MOBILE AGENT COOPERATION METHODS IN HYBRID QUERY OPTIMIZATION

Thanda Win, Khin Mar Lar Tun
University of Computer Studies, Yangon
ucsy24@most.gov.mm

ABSTRACT

The database field has developed very powerful technologies for finding efficient execution plans for declaratively specified queries. Moreover, database queries have increasingly complex in the age of the distributed DBMS (DDBMS). In order to optimize queries accurately, sufficient information must be available to determine which data access techniques are most effective. The role of query optimization is to find a strategy close to optimal. To optimize the query efficiently, it is important to choose the site to execute. For site selection, the statistical information of underlining relations is essential. In this paper, we propose the query optimization model based on mobile agents. This model provides a strategy for executing each query over the network in the most cost-effective way. Our proposed system focuses on these modules: *Query Decomposition*: decomposes the input query into mono-relation query using the detachment techniques, *Optimization Plan*: computes all possible plans using statistical database information and intermediate relation size by cooperating with mobile agents and chooses the optimized plan and *Execution Plan*: executes the optimized plan by using mobile agent and send results to client.

Keywords: Distributed Database, Multidatabase, Mobile Agents, Distributed Computing

1. INTRODUCTION

Multidatabase systems are widely used for distributed, heterogeneous, and autonomous data source integration. Typically, such a system consists of a global component and a collection of local components. The global component hides the underlying heterogeneity and provides users a uniform global information access method. In order to preserve local data source autonomy while providing full database functionality, the global component usually maintains a global schema that contains local schema information. Problems arise as the size of the multidatabase grows. Maintaining and manipulating the larger and larger global schema becomes difficult.

The query processing problem is much more difficult in distributed environments than in centralized ones because a larger number of parameters affect the performance of distributed queries. In particular, the relations involved in a distributed query may be fragmented and/or replicated, thereby inducing communication overhead costs. Furthermore, with many sites to access, query response time may become very high. Many of the researchers have described query processing and optimization, and used the mobile agents and agents in distributed query processes.

Mobile agents are software processes which can autonomously migrate from one host to another during their execution [9]. By transmitting executable programs between (possibly heterogeneous) machines, agent-based computing introduces an important new paradigm for the implementation of distributed applications in an open and dynamically changing environment.

In this paper, we present mobile agent-based hybrid query optimization system. Firstly, each input query is decomposed into mono-relation queries using detachment and substitution techniques. Secondly, all mono-relation queries are used as input. Then constructs all possible query execution plans. We use mobile agent technology to get the statistical database information of distributed databases and to carry mono-relation query. Those agents are send to corresponding databases locate at different sites. Then the local databases computes estimate cost of all mono-relation queries and size of intermediate relations. These estimate results are then sent back to optimizer and computes estimate cost of all possible query plan and chooses the best plan. Finally, executes the best plan by using mobile agent and sends results back to the user.

The rest of the paper is organized as follows: in section 2, we briefly present agent-based query optimization. Section 3 describes our proposed system architecture. Section 4 concludes the paper.

2. AGENT-BASED QUERY OPTIMIZATION

The query optimizer is usually seen as three components: a search space, a cost model, and a search strategy [11]. Figure1 shows the query optimization process. The last layer is performed by all the sites having fragments involved in the query. Each subquery executing at one site, called a local query, is then optimized using the local schema of the site. At this time, the algorithms to perform the relational operations may be chosen.

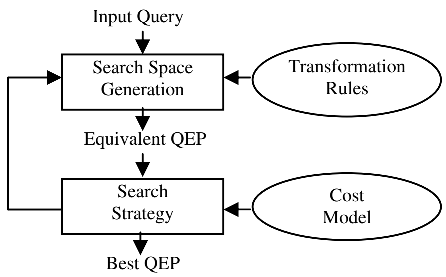

Figure1. Query Optimization Process

Three are three optimization timing in [11]: *static*, *dynamic* and *hybrid* query optimizations.

- *Static query optimization*: is performed at query compilation time and uses the database statistics to estimate the sizes of the intermediate relations.
- *Dynamic query optimization*: proceeds at query execution time. For any execution, the choice of the best next operation can be supported accurate results of the operations executed previously. Therefore, database statistics are not needed to estimate the size of intermediate results.
- *Hybrid query optimization*: is combined static and dynamic query optimization.

Several researchers proposed mobile agent contributions to the distributed dynamic query optimization. In [10], F. Morvan et. al. studied the contribution of the execution model based on mobile agents to the distributed dynamic query optimization. They proposed and evaluated three methods allowing cooperation between the agents participating in the query evaluation process. These methods integrated in the agent decision policy allow the agents to react to estimation errors or to the resource unavailability. The principal contribution of this study is that the proposed strategies minimize the number of messages exchanged between the agents.

In [1], S. Das et. al. presented a mobile agent based engine, Agent-based Complex QUerying and Information Retrieval Engine (ACQUIRE), for retrieving data from heterogeneous, distributed data sources. ACQUIRE acts as an interface agent or softbot by accepting a high-level user query and decomposing this query into a series of subqueries. An optimized data retrieval plan is then generated from these subqueries and a series of mobile agents is spawned that collect and return the required data from the distributed data archives. The system reduces the amount of data transferred across the network by optimizing the retrieval strategy and by utilizing a mobile agent's ability to process data remotely.

Mariposa system [14] proposed a large scale distributed and static query optimization. In this system, only the optimization phase is decentralized, they have no run time optimization. Many researchers presented their static or dynamic query optimization model using mobile agents.

In our proposed system, we use hybrid query optimization because we use both the database statistics data and results of the mono-relation query which are executed at each local site. The optimizer chooses the best plan from all possible query execution plans based on the size of intermediate mono-relation queries.

Mobile agents have several advantages such as reducing network load, overcoming network latency, executing asynchronously and autonomously, adapting dynamically, and robust and fault tolerant. Because of these advantages, we use mobile agent technology to collect the statistical database information of distributed databases and carry mono-relation query and send the corresponding database locate at different sites and executes the best query plan.

2.1. Proposed Query Optimization Algorithm

```
Input: MRQ: multi-relation query
Output: result of the optimized multi-relation
query

Begin
   for each detachable OVQi in MRQ do
           run (OVQi)   ------------------------- (1)
   end-for
   for each database location sites (s-1) do
     { s is the number of databases located at
different sites }
      Construct all possible query plans [QPs]
                              ------------------------- (2)
   end-for
   while n≠0 do  {n is the number of mono-
                  relation queries) -----------(3)
    begin
      for each pair (MQ, S) in Mono-relation
      query-site –list by using mobile agent do

        Move mono-relation query MQ to site S
                              ----------------------- (3.1)
        Get size of intermediate relation [SIR]
                              ----------------------- (3.2)
      end-for
      n←n-1
    end-while
    for each [QPs] do
    compute the estimate cost of all mono-
    relation  queries using [SIR] -------------- (4)
    end-for
    choose the best execution plan (BEP)----(5)
    run (BEP) ----------------------------------- (6)
End. {QOA}
```

In our proposed query optimization algorithm, firstly detaches a given multi-relation query (MRQ) to obtain one variable query (OVQ) called mono-relation queries (MQ) (step 1). Then construct all possible query plans (step 2). In (step 3.1), mono-relation query is moved to corresponding site S by using mobile agent. Get size of intermediate relation (SIR) in (step 3.2). Compute the estimate cost of all mono-relation queries using SIR (step 4). In (step 5), chooses the best query execution plan. (Step 6) executes that optimized plan.

2.2. Case Study

Consider the expression of the query in relational calculus using the SQL syntax is [11]:

```
SELECT  ENAME
FROM    EMP, ASG, PROJ
WHERE   EMP.ENO=ASG.ENO
AND     ASG.PNO=PROJ.PNO
AND     PNAME= "CAD/CAM"
```

Assumption: There are no fragmented databases and each database locates at different sites. Figure2 shows the join graph of distributed query.

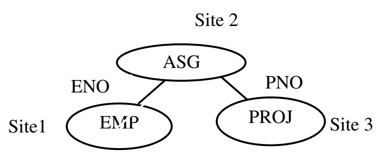

Figure2. Join Graph of Distributed Query

According to the proposed algorithm, given multi-relation query is detached into three mono-relation queries. They are as follows:

The relational algebra query for mono-relation **query1**:

$$\boldsymbol{\Pi_{ENO,\,ENAME}\ (EMP)}$$

The relational algebra query for mono-relation **query2**:

$$\boldsymbol{\Pi_{PNO,\,ENO}\ (ASG)}$$

The relational algebra query for mono-relation **query3**:

$$\boldsymbol{\sigma_{PNAME=\ \text{"CAD/CAM"}}\ (PROJ)}$$

Then construct all possible query execution plans. Four possible execution plans obtained. They are as follows:

Plan 1:

The relational algebra query for **plan 1** is:

$$\boldsymbol{\Pi_{ENAME}\ (EMP \bowtie_{ENO} (ASG \bowtie_{PNO} PROJ'))}$$

where PROJ'=$\sigma_{PNAME=\ \text{"CAD/CAM"}}$ (PROJ)

Step 1: Site3 → Site2

Site 3 computes **PROJ'=$\boldsymbol{\sigma_{PNAME=\ CAD/CAM"}}$(PROJ)**

In step1, mobile agent sends PROJ' to site2.

Site 2 computes **ASG' = ASG ⋈ PROJ'**

Step 2: ASG '→ Site1

In step2, mobile agent carries the intermediate relation ASG' to site1.

Site1 computes **ASG' ⋈ EMP**

Plan 2:

The relational algebra query for **plan 2** is:

$$\boldsymbol{\Pi_{ENAME}\ (EMP \bowtie_{ENO} (ASG \bowtie_{PNO} PROJ'))}$$

where PROJ'=$\sigma_{PNAME=\ \text{"CAD/CAM"}}$ (PROJ)

Step 1: ASG → Site3

In step1, mobile agent is used to send relation ASG to site3.

Site3 computes **ASG'=ASG ⋈ PROJ'**

Step 2: ASG'→Site1

In step2, mobile agent carries intermediate relation ASG' to site1.

Site1 computes **ASG' ⋈ EMP**

Plan 3:

The relational algebra query for **plan3** is:

$$\boldsymbol{\Pi_{ENAME}\ (PROJ' \bowtie_{PNO} (EMP \bowtie_{ENO} ASG))}$$

where PROJ'=$\sigma_{PNAME=\ \text{"CAD/CAM"}}$ (PROJ)

Step 1: EMP→Site2

In step1, mobile agent carries relation EMP to site2.

Site2 computes **EMP'=EMP ⋈ ASG**

Step 2: EMP'→Site3

In step2, mobile agent is used to send the intermediate relation EMP' to site3.

Site3 computes **EMP' ⋈ PROJ'**

Plan 4:

The relational algebra query for **plan 4** is:

$$\boldsymbol{\Pi_{ENAME}\ (EMP \bowtie_{ENO} (PROJ' \bowtie_{PNO} ASG))}$$

where PROJ'=$\sigma_{PNAME=\ \text{"CAD/CAM"}}$ (PROJ)

Step 1: PROJ'→Site2

In step1, mobile agent sends the mono-relation query PROJ' to site2.

Step 2: EMP→Site2

In step2, another mobile agent carries relation EMP to site2.

Site2 computes **PROJ' ⋈ ASG ⋈ EMP**

To select one optimized plan from these possible strategies, sizes of relations and intermediate relations are estimated using mobile agents. Assume the sizes of relations as size (EMP) =8, size (ASG) =10 and size (PROJ) =4 and size (PROJ') =1. Strategy 4 is the least total cost considering data transfer only. Therefore, plan 4 is the optimized plan in this case.

3. PROPOSED MOBILE AGENT COOPERATION IN QUERY OPTIMIZATION SYSTEM

In order to measure the performance of mobile agent cooperation in query optimization, the prototype system is built. Figure3 shows Overall Design of Proposed Query Optimization Model. In our proposed model, we use hybrid query optimization. We use database statistic information and results of mono-relation query. This prototype system contains three main modules: *Query Decomposition*, *Optimization Plan* and *Execution Plan*.

In ***Query Decomposition module***, the input query is decomposed into mono-relation query using detachment and substitution techniques.

Optimization Plan step uses all mono-relation queries as input and constructs all possible query plans. Then compute the estimate cost of each query execution plan. In order to estimate cost of query plan, mobile agent is used to collect the statistical database information of distributed databases. Each mobile agent carries mono-relation query and is sent to corresponding databases locate at different sites. Agents assign the mono-relation query to local database which execute the input query and returns result to agent. Agent then sends back the size of immediate result of mono-relation query. When all the necessary information have been received, estimate cost of all mono-relation queries are computed and optimizer chooses the best execution plan.

Execution Plan accepts the best execution plan and decides whether the input optimized plan can be performed in serial or parallel plan. Executes optimized plan using mobile agent and sends results back to client.

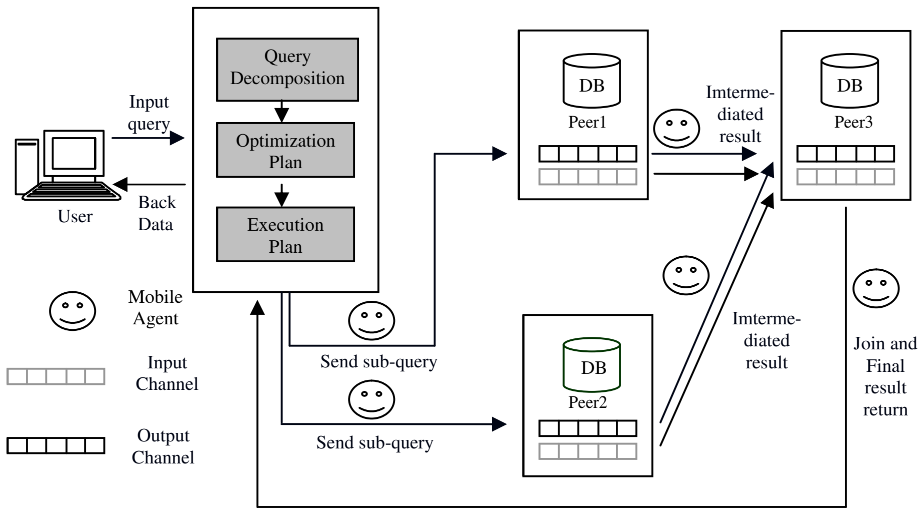

Figure3. Overall Design of Proposed Query Optimization Model

4. CONCLUSION

We discuss the distributed query processing steps, agent-based query optimization, and our proposed system architecture. The most vital part of distributed query system is decomposition of the input query into multiple fragment queries to retrieve the information at different sites according to data location. Another important step to consider is how to execute the algebraic operations such as JOIN and send the partial result data to accumulate final data and return to the user. In our proposed system, we use hybrid query optimization to get the statistical database information of distributed databases, carry mono-relation query to corresponding site and compute all possible query execution plans by cooperating mobile agents. We construct our new algorithm that make use of mobile agent to accumulate the data fragment information to get the optimize query method. We are now currently under construction of proposed prototype and in future, we will evaluate the performance of our query optimization method using mobile agents.

REFERENCES

[1] S. Das, K. Shuster, C. Wu, “Agent-based Complex Querying and Information Retrieval Engine”, Proceedings of the First International Joint Conference on Autonomous Agents and Multi-agent Systems (AAMAS), Bologna, Italy, 2002.

[2] C. Evrendilek and A. Dogac. “Query Decomposition, Optimization and Processing in Multidatabase Systems”. Software Research and Development Center, Scientific and Technical Research Council of Turkiye, Middle East Technical University.

[3] T. Howes, M. C. Smith, G. S. Good, and T. A. Hwes, “Understanding and Deploying LDAP Directory Services”, MacMillan, January 1999.

[4] M. N. Huhns, and M.P. Singh, “Ontologies of Agents”, IEEE Internet Computing, 1997.

[5] M. N. Huns, and M.P. Singh, “Managing Heterogeneous Transaction Workflows with Cooperating Agents”, N.R. Jennings and

M.Woolridge (eds.), Agent Technology, Sprinfer, 1998.

[6] R. Jones and J. Brown "Distributed Query Processing via mobile Agents", http://www.cs.umd.edu/~keleher/818s97/ryan/paper.html, 1997.

[7] D. Kossmann, "The State of the Art in Distributed Query Processing", Submitted to ACM Computing Surveys.

[8] Y. Labrou and T. Finin, " A Proposal for a New KQML Specification", Technical Report TR CS-97-03, Computer Science and Electrical Engineering Department, University of Marylond Baltimore County, Baltimore, MD 21250, February 1997.

[9] D. B. Lange and M. Oshima, "Programming and Deploying Java™ Mobile Agents with Aglets™", Addison Wesley, Reading, MA, 1998.

[10] F. Morvan, M. Hussein, A. Hameurlain. "Mobile Agent Cooperation Methods for Large Scale Distributed Dynamic Query Optimization". In Proceedings of the 14th International Workshop on Databse and Expert Systems Applications (DEXA'03), 2003.

[11] M. T. Ozsu and P. Valduriez, "Principles of Distributed Database Systems", Second Edition, 1999.

[12] A. Sahuguet, B. C. Pierce and V. Tannen, "Distributed Query Optimization: Can Mobile Agents Help?", VLDB conference, 2000.

[13] L. M. Stephens and M. N. Huhns, "Database Connectivity Using Agent-based mediator System", *Paper No.* A38, 1999.

[14] M Stonebraker., et al., "Mariposa: A Wide Area Distributed Database System", The VLDB journal, Vol. 5, 1996, pp. 48-63.

[15] R. Vlach and L. J. Navara, D. MDBAS_ "A Prototype of a Multidatabase Mnagement System based on Mobile Agents", In Proceedings of SOFEM'00, Springer-VErlag, 2000.