2010 16th International Conference on Parallel and Distributed Systems

A New Structured Peer-to-Peer Architecture Based On Physical Distance

Song Wang[1,2], Yong Ma[1], Gang Wang[1] and Xiaoguang Liu[1]
[1]*College of Information Technical Science,Nankai University Nankai-Baidu Joint Lab, Weijin Road 94, Tianjin, China*
[2]*Military Transportation University, The Equipment support Department, Tianjin, 300161, China*
jackws66@yahoo.com

***Abstract*—Recently, structured P2P (Peer-to-Peer) system has become more and more popular. However, P2P systems' large scale and high dynamics have brought a great challenge to data availability and accessing performance. Redundancy techniques are used to solve these problems. However, in structured P2P systems, due to the consistent hash, the overlay network could not match underlying physical network well. The nodes close to each other in the overlay network may have long distances of physical network. In this paper, we put forward a new P2P architecture that constructs node identifiers and places redundant data according to physical location information. It can provide better load balance and access performance. The node's identifier is divided into four parts, representing the node's state, ISP, city and IP respectively, so the nodes having similar identifiers are close to each other in the physical network. Moreover, a query tree is used to help a node routing queries quickly in the physical network. In addition, we maintain an access list for each file. When a node becomes overloaded, replicas are placed on another node selected in the routing path according to the access list, so subsequent access requests could be met in advance.**

***Keywords*-P2P, physical distance, redundancy**

I. Introduction

Recently, P2P has become more and more popular. It has been widely used in file exchanging, peer-to-peer computing and so on. P2P systems have large scale and highly dynamics, that have brought a great challenge. In large-scale P2P systems, hundreds of millions of nodes may join and leave unpredictably, the data availability and access performance will be greatly reduced.

At present, redundancy techniques are used to improve the data reliability, the access performance and the load balance. The crucial problem is how to place and lookup the redundant information. However, In the current structured P2P networks (such as Chord [1]) the routing process and the placement of redundant information does not take into account the physical distance between nodes, leading to high delay.

This paper proposes a method to improve Chord protocol, in which routing and replica placement are based on the physical location information. Based on the improved protocol, the majority of communication is limited in the physically proximal nodes. Our method divides the node's identifier into four parts, representing the node's state, ISP, city and IP respectively, so the nodes whose identifiers are more similar will be more proximal to each other in the physical network. For each node, we use a query tree to deliver requests to it physically adjacent nodes. We also maintain an access list. When a node becomes overloaded, it will place replicas on the routing path according to the access list, so subsequent access requests will be met in advance. At last, we design several simulating experiments to verify the method.

II. Related Work

A. Redundancy mechanism

Redundancy techniques, mainly replication and erasure coding, are the general approaches to improve P2P systems' data availability.

Replication is the most simple redundancy technique. Many replicas of the original data are distributed to the P2P network. Recently, there are many replication strategies in the context of both unstructured [2] [3] and structured overlay networks [4] [5]. An important issue is where to place the replicas. There are three strategies. The first method places fixed number of replicas in successor nodes evenly. If the host node fails, it can be replaced by the subsequent nodes automatically. CFS [6] uses this approach. This approach can enhance data persistence. However, the load on the host node is too heavy and the replicas are not utilized sufficiently. The second method selects replica nodes using several extra hash functions. This approach not only increases data reliability, but also improves the utilization of a replica by evenly distributing the requests on a data object over all replica nodes. And reference [7] [8] both use this approach. The third method maintains a query tree for each node [9]. Queries will be routed along the path from the source node to the root of the tree (the destination node). It could pre-position replicas on the query path to meet the requests of query as soon as possible, that reduces the load of the root and gets a better load balance.

Erasure coding strategy is the strategy that divides the original data into m blocks firstly, and then encodes them into the n blocks ($n > m$), finally, places n blocks to the network independently. If we want to search the original data, just any m blocks can restore the original data. Related research and analysis [10] show that under the same data failure probability, compared with replication, erasure coding occupies less storage space and communication bandwidth, but requires more computation.

B. Physical location information

In P2P networks, network measurement methods optimize the routing delay and redundancy costs by calculating the physical distance between nodes. GNP [11] measures the network distance based on coordinates. It maps the network nodes to the geometric space (such as three-dimensional Euclidean space). GNP selects a group of nodes in fixed locations as landmark nodes, and calculates their coordinates. Then it calculates the coordinates of other nodes based on round-trip delay (RTT) between the node and landmark node. Finally, the physical distance of two nodes can be got by calculating their geometric distance. Vivaldi [12] algorithm also maps the network nodes to the geometry of space dispersedly. It assumes that the connection between each pair of nodes like a spring. The length of rested spring between nodes is the calculative distance by RTT. Then this algorithm optimizes the distance by adjusting the coordinates of nodes to achieve the stability of the spring system under hypothetical spring action, which also makes the nodes' coordinates optimized.

III. P2P ARCHITECTURE BASED ON PHYSICAL DISTANCE

The current structured P2P networks (such as Chord) do not take the physical delay into account in routing and redundant information placement leading to inefficient query. This paper proposes a novel P2P architecture based on physical distance. It connects the logic distance with physical distance, by constructing nodes' identifiers and routing queries using the physical location information. Fig.1 illustrates Chord and our architecture. Chord uses consistent hashing algorithm resulting in that logically proximal nodes may be far away in physical distance. While in our new architecture, proximal nodes in the overlay network will be proximal in the physical network. The redundant information is placed in the physical routing path, so the queries can be satisfied at nearer nodes and hot spots can be eliminated.

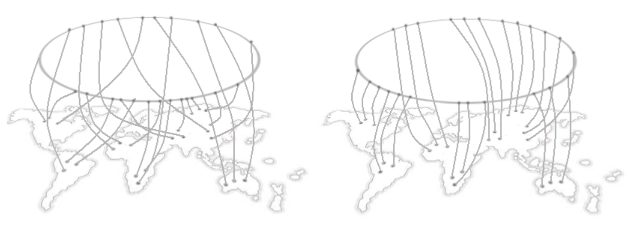


Figure 1: Comparison with Chord and New architecture

A. Dividing overlay network according to the physical distance

In the Chord structure, the node identifiers are generated by the consistent hash function, which maps nodes' properties (such as IPs) to the value space $[0, 2^m)$. The distribution of node identifiers is random and there is no relationship among them. In our architecture based on physical distance, the node identifiers describe the characteristics of the physical network. Fig.2 shows the tree established according to node's physical locations of the nodes. The descendant nodes of roots denote countries, ISPs, cities and IP addresses respectively. For example, the node www.nankai.edu.cn has IP address 202.113.16.33, and its physical location information is the China / CERNET / Tian Jin can be found in Fig.2.

In this paper, we use hierarchy "country-ISP-city-IP" to describe physical distance. This hierarchy may not accord with real physical distance very well. For example, it is possible that a node is closer to a node in another country than another node in the same country. However, our approach is independent of this hierarchy. Other methods depicting physical distance more accurately, such as network measure and clustering, can be easily introduced into our approach.

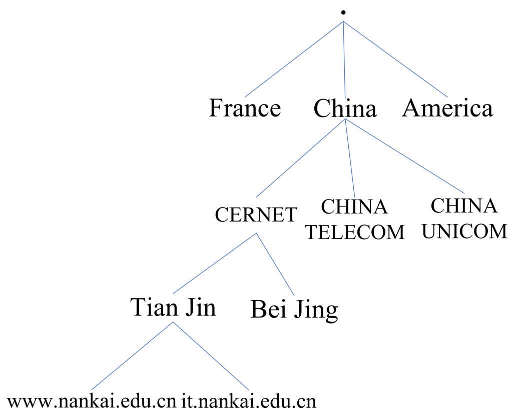


Figure 2: Tree network

B. Generating of node identifier

In the proposed P2P architecture, a node identifier is composed of four parts. Fig.3 shows how to generate a identifier, H_1, H_2, H_3, H_4 are four hash functions, used to generate hash values of country, ISP, city and IP address, N_1, N_2, N_3, N_4 respectively. We concatenate these hash values into node's identifier $N_1N_2N_3N_4$. For example, fig.2 shows that the country, ISP, city and IP addresses of the node www.nankai.edu.cn are China, CERNET, Tianjin and 202.113.16.33 respectively. If their hash values are 1010, 011010, 10011100 and 1110110011 severally, the node identifier is 1010011010100111001110110011. We define that:

$dist(A, B)$ is the physical distance of node A and B and $prefix(A, B)$ is the length of the longest common prefix of the identifiers of A and B. For example, if N_1 and N_5 are the hash values of two different countries, N_2 and N_6 are the hash values of two different ISPs, N_3, N_7 are the hash values of two different cities and N_4, N_8 are the hash values of different IP addresses, $prefix(N_1N_2N_3N_4, N_1N_2N_7N_8)$

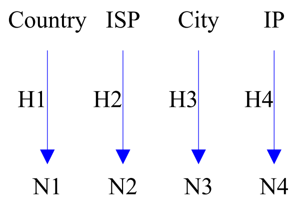


Figure 3: Generation of node identifier

is 2, while $prefix(N_1N_2N_3N_4, N_5N_6N_7N_8)$ is 0. So we can claim:

- The nodes in the same sub-network have the longest common identifier prefix. For example, www.nankai.edu.cn and it.nankai.edu.cn in Figure 2 have the common country, ISP and city parts in their identifiers.
- For any three nodes A, B and C, if $dist(A, B) < dist(A, C)$, $prefix(A, B) > prefix(A, C)$.
- For any three nodes A, B and C, if $prefix(A, B) < prefix(A, C)$, $dist(A, B) > dist(A, C)$.

From the three features described above we can conclude that nodes with more similar identifiers are nearer. One of important tasks in a Chord system is to maintain the connectivity of the entire ring network. Each node needs to send heartbeat packets periodically to subsequent node to detect whether it is online, and the heartbeat packets produce most of the network traffic in the system. Our new architecture reduces this overhead effectively.

To analyze the data packet delay, we make the assumptions: all the nodes are distributed evenly in m_1 countries, each country contains m_2 ISP, each ISP contains m_3 cities, and the distributions of nodes among ISPs and cities are also uniform; the number of nodes is N; the delays among the different countries, different ISPs in the same country, different cities belonging to the same ISP, and different nodes in the same cities are d_1, d_2, d_3, d_4 . In this paper, we suppose that $d_1 > d_2 > d_3 > d_4$. Because in P2P systems the heartbeat packets between logically adjacent nodes produce most of the network traffic, we analyze the average delay of heartbeat packets to evaluate the network overhead.

In a Chord system which physical distance is not taken into account, for two logically adjacent nodes, the probability of they in different countries is $\frac{m_1-1}{m_1}$; the probability of they in different ISPs of the same country is $\frac{1}{m_1} \cdot \frac{m_2-1}{m_2}$; the probability of they in different cities of same ISP is $\frac{1}{m_1} \cdot \frac{1}{m_2} \cdot \frac{m_3-1}{m_3}$; the probability of they in the same city is $\frac{1}{m_1} \cdot \frac{1}{m_2} \cdot \frac{1}{m_3}$. Therefore, the average delay of heartbeat packets is:

$$\begin{aligned} L_i &= \frac{m_1-1}{m_1} d_1 + \frac{1}{m_1} \cdot \frac{m_2-1}{m_2} \cdot d_2 \\ &+ \frac{1}{m_1} \cdot \frac{1}{m_2} \cdot \frac{m_3-1}{m_3} \cdot d_3 + \frac{1}{m_1} \cdot \frac{1}{m_2} \cdot \frac{1}{m_3} \cdot d_4 \\ &\approx d_1 \qquad (m_1 \gg 1) \end{aligned} \tag{1}$$

We can also conclude that in the Chord system which physical distance is not taken into account, the average delay among all the nodes is L_i in that the logical distance is irrelevant to the physical distance.

In the proposed architecture based on physical distance, the Chord ring is divided into m_1 arcs (sub-networks) denoting countries, each containing m_2 sub-arcs denoting ISPs. Each sub-arc consists of m_3 sub-sub-arcs denoting cities. Assume that, the nodes are uniformly distributed in the ring space. If there are enough nodes, the number of adjacent node pairs in different countries is m_1; the number of adjacent node pairs with different ISP is $m_1 \cdot (m_2 - 1)$; the number of adjacent node pairs in the different cities is $m_1 \cdot m_2 \cdot (m_3 - 1)$; the number of adjacent node pairs in the same city is $m_1 \cdot m_2 \cdot m_3 \cdot (\frac{N}{m_1 \cdot m_2 \cdot m_3} - 1)$. Therefore, the average delay of heartbeat packets is:

$$L_d = \begin{cases} \frac{m_1 \cdot d_1 + m_1 \cdot (m_2-1) \cdot d_2 + m_1 \cdot m_2 (\frac{N}{m_1 \cdot m_2} - 1) \cdot d_3}{N} \\ \qquad , m_1 \cdot m_2 < N \leq m_1 \cdot m_2 \cdot m_3 \\ \\ \frac{m_1 \cdot d_1 + m_1 \cdot (m_2-1) \cdot d_2 + m_1 \cdot m_2 \cdot (m_3-1) \cdot d_3}{N} + \\ \frac{m_1 \cdot m_2 \cdot m_3 (\frac{N}{m_1 \cdot m_2 \cdot m_3} - 1) \cdot d_4}{N} \\ \qquad , N > m_1 \cdot m_2 \cdot m_3 \end{cases} \tag{2}$$

The equation 2 can be simplified:

$$L_d \approx \begin{cases} \frac{m_1 \cdot m_2}{N} \cdot (d_2 - d_3) + d_3 \\ \qquad , m_1 \cdot m_2 < N \leq m_1 \cdot m_2 \cdot m_3 \\ \\ \frac{m_1 \cdot m_2 \cdot m_3}{N} \cdot (d_3 - d_4) + d_4 \\ \qquad , N > m_1 \cdot m_2 \cdot m_3 \end{cases} \tag{3}$$

From equation 1 we can see that if physical distance is not taken into account, the average packet delay is unrelated to the number of nodes N. However, we can find from equation 3 that in the architecture based on physical distance the data packet delay increases with increasing N. We also can find that $L_d < L_i$ by the simple proof. Therefore, the improved method could reduce the network overhead effectively.

Now we analysis query packet delay of Chord protocol and improved protocol. In the Chord routing algorithm [1], routing hops of query packet is smaller than $log_2 N$. We can assume that the average routing hops of query packet is $\frac{\log_2 N}{2}$. So the average delay of query packet is:

$$F_i \approx \frac{\log_2 N}{2} \cdot d_1 \tag{4}$$

In the architecture based on physical distance, the average routing hops of query packet among the different countries is $\frac{\log_2 m_1}{2}$, among the different ISPs is $\frac{\log_2 m_2}{2}$, among the different cities is $\frac{\log_2 m_3}{2}$ and in the same city is $\frac{\log_2 N - \log_2 m_1 - \log_2 m_2 - \log_2 m_3}{2}$. The average delay of query packet is:

$$F_d \approx \frac{\log_2 m_1}{2} \cdot d_1 + \frac{\log_2 m_2}{2} \cdot d_2 + \frac{\log_2 m_3}{2} \cdot d_3 + \frac{\log_2 N - \log_2 m_1 - \log_2 m_2 - \log_2 m_3}{2} \cdot d_4 \quad (5)$$

From equation 4 and 5 we can see that the average delays of query packet in the two architectures both increase as N increases; We obviously find that $F_d < F_i$ and the growth rate of F_d is smaller than F_i as N increases.

C. Query tree

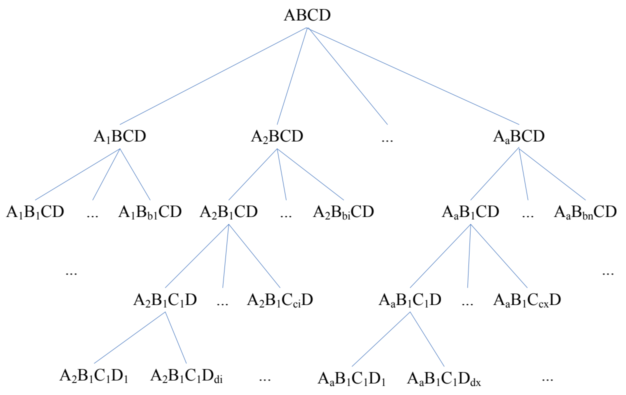


Figure 4: Query tree

In the proposed architecture, a special tree call *query tree* is used to help routing. A query tree has a height of 5 as fig.4 shows. Each node identifier includes four parts, which are hash values of country, ISP, city and IP address. For convenience, we use $c(X)$, $i(X)$, $t(X)$ and $p(X)$ to denote the four hash values of the node X respectively. For example, as shown in fig.4, if the identifier of root is $ABCD$, $c(ABCD) = A$, $i(ABCD) = B$, $t(ABCD) = C$, and $p(ABCD) = D$. A query tree satisfies:

- The root R and its child C satisfy: $c(R) \neq c(C)$, $i(R) = i(C)$, $t(R) = t(C)$ and $p(R) = p(C)$;
- Each second level node P and its child C satisfy: $c(P) = c(C)$, $i(P) \neq i(C)$, $t(P) = t(C)$ and $p(P) = p(C)$;
- Each third layer node P and its child C satisfy: $c(P) = c(C)$, $i(P) = i(C)$, $t(P) \neq t(C)$ and $p(P) = p(C)$;
- Each fourth layer node P and its child C satisfy: $c(P) = c(C)$, $i(P) = i(C)$, $t(P) = t(C)$ and $p(P) \neq p(C)$.

IV. Meta-data operation

This section will introduce meta-data related operations, including publishing, query and replication of the meta-data.

A. Meta-data publishing

The meta-data with identifier k, that designates the destination node, is published by a node. Afterward, the meta-data is delivered to the destination node by Chord routing protocol. If node k is on-line, then node k is the destination node; if node k is not on-line, then the successor of k will be the destination node. The actual destination node is called the *primary node* for the meta-data. Finally, the primary node receives the meta-data and then stores it locally.

B. Meta-data query

When node n receives the query request, the essential work is to identify the destination node k of the query request. Then find the parent node f of node n in the query tree with root k and forward the request to f. When f receives the request, it checks whether the requested meta-data is stored locally, that is, whether it is the primary node of the meta-data. If so, the result is sent to node n directly; otherwise, the algorithm continues to forward the request to f's parent node until reaching the root node. Fig.5 shows a simple example of query routing. A query is initiated from node 5672 to node 1234, so the root of the query tree is 1234. The query path is shown using dotted line.

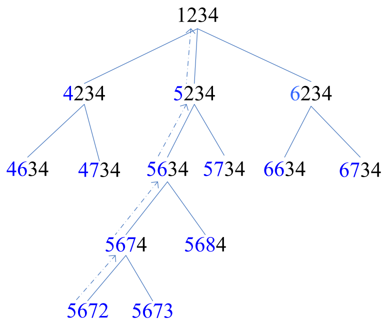
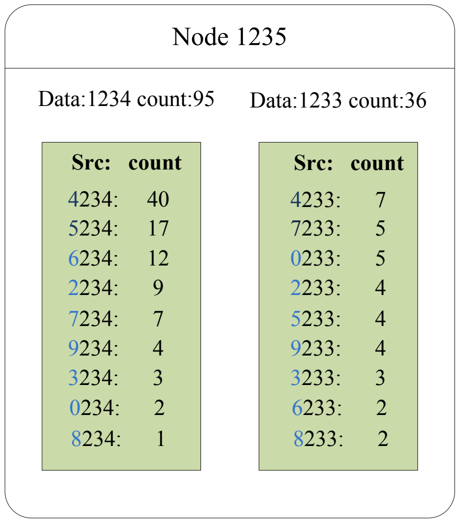


Figure 5: Routing using query tree

Query tree has an important property: for queries to the same destination node, source nodes in the same sub-network (physical area) will share a specific routing path. Fig.5 shows an examples, if the destination node is 1234, the query launched by the nodes from the sub-network 567* will be routed to node 5674; the query launched by the nodes from the sub-network 56** will be routed to node 5634; the query launched by the nodes from the sub-network 5*** will be routed to node 5234. This property is very useful for meta-data replication which will be introduced in the following parts.

C. Meta-data replication

the heterogeneity and load imbalance will overload node. Replication is a common method to solve this problem. In our proposed architecture, query tree can be used to improve the performance of replication mechanism. As mentioned above, nodes in the same sub-network will route queries to the same destination along the same path. This property can help replica placing. Every node maintains an access list for each meta-data to record its recent access information. Assume that node n is overloaded,it will select the meta-data k from access list that is accessed most frequently, and then arranges the replica for k in the sub-network that forwards most of accesses to k.



Node 1235

Data:1234 count:95

Src:	count
4234:	40
5234:	17
6234:	12
2234:	9
7234:	7
9234:	4
3234:	3
0234:	2
8234:	1

Data:1233 count:36

Src:	count
4233:	7
7233:	5
0233:	5
2233:	4
5233:	4
9233:	4
3233:	3
6233:	2
8233:	2

Figure 6: Access list of node 1235

Assume that the primary node of meta-data 1234 is 1235. Fig.6 shows node 1235's two access lists for meta-data 1233 and 1234 with recent visit counts 36 and 95 respectively. When the node 1235 is overloaded, the sub-network 4***, that forwards most of accesses to the meta-data 1234, will be chosen. And the replica of 1234 is placed at the successor of 4234, said R, which will become the primary node of 1234 in the sub-network 4***. So the primary node could meet the query from the sub-network 4***. Therefore the load on 1235 is reduced. R also maintains an access list for meta-data 1234. If the accesses to meta-data 1234 cause the overload of R, it will arrange a replica at the sub-network that most frequently accesses 1234.

The method of arranging replica in the "hot" sub-network can solve the problem of node overload effectively. Moreover, with the spread of replicas, the replica and the source node of the query become closer. This can reduce delay greatly.

In order to analyze the rate of successful queries and the access performance, we make assumptions just like section III-B and assume that the probability of data failure is p_s; in the query tree, the probability of root node creating replica is p_t. In the ordinary Chord architecture the rate of successful queries is:

$$S_i = 1 - p_s \tag{6}$$

In our proposed architecture using query tree, the probability of root node creating replica is p_t. Suppose the query source is uniformly distributed, then the number of queries received in the second layer of the query tree accounts for $\frac{1}{m_1}$ of the total, the number of queries received in the third layer of the query tree accounts for $\frac{1}{m_1 \cdot m_2}$ of the total. Therefore, the probability of creating replica in the second layer is $\frac{p_t}{m_1}$, the probability of creating replica in the third layer is $\frac{p_t}{m_1 \cdot m_2}$. The probability of existence of replica in the second level is p_t ; The probability of existence of replica in the third level is $p_t \cdot \frac{p_t}{m_1}$; The probability of existence of replica in the third level is $p_t \cdot \frac{p_t}{m_1} \cdot \frac{p_t}{m_1 \cdot m_2}$. Therefore, the rate of successful queries is:

$$\begin{aligned} S_d &= 1 - [(1-p_t) \cdot p_s + p_s \cdot (p_t \cdot p_s) \cdot (p_t \cdot \frac{p_t}{m_1} \cdot \\ &\quad p_s) + \ldots] \\ &\approx 1 - [(1-p_t) \cdot p_s + p_s \cdot (p_t \cdot p_s)] \\ &\approx (1-p_s) + p_t \cdot (p_s - p_s^2) \end{aligned} \tag{7}$$

From the comparison with equation 6 and 7, we can find that our proposed architecture improves the rate of successful queries by $p_t \cdot (p_s - p_s^2)$.

In the ordinary Chord architecture, the total query delay is:

$$D_i \approx d_1 \tag{8}$$

And in our proposed architecture, the total query delay is:

$$\begin{aligned} D_d &= (1-p_t) \cdot d_1 + p_t \cdot (1 - \frac{p_t}{m_1}) \cdot d_2 + \\ &\quad mp_t \cdot \frac{p_t}{m_1}(1 - \frac{p_t}{m_2}) \cdot d_3 + p_t \cdot \frac{p_t}{m_1} \cdot \\ &\quad \frac{p_t}{m_1 \cdot m_2} \cdot d_4 \\ &\approx (1-p_t) \cdot d_1 + p_t \cdot d_2 \\ &\approx d_1 - p_t \cdot (d_1 - d_2) \end{aligned} \tag{9}$$

From the comparison with equation 8 and 9, we can see our proposed architecture reduces the query delay by $p_t \cdot (d_1 - d_2)$.

V. Experiment

A. Experimental environment

In order to verify the advantage of the method based on physical distance, we carried out some simulation experiments. Simulator used in this paper is The Chord Simulator [13], which is designed based on event mechanism by MIT CSAIL (Computer Science and Artificial Intelligence Laboratory) in 2003. It can simulate the events of node joining, node leaving, adding resources and searching resources.

B. Simulation experiments and results analysis

Experiments in the input file are the events of node joining, node leaving, adding resources and searching resources. It is generated according to some parameters:

- *The number of nodes*. The nodes evenly distributed in 20 countries, while each country has 20 ISP and each ISP has 50 cities.
- *Number of resources*.
- *Number of Queries*.
- *Distribution of node joining and leaving*. The time interval of joining and leaving meets the Pareto distribution, which has two parameters shape and scale. Shape is related to the system's dynamics which is the dynamic changes of the system as nodes join and leave. The smaller value of shape is, the more frequent nodes join and leave, that leads to the higher dynamics of the system. The simulation experiments changed the dynamics of the system by changing the value of shape.
- *Query distribution*. Uniform or Zipf.

C. *Comparison of packet delays*

In P2P system based on physical distance, nodes identified by four parts, country, ISP, city and IP address. The nodes with short physical distance are close to each other in the logical locations. Therefore, the majority of communication is limited in the location where the nodes close to each other in the physical location. To verify the effect of reducing the packet delay in the improved system, we performed two groups of experiments to test node joining and leaving. The test time is 1000 seconds, the number of resources is 1000, the number of queries is 1000. The query distribution was uniform distribution. In the simulation, we assumed that the delay of nodes between different countries is 200ms; the delay of nodes between different ISP in the same country is 100ms; the delay of nodes between different cities with same ISP in the same country is 50ms; the delay of nodes with the same ISP in the same city is 10ms.

Figure 7 shows the results with the constant dynamics of the system. There are four curves, "chord-all" and "chord-query" severally represent the average delay of all data packets and query packets in the ordinary Chord system; "phys-all" and "phys-query" represent the average delay of all data packets and query packets in the improved Chord system based on physical distance. As can be seen from the results:

- There is no significant change of average delay of all data packets with two methods as the increment of nodes.
- The average delays of query packets with both methods increases as the increment of nodes.
- The average delay of all packets with proposed architecture is reduced by $55\% \sim 65\%$ compared with Chord.
- The average delay of query packets in the proposed architecture increases more slowly than Chord as the increment of nodes.

In the second experiment, with a constant number of nodes 2000, we change the dynamics of the system to test

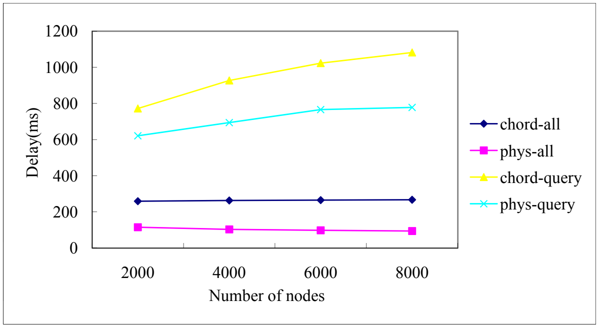


Figure 7: Average packet delay with the constant dynamic of the system

the packet delay. The results are shown in fig.8. As can be seen from the results:

- With the dynamics change of the system, there is no significant change of average delay of all data packets and query data packets with two methods.
- The average delay of all packets with proposed architecture is reduced by $50\% \sim 55\%$ compared with Chord.

As can be seen from the above two experiments, the number of nodes and dynamics of the system have limited impact on the average delay of all data packets. Meanwhile, the average delay of query data packets increases as the increment of nodes because routing hops are related to the number of nodes. The proposed architecture can effectively reduce the average packet delay and shorten the transmission distance of data packets.

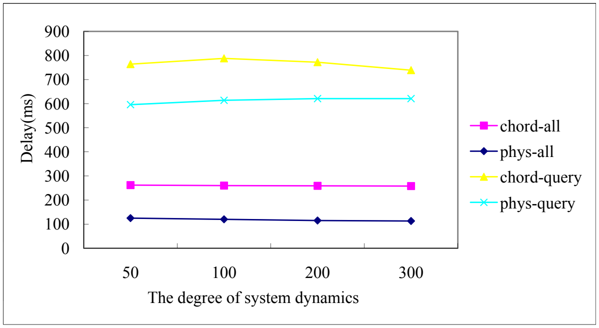


Figure 8: Average packet delay with the constant number of nodes

D. *Comparison of successful query rate*

In this paper, the proposed method divides the node ID into four parts. Simultaneously, queries are no longer sent to the destination node directly but to reach the destination node using the query tree, while the replicas of the query are placed to some appropriate areas. In order to test its rate of successful queries, this section has two comparative experiments. The test time is 1000 seconds, the number of resources is 1000 in the system, the number of query is 1000, and the number of nodes is 2000. The results of two experiments are described as followed:

In the first experiments, the resources queried are evenly distributed. With the constant number of nodes, we change the dynamics of the system to test the rate of successful queries. The experimental results are shown in fig.9. The results can be seen that there was no significant difference in rate of successful queries with two methods and the rates both increase as the reduction of the system's dynamics. That is because the resources queried are uniformly distributed leading to the root rarely spreads a replica.

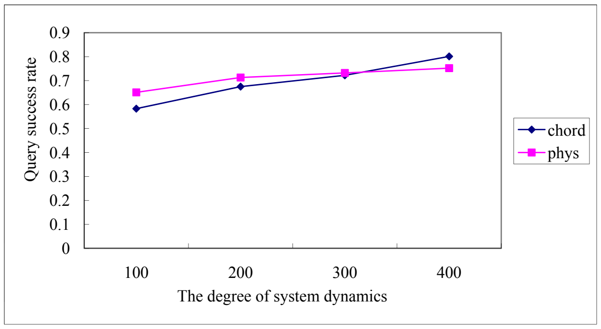


Figure 9: Rate of successful query with uniform distribution

In the second experiments, the distribution of resources queried is Zipf distribution. The experimental results are shown in fig.10. As can be seen from the experimental results, the rate of successful queries with two methods both increase as the reduction of the system's dynamics. Furthermore, the rate of query tree method is higher than that of Chord. Since query target meets Zipf distribution, hot spots exist in the system resulting in queries on the 20% resources accounting for 80% of all the queries, and the probability of the root spreading replica is 0.8. According to Formula 6 and 7, the rate of successful queries with query tree method increases about $0.8p_s(1-p_s)$. In fig.10, query tree method increases successful rate by $13\% \sim 23\%$. It has been shown that the experimental results are in good agreement with the theoretical analysis.

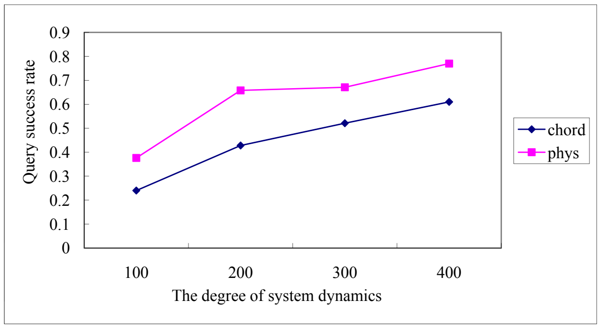


Figure 10: Success rate with Zipf distribution

E. Comparison of the performance of access requests

Using the query tree and replication can improve the performance of access requests. To verify the improvement, this section does comparative experiments. The evaluation metric is the physical delay between the initiated node and the query hit. In the experiments, the simulation time is 1000 seconds, the number of resources is 1000, the number of queries is 1000, and the number of nodes is 2000.

In the first experiment, the query target is uniformly distributed. With the constant number of nodes, we change the dynamics of the system to test the performance of access requests. The experimental results are shown in fig.11. We can see that, with the dynamics of the system reduced, the physical delay between the initiated node and the query hit node in both systems don't change significantly. The two systems perform close, however the proposed architecture is slightly better.

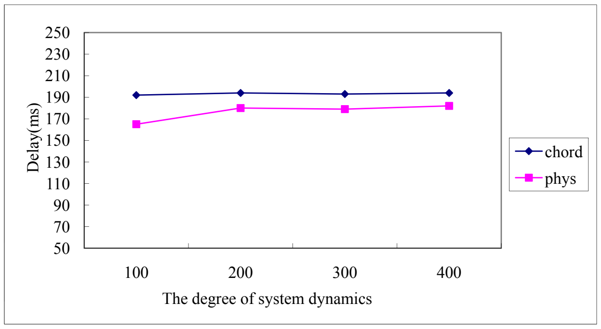


Figure 11: Delay with uniform distribution

In the second experiment, the query target meet the Zipf distribution. With the constant number of nodes, we change the dynamics of the system to test the performance of access requests. The experimental results are shown in fig.12. We can see that, with the degree of system dynamics reduced, the physical delay between the initiated node and the query hit node in both systems don't change significantly. The proposed architecture decreases delay remarkably. When the query target meets the Zipf distribution, there is hot spot issues and the queries of 20% of the resources occupied 80% of all queries, so the probability of distributing replicas is 0.8. As mentioned in section IV-C, compared with Chord, the proposed architecture theoretically reduces physical delay by $0.8*(200-100)$ (about 80ms). As can be seen from fig.12, when using query tree, the physical delay is reduced about $60 \sim 85$ ms, about $30\% \sim 45\%$, that is well agreed with the theoretical analysis.

VI. Conclusions

In the proposed P2P systems based on physical location, when using the query tree in accordance with the physical location, the queries from the same sub-network converge to the primary node of the sub-network. The simulation

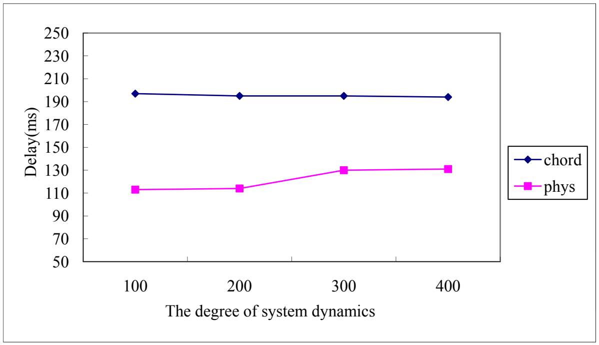


Figure 12: Delay with Zipf distribution

experiments shows that placing a replica on the sub-network root node makes the queries from the same sub-network acknowledged by the primary node in advance, which reduce the time required for data transmission. This will not only increase the reliability of data but also greatly improve the system performance of access requests.

In the following study, we will continue to improve our system. For example, to generate the node identifier we can use such as Huffman encoding which can solve the problem of namespace imbalance. Furthermore, besides the hierarchy "country-ISP-city-IP", we will use new methods, such as network measure and clustering, to construct distance hierarchy that describes the physical delay more precisely. Besides, we can use the method based physical distance on other P2P system to verify its effect.

Acknowledgment

This work was supported in part by the National High Technology Research and Development Program of China (2008AA01Z401), NSFC of China (60903028,61070014), SRFDP of China (20070055054), and Science and Technology Development Plan of Tianjin (08JCYBJC13000).

References

[1] I. Stoica, R. Morris, D. Karger, M. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," in *the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*. ACM, 2001, pp. 149–160.

[2] E. Cohen and S. Shenker, "Replication strategies in unstructured peer-to-peer networks," in *Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*. ACM, 2002, pp. 177–190.

[3] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker, "Search and replication in unstructured peer-to-peer networks," in *Proceedings of the 16th international conference on Supercomputing*. ACM, 2002, pp. 84–95.

[4] V. Gopalakrishnan, B. Silaghi, B. Bhattacharjee, and P. Keleher, "Adaptive replication in peer-to-peer systems," in *Distributed Computing Systems, 2004. Proceedings. 24th International Conference on*, 2004, pp. 360–369.

[5] A. Rowstron and P. Druschel, "Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility," vol. 35, no. 5. ACM, 2001, pp. 188–201.

[6] F. Dabek, M. Kaashoek, D. Karger, R. Morris, and I. Stoica, "Wide-area cooperative storage with CFS," in *Proceedings of the eighteenth ACM symposium on Operating systems principles*. ACM, 2001, pp. 202–215.

[7] B. Zhao, J. Kubiatowicz, and A. Joseph, "Tapestry: An infrastructure for fault-tolerant wide-area location and routing," *Computer*, vol. 74, pp. 11–20, 2001.

[8] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker, "A scalable content-addressable network," in *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*. ACM, 2001, pp. 161–172.

[9] J. C. Kuang-li Huang, Tai-yi Huang, "LessLog: A logless file replication algorithm for peer-to-peer distributed systems," in *Parallel and Distributed Processing Symposium*, 2004, p. 82.

[10] J. K. Hakim Weatherspoon, "Erasure coding vs. replication: A quantitative comparison," in *IPTPS*, 2002, pp. 328–337.

[11] T. Ng and H. Zhang, "Predicting Internet network distance with coordinates-based approaches," in *IEEE INFOCOM*, vol. 2, 2002, pp. 170–179.

[12] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: A decentralized network coordinate system," in *Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*. ACM, 2004, pp. 15–26.

[13] "The chord simulator," http://pdos.csail.mit.edu/chord/sim.html.