Evaluating RDF Queries Over DHTs

Source Paper
Evaluating Conjunctive Triple Patterns
over Large Structured Overlay Network

Seminar report
Done By

Waled Almukawi
Computer Networks and Telematics
Albert-Ludwigs Uneversity Freiburg

July 29, 2009

Abstract

We describe the problem of evaluating conjunctive queries consist of RDF data triple patterns on top of Large structured peer to peer network, We try to construct evaluate algorithms that are scalable and can deal with big amounts of RDF data, distribute the query processing load evenly and optimize network traffic. So we will introduce two novel query processing algorithms then we will compare between these two algorithm through a detailed experimental evaluation.

1 Introduction

In this paper the we focus on the problem of evaluating conjunctive queries consist of multi predicates of RDF triple patterns stored in distributed hash tables, Distributed hash tables (DHTs) are a type of P2P networks that use hash function to store data in distributed hash tables, and enable us to build scalable, robust and fault-tolerant distributed applications[1]. The previous work on RDFPeers which use queries with atomic formula or conjuncted atomic formula with the same subject or the same object and might have deferent constant values in the queries[6], therefore it was necessary to use another type of query called Conjunctive query that allow us to pose flexible queries where the triples pattern can come in different subjects, predicates and objects in the triple patterns such that the subject variable in a triple pattern could be used as a predicate or an object in another triple pattern, it exploit the power of Select-Project-Join queries that allow us to pose more complicated queries and process a huge amount of RDF triple stored in p2p network.We present two novel algorithms for the evaluation of conjunctive queries over RDf triple patterns stored in Distributed hash tables Chord .

Outline The remainder of this article is organized as follows. part2 we will have a look on conjunctive queries what are the conjunctive queries? and how do they look like ? And what is the deferent between the conjunctive queries and queries used over RDF data in the previous work?, part3 we will explain the System model and Data Model, in part4 we will present two novel algorithms Query Chain(QC) and Spread By v value(SBV)algorithm then we will compare them and discuss several points that come through the experimental evaluation of these two algorithms, Part5 will talk about the Network traffic optimization such that what can we do in order to reduce the network traffics, part6 will talk about the experimental evaluation for the two algorithms used and show the main deferent between them using a statistical graph, In part6 we make a conclusion on our paper and what is the possible future work then part7 will list the references used.

2 Conjunctive Queires

Conjunctive queries are simply the fragments of first order logic given by a set of atomic formula using Conjunction, The general form of conjunctive query is
?x1 , ... , ?xn : (s1 , p1 , o1) , (s2 , p2 , o2) , ..., (sn , pn , on)
where the right side is called Body of the conjunctive query and the left side called the Head of the conjunctive query and ?x1 , . . . , ?xn are variables,, each (si , pi , oi) is a triple pattern, and each variable ?xi appears in at least one triple pattern (si , pi , oi). Variables has the sign ?, ?x1 ,..., ?xn are the set of answer variables (Free Variables) and the variable appear just in the body of the conjunctive query called bound variables, to evaluate

the conjunctive query we make mapping for the variables in the Conjunctive queries to the values in the corresponding relation in the database schema[3], we focus in this paper on three critical points in distributed setting such as the size of data stored in the network , load distribution and generated network traffic, the algorithms force as many networks nodes as possible to evaluate the query, store just a few number of items , optimize network traffic[1].

3 System Model and Data Model

system model is runing Chord protocol where all node in the network equal they have the same right and run the same software.every node has akey denoted as key(n),we apply hash function on the key to generate the m-bit identifier.the item also has unique key and by hash function we can generate the item ID .the look up operation take O(logN)nodes .becuase every node has a finger table use it to find the closer node to the item ,and dont have to go through all the nodes.
Data Model
every node can publish its data by creating the data into rdf triples and make it available to the rest nodes of the networks.Rdf triple is a way to model data as (Subject, Predicate, Object) such a kind of data can be understandable for both human being and computer, and provide data entigrity. The subject of a triple denotes the resource that the statement is about, the predicate denotes a property or a characteristic of the subject, and the object denote the value of the property

4 Query Chain Algorithm

The basic idea of QC is that the query is evaluated by a chain of nodes. The Intermediate results flow through the nodes of this chain and finally the last node in the chain send the result back to the node that has posed the query[2]. the query chain algorithm has three main phase
Indexing a new triple
Every triple pattern will be stored three times, the algorithm will hash the value of the subject and store the triple in the successor node of it
i.e I1=hash(key(subject)), and do the same with the predicate I2=hash(key(predicate)) and the object I3=hash(key(object)).

Evaluating a query

Suppose a node X submitted the conjunctive query

q:ans(x):- q1 ,q2, . . . , qc .

Each triple pattern of q might be evaluated by different node sequentially, although the order of choosing a pattern is crucial, these nodes form the query chain for q, Lets have this example to show how does QC-algorithm work.

Example1

let Tt1,t2,t3 where t1=(s1,p1,s2), t2=(s2,p2,s3), t3(s3,p3,o3). We can store it as follow: r1=Successor(hash(p1)), r2=Successor(hash(p2)), r3=Successor(hash(o3)). Actually every triple stored three times but here we show just the case we need in this example. Suppose node x submits the conjunctive query q: ans(x):- (?x,p1,?y),(?y,p2,?z),(?z,p3,o3) So x will choose the first triple pattern and compute Succ(h(p1)) to find the responsible node and send the message QEval (q,i,R',ip(x)) to this node, Where q is the Conjunctive Query, i the ith-triple pattern, R' is temporal set that will accumulate the intermediate result, ip(x) is the IP address for the node that submitted q. Suppose the responsible node for the first triple pattern is r1, then r1 will search in its local triple table end evaluate the first triple pattern, because we have just on triple stored, the result will be x=s1,y=s2, then r will pick the second triple pattern computing Succ(h(p2)), and send the message Qeval (q, 2, R', ip(x)) to the responsible node. Suppose r2 is the responsible node, So r2 will do the same action find that y=s2, z=s3, then send the message Qeval (q,3,R',ip(x)) to the responsible node, Suppose r3 is the responsible node so r3 will look in its local triple table , then evaluate r3so the answer is x=S3, after that r3 will send the result back to x.

-local processing at each chain node: here we will show the action happened during the evaluation of q. If a node n receives a message QEval(q, i, R, IP (x)). then node n will: First, n evaluates the i-th triple pattern of q using its local triple table i.e.it calculates the relation L = X (F (T T)) where F is a selection condition and X is a (possibly empty) list of natural numbers between 1 and 3. F and X are formed in the natural way by taking into account the constants and variables of qi e.g., if qi is (si , pi , ?oi) then L = 1 (1=si 2=pi (T T)). Then, n computes a new relation with intermediate results R = Y (R L) where Y is the (possibly empty) list of positive integers, identifying columns of R and L that correspond to answer variables or variables, Note that the special case of i =1 (when R = Y (L)) is covered by the above formula for R , given the initial value () of R. If R is not the empty relation then n creates a message QEval(q, i + 1, R , IP (x)) and sends it to the node that will evaluate triple pat-tern qi+1, If R is the empty relation then the computation stops and an empty answer is returned to node x. In the last triple pattern where i=k of q, n simply returns relation R back to x using a message Answer(q, R). R here contains the final answer to query q over the database of triples in the network. In the current implementation, R = Y (R X (F (T T))).

5 Spread By Value Algorithm

This algorithm is improving the idea of QC algorithm it distribute the query processing load evenly, Such that it takes the values generated immediately and send it on fly to deferent responsible nodes that can evaluate the triple pattern, so every node will evaluate the instance received and send the result back to the submitted node, In this algorithm we store the triple more than times such tha it store the triple by hashing the value of the subject, predicate, object and the possible combination by concatenating the value of the subject with predicate or the value of subject with the object and the other possible combination. We will here make example show how does SBV algorithm work and what is the deferent between this algorithm and the QC algorithm.

Example2
If we have three triple patterns and we want to index and store them, We do the following
Put(s1,p1,s2), (s2,p2,s3), (s3,p3,o3), (s1,p1,o1). -Index new triples: These triple patterns indexed many times, but we will show just the indexing we need in our example : r1=Successor(hash(p1)), r1=Successor(hash(s2+p2)), r1=Successor(hash(s3+o3)). Evaluating a query If node x submits q: ans(x):-(?x,p1,?y),(?y,p2,?z),(?z,p3,o3). First step x will determine the responsible node for the first triple pattern it will compute(h(p1)) and send the message QEval(q,V,u,ip(x)), where u is the empty valuation. Then will r1 will evaluate the triple pattern and find out that x=s1,s2,s1,o1.now r1 will take the result and find the responsible nodes for these triple:as follow qa=(s2p2,?z)(?z,p3,o3) r1 computes h(s2+p2)and find that r2 is responsible

node for this then r1 send the message QEval(‘qa,V,‘va,ip(x)).and do the same ‘qb= (o1p2,?z)(?z,p3,o3) it compute h(o1+p2) and find that the successor node is r4 then r1 snd the message QEval(‘qb,V,‘vb,ip(x)) to r4, when r4 revieve the message r4 evaluate the triple pattern, since r4 has no stored triples it will get the empty result, at the same time r2 receive the message then r2 try to evaluate the triple pattern and find that z=s3.
After that r2 will determine the successor node for the 3rd triple pattern h(s3+p3+o3),and find that r3 is the successor node ,then r2 assign
“Va=(x=s1,y=s2,z=s3),“qa=(s3,p3,o3) then sends the message
QEval(“qa,V,“va,ip(x)) to r3, then r3 evaluate the triple and find that x=s1, then r1 deliver the answer s1 to node x back.

6 Optimizing network traffic

To optimize the network traffic we will use Ip cache that will store the ip adrees of the node that contain the desired triple in the query ,such that in the next look up operation we can go to this node directly without passing the node in-between so this will reduce the network traffic. Als by using IPC we can reduce the routing load incurred by the node in the network.

7 Experiments

We here implement and simulate the chord using Java code, and run the agorithms used in theis paper on this simulator, our metric is
-The amount of network traffic that is created. -the distribution of the query procesing load among the network nodes.We stored 10.000 rdf triples and we have 10.000 networks nodes.we poses many conjunctive query to the algorithms in the simulator from 5 and up to 640 queries then we compare the result of these algorithms using a statistical graph.

8 Conclusion And Future Work

In this paper we have used two novel algorithms to evaluate the queries over RDF triples stored in P2P network, these algorithms consider the storage load, query processing load and the network traffic, and the basic idea in both algorithm was by splitting the conjunctive query into the triples it consists of and evaluate every triple separately at deferent nodes then we find the final result by matching the separated result and send the result again to the node that submitted the query, then we see that QC algorithm works sequentially which is time consuming and the SBV algorithm works in parallel so it is faster but it yields more storage cost, as future work we can improve the algorithm such that at every evaluation step it choose the triple pattern with less selectivity in order to speed up the query, and also we can improve the algorithm to make reasoning in the rdfpeers sideand also we want to support the whole function of the SPARQL, RDQL, RQL such that the system that used this algorithms would be more powerfull.

9 References

[1] E. Liarou, S. Idreos, and M. Koubarakis. Evaluating Conjunctive Triple Pattern Queries over Large Structured Overlay Networks. In I. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, and L. Aroyo.

[2] E. Liarou, S. Idreos, and M. Koubarakis. Publish-Subscribe with RDF Data over Large Structured Overlay Networks. In DBISP2P 05.

[3] S. Abiteboul, R. Hull, V.Vianu: Foundations of Databases, Addison-Wesley, 1995. ISBN 0-201-53771-0.

[4] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A Scalable peer-to-peer lookup service for internet applications.

[5] E. Prudhommeaux and A. Seaborn. SPARQL Query Language for RDF. http://www.w3.org/TR/rdf-sparql-query/, 2005.

[6] M.Cai, M. R. Frank, B. Yan, and R. M. MacGregor. A Subscribable Peer-to-Peer RDF Repository for Distributed Metadata Management.