Overview of Query Optimization in Relational Systems

Presenter: Cong Lin
Discussion: Wenhao Xu
Surajit Chaudhuri
Microsoft Research

Outline

• Part I Motivation
• Part II Introduction
• Part III Search Space
• Part IV Statistics & Cost Estimation
• Part V Enumeration Architecture
• Part VI Summary

Part I Motivation

Motivation

• An overview of current query optimization techniques
• Gives fundamentals of query optimization

Part II Introduction

Query can be complicated
But Optimizer does a lot!

2 key components for query evaluation
– Query optimizer
– Query execution engine

Query Execution Engine

• Implements a set of physical operators
• Physical operator
  – Input: one or more data streams
  – Output: (through operation) one output data stream
  – Operator Types: (external) sort, sequential scan, index scan, nested loop join, sort-merge join
  – Responsibility: Execution of operator tree that generates answers to the query
Example Operator Tree

Key Idea: Query Optimization as a Search Problem
- Requirement
  - Search space
  - Cost estimation technique
  - Enumeration algorithm
- Aim
  - Search for the best (or not the worst) plan

Transformation: Commuting Between Operators
- Generalized Join Sequencing
  - Join operations are commutative and associative
    - Linear joins vs Bushy join

Transformation: Commuting Between Operators
- Outer Join and Join

Query Optimizer
- Input: Parsed representation of SQL query
- Output: An execution plan to Query Execution Engine
- Responsibility: Generate an efficient execution plan from the Search Space of possible execution plans

Part III Search Space
- Depends on:
  - Equivalence performing algebraic transformations
  - Physical operators supported in an optimizer
- Transformations may not reduce cost and therefore must be applied in a cost-based manner to ensure a positive benefit

Can’t explore all options
Transformation: Commuting Between Operators

- Group-By and Join

![Diagram of Group-By and Join]

A Hairy Discussion

- Does the formulation of a query affect the execution of that query? Can users optimize their queries’ execution through better syntax?
- Bushy Joins: Is it naive to just leave them out of the search? Why do we always only consider linear joins? When would this cause problems? How could we incorporate bushy joins into our search?

Transformation: Multi-Block Query to Single-Block

- Merging Views

\[
Q = \text{Join}(R, V) \quad \text{View} \quad V = \text{Join}(S, T)
\]

\[
Q = \text{Join}(R, \text{Join}(S, T))
\]

- Merging Nested Subqueries

![Diagram of Merging Nested Subqueries]

Part IV Statistics and Cost Estimation

Statistics

- Example
  - # tuples in table, # physical pages used by table, histograms
- Estimation
  - Method: Sampling
  - Problem: Estimating distinct values is provably error prone
- Propagation
  - Propagate information through operators
  - Problem: Can be difficult as some inaccurate assumptions might be made

Cost Estimation

- Principles
  - Must be accurate because optimization is only as good as its cost estimates
  - Must be efficient as it is repeatedly invoked by the optimizer
- Frameworks
  - Collect statistical summaries of data stored
  - Given an operator and statistical summaries of its input streams, determine
  - statistical summary of output data stream
  - estimated cost of executing the operation

A Statistical Discussion

- Some estimated statistics are provably erroneous. Is it then worth estimating? If so, what sort of strategy should we adopt when using estimates with known problems?
Cost Computation

• Costs
  – CPU
  – I/O
  – Communication costs (parallel & distributed)
• Problem
  – Difficulty in determining best cost estimator
  – Inaccuracy of statistical summary propagation

Part V Enumeration Architectures

• Responsibility
  – Explores search space to pick cheap execution plan
    • Enumerators concentrate on linear join sequences
      rather than bushy join sequences
      due to the size of the search space including bushy join sequences
  – Be extensible

Extensible Optimizers

• Adapt to changes in search space
  – New transformations
  – Addition of new physical operators
  – Changes in cost estimation techniques
• Solutions
  – Use generalized cost functions & physical properties with operator nodes
  – Use rule engine that allows transformations to modify the query
    expression or the operator trees
  – Expose “knobs” to tune behavior of system

Materialized Views

• Views cached by database system
• Advantage
  – Reduce the cost of executing the query
• Problems
  – Reformulating query
    • General problem is undecidable
    • Determining effective sufficient conditions is nontrivial
    • To be extended for complex queries

Part VI Summary

• Query optimization as a search problem
  A search space-hard
  Cost estimation technique
  An enumeration algorithm

No one knows the best execution plan

Ending Discussions

• Most of us have decided that the Relational Model is the way to go. These papers, however, show that under the hood are some scary problems and black magic. Is this surprising? Why (not)?
• This paper is from 98, System R paper is from ~ 20 years earlier. How has query optimization changed in that span of time? Is the amount/direction of progress surprising? Do you expect much change in the years since the printing of this paper?
• What other areas of Computer Science is query optimization like? Could it benefit from ideas from other areas? How?
Thanks~