Access Path Selection in a Relational Database Management System

Presenter: Dutch Meyer
Discussion: Mike "Debo" DiBernardo

Somewhat based on slides from Stephen Ingram, modified by Rachel Pottinger.

Key Points

- Finding the right path
- Optimization based on estimation
- Tackling the problem space
- System R - the basis for the modern query optimizer

Find the Best path

- SELECT * FROM A,B,C WHERE A.n = B.n AND B.m = C.m
- A = 100 tuples
- B = 50 tuples
- C = 2 tuples
- Which plan is cheaper?
  - Join(C, Join(A, B))
  - Join(A, Join(B, C))

How hard is this problem?

Relation Optimization

- With a single relation, it's not so bad
  - We can consider all paths

Join Optimization

- 2-Way joins
- N-Way joins
- How many permutations of N-Way joins could there possibly be – N! !

Optimization

- So the general case search space is big.
- Possible aids:
  - Heuristics
  - Statistics
  - Dynamic Programming

Heuristics!

- Hard coded rules
- Easy to understand
- Cheap to compile, use
- Example: Predicates

Or no heuristics?

- Inflexible
- Non General

Statistics!

- Generalized solution
- Example: Data size, type, distribution

Or no Statistics?

- Difficult to compile
- Difficult to store

Dynamic Programming

- Simplify search space by reusing solutions
- Reduce storage costs to 2^N
  - Not so bad, coming from N!

Or…

- Dynamic programming improves the effectiveness of statistics.
- Careful heuristics further limit the search space.
- Proper ordering allows the methods to work together.

Ordering

- Defn: Interesting Ordering – Orderings that we are interested in returning
- Example: Group-By. Order-By.

Key Contributions

- Cost based optimization
  - Statistics
  - CPU utilization (for sorts, etc.)
- Dynamic programming approach
- Interesting Orders