.A$

$= \ldots$

Size of $W = \sigma_{a = b}(R)$ in general:

Assume: Only existing $Z$ values $a_1, a_2, \ldots$ are used in a select expression $Z = a_i$, each with equal probability $1/V(R,Z)$

$\Rightarrow$ the expected size of $\sigma_{Z = a_i}(R)$ is

$E(T(W)) = \sum 1/V(R,Z) \times T(\sigma_{Z = a_i}(R))$

$= T(R)/V(R,Z)$

Example

R | A | B | C | D
---|---|---|---|---
1  | 1 | 0 | 1 | 0
2  | 1 | 2 | 0 | 1
3  | 0 | 1 | 0 | 0
4  | 1 | 0 | 0 | 0
5  | 0 | 0 | 0 | 0

$V(R, A) = 3$

$V(R, B) = 1$

$V(R, C) = 5$

$V(R, D) = 4$

$T(\sigma_{A = a}(R)) = 2, T(\sigma_{A = b}(R)) = 2, T(\sigma_{A = c}(R)) = 1$

$\Rightarrow (2 + 2 + 1)/3 = 5/3$

What about selection $W = \sigma_{Z \geq \text{val}}(R)$?

$T(W) = \ ?$

- Estimate 1: $T(W) = T(R)/2$
  - Rationale: On avg, 1/2 of tuples satisfy the condition

- Estimate 2: $T(W) = T(R)/3$
  - Rationale: Acknowledges the tendency of selecting “interesting” (e.g., rare tuples) more frequently

Size estimates for $W = R_1 \bowtie R_2$

Let $X =$ attributes of $R_1$

$Y =$ attributes of $R_2$

Case 1

$X \cap Y = \emptyset$

Same as $R_1 \times R_2$

Case 2

$W = R_1 \bowtie R_2$

$X \cap Y = A$

Assumption:

$V(R_1, A) \leq V(R_2, A) \Rightarrow \pi_A(R_1) \subseteq \pi_A(R_2)$

$V(R_2, A) \leq V(R_1, A) \Rightarrow \pi_A(R_2) \subseteq \pi_A(R_1)$

“Containment of value sets” [Sec. 7.4.4]
Why should “containment of value sets” hold?

Consider joining relations
Faculties(FName, ...) and
Depts(DName, FName, ...)
where Depts.FName is a foreign key, and
faculties can have 0,...,n departments.
Now \( V(\text{Depts, FName}) \leq V(\text{Faculties, FName}) \),
and referential integrity requires that
\[
\pi_{\text{FName}}(\text{Depts}) \subseteq \pi_{\text{FName}}(\text{Faculties})
\]

Estimating \( T(W) \) when \( V(\text{R1,A}) \leq V(\text{R2,A}) \)

\[
R_1 \quad A \quad B \quad C
\]
\[
R_2 \quad A \quad D
\]
Take 1 tuple
Match
Each estimated to match with \( T(\text{R2})/V(\text{R2,A}) \) tuples ...
so \( T(W) = T(\text{R1}) \times T(\text{R2}) / V(\text{R2,A}) \)

With similar ideas, can estimate sizes of:

- \( W = \Pi_{ab} (R) \) ...... [Sec. 7.4.2]
- \( W = \sigma_{a \geq b} (R) = \sigma_{b < a} (R) \);
  Ass. A and B independent =>
  \( T(W) = T(R)/(V(R, A) \times V(R, B)) \)
- \( W=R(A,X,Y) \sigma_{S(X,Y,B)} \); [Sec. 7.4.5]
  Ass. X and Y independent => ...
  \( T(W) = T(R)/T(S)/(\max(V(R,X), V(S,X)) \times \\
  \max(V(R,Y), V(S,Y))) \)
- Union, intersection, diff, .... [Sec. 7.4.7]

Note: for complex expressions, need intermediate estimates.
E.g. \( W = [\sigma_{A=a} (R1)]_j \bowtie R2 \)

Treat as relation U
\( T(U) = T(R1)/V(R1,A) \)
\( L(U) = L(R1) \)
Also need an estimate for \( V(U, A_i) \) !
Example:

\[ R \]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>cat</td>
<td>1</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>cat</td>
<td>2</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>dog</td>
<td>1</td>
<td>30</td>
<td>10</td>
</tr>
<tr>
<td>dog</td>
<td>1</td>
<td>40</td>
<td>20</td>
</tr>
<tr>
<td>bat</td>
<td>1</td>
<td>50</td>
<td>10</td>
</tr>
</tbody>
</table>

\[ V(R,A) = 3 \]
\[ V(R,B) = 1 \]
\[ V(R,C) = T(R) = 5 \]
\[ V(R,D) = 3 \]

\[ U = \sigma_{A=x}(R) \]

\[ V(U, D) = V(R, D) / T(R, A) \]

Outline/Summary:

- Estimating cost of query plan
  - Estimating size of results - done!
  - Estimating # of IOs - last week
- Generate and compare plans
  - skip this
- Execute physical operations of query plan
  - Sketched last week