TELKOMNIKA, Vol. 11, No. 10, October 2013, pp. 6166 ~ 6172
ISSN: 2302-4046

An Improved Resource Query and Location Algorithm Based on Cloud Computing

Wuxue Jiang*[1,2], Jing Zhang[1], Junhuai Li[1] , [3]Hui Hu
[1]School of Computer Science and Engineering, Xi'an University of Technology, Xi'an 710048, Shaanxi, China
[2]Department of Computer Engineering, Dongguan Polytechnic, Dongguan 523808, Guangdong, China
[3]Department of Science and Research, Huizhou Univeresity, Huizhou 516007, Guangdong, China
*Corresponding author, e-mail: jiangwuxue_1@163.com

Abstract
With the continuous development of cloud computing applications, the problems of cloud computing are more and more obvious, such as the lower efficiency of resource search, the lack of network scalability, and some difficulties in management. In this paper, the problems of P2P network and the structural characteristics of the IS-P2P network are studied. On the basis of traditional Chord algorithm, the improvement of the speed of search is made. BiChord: a resource locator and query algorithm based on cloud computing is put forward. The bidirectional routing tables FingerTable and RefingerTable are designed by this algorithm, which adopts the query strategy of bidirectional location of keywords. In addition, the paper also proposes XP query: a message query mechanism based on the structure of the cloud computing. Simulation results show that BiChord has better routing performance than traditional Chord algorithm and the message query mechanism also improves the efficiency of resource queries.

Keywords: *BiChord Algorithm; Resource Location and Query; XP query; Cloud Computing*

Copyright © 2013 Universitas Ahmad Dahlan. All rights reserved.

1. Introduction

At present, there are obvious advantages and broad prospects in the application and development of cloud computing. With the rapid growth of network users, many problems that exist in the network model are becoming serious obstacles to the further development of cloud computing technology, although there is development in cloud computing technology. According to analysis, the main problems of cloud computing technology include:

1.1. Lack of Scalability of Routing Network Model

Scalability has always caused serious troubles to cloud computing technology researchers and developers. The Gnutella model is the most widely used model. According to research made by Clip2, users of 56k modem can query no more than 20 messages within one second. If there are more than 1000 network nodes, then the processing limit will be broken. The node failure on this part will inevitably result in the fragmentation of Gnutella network [1, 2]. Thus access and query can only be conducted within a very small part of the network, causing a serious lack of scalability of the network.

1.2. Differences of Physical Topology between Cloud Computing Overlay Network and Underlying Network

The problem is described based on the Gnutella model. The detour problem occurred in the Chord network

model is caused by the mapping method designed by the hash algorithm, which causes two adjacent nodes in hash ring look close to each other, but they may be far away from each other in the real network topology, thus resulting in seriously low efficiency of routing.

1.3. Problems in Management

Due to too much freedom of network users in applicable P2P technology, there is a tendency of "anarchism" that is difficult for effective management, resulting in pornography and virus on P2P network and facilitating illegal trade [3, 4]; the problems of traffic calculation,

Received May 31, 2013; Revised July 14, 2013; Accepted July 24, 2013

charges, verification of the value of goods are all difficult issues that need to be solved, resulting restrictions of P2P technology in specific applications. The researches on P2P technology are mainly focused on network performance, search mechanism and security[5]. In this paper, the new structure of P2P network is studied and the method of improving the efficiency of search and location is put forward from the users' point of view. On the basis of the IS-P2P network model, the query mechanism is studied. The inefficiency of query in P2P network is improved by the use of multi-level XP query mechanism.

2. Cloud Computing and P2P Technology

Cloud computing is a new IT technology following grid computing. It has a certain risk while it brings the new opportunities for supply chain information coordination mode. Based on the cloud computing technology, this paper takes the advantages and disadvantages of its applications in supply chain information coordination into a comprehensive consideration and proposes a supply chain coordination mode which is capable of resisting the cloud computing risk. Cloud computing system takes advantage of the tens of millions of idle computing resources on the Internet so that it can be more powerful than the centralized computing system. However, its resource scheduling faces challenges due to its massive resources, heterogeneous nature, and network communication delay.

In order to maximize the advantages of P2P technology and overcome the weaknesses of P2P technology, this paper applies the IS-P2P network model which creates a tier of virtual logical topology based on a distributed hash table (DHT) on the basis of physical topology of the Internet, which effectively establishes a P2P routing network model with a distributed structure [6, 7].

In the IS-P2P network model, in order to achieve high dynamics and efficient search function, the control tier is designed into two tiers in which the first tier is an index network, a P2P network composed by index peers, and the second tier is a P2P network composed by ordinary peers.

The index network composed by inodes is located in the upper side, which is a relatively complete structured P2P network. Its function is to provide index service to the underlying layer. Each inode has its own ID number in the query and no longer takes the flooding method in information query and release. Instead, information can travel from any node of ID0 to the target node ID1, which makes it possible to avoid information overload for easy control. Thus, IS-PSP enjoys the advantage of convenient management. In the DHT network mode, if instability of the network is caused by leaving or joining a new inode, further operation is necessary to ensure the stability of inodes in dynamic changes.

3. Location of IS-P2P Resource and Query Algorithm

3.1. Improvement of Chord Algorithm

In the Chord algorithm, each BP maintains the adjacent relationship between itself and its successor node, so it is always possible to correctly locate any node in the ring. But there are serious deficiencies in the Chord algorithm: adjacent nodes in counterclockwise direction need to traverse the entire Chord ring, resulting in great inconvenience in searching nodes [8, 9].

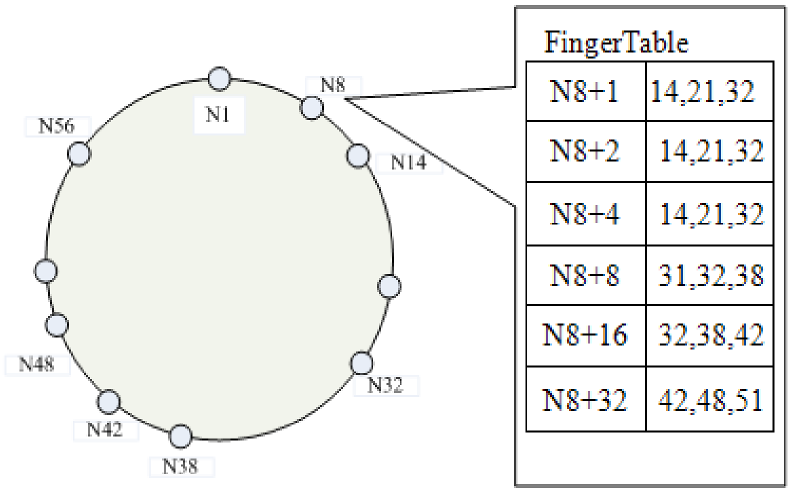
In the BiChord algorithm, each node is responsible for the maintenance of both routing tables and saves the information of some nodes in the ring, including information in clockwise and counterclockwise directions. Bidirectional search and location in both clockwise and counterclockwise direction are started from the initial node of query, which allows for rapid localization.

Its main composition includes:

A. Construction of a Bidirectional Routing Table

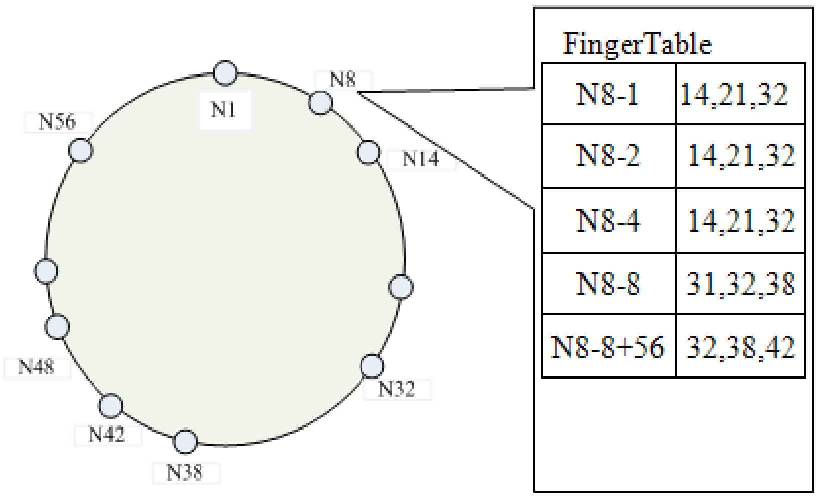
When nodes are added to the improved Chord ring, the construction of two routing tables – FingerTable and ReFingerTable – are required for each node. Both routing tables save the information of the successor nodes in the clockwise and counterclockwise directions respectively. Each table entry maintains j available successor nodes. The SuccessorList of entry i in the FingerTable of the node n saves j successor nodes from n+2i, here they are marked as finger[i].node[1], finger[i].node[2]…finger[i].node[j]. If finger[i].node[1] is disabled, finger[i].node[2] will replaces it to continue the function of node. The method ensures the continuity of nodes. The structure of FingerTable and ReFingerTalbe are shown in Figure. 1 and Figure. 2 respectively.

In the above process, if it needs to search the successor node of each node, the time required to calculate m entrances of TableFinger and ReFingerTable is O(m*logN). Here, to reduce the time overhead in the query as much as possible, the improved Chord is able to check the difference between the entrances of an interval and its successor or determine whether they are identical, which reduces the time needed to check the number of nodes to O(logN), reducing the time overhead. At the same time, it is necessary to update the routing tables, i.e., synchronous updating of the two tables is required to ensure their rapidness and accuracy.



N8+1	14,21,32
N8+2	14,21,32
N8+4	14,21,32
N8+8	31,32,38
N8+16	32,38,42
N8+32	42,48,51

Figure 1. FingerTable Structure



N8-1	14,21,32
N8-2	14,21,32
N8-4	14,21,32
N8-8	31,32,38
N8-8+56	32,38,42

Figure 2. ReFingerTable Structure

B. Updating of FingerTable and ReFingerTable

The process of updating starts in the sequence of 1 to m in the FingerTable. The updating of entry i in the FingerTable is decribed as follows:

If a new node n is added to the system, first find n' – the predecessor of n-2i and request n' for the updating of the FingerTable. After n' receives the request, first compare the value of Finger[i].node[1] of the first node of entry i in the FingerTable with that of the node n'. If the former is higher than the latter, then compare the value of Finger[i].node[1] of the last node of entry i in the FingerTable with that of the node n'; if (a) the former is higher than the latter, then check whether the value of the new node n is between the values of two entries of the FingerTable; if it is, replace the entry of higher value with the new node n and information updating is finished; if it is not, send a request to the precursor node of n-2i for the updating of entry i of the FingerTable.

If the node n is not the last node in the ring and the value of Finger[i].node[1] is higher than that of the new node n, then: (a) compare the difference between the new node n and 2m-Value(n'), if the value of the new node n is higher, check whether it is between the values of two entries of the FingerTable, if it is, replace the entry of higher value with the new node n and the information updating is finished; (b) if it is not, send a request to the precursor node for the updating of entry i of the FingerTable.

Similar to the updating of the FingerTable, the improved Chord also requires the updating of the ReFingerTable, but the process is reversed.

Finally, the joining of the new node may result in the mapping changes between keyword and node, which requires the transfer of keywords.

At the same time, the process of joining and leaving of node is completely reversed. During the leaving process, the ReFingerTable and the FingerTable must be updated to transfer the key values stored in a node that has left to the successor node.

3.2. Bidirectional Location of Keyword in Clockwise and Counterclockwise Directions

In the improved Chord, if it is to locate the key value k corresponding to data, send requests from the clockwise and counterclockwise directions to locate the node corresponding to the value of k from two directions. Once the process succeeds, relevant information will be sent to the initial node.

The query algorithm is described as follows:

n: network node; Value (n): identification of node n;

predecessor (n): a previous (precursor) node of node n;

successor (n): the next (succeessor) node of node n;

m: number of entries in the FingerTable;

Finger[i].node[i]: the ith successor node of entry i in the FingerTable;

ReFinger[i].node[i]: the ith successor node of entry i in the RefingerTable.

The steps of the BiChord bidirectional location algorithm are as follows:

Step 1: If Value (n) is in the current node n, it returns the current node address and the query is successful;

Step 2: If Value (n) is between the hash value of the current node n and that of the successor node of node n, then continue to query clockwise with the successor node of n as the next hop, i.e., find node n' with the hash value closest to keyword in the FingerTable and node n'' after node n' but with unequal value. Repeat the cycle.

Step 3: If Value(n) is between the hash value of the current node n and that of the predecessor node of node n, then query counterclockwise with the successor node of n as the next hop, i.e., find node n' with the hash value closest to keyword in RefingerTable and node n'' after node n' but with unequal value. Repeat the cycle.

In the bidirectional location algorithm, the switching of clockwise and counterclockwise direction is determined by the position of the keyword.

4. Search Processes of IS-P2P Structural Resources

The special two-tier structure of the IS-P2P system makes the query process of node differ much with other P2P systems. The complexity of query is closely related to the number of indexes of the resources that satisfies the query.

The XP query is a query mechanism based on the IS-P2P structure with applications gradually nested and has a good adaptability to the index-based peer-to-peer network resource query [10].

If the inode has abundant index resources that satisfy the node query demand, XP can just perform local query to meet users' demand instead of forwarding the query request. Peer sends a query request to Query. XP obtains the result that satisfies the query requirements through local query, sets up a query result message Result, and sends it to Peer. The procedure of XP complex query is shown in Figure. 3.

If the resources on XP cannot satisfy the query request of the node, XP needs to modify query request of the node. If the original query depth is d and the required number of results is m, then XP needs to construct a query message with unchanged keyword, query depth of d-1 and required number of results of m/n, and forwards query message to n subsequent inodes in the RID list. The satisfied query results are integrated by XP into a query result message and respond to the requesting node.

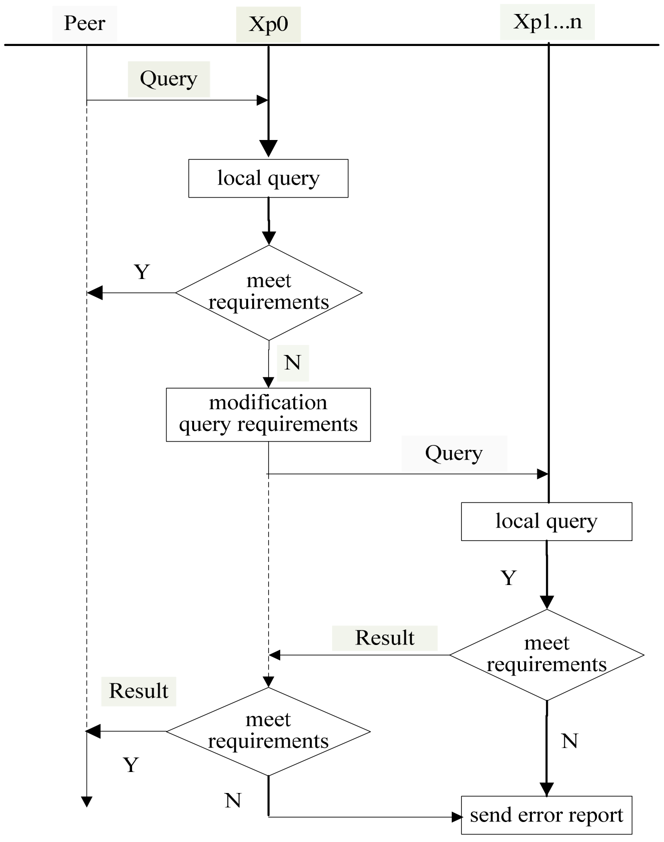


Figure 3. The procedure of XP complex query

5. Simulation Experiment of Model

The simulation experiment is performed on a computer (Intel 3GHz, 2M) and a P2Psim simulation platform. The experiment compared and tested the routing hops and query time in the resource location process. In order to highlight the superiority of the BiChord algorithm performance, Segment processing is done on the selection of the number of nodes.

The number of routing hops refers to the number of intermediate nodes passed by routing path involved in each query event [11]. Document query time refers to the total of all time delays on each node in the path of document query event [12]. The comparison of the average number of routing hops between BiChord and Chord is showed in Figure 4.

A. When the number of nodes is no more than 1000, there is no difference of the average number of routing hops between BiChord and Chord. The number of hops is normally 3 because the number of hops required to query the traverse of the initial node in the entire ring by unidirectional routing is 3 when the number of nodes is small.

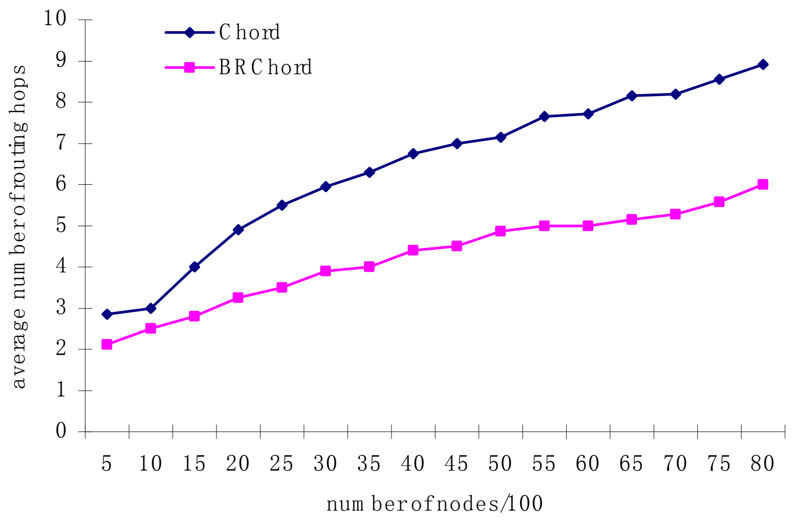


Figure 4. Average Routing Hops Graph of BRChord and Chord

B. When the number of nodes is over 3000, the average number of routing hops in the query event in Chord will be more than 6, while that of BiChord is less than 4. This is because that most query events are finished by only traversing half of the ring, instead of the entire ring.

The distribution of the average query time of BiChord and Chord is shown in Figure. 5.

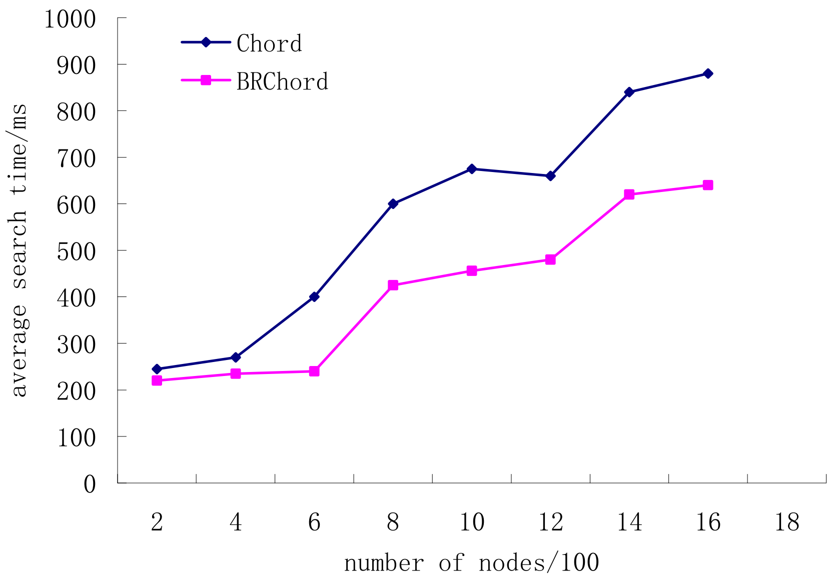


Figure 5. Average Search Time Graph of BRChord and Chord

As can be seen in Figure 5:

A. When the number of system nodes is less than 600, the average query time of BiChord and Chord for each document is close, both within 300ms.

B. When the number of system nodes reaches 800, the query time of BiChord has a nearly 33% reduction as compared with that of Chord, which demonstrates significant improvement in the efficiency of BiChord query.

C. When the number of the system nodes is over 1400, with the increase of the number of nodes in the system, the increase of the query time of BiChord tends to be smooth as compared with that of Chord.

6. Conclusion

This paper proposes an improved network resource location algorithm - BiChord algorithm based on the characteristics of the IS-P2P network model. The algorithm enjoys theoretical feasibility in terms of construction, updating and bidirectional query mechanism of routing tables. The introduction of XP query mechanism provides good adaptability for the index-based in-depth resource query. The example simulation experiment proves that the combination of the BiChord algorithm and the XP query shows significant superiority in terms of the number of hops of resource location and the distribution of average query time, which improves the routing performance and the efficiency of resource location.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Grant No. 61172018), and by the National Natural Science Foundation of China (Grant No. 61172124), and by by Science and Technological Program for Dongguan′s Higher Education, Science and Research, and Health Care Institutions (No.2011108101010), and by the Natural Science Foundation of Guangdong Province (Grant No. S2012010009865) and Guangdong Province Science and technology plan project (Grant No. 2012B010100040).

References

[1] Schmidt C, Parashar M. Enabling flexible queries with guarantees in p2p systems. *IEEE Internet Computing*. 2004; 8(3): 19-26.
[2] I Stoica, et al. Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications. *IEEE/ACM Transaction on Networking*. 2003; 11(1): 17-32.
[3] Yuh-Jzer Joung, Terry Hui-Ye Chiu, Shy-Min Chen. Cooperating with free riders in unstructured P2P networks. *Computer Networks*. 2012; 56(1): 198-212.
[4] T GU, HK Pung and DG Zhang. Information retrieval in schema-based P2P systems using one-dimensional semantic space. *Computer Networks*. 2007; 5(16): 4543-4560.
[5] P Haase, R Siebes, F Harmelen. Expertise-based peer selection in peer-to-peer networks. *Knowledge and Information Systems*. 15(1): 75-107.
[6] Chahita Taank, Rajesh Bharati. Load Balancing Algorithm for DHT Based Structured Peer to Peer System. *International Journal of Emerging Technology and Advanced Engineering*. 2013; 3(1): 356-359.
[7] S Surana, B Godfrey, K Lakshminarayanan, R Karp, I Stocia. Load Balancing in Dynamic Structured P2P Systems. *Performance Evaluation*. 2006; 63(6): 217-240.
[8] Y Zhu, Y Hu. Efficient Proximity-Aware Load Balancing for DHT-Based P2P Systems. *IEEE Trans. Parallel and Distributed Systems*. 2005; 6(4): 349-361.
[9] H Shen, CZ Xu. Locality-Aware and Churn-Resilient Load Balancing Algorithms in Structured P2P Networks. *IEEE Transaction on Parallel and Distributed Systems*. 2007; 18(6): 849-862.
[10] TL Casavant, JG Kuhl. A Taxonomy of Scheduling in General-Purpose Distributed Computing Systems. *IEEE Transaction Software Engineering*. 2008; 14(2): 234-246.
[11] M Cai, M Frank, J Chen, P Szekely. MAAN: A multi-attribute addressable network for grid information services. *Journal of Grid Computing*. 2004; 2(1): 3-14.
[12] M Kavitha, B Chellaprabha. An efficient message passing technique using route path information in unstructured peer to peer network. *International Journal of Computer Science and Information Technology & Security.* 2012; 2(2): 456-458.