Data Engineering
Query Optimization (Cost-based optimization)

Shivnath Babu
Query Optimization Problem

Pick the best plan from the space of physical plans
Cost-Based Optimization

• Prune the space of plans using heuristics
• Estimate cost for remaining plans
  – Be smart about how you iterate through plans
• Pick the plan with least cost

Focus on queries with joins
Heuristics for pruning plan space

• Predicates as early as possible
• Avoid plans with cross products
• Only left-deep join trees
Physical Plan Selection

Logical Query Plan

P1 | P2 | .... | Pn

C1 | C2 | .... | Cn

Pick minimum cost one

Physical plans

Costs
Review of Notation

- \( T(R) \) : Number of tuples in \( R \)
- \( B(R) \) : Number of blocks in \( R \)
Simple Cost Model

\[
\text{Cost} \ (R \ \bowtie \ S) = T(R) + T(S)
\]

All other operators have 0 cost

Note: The simple cost model used for illustration only
Cost Model Example

Total Cost: \( T(R) + T(S) + T(T) + T(X) \)
Selinger Algorithm

• *Dynamic Programming* based

• Dynamic Programming:
  – General algorithmic paradigm
  – Exploits “principle of optimality”
  – Useful reading:
    • Chapter 16, *Introduction to Algorithms*, Cormen, Leiserson, Rivest
Principle of Optimality

Optimal for “whole” made up from optimal for “parts”
Principle of Optimality

Query:  \( R1 \bowtie R2 \bowtie R3 \bowtie R4 \bowtie R5 \)

Optimal Plan:
Principle of Optimality

Query: $R1 \bowtie R2 \bowtie R3 \bowtie R4 \bowtie R5$

Optimal Plan:

Optimal plan for joining $R3, R2, R4, R1$
Principle of Optimality

Query:  $R1 \bowtie R2 \bowtie R3 \bowtie R4 \bowtie R5$

Optimal Plan:

Optimal plan for joining $R3, R2, R4$
Exploiting Principle of Optimality

Query: \( R1 \Join R2 \Join \ldots \Join Rn \)

- Optimal for joining \( R1, R2, R3 \)
- Sub-Optimal for joining \( R1, R2, R3 \)
Exploiting Principle of Optimality

A sub-optimal sub-plan cannot lead to an optimal plan
Selinger Algorithm:

Query:  $R1 \bowtie R2 \bowtie R3 \bowtie R4$

Progress of algorithm
Notation

OPT ( \{ R1, R2, R3 \}):

Cost of optimal plan to join $R1, R2, R3$

T ( \{ R1, R2, R3 \}):

Number of tuples in $R1 \bowtie R2 \bowtie R3$
Selinger Algorithm:

OPT ( \{ R1, R2, R3 \} ):

\[
\min \begin{cases}
OPT ( \{ R1, R2 \} ) + T ( \{ R1, R2 \} ) + T(R3) \\
OPT ( \{ R2, R3 \} ) + T ( \{ R2, R3 \} ) + T(R1) \\
OPT ( \{ R1, R3 \} ) + T ( \{ R1, R3 \} ) + T(R2)
\end{cases}
\]

Note: Valid only for the simple cost model
Selinger Algorithm:

Query:  $R1 \bowtie R2 \bowtie R3 \bowtie R4$

Progress of algorithm
Selinger Algorithm:

Query:  \( R1 \bowtie R2 \bowtie R3 \bowtie R4 \)

Progress of algorithm
Selinger Algorithm:

Query:  \( R1 \bowtie R2 \bowtie R3 \bowtie R4 \)

Optimal plan:
More Complex Cost Model

• DB System:
  – Two join algorithms:
    • Tuple-based nested loop join
    • Sort-Merge join
  – Two access methods
    • Table Scan
    • Index Scan (all indexes are in memory)
  – Plans pipelined as much as possible

• Cost: Number of disk I/O s
Cost of Table Scan

Table Scan

Cost: B (R)
Cost of Clustered Index Scan

Cost: B (R)
Cost of Clustered Index Scan

R.A > 50

Index Scan

Cost: B (X)
Cost of Non-Clustered Index Scan

Index Scan

Cost: $T(R)$
Cost of Non-Clustered Index Scan

Index Scan

Cost: \( T(X) \)

R

R.A > 50

X
Cost of Tuple-Based NLJ

Cost for entire plan:

\[ \text{Cost (Outer)} + T(X) \times \text{Cost (Inner)} \]
Cost of Sort-Merge Join

Cost for entire plan:

\[ \text{Cost (Right)} + \text{Cost (Left)} + 2 (B (X) + B (Y)) \]
Cost of Sort-Merge Join

Cost for entire plan:

\[ \text{Cost (Right)} + \text{Cost (Left)} + 2 \frac{B(Y)}{Y} \]
Cost of Sort-Merge Join

Cost for entire plan:

Cost (Right) + Cost (Left)

Sorted on R2.A

Sorted on R1.A

R1.A = R2.A

R1

R2
Cost of Sort-Merge Join

Bottom Line: Cost depends on sorted-ness of inputs
Principle of Optimality?

Query: \( R1 \bowtie R2 \bowtie R3 \bowtie R4 \bowtie R5 \)

Optimal plan:

Plan X

\( \text{SMJ} \)

(R1.A = R2.A)

Scan

R1

Is Plan X the optimal plan for joining R2, R3, R4, R5?
Violation of Principle of Optimality

Suboptimal plan for joining R2,R3,R4,R5

Optimal plan for joining R2,R3,R4,R4
Principle of Optimality?

Query: \( R1 \bowtie R2 \bowtie R3 \bowtie R4 \bowtie R5 \)

Optimal plan:

- **SMJ**
  - \( (R1.A = R2.A) \)
- **Scan**

Can we assert anything about plan X?
Weaker Principle of Optimality

If plan X produces output sorted on R2.A then plan X is the **optimal plan** for joining R2, R3, R4, R5 that produces output sorted on R2.A

If plan X produces output unsorted on R2.A then plan X is the **optimal plan** for joining R2, R3, R4, R5
Interesting Order

• An attribute is an interesting order if:
  – participates in a join predicate
  – Occurs in the Group By clause
  – Occurs in the Order By clause
Interesting Order: Example

Select *  
From R1(A,B), R2(A,B), R3(B,C)  

Modified Selinger Algorithm

{R1,R2,R3}

{R1,R2}  {R1,R2}(A)  {R1,R2}(B)  {R2,R3}  {R2,R3}(A)  {R2,R3}(B)

{R1}  {R1}(A)  {R2}  {R2}(A)  {R2}(B)  {R3}  {R3}(B)
Notation

\{R_1,R_2\} (C)

Optimal way of joining R1, R2 so that output is sorted on attribute R2.C
Modified Selinger Algorithm