Research on architecture and query performance based on distributed graph database Neo4j

Hongcheng Huang, Ziyu Dong
Chongqing University of Posts and Telecommunications

***Abstract*—Neo4j database based on graph model is different from other database on implementation.Firstly, this paper explores the architecture of the system and its internal mechanism.Secondly, Neo4j offers three ways to query, and evaluate its query performance from several dimensions: data size, query complexity, query number, etc. The results show that there are obvious difference in performance under different scenarios. This paper analyzes the experimental results and selection suggestions of query ways are recommended later. It provides a reference of query performance optimization for specific business applications.**

Keywords- graph database; architecture; query performance

I. INTRODUCTION

With the development of the Internet and the advent of the era of big data, social networks[1], cloud services[2] and other business applications generated massive amounts of data, and the degree of correlation between data is more complicated and flexible, so scalability and flexibility of the relational database which is based on relational model is difficult to fully adapt to different application scenarios[3].No SQL database according to the data model can be divided into four categories[4]:K-V, Big Table, Document, Graph. Graph database can effectively utilize the graph model to store, manage, update data and relationships, whose schemas are free and flexible.Neo4j is a robust (fully ACID) transactional property graph database, which is open-source and written by Java programming language[5].In Neo4j,graph is composed of a large number of nodes and relationships. Both nodes and relationships can have properties which expressed in the form of key-value pairs.Neo4j ' s query methods include Java Core API, Traverser Frame Work and Cypher query Language[6].A comparison on query performance between Neo4j and MySQL is documented in [7].They reported query times of Neo4j being 2-5 times lower than MySQL for their 100 object data set and 15-30 times lower for their 500 objects data set. And it demonstrates that the graph model is superior to the relational model.[8] demonstrates that Neo4j has a higher query performance when running in embedded mode than in standalone mode. This article will evaluate and analyze the performance of different query ways from the data size, the complexity of relationships, and the quering stability, etc.

II. NEO4J SYSTEM ARCHITECTURE

A. System Structure

The Neo4j graph data is stored on the disk in the form of record files. In order to improve the speed of retrieval,Neo4j uses two layers of caching mechanism(file system cache and object cache).On disk, relationships contain most of the information, and each node only holds their first reference to the relationships.But on the contrary in the object cache,a node holds all relationships, and a relationship becomes very simple, just keep all its properties.A node ' s relationships are classified according to RelationshipType and it ' s good to traverse rapidly.Transaction modules which include transaction log and transaction management guarantee the ACID of transaction.HA module characterizes clustering capabilities of the system.The top layer is APIs providing the corresponding interface for the caller of the business.In HA cluster mode,transaction log and Core API are shared by all instances, while other modules and functions are based on local data of each running instance.

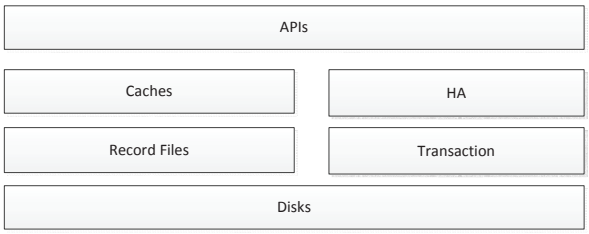

Figure 1. System structure

B. Storage Structure

1) Node

The data of nodes in Neo4j database is stored in the file named neostore. nodestore. db on disk.Like most of the Neo4j store files, the node store is a fixed-size record store where each record is 9 bytes in length. Fixed record sizes enable fast lookups for nodes within the store file—if we have a node with id 100 then we know its record begins 900 bytes into the file and so the database can directly compute the record location at cost O(1) without performing a search at cost O(log n)[9].Inside a node record, the first byte is the in-use flag which tells the database whether the record is currently used or can be reclaimed to store new records.The next four bytes are the of the first relationship connected to the node. The last four nodes bytes are the id of the first property for the node.

2) Relationship

The data of relationships in Neo4j database is stored in the file named neostore.relationshipstore.db on disk.Like the node store file, the relationship store contains fixed-size(33 bytes) records and each record contains the id of the nodes at the start and end of each relationship.

3) Property

The data of properties in Neo4j database is stored in the file named neostore.propertystore.db on disk.As well,it contains the

978-1-4799-2860-6/13/$31.00 ©2013 IEEE

property index store file(neostore.propertystore.db.index) and the dynamicStore record(being either in the DynamicStringStore in neostore.propertystore.db.strings or the DynamicArrayStore in neostore.propertystore.db.arrays depending on the property type).

C. HA Cluster Structure

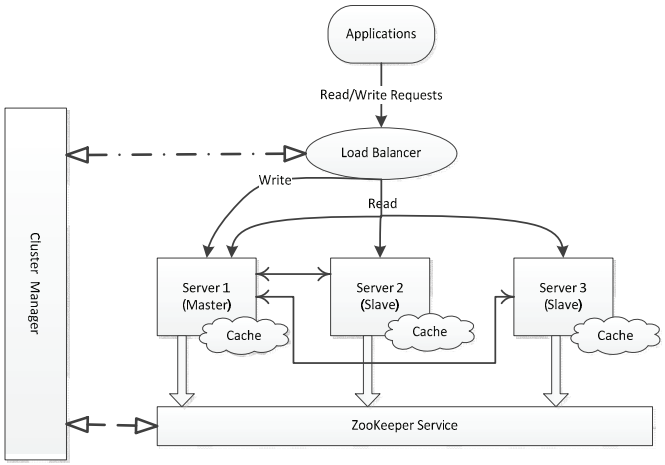

Figure 2. Neo4j HA cluster structure

Neo4j HA cluster is a Master-Slave structure.A load balancer deployed in the front of cluster directs read and write requests of applications.Usually write transactions are implemented by the Master node,each Slave node synchronizes with the Master node through the HA cluster protocol.So that each node in the cluster can maintain the data consistency.If write transactions are distributed to a Slave node,the transaction is synchronized to Master node firstly and then the results returned to the client.The latter approach can ensure data persistent every time(involves two instances, rather than one),but its response speed is slower than the former due to the network load and synchronization protocol overhead.The read requests are much simpler, they do not involve transactions, any node can handle the read requests.In addition,there will be Master node election in cluster,and this is mainly done through ZooKeeper Service component.ZooKeeper is equivalent to a distributed coordinator[10].Each active node in the cluster can be registered to the component.When the cluster started or the current Master node failed,ZooKeeper Service component will monitor these changes and elect a new Master timely.Cluster Manager provides a management method of Cluster,such as instructing the cluster configuration information, pointing out that what are the current available instances, whether need to elect the Master node,and so on.

III. QUERY PERFORMANCE TEST

A. Test Environment

Hardware environment: CPU Intel core i3-3220 (3.30 GHz), memory 4GB (Kingston DDR3 1600 MHZ).Software environment:using the enterprise version 1.9.1 and writting test applications in embedded mode that JVM's maximum available memory is 247M.The basic model of test data is shown in figure 3. There are two types of nodes: author nodes and paper nodes,and relationship between author node and paper node is [author] and relationship between paper nodes is [ref].

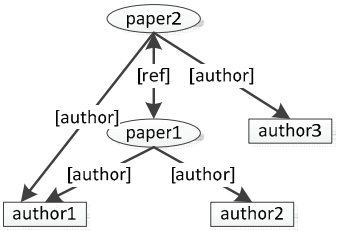

Figure 3. Basic Model of test data

B. Test Method

TABLE I. TEST CONDITIONS

Data1	1297810nodes,8929981properties,6963373 relationships,4053MB database disk usage
Data2	2566nodes,69525properties,49843relations hips,47MB database disk usage
Complexity1	find coAuthor of a paper written by an author
Complexity2	find a paper written by an author

All queries have been executed 11 times where the first time was thrown away since it warms up neo4j caches. The values are average values over the other 10 executions.There are two kinds of data size and complexity in the process of testing,as shown in table 1.

C. Test Results

1) Test case

Test case 1:Data1 and Complexity1,"queries numbers" (the number of author nodes) as variable,the time unit is second,the result is shown in figure 4.In the figure,both first figure and second figure are tested in the same conditions, but with different horizontal axis.

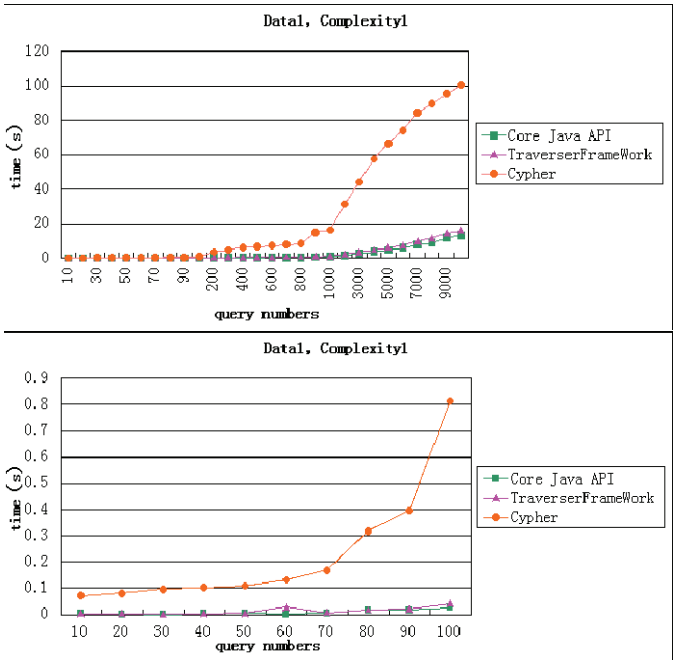

Figure 4. TestCase1

Test case 2:Data1 and Complexity1,measuring query performance of two kinds of the number of author nodes 100,1000,the time unit is second,the result is shown in figure 5.

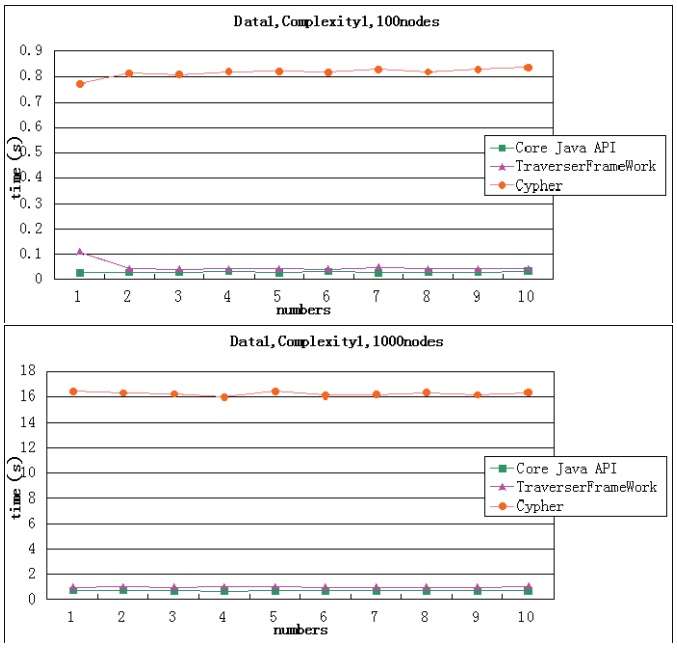

Figure 5. TestCase2

Test case 3:Data1 and Complexity2,” queries numbers” (the number of author nodes) as variable,the time unit is second,the result is shown in figure 6.

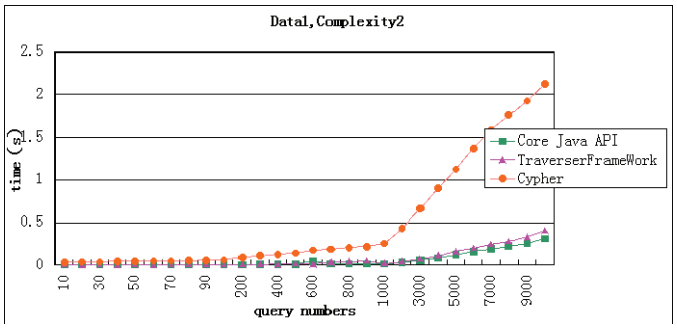

Figure 6. TestCase3

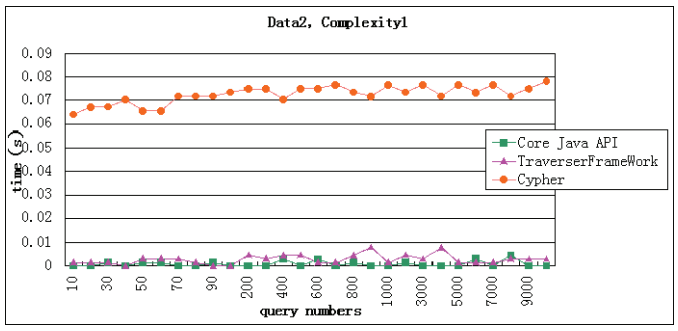

Figure 7. TestCase4

Test case 4:Data2 and Complexity1,” queries numbers” (the number of author nodes) as variable,the time unit is second,the result is shown in figure 7.

Test case 5:Data2 and Complexity1,measuring query performance of two kinds of the number of author nodes 100,1000,the time unit is second,the results is shown in figure 8.

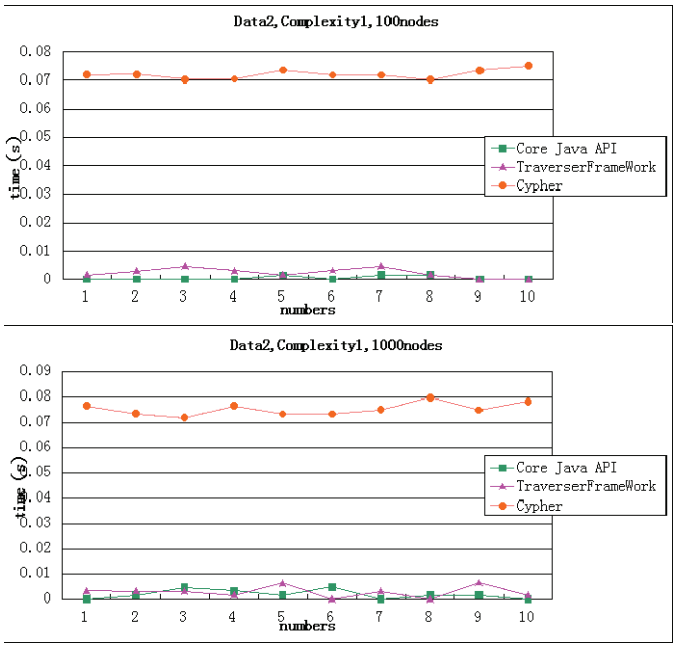

Figure 8. TestCase5

Test case 6:Data2 and Complexity2,” queries numbers” (the number of author nodes) as variable,the time unit is second,the result is shown in figure 9.

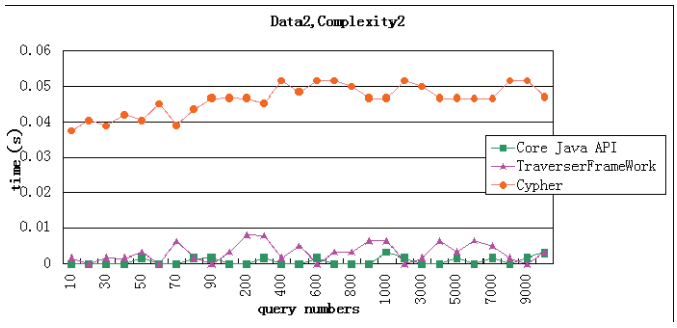

Figure 9. TestCase6

IV. ANALYSIS

A. The influence of query method to query performance

Cypher is a query language that having SQL-like syntax and easy to learn.The above test codes indicate that the Core Java API in use is the most complicated in these three ways, followed by TraverserFrameWork, Cypher.However, the above figures show that the performance of the three kinds of query methods, in contrast,with preferential treatment to the bad sequence is:Core Java API, TraverserFrameWork,Cypher.This is because that kernel is the center of Neo4j from the point of view of system structure and apis are outside of kernel.TraverserFrameWork is built on top of Core Java API, and Cypher is encapsulated by TraverserFrameWork.If pursue a shorter response time,we can choose Core Java API to query.If focus on easy and maintainable, we may use Cypher.

TraverserFrameWork can be regarded as a good compromise because of its outstanding performance close to Core Java API and easier to use than it.

B. The influence of data size to query performance

Compared figure 4 and figure 7,we can see that if the value of "query numbers" is small(e.g. in 50), the query performance of these two methods(Core Java API and TraverserFrameWork) is basically independent of the amount of data, and both query time is very close with the same trend when data size increases gradually.Increasing with data size, the overall trend is that the query time of these three ways will increase. For Cypher, data size has a greater impact than Core Java API and TraverserFrameWork.We can see that, the memory for the query performance is very much affected. Whether single instance or Neo4j's HA structure, increasing exclusive memory helps improve query performance.

C. Query performance stability

Figure 5 and figure 8 show that Neo4j's query performance of each query way is very stable when in the same of complexity.Wether the number of nodes(1000 nodes and 100 nodes) is 10 times, consuming time of the former(1000 nodes) of each query way is 20 times than the latter(100 nodes) in figure 5 and 1 time in figure 8.Therefore,particularly in the case of large data size,decompositing one query task into multiple sub-tasks can be properly considered to improve read performance.

D. The influence of query complexity to query performance

Figure 4, figure 6, figure 7 and figure 9 show that within the same data size, three query ways remain the same query time and do not matter with different query complexities when query number is small.But When query number exceeds a certain value,the more complex of query, the more time consuming. The trend of query performance of different query complexities is consistent increasing with query numbers in the same data size.

CONCLUSION

The data of Neo4j is stored on disk and data will be loaded into memory when originally query and then cached.For a Neo4j's HA cluster, each Neo4j instance data is consistent and their respective queries are based on each instance of independent occupied memory space.It means that when distributing query requests of different business applications, the query performance of HA cluster is determined by each Neo4j instance's query performance. So just evaluating a Neo4j system's query performance can know the query performance of HA cluster. As can be seen through the above evaluation, for Neo4j system, selection of query ways need to be taken full account of data size,query complexity,query numbers, the cost of development or maintenance, and other factors in every specific application scenario.The test cases above provide a certain reference.

REFERENCES

[1] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee.Measurement and analysis of online social networks.The 5th ACM/USENIX Internet Measurement Conference, 2007.

[2] Yanmei Huo, Hongyuan Wang, Liang Hu, Hongji Yang. A Cloud Storage Architecture Model for Data-Intentsive Applications. 2011.

[3] Szabolcs Rozsnyai, Aleksander Slominski, Yurdaer Doganata.Large-Scale Distributed Storage System for Business Provenance.Cloud Computing (CLOUD), IEEE International Conference,2011.

[4] Ilya Katsov. NoSql Data Modeling Techniques. March 2012.

[5] Neo technology, Inc.Neo4j Home. http://neo4j.org, 2012.

[6] Neo technology, Inc.The Neo4j Manual v2.0.0,2013.http://docs.neo4j.org/chunked/milestone.

[7] Shalini Batra, Charu Tyagi. Comparative Analysis of Relational And Graph Databases. International Journal of Soft Computing and Engineering (IJSCE), 2(2), May 2012.

[8] Florian Holzschuher,Prof. Dr. René Peinl. Performance of Graph Query Languages:Comparison of Cypher, Gremlin and Native Access in Neo4j.EDBT/ICDT '13March 18 - 22 2013.

[9] Ian Robinson, Jim Webber, Emil Eifrem. Graph Databases(Early release revision 1).O' Reilly Media, Inc., 02-25 2013.

[10] Apache Software Foundation.Home.http://zookeeper.apache.org, 2013.