Contemporary Engineering Sciences, Vol. 7, 2014, no. 29, 1603 - 1608
HIKARI Ltd, www.m-hikari.com
http://dx.doi.org/10.12988/ces.2014.411216

Development of Cloud P2P OLAP System Based on Balanced Quadtree

Kil Hong Joo

Dept. of Computer Education
Gyeongin National University of Education
Anyang Gyeonggido, Korea

Abstract

To make use of the recent distributed OLAP environment, researches on P2P (Peer-to-Peer) OLAP with the dynamic hybrid tree (DHT) and Grid OLAP are actively proposed. In addition, OLAP query execution costs many minutes by its enormous data and OLAP query properties. To adapt OLAP systems to the cloud computing environment, P2P OLAP has problems on executing multi-dimensional range queries because of P2P properties. In this paper, the central management Cloud P2P OLAP is proposed to provide an efficient cloud computing environment which query results are able to be reused with the time-series properties by various users. The Cloud P2P OLAP is grafted onto hierarchical hybrid P2P for fast query results and multi-dimensional range query, so that the index loads are distributed and the performance is improved.

Keywords: P2P, OLAP, Cloud Computing, Query index, Balanced Quadtree

1. Introduction

On-line analytical processing (OLAP) systems are at the heart of many *business analytics* applications [1, 2]. In contrast to queries for *on-line transaction processing* (OLTP) systems which typically access only a small portion of the database (e.g. update a customer record), OLAP queries may need to aggregate large portions of the database (e.g. calculate the total sales of a certain type of items during a certain time period) which may lead to performance issues.

Therefore, most of the traditional OLAP research, and most of the commercial systems, follow the *static* data cube approach and materialize all or a subset of the cuboids of the data cube in order to ensure adequate query performance [3, 4, 5]. However, the traditional *static* data cube approach has several disadvantages. Google's Bigtable [6] extended the index concept like B+ tree into multi-level distributed system. This paper focuses on such a distributed central managed index. As a result, the multi-level hybrid P2P is grafted on OLAP environment. Although OLAP researches on P2P or Grid with limitedly managed indexes have already progressed, this paper proposes improved algorithms of range queries and time series management which are passed over by existing works. This paper proposed the multi-dimensional range query execution by quad tree grafted with interval tree additionally on time-series index. Furthermore, the advanced index with multi-level hybrid P2P is used to work on Cloud P2P OLAP which includes hierarchical nodes and time-series concepts.

2. Cloud P2P OLAP Index and Algorithm

In this paper, the multi-layer (N levels) hybrid P2P is proposed with indexes shown in Figure 1. N means thee level layers are able to be expanded by the size of indexes. The minimum value of N is 2, which is P2P composed of only the central server and 1 level client - servers. Under the corresponding cloud platform, an instance insertion or deletion doesn't require to reboot OS and service instances are increased even when the system is working on the service. Therefore, these infinite expandability and flexibility make a linear performance improvement. The reason Cloud P2P OLAP system has an index server in the cloud system is that the multi-level index expansion is able without any specified hardware and the usage fee is just as much as how much it is used.

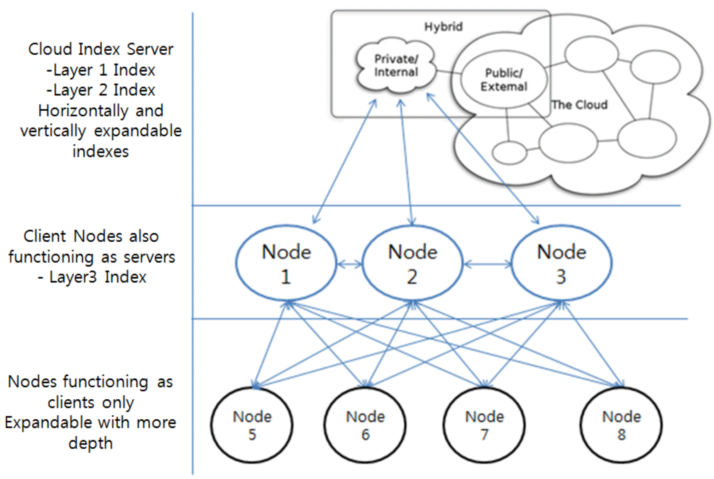

Figure 1: Cloud P2P OLAP system

In this paper, two-dimensional adjacency information is proposed to represent division similarity information together with one-dimensional physical information. The physical adjacency is important considering network transfer efficiency, and the logical adjacency is important to find existing data in a similar department and to reuse data afterward. For an index structure to show two-dimensional information, R-tree, KD-tree, etc. are used. However, to adapt time-series indexes together, a quadtree index is proposed in this paper. After determining the adjacency, nodes within the queried time-series are examined from the most adjacent nodes by expanded recursive calls. If the balance mechanism works, the time-series information at each depth are summarized and used to reconstruct the entire time-series tree. In this case, the quadtree is very advantageous. However, the balance is not guaranteed, so the inquiry performance can be a drawback. In this paper, to overcome the unbalanced quadtree drawbacks, the preprocessing algorithm is additionally proposed to reconstruct the index. When OLAP updates the materialized view at nighttime periodically, P2P OLAP on cloud computing also reconstructs the index to maintain day-unit balance. The result of this mechanism is shown at Figure 2. Figure 3 shows the result of the balance mechanism even though the preprocessing is executed once.

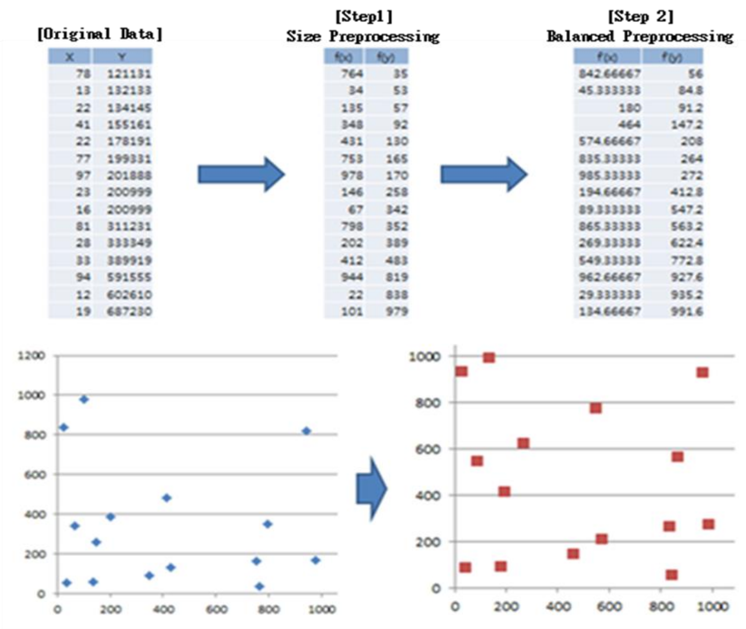

Figure 2: Balanced preprocessing mechanism

In this paper, end-Cloud P2P OLAP index in Layer 1 expresses relationship of cubes as a tree. In addition, both the physical and logical adjacency indexes in Layer 2 are managed to the Interval-tree by Quadtree. Hence, if query index is not included to parent node, it is not included to the set of child nodes. In this paper, weaknesses of bit index are overcome by the managing time series limits. Figure 4 shows the Cloud P2P OLAP algorithm. If any node in the node 12 is not satisfied to the value of time series, this algorithm checks interval of any node and it finds the candidate node in the set of parent adjacency nodes.

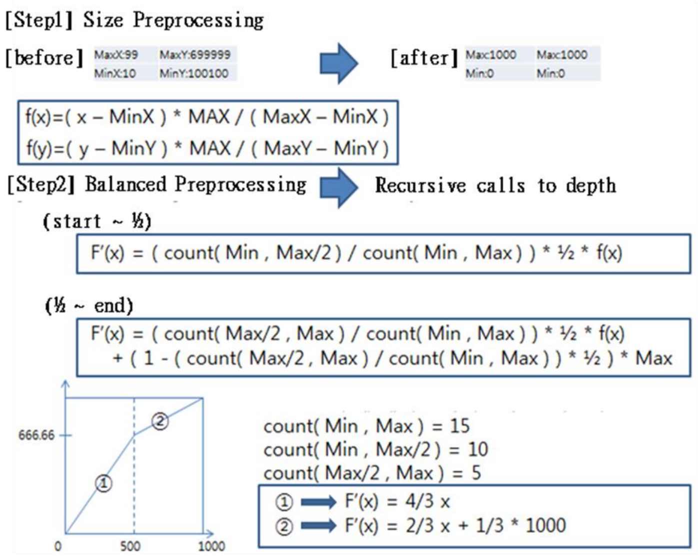

Figure 3: Balanced Quadtree Algorithm

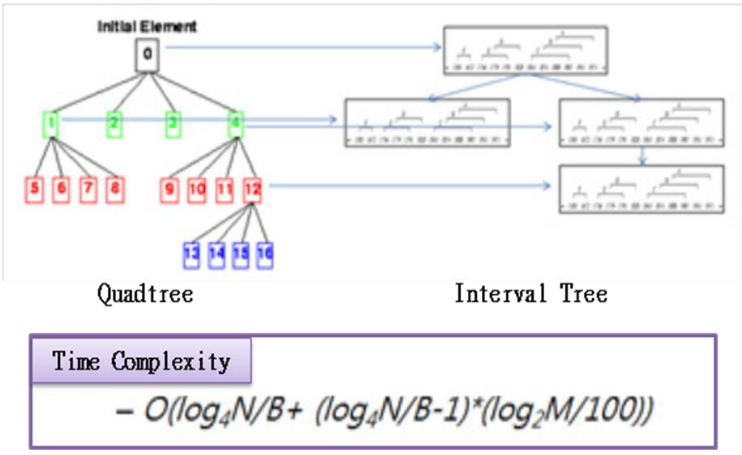

Figure 4: Cloud P2P OLAP Algorithm

3. Experiments

The proposed system is implemented on .NET Framework 3.5 with C#. The SOA service communication protocols in P2P and Cloud are developed with WCF and SOAP protocols. The 4 dimensional data set with 2,930,459 row is used to analyze the performance. In Figure 5, the searching efficient of the compound balanced quadtree and interval tree is compared with those of the compound ubbalanced quadtree and interval tree for the from 100 to 10000 nodes. The balanced quadtree is more efficient than unbalanced quadtree about 40 percent. In Figure 6, the searching efficient of the compound balanced quadtree and interval tree is compared with those of the compound PR-tree and interval tree for the from 10 to 1000 nodes. The balanced quadtree is more efficient than PR-tree in less than 600 nodes. However, if critical value is 8, 600 nodes mean 4800 server. Since this interval can satisfy the servers and client nodes any company, proposed balanced quadtree is very efficient.

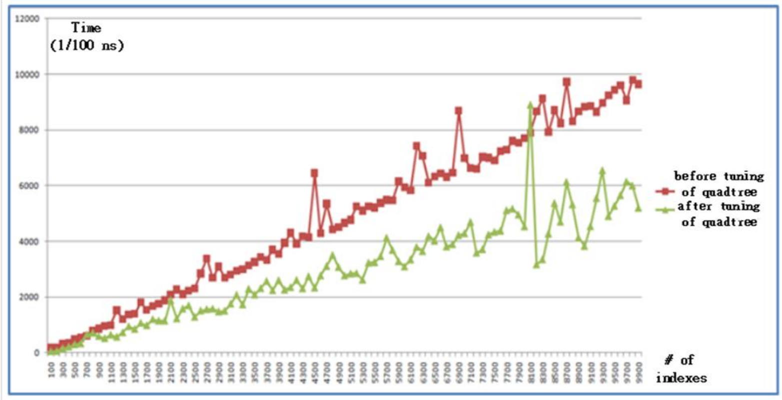

Figure 5: Comparison searching efficiency between Balanced Quadtree + Interval tree and Unbalanced Quadtree + Interval tree

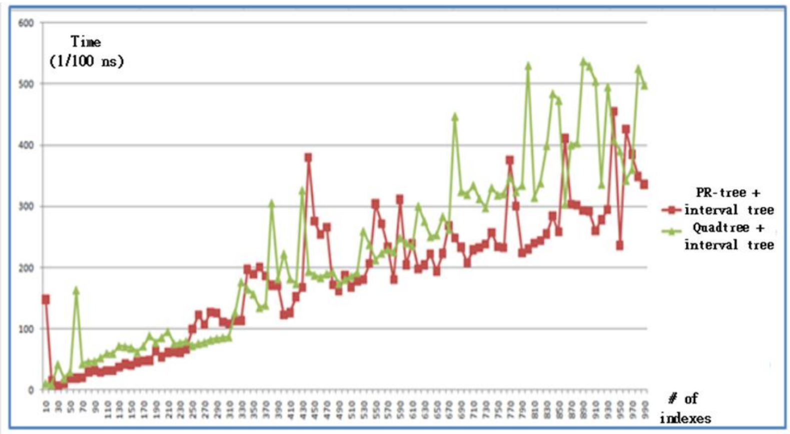

Figure 6: Comparison searching efficiency between Balanced Quadtree + Interval tree and PR-tree + Interval tree

4. Conclusion

The optimized algorithm are proposed and proved by experiments. Also, on hybrid P2P which indexes are managed centrally, the usability of index servers in cloud system are proved to maximize the performance of servers and to manage effectively. P2P OLAP should be able to reuse in time-series and to query multi-dimensional range on cube caches. Cloud P2P OLAP is proposed to add/removes dynamically instances to maximize expandability and usability. The architecture including cloud systems and P2P together are newly proposed and proved to have better performance than general OLAP systems by experiments.

References

[1] McWherter, D. T.; Schroeder, B.; Ailamaki, A.; Harchol-Balter, M.: Priority mechanisms for OLTP and transactional Web applications, *International conference on Data Engineering* (2004) 535 - 546.
http://dx.doi.org/10.1109/icde.2004.1320025

[2] Hasso Plattner: A common database approach for OLTP and OLAP using an in-memory column database**,** *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data* (2009) 1 - 2.
http://dx.doi.org/10.1145/1559845.1559846

[3] Gupta, H.; Dept. of Comput. Sci., Stanford Univ., CA, USA: Harinarayan, V.; Rajaraman, A.; Ullman, J. D.: Index selection for OLAP, *International conference on Data Engineering* (1997) 208 - 219.
http://dx.doi.org/10.1109/icde.1997.581755

[4] Uwe Röhm, Klemens Böhm, Hans-Jörg Schek: OLAP Query Routing and Physical Design in a Database Cluster, *International Conference on Extending Database Technology* (2000) 254 - 268.
http://dx.doi.org/10.1007/3-540-46439-5_18

[5] Kil Hong Joo, Won Suk Lee**:** An Adaptive Materialization Method for Globally Optimized Query Evaluation in a Hierarchical Mediator System, *Computer Journal* 50 (1), (2007) 68 - 80. http://dx.doi.org/10.1093/comjnl/bxl047

[6] F. Chang, J. Dean, S. Ghemawat, WC. Hsieh, DA. Wallach, M. Burrows, T. Chandra, A. Fikes and RE. Gruber, “Bigtable: A distributed storage system for structured data”, *Journal ACM Transaction on computer systems* 26 (2), (2008) 205 - 218.

Received: June 20, 2014; Published: December 2, 2014