Query Optimization Issues for Data Retrieval in Cloud Computing

N.Samatha[1], K.Vijay Chandu [2], P.Raja Sekhar Reddy[3]
[1, 3] CSE, Anurag Group of Institutions, Hyderabad, A.P, India
[2] CSE, Viswa Bharathi Group of Institutions, Hyderabad, A.P, India

Abstract
Cloud data storage redefines the issues targeted on customer's out-sourced data (data that is not stored/retrieved from the customers own servers). In this work we observed that, from a customer's point of view, the data need to be accessed with in no time the user given the request ,Even though the data is stored on cloud severs the effective query optimizations may not be defined to access the data in an efficient way. This paper proposes the efficient available Query optimization techniques for efficient retrieval of data to satisfy the customer needs.

Keywords: Cloud computing, storage, Cloud *service* provider, Query Optimization.

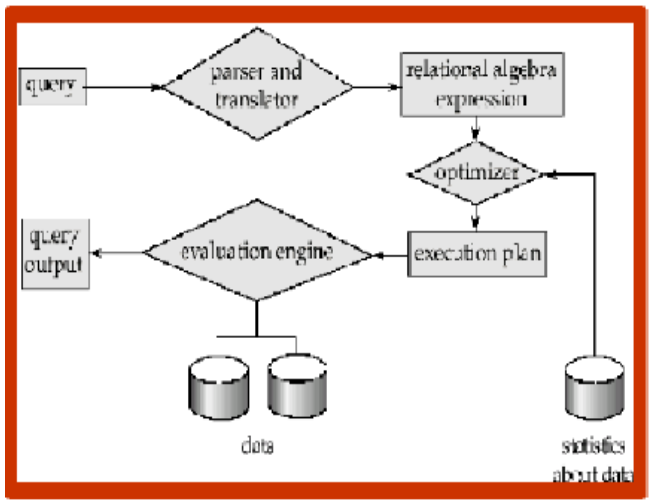
Introduction
The industrial information technology towards a subscription based or pay-per-use service business model known as *cloud computing*. This paradigm provides users with a long list of advantages, such as provision computing capabilities; broad, heterogeneous network access; resource pooling and rapid elas-ticity with measured services .Huge amounts of data being retrieved from geographically distributed data sources, and non-localized data-handling requirements, creates such a change in technological as well as business model. One of the prominent services offered in *cloud computing* is the *cloud data storage*, in which, subscribers do not have to store their own data on their servers, where instead their data will be stored on the cloud service provider's servers. In cloud computing, subscribers have to pay the providers for this storage service. This service does not only provides flexibility and scalability data storage, it also provides customers with the benefit of paying only for the amount of data they needs to store for a particular period of time, without any concerns of efficient storage mechanisms and maintainability issues with large amounts of data storage[3]. In addition to these benefits, customers can easily access their data from any geographical region where the Cloud Service Provider's network or Internet can be accessed [1]. An example of the cloud computing is shown in Fig. 1. Since *cloud service providers* (*SP*) are separate market entities, data integrity and privacy and retrieval are the most critical issues that need to be addressed in *cloud computing*. Even though the cloud service providers have standard regulations and powerful infrastructure to ensure

Customer's data privacy, data retrieval and provide a better availability [5], the reports of privacy breach and service outage have been apparent in last few years

Fig. 1. Cloud computing architecture example

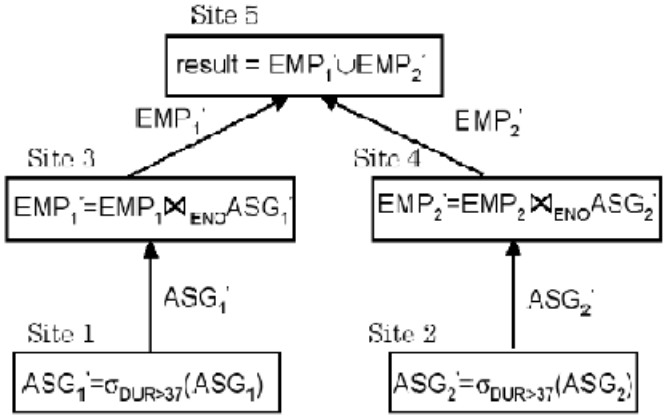
In this work we observed that, from a customer's point of view, relying upon data retrieving which he needs by performing an effective query optimization In addition, providing reliability ,availability are crucial and equally important to query optimization. Query Optimization can be achieved by implementing the optimization techniques for effective retrieval of data.

To address optimization issues in this paper, we proposed the techniques for optimizing the queries to provide customers with fast data retreival [2]. In our model,
Query processing: A 3-step process that transforms a high-level query (of relationa lcalculus/SQL) into an **equivalent** and **more efficient** lower-level query (of relational algebra).
1. **Parsing and translation**
– Check syntax and verify relations.
– Translate the query into an equivalent relational algebra expression.
2. **Optimization**
– Generate an optimal evaluation plan (with lowest cost) for the query plan.
3. **Evaluation**
– The query-execution engine takes an (optimal) evaluation plan, executes that plan and returns the answers to the query.



The success of storage is due, in part to the availability
– of declarative query languages that allow to easily express complex queries without knowing about the details of the physical data organization and of advanced query processing technology that transforms the high-level user/application queries into efficient lower-level query execution strategies.
The query transformation should achieve both **correctness** and **efficiency**
– The main difficulty is to achieve the efficiency
– This is also one of the most important tasks of any distributed system
• **Cloud storage system query processing**: Transform a high-level query (of relational calculus/SQL) on a stored database (i.e., a set of global relations) into an **equivalent** and **efficient** lower-level query (of relational algebra) on relation fragments.
• Cloud storage system query processing is more complex
– Fragmentation/replication of relations
– Additional communication costs
– Parallel execution
Example: Transformation of an SQL-query into an RA-query.
Relations: EMP(ENO, ENAME, TITLE), ASG(ENO,PNO,RESP,DUR)
Query: *Find the names of employees who are managing a project*?
– High level query
SELECT ENAME **FROM** EMP,ASG
WHERE EMP.ENO = ASG.ENO **AND** DUR > 37
– Two possible transformations of the query are:
_ Expression 1: _ENAME(_DUR>37∧ EMP.ENO=ASG.ENO(EMP × ASG))
_ Expression 2: _ENAME(EMP ⋊⋉ENO (_DUR>37(ASG)))
– Expression 2 avoids the expensive and large intermediate Cartesian product, and therefore typically is better.

We make the following assumptions about the data fragmentation[8]
– Data is (horizontally) fragmented and distributed
_ Site1: ASG1 = _ENO≤"E3"(ASG)
_ Site2: ASG2 = _ENO>"E3"(ASG)
_ Site3: EMP1 = _ENO≤"E3"(EMP)
_ Site4: EMP2 = _ENO>"E3"(EMP)
_ Site5: Result
– Relations ASG and EMP are fragmented and distributed in the same way
– Relations ASG and EMP are locally clustered on attributes RESP and ENO respectively
Now consider the expression _ENAME(EMP ⋊⋉ENO (_DUR>37(ASG)))
• Strategy 1 (partially parallel execution)[2]:
– Produce ASG′1 and move to Site 3
– Produce ASG′ 2 and move to Site 4
– Join ASG′ 1 with EMP1 at Site 3 and move the result to Site 5
– Join ASG′ 2 with EMP2 at Site 4 and move the result to Site 5
– Union the result in Site 5
• Strategy 2:
– Move ASG1 and ASG2 to Site 5
– Move EMP1 and EMP2 to Site 5
– Select and join at Site 5
• For simplicity, the final projection is omitted.



Calculating the cost of the two strategies under the following assumptions:
– Tuples are uniformly distributed to the fragments; 20 tuples satisfy DUR>37
– size(EMP) = 400, size(ASG) = 1000
– tuple access cost = 1 unit; tuple transfer cost = 10 units
– ASG and EMP have a local index on DUR and ENO

• Strategy 1:

– Produce ASG's: (10+10) * tuple access cost 20
– Transfer ASG's to the sites of EMPs: (10+10) * tuple transfer cost 200
– Produce EMP's: (10+10) * tuple access cost * 2 40
– Transfer EMP's to result site: (10+10) * tuple transfer cost 200

– Total cost 460

• Strategy 2:

– Transfer EMP1, EMP2 to site 5: 400 * tuple transfer cost 4,000
– Transfer ASG1, ASG2 to site 5: 1000 * tuple transfer cost 10,000
– Select tuples from ASG1 [ASG2: 1000 * tuple access cost 1,000
– Join EMP and ASG': 400 * 20 * tuple access cost 8,000
– Total cost 23,000

Query optimization is a crucial and difficult part of the overall query processing[2]
• Objective of query optimization is to **minimize** the following cost function: I/O cost + CPU cost + communication cost
• Two different scenarios are considered:
– Wide area networks
Communication cost dominates
low bandwidth
low speed
high protocol overhead
Most algorithms ignore all other cost components
– Local area networks
Communication cost not that dominant
Total cost function should be considered
Ordering of the operators of relational algebra is crucial for efficient query processing
• Rule of thumb: move expensive operators at the end of query processing
• Cost of RA operations:

Operation	Complexity
Select, Project (withoutduplicate elimination)	O(n)
Project (with duplicate elimination)	O(n log n)
Group ,Join,Semi-join Division,Set Operators	O(n log n)
Cartesian Product	O(n2)

Query Optimization Issues[2]
Several issues have to be considered in query optimization
• Types of query optimizers
– Wrt the search techniques (exhaustive search, heuristics)
– Wrt the time when the query is optimized (static, dynamic)
• Statistics
• Decision sites
• Network topology
• Use of semi joins

Types of Query Optimizers wrt Search Techniques[2]
– Exhaustive search
Cost-based
Optimal
Combinatorial complexity in the number of relations
– Heuristics
Not optimal
Regroups common sub-expressions
Performs selection, projection first
Replaces a join by a series of semijoins
Reorders operations to reduce intermediate relation size
Optimizes individual operations

Types of Query Optimizers wrt Optimization Timing
– Static
Query is optimized prior to the execution
As a consequence it is difficult to estimate the size of the intermediate results
Typically amortizes over many executions
– Dynamic
Optimization is done at run time
Provides exact information on the intermediate relation sizes
Have to re-optimize for multiple executions
– Hybrid
First, the query is compiled using a static algorithm
Then, if the error in estimate sizes greater than threshold, the query is re-optimized at run time.

Statistics
– Relation/fragments
Cardinality
Size of a tuple
Fraction of tuples participating in a join with another relation/fragment
– Attribute
Cardinality of domain
Actual number of distinct values
Distribution of attribute values (e.g., histograms)
– Common assumptions
Independence between different attribute values
Uniform distribution of attribute values within their domain

Decision sites
– Centralized
Single site determines the "best" schedule
Simple
Knowledge about the entire distributed database is needed
– Distributed
Cooperation among sites to determine the schedule
Only local information is needed
Cooperation comes with an overhead cost
– Hybrid
One site determines the global schedule
Each site optimizes the local sub-queries

Network topology

– Wide area networks (WAN) point-to-point

Characteristics

· Low bandwidth

· Low speed

· High protocol overhead

Communication cost dominate; all other cost factors are ignored

Global schedule to minimize communication cost

Local schedules according to centralized query optimization

– Local area networks (LAN)

Communication cost not that dominant

Total cost function should be considered

Broadcasting can be exploited (joins)

Special algorithms exist for star networks

Use of Semi joins[2]

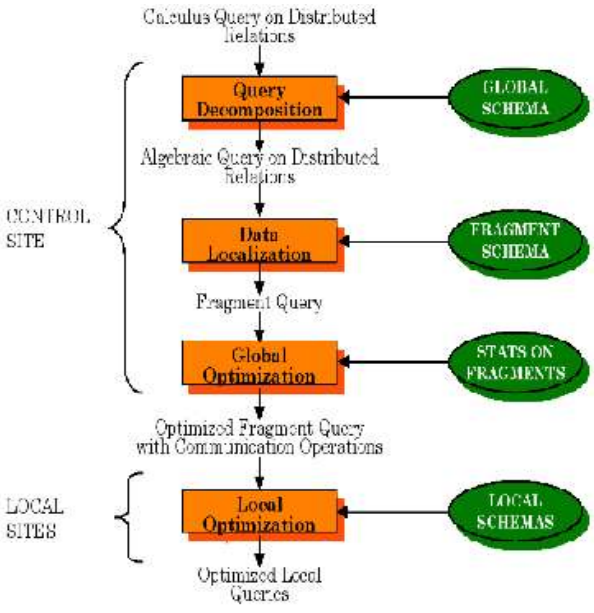
Reduce the size of the join operands by first computing semijoins

Particularly relevant when the main cost is the communication cost

Improves the processing of distributed join operations by reducing the size of data

exchange between sites

However, the number of messages as well as local processing time is increased



Query processing transforms a high level query (relational calculus) into an equivalent lower level query (relational algebra). The main difficulty is to achieve the efficiency in the transformation

• Query optimization aims to minimize the cost function:

I/O cost + CPU cost + communication cost

• Query optimizers vary by search type (exhaustive search, heuristics) and by type of the

algorithm (dynamic, static, hybrid). Different statistics are collected to support the query

optimization process

• Query optimizers vary by decision sites (centralized, distributed, hybrid)

• Query processing is done in the following sequence: query decomposition data localization global optimization local optimization.

Conclusion

In this paper, we proposed a various issues related to the Query Optimization in cloud computing, which seeks to provide each customer with better data retrieval from cloud data storage.

Acknowledgement

We are very much thankful to Prof. G. Vishnu Murthy, Head of the CSE Dept who had given valuable suggestions in carrying out this paper.

Reference

[1] Amazon.com, “Amazon s3 availablity event: July 20, 2008”, Online at http://status.aws.amazon.com/s3-20080720.html, 2008.

[2] M. Tamer Oezsu, Patrick Valduriez ``Principles of Distributed Database Systems, Second Edition" Prentice Hall, ISBN 0-13-659707-6, 1999

[3] R. Gellman, “Privacy in the clouds: Risks to privacy and confidentiality from cloud computing”, Prepared for *the World Privacy Forum*, online at http://www.worldprivacyforum.org/pdf/WPF Cloud Privacy Report.pdf,Feb 2009.

[4] W. Itani, A. Kayssi, A. Chehab, “Privacy as a Service: Privacy-Aware Data Storage and Processing in Cloud Computing Architectures,” *Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing*, Dec 2009

[5] M. Jensen, J. Schwenk, N. Gruschka, L.L. Iacono, “On Technical Security Issues in Cloud Computing”, *IEEE International Conference on Cloud Computing, (CLOUD II 2009)*, Banglore, India, September 2009, 109-116

[6] P. F. Oliveira, L. Lima, T. T. V. Vinhoza, J. Barros, M. M´edard,“Trusted storage over untrusted networks”, *IEEE GLOBECOM 2010*, Miami, FL. USA.

[7] S. H. Shin, K. Kobara, “Towards secure cloud storage”, *Demo forCloudCom2010*, Dec 2010.

[8] Stefano ceri Giuseppe pelagati “Distributed Databases-Principles and systems”Tata MC Graw Hill 2008

[9] J. Du, W. Wei, X. Gu, T. Yu, “RunTest: assuring integrity of dataflow processing in cloud computing infrastructures”, In Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security (ASIACCS ’10), ACM, New York, NY, USA, 293-304