ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

Single Query Optimization of SPARQL Using Ant Colony Optimization

Ms.P R Rupashini[1], Mrs. R Gomathi[2]

ME Department of CSE,Bannari Amman Institute of Technology Sathyamangalam,India[1]

Associate Professor/Sr. grade, Department of CSE, Bannari Amman Institute of Technology, Sathyamangalam, India[2]

ABSTRACT— As increasingly large RDF datasets are being published on the web, efficient RDF querying has become an essential factor in realizing the semantic web. One of the reasons of slow speeds of executions could be excess normalization leading to single tables. More the number of tables more are the complex nature of joins, and thus leading to more execution times. So to retrieve data quickly, there is a need to optimize the query. Query optimization is the refining process in database administration and it helps to bring down speed of execution Query optimization is performed translating the SQL queries in to query tree. Then pick the best algorithm for optimization of queries. In this paper we are used a meta heuristic technique Ant Colony Optimization (ACO) .The proposed technique is efficient and scalable for SPARQL query by using with ACO.

KEYWORDS— RDF, SPARQL, Query optimization, ACO

I. INTRODUCTION

RDF is the data format of interlinked data. RDF is a directed, labeled graph data format for representing information in the Web. RDF is an essence of triple format namely subject, predicate and object.

SPARQL is a query language and a protocol for retrieving RDF data which has been formulated and designed by the W3C RDF Data Access Working Group. SPARQL is a query language for pattern matching for RDF graphs. SPARQL syntax is similar to SQL, but SPARQL is more powerful, which enables queries spanning single disparate (local or remote) data sources containing heterogeneous semistructured data.

The SPARQL query language is related to the following specifications:

- The SPARQL Protocol for RDF [SPROT] specification defines the remote protocol for issuing SPARQL queries and receiving the results.
- The SPARQL Query Results XML Format [RESULTS] specification defines an XMLdocument format for representing the results of SPARQL SELECT and ASK queries.

SPARQL takes the description of what the application wants, in the form of a query, and returns that information, in the form of a set of bindings or an RDF graph. Query optimization is the most critical phase in query processing. Multi Query optimization is a technique in which single query plans for satisfying a query are examined and a good query plan is identified. Complexity arises in MQO which leads to NP-Hard. There may be many plans to find the best strategy. Cost based query optimizers evaluate the resource of various query plans and use the basis for plan selection using algorithms. The search space can become quite large depending on the complexity of the SPARQL query.

Complex queries are becoming common, due to the advent of technological tools that help examine information from large data stores. These complex queries share a lot of common sub-expressions since i) extensive views for different query that share a common value ii) There are nested queries that are correlated where outer query and inner query variables are not common but form a common sub-expression. Keeping the above challenges we design a framework for MQO with the following contributions:

1. Summarize the similar pattern in the SPARQL query.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

2. Summarized patterns will be clustered based on the common substructures.
3. Clustered queries will be rewritten and finally query execution is performed.
4. Experiments prove that the model is very efficient and scalable.

II. RELATED WORK

Diwan et al., [5] Database systems frequently have to execute a batch of related queries. Multi-query optimization exploits evaluation plans that share common results. Current approaches to multi-query optimization assume there is infinite disk space, and very limited memory space. Pipelining was the only option considered for avoiding expensive disk writes. The availability of fairly large and inexpensive main memory motivates the need to make best use of available main memory for caching shared results, and scheduling queries in a manner that facilitates caching. Pipelining needs to be exploited at the same time.

The problem of multi-query optimization taking into account query scheduling, caching and pipelining is discussed. MQO with either just query scheduling or just caching is NP-complete is proved. The first known algorithm for the most general MQO problem with scheduling, caching and pipelining is provided. After showing the connections of this problem with other traditional scheduling problems and graph theoretic problems we outline heuristics for MQO with scheduling, caching and pipelining.

Markus et al. [1] presented the problem of Basic Graph Pattern (BGP) optimization for SPARQL queries and main memory graph implementations of RDF data is formalized. The characteristics of heuristics for selectivity based static BGP optimization is defined and analyzed.

Hawash et. al. [3] proposed two disk based versions of trace equivalence and bi-similarity algorithms for summarizing data graphs. Authors adapted the algorithms to the RDF data model.

Michael Schmidt et al. [9] presented Fundamental aspects related to the efficient processing of the SPARQL query language for RDF, proposed by theW3C to encode machine-readable information in the Semantic Web is proposed.

Mohammad et al. [8] proposed Semantic web is an emerging area to augment human reasoning. Various technologies are being developed in the arena which have been standardized by the World Wide Web Consortium (W3C). One such standard is the Resource Description Framework (RDF). Semantic web technologies can be utilized to build efficient and scalable systems for Cloud Computing. With the explosion of semantic web technologies, large RDF graphs are common place.

Parthasarathy et al. [11] presented RDF and RDFS have recently become very popular as frameworks for representing data and meta-data in form of a domain description, respectively. RDF data can also be thought of as graph data. Keyword based querying of RDF data was focused.

Jiewen et al. [8] proposed , A scalable RDF data management system that is up to three orders of magnitude more efficient than popular multi-node RDF data management systems was introduced. In so doing, techniques for leveraging state-of-the-art single node RDF-store technology partitioning the data across nodes in a manner that helps accelerate query processing through locality optimizations and decomposing SPARQL queries into high performance fragments that take advantage of how data is partitioned in a cluster was also discussed.

Wangchao et al [12] developed The classical problem of query optimization in the context of RDF/SPARQL is revisited. The techniques developed for relational and semi-structured data/query languages are hard, if not impossible, to be extended to account for RDF data model and graph query patterns expressed in SPARQL is shown. In light of the NP-hardness of the multi-query optimization for SPARQL, heuristic algorithms that partition the input batch of queries into groups such that each group of queries can be optimized together is been proposed.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

Tang et al. [13] proposed a constraint-based optimization framework. Specifically, the expertise matching problem was transformed to a convex cost flow problem and the objective was then to find a feasible flow with minimum cost under certain constraints. According to the authors, the framework could achieve an optimal solution.

III. PROPOSED SYSTEM

Single Query with Ant Colony Optimization

Ant Colony System (ACS) is an agent-based system, which is based on the biological ants and their social behaviour. The Ants can choose any path to reach its destination. Initially it could be nearest or it could be longest. This new approach is called Ant Colony Optimization (ACO). ACO is a soft computing technique inspired by the foraging behavior of ant colonies. In such colonies, ants walking between their nest and a food source mark their paths with pheromone traces. Foraging ants make use of these traces, as they tend to follow paths where the pheromone concentration is highest. Over time, shorter paths attract more and more pheromones as they are traversed with increasing frequency because of their length as well as the pheromone traces on these paths. The colony thus converges to using a short path.

The Ants are best for travel sales man problem. And these inherit parallelism and works in a team. The ACO is best for dynamic application. The idea behind ACO is to produce a model to search for a minimum cost path. The behavior of artificial ants is inspired from real ants. Each ant builds a path from its source node to its destination. Ants are able to find the shortest path for its source of food. These deposit pheromone trails in their travelling, as a communication with others. After carrying the food and start returning back, following their pheromone trails, and still depositing more pheromone. While an ant builds a path, it gets quantitative information about the path cost and qualitative information about the amount of traffic in the network. Then, another ant travelling through the same path will carry this information.

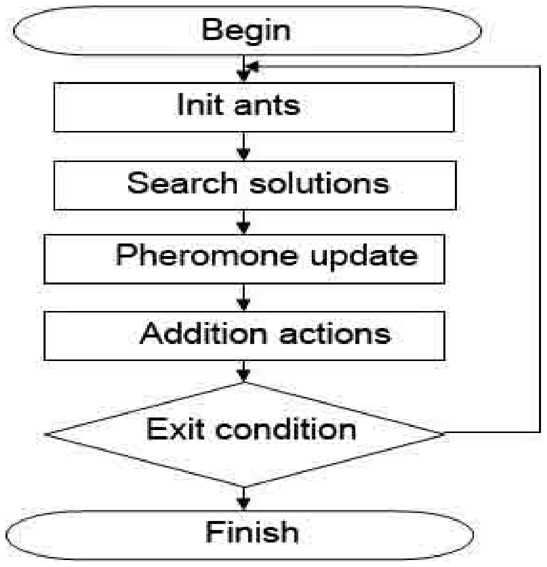

Fig.1 Flow chat of Ant Colony Algorithm

Let us consider an RDF model of the CIAWorld Factbook generated by using QMap. This model contains data about 250 countries, defined in over 100,000 triples.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

The query can be expressed in SPARQL as

```
1.  PREFIXc: <http://www.daml.org/2001/09/countries/fips#>
2.  PREFIXo: <http://www.daml.org/2003/09/factbook/factbook-ont#>
3.  SELECT ?partner
4.  WHERE { c:NL o:exportPartner ?expPartner
5.  ?expPartner o:country ?partner
6.  ?partner o:dependentArea ?area
7.  ?area o:internationalDispute ?conflict }
```

This query can be subdivided into four subqueries:

- ✓ a query for information on the export partners of The Netherlands (line 4),
- ✓ a query for countries associated with other countries export partners (line 5),
- ✓ a query for dependent areas (line 6),
- ✓ a query for international disputes (line 7).

In order to resolve the complete query, the results of the individual subqueries can be joined in any order. Here, the number of statements resulting from a join is equal to the number of statements compliant with both operands' constraints. A sequence of joins in such a query can be visualized as a tree. The leaf nodes of an RDF query tree typically represent inputs, whereas the internal nodes represent algebra operations, enabling one to specify basic retrieval requests on the inputs. The tree can be given as

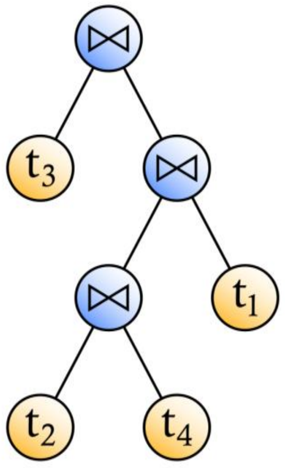

Fig. 2 Left-deep query tree

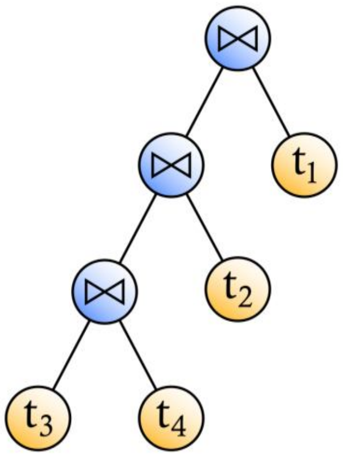

Fig. 3 Bushy query tree

IV. RESULT AND DISCUSSION

The Lehigh University Benchmark (LUBM) is developed to facilitate the evaluation of Semantic Web repositories in a standard and systematic way. The LUBM data set is a benchmark data set designed to enable researchers to evaluate a semantic web repository's performance . The LUBM data generator generates data in RDF/XML serialization format. Therefore, we convert the data to N-Triples to store the data, because with that format, we have a complete RDF triple (Subject, Predicate, and Object) in one line of a file .The benchmark is to evaluate the performance of those repositories with respect to extensional queries over a large data set that commits to a single realistic ontology. It consists of domain ontology of university, repeatable and customizable synthetic data, a set of test queries, and large performance metrics. The table 1 shows execution time of each query is measured in milliseconds by using Single query with ACO.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

Table 1 Execution Time for Single SPARQL Query with ACO

Query	Execution time (ms)
1	110
2	128
3	97
4	102
5	84

LUBM data set is used as input which has triple patterns (subject, predicate and object). SPARQL query from the user is passed to the query processing engine. Query optimization is implemented where best plan algorithm is generated to select the best execution plan and the final output to the SPARQL query is retrieved from RDF store and returned to the user. The execution time of each query is measured in milliseconds and is shown in the figure 4. It retrieves the queries given by the user effectively and efficiently.

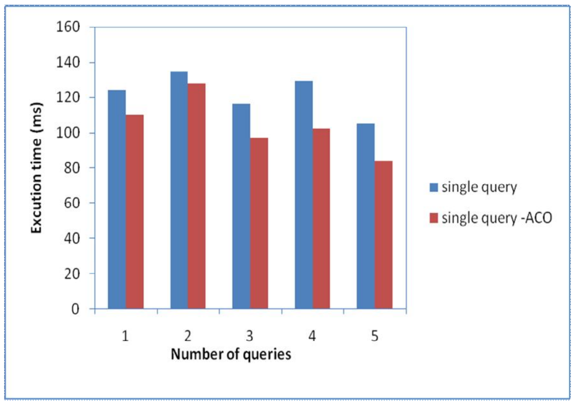

Fig. 4 Execution Time for Single SPARQL Query and Single SPARQL Query with ACO

VI. CONCLUSION

With the drawbacks in the single query system, the study of storing and retrieval of SPARQL queries over RDF data, with the objective to minimize total query evaluation time is implemented. In order to decrease the delay time to answer the query we go for optimization algorithm. Single queries outperforms for large amount of data can be stored and retrieved by using ACO optimizing algorithm. In future by using the optimization algorithm multiple queries and complex queries can also be implemented.

REFERENCES

1 Abadi, D.J. Marcus, A. Madden, S.R. and Hollenbach, K. (2007) 'Scalable semantic web data management using vertical partitioning', *VLDB*, pp. 411-422.
2 Angles, R. and Gutierrez, C. (2008) 'The expressive power of SPARQL', *ISWC*, pp 114-129.
3 Anton Deik, Bilal Faraj, Ala Hawash, Mustafa Jarrar 2010. 'Towards Query Optimization for the Data Web - Two Disk-Based algorithms: Trace Equivalence and Bisimilarity', *In proceedings of the International Conference on Intelligent Semantic Web Applications and Services*. pp 131-137.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

4 Dalvi, NN, Sanghai, SK, Roy, P and Sudarshan, S 2001,'Pipelining in multi-query optimization', PODS, pp. 59-70.
5 Diwan, S, Sudarshan, and Thomas, D 2006, 'Scheduling and caching in multi-query optimization', COMAD, pp 150-153.
6 M. Dorigo K. Socha 2007 ,'An Introduction to Ant Colony Optimization', T. F. Gonzalez, Approximation Algorithms and Metaheuristics, CRC Press, pp 6-22.
7 Jain, K, Murty, MN and Flynn, PJ 1999, 'Data clustering: a review. ACM Comput. Surv', vol. 31, no. 3, pp. 264-323, 1999.
8 Jiewen Huang ,Daniel,J,Abadi ,Kun Ren,2011, 'Scalable SPARQL Querying of Large RDF Graphs' ,Proceedings of the VLDB Endowment, vol. 4, no. 11, pp 1123-1134.
9 M. Schmidt, M. Meier, and G. Lausen. (2010) Foundations of SPARQL query optimization, *ICDT*, pp 4-33.
10 Mohammad Farhan Husain, James McGlothlin, Mohammad Mehedy Masud, Latifur R. Khan, and Bhavani Thuraisingham, 2011,'Heuristics-Based Query Processing for Large RDF Graphs Using Cloud Computing', IEEE transactions on knowledge and data engineering, vol. 23, no. 9, pp 1312-132.
11 Parthasarathy K ,Sreenivasa Kumar P, Dominic Damien 2011, 'Ranked Answer Graph Construction for Keyword Queries on RDF Graphs without Distance Neighbourhood Restriction' , WWW, ACM 978-1-4503-0637-9/11/03.
12 Wangchao Le,Anastasios Kementsietsidis, Songyun Duan, Feifei Li, 'Scalable Multi-Query Optimization for SPARQL' 2012, IEEE 28th International Conference on Data Engineering, pp. 666-677.
13 Wenbin Tang, Jie Tang, and Chenhao Tan 2010, 'Expertise Matching via Constraint-based Optimization', IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, 978-0-7695-4191-4/10.
14 http://webcache.googleusercontent.com/search?q=cache:0zSkTIfXRAUJ:swat.cse.lehigh.edu/projects/lubm/query.htm+http://swat.cse.lehigh.edu/projects/lubm/query.html