Local Cost Estimation for Global Query Optimization in a Multidatabase System

Dr. Qiang Zhu
The University of Michigan - Dearborn

Outline

• Introduction
• Challenges for QO in MDBS
• Query Sampling Method
• Qualitative Approach
• Fractional Analysis and Probabilistic Approach
• Conclusions
1. Introduction

- **Multidatabase System (MDBS)**
  - **What:** a distributed system that integrates data from various *pre-existing* databases managed by *heterogeneous* local DBMSs
  - **Key feature:** *local autonomy*

- **Why Global Query Optimization (GQO)**
  
  MDBS $\Rightarrow$ Global query
  $\Rightarrow$ Global query optimization
  $\Rightarrow$ Good overall *system performance*
2. Challenges for Global Query Optimization in MDBS

• **GQO for Traditional DDBS**
  
  developed for a *homogeneous* environment
  
  – **Techniques:**
    - optimal vs heuristic searches
    - join vs semijoin strategies
    - static vs dynamic optimization
    - sequential vs parallel execution
  
  ⇒ many *not suitable* for an MDBS

• **Challenges for GQO in MDBS**

  Caused by *local autonomy*:
  
  – Some local optimization information may *not be available* at global level
  – *Different and changing* local capabilities are assumed
  
  – *Heterogeneous* data formats and models may be used
  – Implementation of local DBMSs *cannot be changed*
  – *More constraints* need to be considered during global query optimization
  
  ⇒ **Crucial challenge:** *incomplete local information*
• **Proposed Techniques**
  – Calibration method (*Du et al. 92*)
  – Fuzzy approach (*Zhu et al. 94*)
  – Extended calibration method (*Gardarin et al. 96*)
  – Cost vector database approach (*Adali et al. 96*)
  – Generic cost model approach (*Naache et al. 98*)
  – Garlic approach (*Roth et al. 99*)
  – Query sampling method (*Zhu et al. 94 & 98*)
  – Qualitative approach (*Zhu et al. 00*)
  – Fractional analysis approach (*Zhu et al. 00*)
  – Probabilistic approach (*Zhu et al. 00*)

---

3. Query Sampling Method

• **Key idea**
• **Classification of Queries**
  – Two extreme cases

   - **Information available**
     - **Characteristics of queries**: e.g., unary queries, join queries
     - **Characteristics of operand tables**: e.g., number of tuples, indexed columns
     - **Characteristics of local DBMSs**: e.g., supported access methods

   – **Classification goal**: each query class corresponds to one access method
– **Classification rules:** based on common rules for access methods, such as
  - A unary query and a join query use **different** access methods
  - A clustered-index-based method is **preferred** to an (non-clustered) index-based method
  - An index-based method is **preferred** to a sequential scan method
  - A clustered-index-based method is **chosen** for a query if it has a conjunctive term that can use a clustered-index, e.g., for query
    \[ \sigma_{R.a=2 \land (R.b<3 \lor R.c<4)}(R) \]
    clustered - indexed
  etc.

– **Classification methods**
  - **Bottom-up** method
  - **Top-down** method

– **Example of classification**

\[ G = \{ \text{ all queries } \} \]

\[ G = G_1 \cup G_2 \]

- unary queries
- join queries

\[ G_1 = G_{11} \cup G_{12} \cup G_{13} \]
\[ G_2 = G_{21} \cup G_{22} \cup G_{23} \]
\[
G_{11} = \{ \pi_{b_{1}, \ldots, b_{y}}(\sigma_{F_{i_{1}} \land \ldots \land F_{u_{1}}}(R)) \mid \text{at least one } F_{i_{j}} (1 \leq i \leq m) \}
\]

is \(R.a = C\), where \(R.a\) is a clustered - indexed column \}
\[
G_{12} = \{ \pi_{b_{1}, \ldots, b_{y}}(\sigma_{F_{i_{1}} \land \ldots \land F_{u_{1}}}(R)) \mid \text{at least one } F_{i_{j}} (1 \leq i \leq m) \}
\]

is \(R.a = C\), where \(R.a\) is an indexed column \} - \(G_{11}\)
\[
G_{13} = G_{1} - (G_{11} \cup G_{12})
\]

….. classification can be further refined

- **Relevant issues**
  - Composition of rules
  - Redundancy of rules
  - Classification algorithms
  - Membership testing

---

- **Sampling and cost formulas**
  - **Sampling method:** mixed judgment and probability sampling
    - Use some knowledge to restrict a query class to a representative subset
    - Apply one or more types of probability sampling, e.g., simple random sampling and cluster sampling, to draw a sample
  - **Example**
    
    For \(G_{11} : \{ \pi_{b_{1}, \ldots, b_{y}}(\sigma_{F_{i_{1}} \land \ldots \land F_{u_{1}}}(R)) \} \)
    
    \(F_{i_{1}} : R.a = C \quad \text{-- key conjunctive term} \)
    
    clustered - indexed
    
    Sample : \(SP_{11} = \{ \pi_{a}(\sigma_{R.a=C \land F}(R)) \} \)
– Derivation of cost formulas

■ Explanatory variables

Basic set:
- Cardinality of operand table(s)
- Cardinality of result table
- Size of intermediate result(s)

Secondary set:
- Operand tuple length(s)
- Result tuple length
- Characteristics of index tree
- ...... etc

■ Selection of variables

A mixed forward and backward procedure
- Backward: remove insignificant variables from the basic set
- Forward: add more significant variables from the secondary set

■ Example of cost formula

For a unary query class:

\[ C_1 = \beta_1 + \beta_2 \times N + \beta_3 \times N \times S \]

\( N \) -- operand table size
\( S \) -- selectivity

coefficient
- **Estimation of coefficients**
  - Multiple regression analysis

- **Validation of cost formulas**
  - Standard error of estimation
  - Coefficient of total determination
  - F-test
  - Test queries

- **Experiment results on Oracle**
• Experiment results on DB2

For more details:

Q. Zhu and P.-A. Larson:

4. Qualitative Approach

- **Motivation:** query cost may change dramatically in a dynamic environment
- **Types of dynamic factors**
  - **Frequently-changing factors**
    E.g., CPU load, I/Os per sec., amount of memory being used
  - **Occasionally-changing factors**
    E.g., DBMS configuration parameters, data physical distribution, physical memory size
  - **Steady factors**
    E.g., CPU speed, DBMS release and type

Table R7(a1, a2, ..., a9) has 50,000 tuples of random numbers

Query:
```
select a1, a5, a7
from R7
where a3 > 300 and a8 < 2000
```
• Capture dynamic factors in cost models
  – Steady factors
    - usually don’t cause problem
  – Occasionally-changing factors
    - periodically rebuild the cost model via the query sampling method
  – Frequently-changing factors
    - infeasible to rebuild frequently
    - difficult to include all dynamic variables: (1) too many; (2) unknown interaction forms

⇒ use a new qualitative approach

• Key idea
  – Consider the combined effect of all dynamic factors on query cost
  – Use the cost of a probing query to measure the system contention level
  – Divide the contention level into a number of discrete states: e.g., high contention, medium contention, low contention, no contention, etc.
  – Use a qualitative variable in a cost model to indicate contention states
• **Cost model with qualitative variable**
  
  – **Qualitative variable**
  
  ▪ a qualitative variable with **m states** is represented by **m-1 indicator variables**
    
    \[
    S_1 : Z_1 = 1, Z_2 = 0, \ldots, Z_{m-1} = 0 \\
    S_2 : Z_1 = 0, Z_2 = 1, \ldots, Z_{m-1} = 0 \\
    \ldots \\
    S_m : Z_1 = 0, Z_2 = 0, \ldots, Z_{m-1} = 0
    \]
  
  – **Cost model**
  
  \[
  Y = (\beta_0^0 + \sum_{j=1}^{m-1} \beta_0^j Z_j) + \sum_{i=1}^n (\beta_i^0 + \sum_{j=1}^{m-1} \beta_i^j Z_j) X_i
  \]

• **System states determination**

  How to determine the system contention states?
  
  – **two extremes**: one state ↔ infinite states
  
  – determination via **iterative uniform partition with merging adjustment**
    
    ▪ **Phase I**: uniformly partition the range of probing query cost with an incremental number of states until \( \left| (R_{new}^2 - R_{old}^2) / R_{old}^2 \right| \) and \( \left| (s_{new} - s_{old}) / s_{old} \right| \) are sufficiently small, where
      
      \[
      R^2 \quad \text{coefficient of total determination} \\
      s \quad \text{standard error of estimation}
      \]
- **Phase II**: merge two states $S_{k-1}$ and $S_k$ if no significant difference in coefficients for the cost model, i.e., if

$$r_k = \max_{r \in \{0, 1, 2, \ldots, n\}} \{|(\theta_i^{k-1} - \theta_i^k) / \theta_i^k|\}$$

is too small, where

$$\theta_i' = \beta_i^0 + \beta_i^j$$

-- adjusted coefficient of $X_i$ for state $S_j$

- **Experiment results on Oracle**

![Graph showing experiment results on Oracle](image)
• Experiment results on DB2

For more details:

Q. Zhu, Y. Sun, S. Motheramgari:

5. Fractional Analysis and Probabilistic Approach

- **Motivation:** a large (cost) query may experience multiple states during its execution, how to estimate its cost?

- **Simple solutions**
  - *single state analysis:* consider only one prevailing contention state, e.g., the initial state
  - *average cost analysis:* take average of costs in all contention states

⇒ **Better solutions?** Yes

- **Fractional Analysis Approach**
  - *Typical load curve*
- **Key idea**
  - Calculate the fraction of cost in each state and add them up
  - Let

\[ \Delta = \{S_1, S_2, \ldots, S_M\} \quad \text{all possible states} \]

\[ S^{(1)}, S^{(2)}, \ldots, S^{(N)} \quad \text{sequence of states occurred along the load curve} \]

\[ t^{(i-1)}, t^{(i)} \quad \text{starting and ending times for } S^{(i)} \]

\[ Q \quad \text{query starting at } t_Q^{(k)} \text{ in state } S^{(k)} \]

\[ T^{(k)} = (t^{(k)} - t_Q^{(k)}) \quad \text{max time interval for } Q \text{ in } S^{(k)} \]

\[ T^{(i)} = (t^{(i)} - t^{(i-1)}) \quad \text{for } i > k \quad \text{time interval for } S^{(i)} \]

\[ C(Q, S^{(i)}) \quad \text{cost estimate of } Q \text{ in } S^{(i)} \]
- **Case 1:** if \( C(Q, S^{(k)}) \leq (t^{(k)} - t_0^{(s)}) \), then \( C(Q) = C(Q, S^{(k)}) \)
- **Case 2:** if \( C(Q, S^{(k)}) > (t^{(k)} - t_0^{(s)}) \), then

estimated fraction of work done for \( Q \) in \( S^{(k)} \) is :

\[
\frac{(t^{(k)} - t_0^{(s)})}{C(Q, S^{(k)})}
\]

remaining fraction of work for \( Q \) is :

\[
[1 - \frac{(t^{(k)} - t_0^{(s)})}{C(Q, S^{(k)})}]
\]

- **Sub-case 1:**

if \( [1 - \frac{(t^{(k)} - t_0^{(s)})}{C(Q, S^{(k)})}] \ast C(Q, S^{(k+1)}) \leq (t^{(k+1)} - t^{(k)}) \),
then

\[
C(Q) = \left( t^{(k)} - t_0^{(s)} \right) + \left[ 1 - \frac{(t^{(k)} - t_0^{(s)})}{C(Q, S^{(k)})} \right] \ast C(Q, S^{(k+1)})
\]

\[
\text{work done in } S^{(k+1)}
\]

\[
\text{work done in } S^{(k+1)}
\]

- **General cost estimation formula:**

\[
C(Q) = \sum_{i=k}^{m} T^{(i)} + [1 - \sum_{i=k}^{m} T^{(i)} / C(Q, S^{(i)})] \ast C(Q, S^{(m+1)})
\]

where \( m \) is the minimum integer such that

\[
[1 - \sum_{i=k}^{m} T^{(i)} / C(Q, S^{(i)})] \ast C(Q, S^{(m+1)}) \leq T^{(m+1)}
\]

- **Assumptions:**
  - Load curve is **prior known**
  - Load changes **gradually**
– Experiment Results on Oracle

![Graph showing query cost over query number]

• **Probabilistic Approach**
  
  – **Motivation:** how to deal with a rapidly and randomly changing environment?
  
  – **Observations**
    
    - Occurrence of a contention state is a random phenomenon and governed by laws of probability
    
    - The sequence of occurrences of contention states can be considered as a Markov chain
    
    - Transition probability $P_{ij}$ for state $Si$ changing to state $Sj$ in the next step is inversely proportional to the distance between $Si$ and $Sj$

  – **Limit probability**
    
    $\pi_i = \lim_{n \to \infty} P_{ij}(n)$ — the limit probability, where $P_{ij}(n)$ is the probability for $Si$ changing to $Sj$ after $n$ steps
Properties

- Independent of initial state $S_i$
- Represent the long-run portion of time for the Markov chain being in the state

Satisfy the system of linear equations:

$$\pi_j = \sum_{i=1}^{M} \pi_i P_{ij} \quad \text{for} \quad j = 1,2,...,M \quad \text{subject to} \quad \sum_{j=1}^{M} \pi_j = 1$$

Cost formula

- Cost incurred in state $S_i$: $\pi_i \ast C(Q)$
- Fraction of work in state $S_i$: $(\pi_i \ast C(Q)) / C(Q, S_i)$
- Identity: $\sum_{i=1}^{M} (\pi_i \ast C(Q)) / C(Q, S_i) = 1$

- Cost formula: $C(Q) = 1/[\sum_{i=1}^{M} \pi_i / C(Q, S_i)]$

- Experiment results on Oracle
• For more details:

Q. Zhu, S. Motheramgari, Y. Sun:

➢ “Cost Estimation for Queries Experiencing Multiple Contention States in Dynamic Multidatabase Environments”, *Knowledge and Information Systems*, Springer Verlag, 2001 (to appear)

6. Conclusions

• A crucial challenge for global query optimization in an MDBS is that some local cost information may not be available at the global level

• A number of techniques have been proposed to estimate local cost parameters at the global level in an MDBS

• Query sampling method is useful in estimating query costs in a static MDBS environment
• **Qualitative approach** is useful in estimating costs of queries experiencing one contention state in a dynamic environment.

• **Fractional analysis approach** is useful in estimating costs of queries experiencing multiple contention states in a gradually changing dynamic environment.

• **Probabilistic approach** is useful in estimating costs of queries experiencing multiple contention states in a rapidly changing dynamic environment.

• **Further research** needs to be done in future.

**For more information:**

http://www.engin.umd.umich.edu/~qzhu