International Journal on Cybernetics & Informatics (IJCI) Vol. 5, No. 2, April 2016

ENHANCING THE PERFORMANCE OF E-COMMERCE SOLUTIONS BY FRIENDS RECOMMENDATION SYSTEM AND NEO4J DATABASE

Shahina C P, Bindu P S and Surekha Mariam Varghese

[1]Department of Computer Science and Engineering, M. A. College of Engineering, Kothamangalam, Kerala, India

ABSTRACT

In the past, selling needs brick and mortar store and it is limited to a local customer base. But today, ecommerce websites make it easy for the customers around the world to shop by virtual store. Performance of ecommerce websites depend not only the quality and price of the product but also the customer centric suggestions about the product and services and retrieval speed of websites. There is more possibility to buy a product brand suggested by friends or relatives than by viewing commends from unknown people. We can implement such a customer centric approach using Neo4j database, which provides easy and fast retrieval of information based on multiple customer attributes and relations than relational databases. Experiment shows the enhancement of performance in terms of time and complexity.

KEYWORDS

RDBMS, NOSQL, GRAPH DATABASE, NEO4J, E-COMMERCE, SOCIAL NETWORK, CYPHER QUERY LANGUAGE, MYSQL.

1. INTRODUCTION

Humans are social creature, so social recommendation within an ecommerce platform is more effective mainly because of the following reasons. Friends of friends recommendation helps customers to connect and build network faster and more organically. Product recommendations [5] help business to maximize their revenue. But most of the reviews are not reliable, the friends or related people's recommendations ensures reliability about a product.

This paper presents an ecommerce social network that contains a social community [2] of customers having relationships with other customers and thus with the products. Nowadays we have various social communities. A person may have different social community [2] membership like community of friends during school education, college education, professional community, other friend's community etc. E-commerce websites can make use of these social communities for implementing friends' recommendation system [3]. A customer who is aware of a particular product makes a review of that product telling that it is good or not. Based on this comment, a friend or a person related to him or a friend of a friend can take a decision to buy a product or not. Neo4j graph [3] database can be used to implement this system, since we can assure reliability, response time, efficiency and accuracy for large datasets and relations compared to relational database.

2. PROBLEM DEFINITION

Ecommerce is an application used for online shopping. It provides more efficient and cheaper distribution of products. It is very helpful for customers because of convenience, selection of product from more varieties than local shops. But sometimes, while accessing ecommerce platforms, we face issues such as

poor customer service after purchasing a product, poor product quality, delay in delivery of the product etc. So while purchasing a product via ecommerce website, we prefer to get feedback of other customers who had already bought that product earlier or who had already accessed that specific ecommerce platform. Some ecommerce websites provide recommendations, but there is no guarantee that the suggestions posted by ecommerce platform itself are reliable. So to provide reliable feedback / suggestions, we propose an ecommerce solution with an integrated friend's recommendation system. In this application a customer will get a review of a product which is suggested by a friend or a friend of friend via various social community [2] networks.

The first choice for building the ecommerce solution integrated with social networks is by using RDBMS. It is commonly used solutions to store the data in almost all applications. Sql is a language used for querying and maintaining the relational database. MySQL database is an open source relational database management system which stores data in the forms of tables. In our application different tables are required for storing personal attributes, products, relation etc. Each table has specified schema and relationship between different tables are represented using foreign keys. Many constraints are also required for this integrated ecommerce - social network solution. Considering all these points, we came to the conclusion that MySQL is not suitable to store large amount of semi structured data with dynamically growing behaviour.

The integrated solution is very powerful, but at the same time very challenging because the ecommerce websites and friend connections / relations are highly dynamic in nature. This solution will also have to handle the high density connections and may need complex queries to handle features such as customer centric personalized shopping. Traditional databases are very good in complex query processing but it cannot handle big data with dynamic schema changes [1]. So this application has to handle huge amount of data and relations with different varieties / categories without compromising the retrieval speed / response time because of the semantic checks [1].

Neo4j database are nosql, open source graph database which also supports ACID transaction [3]. The developers describe Neo4j as "embedded, disk-based, fully transactional that stores data in graph rather than in tables".Neo4j is highly advisable for managing complex relationship between heterogeneous data such as many views of customer problem is being addressed for various clients. In Neo4j, everything is stored in form of an edge, a node or an attribute. Data is represented as nodes; relationships between data can be represented as edges; both nodes and edges can have properties. Neo4j can scale up to billions of nodes and relationships.Neo4j graph databases supports complex querying because it stores data as relation which helps faster retrieval of information using graph traversals. Instead of using relational tables, neo4j uses graph nodes to store data. Similar to attributes in RDBMS, Neo4j uses properties for each node. Each node can have different number and types of properties. For example a node representing a person can have properties like name, DOB, designation etc. and node representing product can have properties like product code, product name etc. Similarly properties may vary person to person and product to product. In RDBMS relationship between different tables are represented as foreign key but here relationships between different nodes can be represented as edges. Similar to nodes, each edge can have properties. For example relationship between person to person may have friend, known since 10 years etc. and relationship between person and product may be buy, suggestions etc.

Another commonly used graph database is flock DB. Flock DB is simpler than neo4j but not capable of doing graph traversal in depth. Flock DB is intended to handle twitters single depth followers[9]. Ecommerce social network requires a complex graph consisting of many customers and item. This, in turn, requires in depth traversals while querying. Since flock DB is capable of only single depth traversal, neo4j is best suited for this application.

Cypher is a declarative graph query language for the Neo4j which allows expressive and efficient querying and updating the graph data. Cypher is a relatively simple but still very powerful language. Very complicated database queries can easily be represented using Cypher. Figure 1 shows ecommerce

application with friends' recommendation system including nodes with different person and products with different relationships between them. The relationships between different nodes are represented by edges.

Name	Neo4j	Mysql
Description	Open source RDBMS	Widely used Open source RDBMS
Database model	Graph DBMS	Relational DBMS
Implementation language	Java	C and C++
License	Open source	Open source
Developer	Neo Technology	Oracle
Transaction concepts	ACID	ACID
Server Operating System	Linux OS X Windows	FreeBSD Linux OS X Solaris Windows
Data scheme	schema-free	Yes
Typing	Yes	Yes
Secondary indexes	Yes	Yes
SQL	No	Yes

3. METHODOLOGY

By using cypher query our application can be implemented in following steps-

1. Creation of graph of social community networks of customers with relationships including friends, family colleagues, and other acquaintances along with their properties.
2. Creation of product nodes with their properties.

3. Associate customer nodes with product nodes with relationship like buy, suggest etc. Figure 1 show the graph generated.

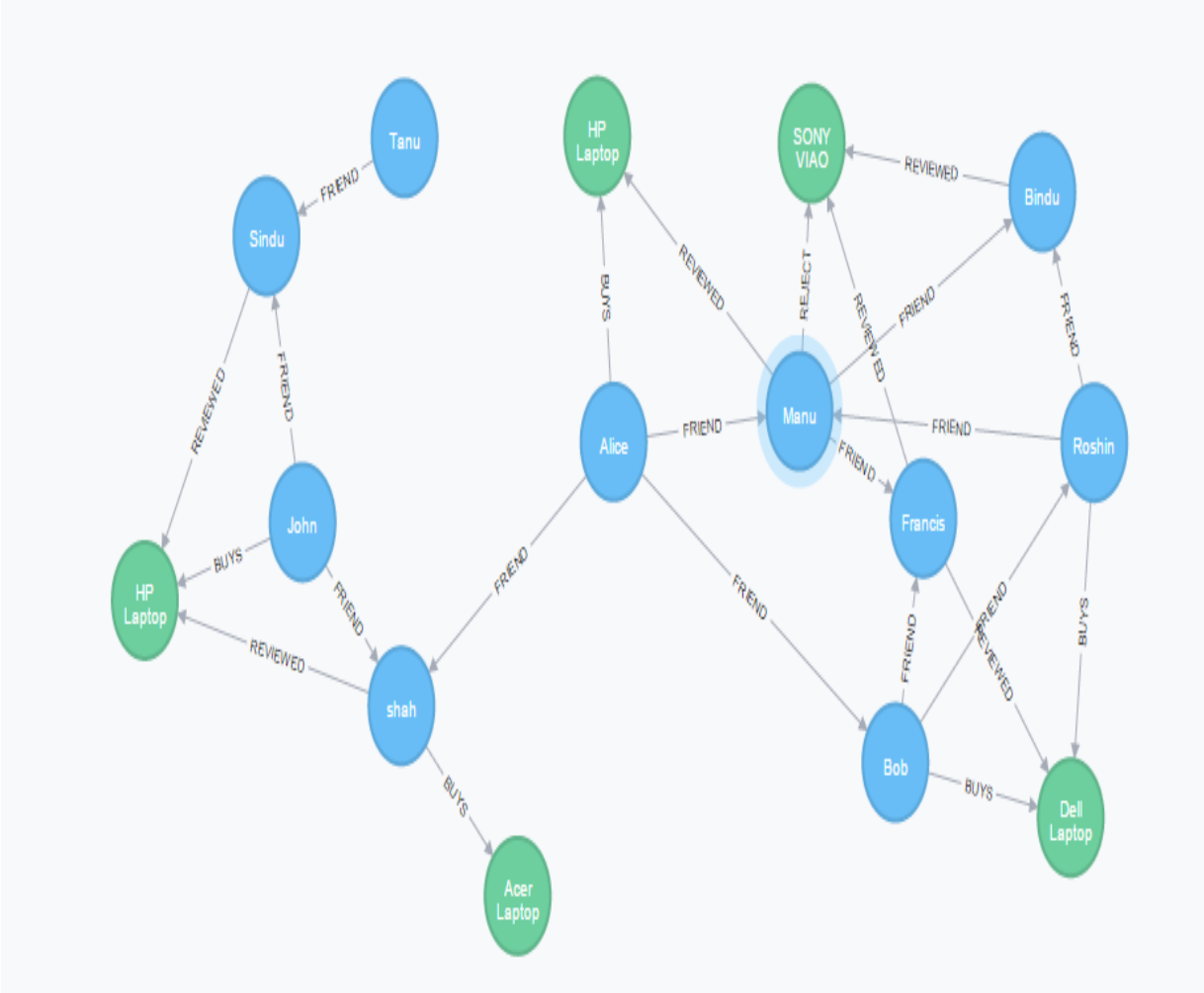

Figure 1. Example of ecommerce social network

Step 1 – Creation of Social Community

A set of customer nodes were created with properties like name, designation, age group, place etc. The edges representing relationships as friend were created. The properties for relationships were expressed as period of friendship and type of friendship such as close, very close, family friend.

Step 2 – Creation of Product Nodes

A set of product nodes with properties like name, price, model were created. The relationships between products and customers were represented with properties like Reviewed, Buys, and Reject. The properties

of Buys can be date of purchase, feedback, purpose etc. The property of Reviewed can be a comment such as Excellent, Bad service, Not good etc. The property of Reject can be Decision such as buy later.

For recommendation of a product, there are 3 factors [6].
1. Model data and data relationship to know how recommendation can be made
2. Make recommendation in real time by querying the relationship
3. Make it richer by adding data and relationship dynamically

4. PERFORMANCE ANALYSIS

4.1. WRITING QUERIES IN NEO4J

Graph data queries are straightforward to write and easy to understand. Graph databases have their own syntax for such queries. Cypher queries are much simpler than SQL queries; a long SQL statement can be divided to many queries using Cypher with much less number of lines.

4.1.1 How does Neo4j traverse a graph during a Query Execution?

Neo4j takes the cypher query as a metadata description of what we want and depending on the statistics, available indices it uses a combination of operations to perform the query. The main operations are look ups nodes, expand, expand into (between two nodes), hash-join, apply and semi apply [7].

4.2. QUERY PERFORMANCE

Query performance in a relational database is impacted by data growth and the number of JOINs [1]. As tables get bigger, so do indexes, which means that joining the same number of entities requires more and more time to execute. As questions / queries get more complex, the challenge increases as there is a huge number of entities to join. Even if the data volume stays constant, computational complexity explodes, which impacts query performance [1].

Neo4j is scalable and in this solution, it shows very small increase in the time to execute the query as data volume grows. This is because it doesn't compute relationships at query time but stores them at insertion time. In addition, graph queries look at the neighbourhood of starting points, so regardless of the total amount of data in the database, the amount of data that is examined remains roughly the same.

For example this project uses Neo4j to make real-time product recommendations by using information about what customer is looking for. Customers can view their friends recommendations and suggestions so can decide whether to purchase that product. In RDBMS, different tables are needed to store customer, product, order, friends etc. and different foreign keys are needed to connect these tables to provide the product recommendations.

Despite their name, relational databases do not store relationships between data elements in the same table i.e. relationship between two customers within a customer table is difficult in RDBMS. More over they are not well suited for ecommerce and social network systems which is having highly connected data with no specific schema. So as the volume, velocity and variety of data increases, the relationships grows even faster. So relational databases do not adapt well to changes.

Unlike a relational database, a graph database is structured entirely around data relationships. Graph databases treat relationships not as schema structure but as data. In Neo4j, the data is structured around data relationships so real-time query performance can be achieved no matter how large or connected the dataset.

For example we have three queries. First we evaluated these queries by 500 objects. The queries and results are shown in Figure 2. The three queries defined were:

q0: Find all friends of Bob.
q1: Find the product bought by Bob's friends.
q2: Find the suggestions of products that Bob's friend bought.

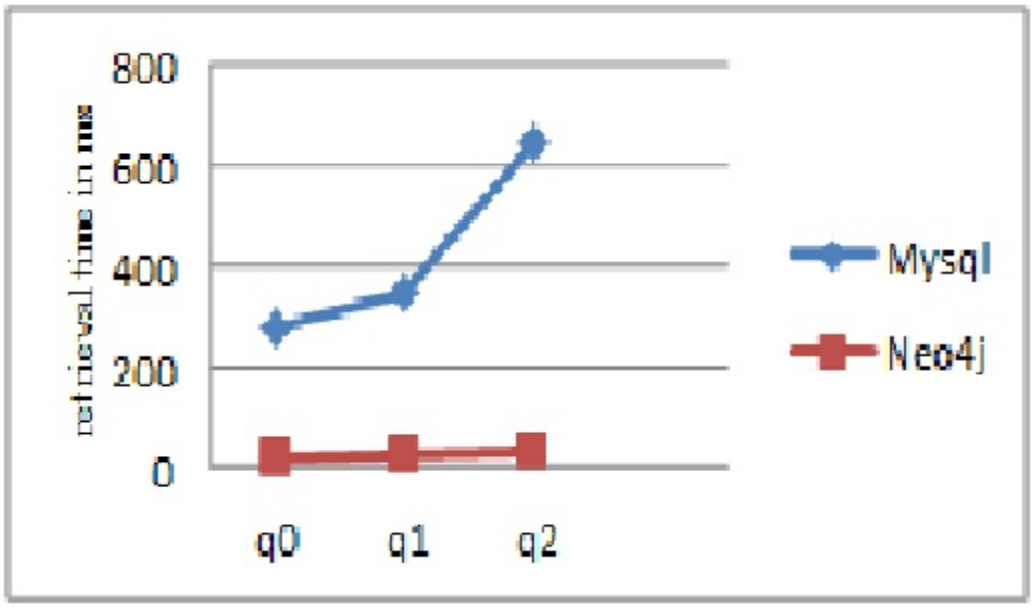

Figure 2. Retrieval time of queries having 500 objects by Neo4j and MySQL

5. CONCLUSIONS

In this paper, we implement an integrated solution to connect friends' recommendation system with ecommerce platforms using Neo4j graph database. We came to a conclusion that when compared with relational database, performance of Neo4j is better in terms of time and complexity. On comparison overall performance of graph databases exceeds the relational database

REFERENCES

[1] "Overcoming SQL Strain and SQL Pain" http://neo4j.com/resources/wp-overcoming-sql-strain

[2] "Social network-Ne4j graph database" http://neo4j.com/use-cases/social-network/

[3] "Nosql Databases", http://nosql-database.org/

[4] "Neo4j graph database" http://neo4j.com/developer/graph-database/

[5] "Real time recommendation with Neo4j" http://neo4j.com/use-cases/real-time-recommendation-engine/

[6] "Building recommendation engine with cypher in two minutes"- http://neo4j.com/resources/wp-recommendations-bu

[7] " Ne4j graph database" http://neo4j.com/blog/introducing-new-cypher-query-optimizer/

[8] "Neo4j" http://neo4j.com

[9] "Flockdb"https://en.wikipedia.org/wiki/FlockDB

Author

Shahina C P. is currently pursuing M.Tech in Computer Science and Engineering in Mar Athanasius College of Engineering. She completed her B.Tech from MES College of Engineering, Kuttippuram. Her areas of research are Modern Database System and Machine Learning .

Bindu.P.S. is currently pursuing M.Tech in Computer Science and Engineering in Mar Athanasius College of Engineering. She completed her B.Tech from SRM College of Engineering, Chennai. Her areas of research are Modern Database System and Data Mining .She is a principal of Govt.Polytechnic College, presently doing M Tech , on deputation. She has an experience of 21 years in various capacities as Lecturer, Head of Department and Principal in the department of Technical Education, Kerala.

Surekha Mariam Varghese is currently heading the Department of Computer Science and Engineering, M.A. College of Engineering, Kothamangalam, Kerala, India. She received her B-Tech Degree in Computer Science and Engineering in 1990 from College Engineering, Trivandrum affiliated to Kerala University and M-Tech in Computer and Information Sciences from Cochin University of Science and Technology, Kochi in 1996. She obtained Ph.D in Computer Security from Cochin University of Science and Technology.Kochi in 2009. She has around 25 years of teaching and research experience in various institutions in India. Her research interests include Machine learning, Network Security, Database Management, Data Structures and Algorithms, Operating Systems and Distributed Computing. She has published 17 papers in international journals and international conference proceedings. She has been in the chair for many international conferences and journals.