

Hier-DHT: Structured P2P-based Hierarchy Search Model for Time-shifted Streaming

JING-YANG YU *, XIAO-JIN REN **

*College of Computer and Information Engineering, Henan University, Kaifeng, Henan, China

**Network Information Center, Henan University, Kaifeng, Henan, China

yjy@henu.edu.cn, rxj@henu.edu.cn

Abstract— In the current, the structured P2P-based time-shifted streaming system exist the problems that the copy of time-shifted streaming cannot be fully found and the maintenance costs are higher. For resolving the problems a structured P2P-based hierarchy search model for time-shifted streaming, namely Hier-DHT, is proposed. In Hier-DHT, the hierarchy is divided according to the first video block in the cache of node. The nodes that cache the same first video block form a layer. Nodes are divided into layer node and layer-in node. All layer nodes form a ring. Layer node is brought by election of nodes in the layer. All register information is registered in layer node. Routing a lookup request is only on the ring formed by layer node. By the design, the video block cached can be fully found and the maintenance costs can be decreased effectively. Experimental results show that compared with other model, Hier-DHT has a better cache hit ratio, the better lookup performance and the lower maintenance overhead.

Keywords— time-shifted streaming, P2P, layer node, lookup, maintenance overhead

I. INTRODUCTION

Time-shifted streaming is a service that allows users to pause, fast forward, rewind, jump and other operations in the process watching live video. The requirement of network bandwidth, storage and other resource is higher in the time-shifted streaming system. The traditional time-shifted streaming service is based on client/server model. In the model the server is a bottleneck. There exist the problems of troubles of single failure and scalability.

In recent years, to resolve the problem, researcher begins to study the time-shifted streaming P2P-based[1][3]. They are self-scalable, lowering server load. But in those systems some problems still exist. P2TSS[1] effectively solves the time-shifting problem and reduces the pressure on the server-end. But P2TSS still has some problems. Firstly, because the underlying routing mechanism uses pseudo-DHT[2], which do not consider the real network delay information, the average

delay is larger. Secondly, P2TSS only cache the video blocks after the first video block. So all participating nodes do not cache the first video block they begin play. Thirdly, the copy of time-shifted streaming cannot be fully found. To resolve the problems, DRP-DHT[3] is proposed.

It improves the average lookup delay and resolve the problem of the first video block caching. But DRP-DHT also brings greater maintenance overhead, because bridge node must index information of all bridge nodes and all normal nodes in the layer. In addition, the problem that the copy of time-shifted streaming cannot be fully found, still do not be resolved.

To resolve those problems above, a structured P2P-based hierarchy search model for time-shifted streaming, namely Hier-DHT, is proposed. In Hier-DHT, the hierarchy is divided according to the first video block in the cache of node. The stronger capacity node is elected as the layer node. The layer node is responsible for all registration information and routing. All cached video block of node is registered. Hier-DHT improves the cache hit ratio, and has the better lookup performance and the lower maintenance overhead.

The structure of the paper is as follows. In section 2, we introduce Hier-DHT. Section 3 provides simulation results. Conclusions are given in section 4.

II. HIER-DHT

In Hier-DHT, node is divided into layer node and layer-in node. Layer node has the stronger capacity. Layer node is brought by election of layer-in nodes. Layer-in node is the normal node in the layer. Each node has two ID. They are layer ID and layer-in ID. All layer-in nodes form a Chord[4] ring. All layer nodes also form a Chord ring, denoted as layer node ring. When a node join the network, firstly selects a number between 0 and 2^{m-1} as its lay ID through hashing the index of the first video block it play, and find out the relevant layer node. Secondly, by the landmark [5] algorithm layer-in ID is achieved. The download speed of the video block can be improved by the landmark algorithm.

A. Hier-DHT Structure

In Hier-DHT, each node own two node IDs, denoted as N_{LI}^{II} . LI is the layer ID. II is the layer-in ID. Layer node is the stronger capacity node in the layer. All layer nodes form a Chord ring. All layer-in nodes that are in the same layer also form a Chord ring. There are two finger tables in the layer node. One is used to route information between layers. Another is used to route information in the layer. Layer-in node has only a finger table that is used to route information in the layer. Each layer-in node has a finger to point the layer node in the same layer. The finger table is formed according to Chord protocol. Fig. 1 shows an example of Hier-DHT that has h layer. N_1^{12} , N_1^{20} , N_1^{31} , N_1^{37} , N_1^{48} are the nodes in layer one. N_2^{10} , N_2^{32} , N_2^{48} , N_2^{53} are the nodes in layer two. N_h^{18} , N_h^{33} , N_h^{66} are the nodes in layer h . N_1^{20} is the layer node of layer one. N_2^{48} is the layer node of layer two. N_h^{66} is layer node of layer h . N_1^{37} is the standby layer node of layer one. N_2^{53} is the standby layer node of layer two. N_h^{18} is the standby layer node of layer h . Standby layer node is a copy of layer node. Fig. 1 also shows the finger information of layer node N_1^{37} and layer-in node N_1^{48} . Two finger tables of N_1^{37} are used to point to layer node and layer-in node in the same layer respectively. N_1^{48} has only a finger table that is used to point to node in the same layer.

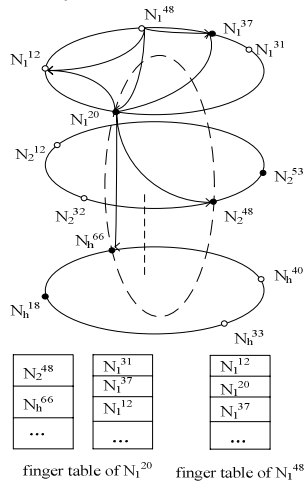


Figure 1. Hier-DHT structure

B. Joins and Departures

When a node n play a time-shifted streaming, firstly a hash value is achieved by hashing the index of the first video block in the time-shifted steaming. Then the node n sends a query that includes the value to the layer node in the same layer. Layer node begins to retrieve for the node n according to the hash value. If the relative layer node is found, the node n will join the layer and set the hash value as it's the layer ID. Then n computes its layer-in ID and joins in the layer by the join protocol of Chord. If the relative layer node could not be found, a new layer will be constructed, and n become the layer node of the new layer.

Leaves protocol is an improvment approach of the leave protocol of Chord. The difference is only when a layer node

leaves system, the standby layer node will turn into the layer node and notify relative layer node to update the finger table. The other is the same as the leave protocol of Chord.

C. Registration

To share the cached video block, the video block in the buffer must be registered. The video streaming is the continuous video block. Each video block has an index. In Hier-DHT, let video block index i denote a key. i is hashed using SHA-1, a hash function. In [1] and [3], node only register the index of the first video block they store in the buffer. In the situation, there exist the problem that some video block in the buffer could not be found. Therefore, in Hier-DHT, to get video block from buffer as much as possible, the node register all video block index in the buffer. All registration information is published on the ring that all layer node form. When a node begins to play a video block, firstly hash the video block index. Then the node runs register(key, value), where value includes its network address, layer address, layer-in address, registration time, cache size, etc. For sharing with other node in time, when node play the video block, the node cache the video block at the same time.

D. Retrieval

When a node n query a video block i , it sends retrieval(i , ii) to the layer node ring, where ii is the layer-in ID. The difference between retrieval(i , ii) and retrieval(i) of Chord is only the difference of the reference. All requests are sent to the layer node ring. When the relative layer node of key i is found, if the layer node has the registration information of the video block, the layer node return a registration information to n according to the shortest distance between ii and layer-in ID of registration node. Then n will contact relative node to download and play video block. Otherwise, the Null is returned to n . For example, in Fig. 1 N_h^{33} query video block 1. Firstly, N_h^{33} sends retrieval(1, 33) to the layer node N_h^{66} of layer h . After N_h^{66} receive the request, it will lookup key 1, and at last the layer node N_1^{20} will be found. N_1^{20} is the layer node of layer 1, and it has all registration information of video block 1. N_1^{20} return the registration information which the layer-in ID of registration node is the shortest distance with 33 to N_h^{33} . N_h^{33} contact the relative node to download and play video. If the layer 1 does not exist, N_h^{33} send the request to sever, and download the video block from server.

E. Layer Node

Layer node has significant resources and bandwidth in the layer. When a new node n joins into the system, if the layer is not constructed, a new layer is formed and n will become the layer node. Otherwise, the node joins into the layer as layer-in node. After a period of time, for example an hour, layer-in node n itself decides whether to apply to become layer node based on its capacity. If the node n has many idle resources, n will contact layer node to consult to become the layer node. The process will compare the resource situation between n and the layer node. If the resource situation of n is obviously better than the layer node, n will copy all finger tables of the layer node and become layer node. Former layer node turns into

layer-in node and add a finger that points to n . When other layer node contacts it, it will return the current layer node n to other layer node. Thus, other layer node will update its relative finger information. If the resource situation of n is not obviously better than the layer node, n will apply to become standby layer node, and contact standby layer node. The process is similar to apply to become layer node. If the apply is success, n become the standby layer node, the former standby layer node become layer-in node.

III. SIMULATIONS

We implemented Hier-DHT to perform simulations within an PlanetSim simulator[6]. The cache algorithm uses IPPI[3] that is the improvement of IPP[1]. IPPI caches all video block that it plays, and resolves the problems that IPP only caches the video blocks after the first video block.

Node arrived in the system at an arbitrary time, independent of each other. The video length L was 60000s and video block size Q was 10s. Video bit rate is 1Mb/s. 50% node request live streaming, 50% node request time-shifted streaming.

Two sets of simulations are carried out. In the first set of simulations, we evaluate the caching performance. We use the cache model which uses the proxy server, only as a baseline system(denoted as Proxy). Proxy model that has infinite cache size is the best caching model. We compare the caching performance of Hier-DHT, DRP-DHT, Pseudo-DHT with the baseline system. The size of client cache is scaled from 10M, 50M, 100M, 500M, 1000M, 5000M, 10000M and up to the infinite cache. The cache size of proxy server is set to infinite. Cache Hit Ratio is used as the metric to evaluate the caching performance. Cache Hit Ratio is the ratio between the number of requests that hit in the client cache or in the proxy cache and the total number of requests. The higher the cache hit ratio is, the better the caching performance is. The query to sever will decrease.

In the second set of simulations, we evaluate the lookup performance and system overhead. For simplicity, the size of client cache is equal. Each node has 500M cache. So each node can store fifty video blocks.

A. Cache Hit Ratio

In the set of simulations, the system caching performance is measured. 1000 nodes join in the system at an arbitrary time, independent of each other. Fig. 2 shows the cache hit ratio as a function of per-node cache size. The dotted horizontal lines represent the cache hit ratio that a proxy server with infinite cache size. From Fig. 2 we can see that, for all models, with the cache size increasing, the cache hit ratio also increase. The cache hit ratio of DRP-DHT is superior to Pseudo-DHT, and Hier-DHT is superior to DRP-DHT. This is because the node of Hier-DHT registers all cached video block. But the node of DRP-DHT and Pseudo-DHT only registers the first video block in the buffer. In addition, at the point of 500M, the cache hit ratio of Hier-DHT is close to that of proxy server. So, we conclude that Hier-DHT can utilize effectively the per-node caches and has very good cache performance that is comparable to a centralized web cache system.

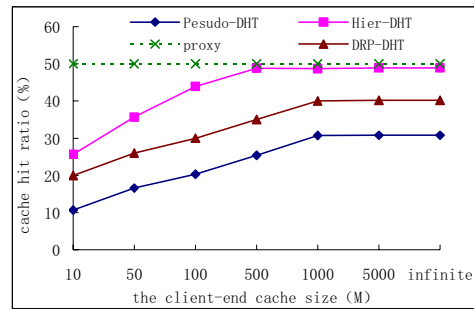


Figure 2. The cache hit ratio as a function of node cache size

B. Lookup Performance and Maintenance Overhead

In the set of simulations, we compare the lookup performance and maintenance overhead of Hier-DHT, DRP-DHT and Pseudo-DHT model. From the first set of simulations, we see that, at the point of 500M, the cache hit ratio of Hier-DHT is close to that of proxy model with infinite cache size. So, in the simulation the cache size of each node is set to 500M. The average lookup hops is used to evaluate the lookup performance. The average maintenance hops is used to evaluate system maintenance overhead. The average maintenance hops is the ratio between the total number of lookup hops that finger table of node construction and maintenance and the total number of node. This is because that nodes in DHT system need to maintain routing table. When a new node joins in system, the new node need build its finger table. When a node leaves system or fails, some relative node need update their relative fingers. No matter which happens, the lookup operation of building finger is necessary. So, the average maintenance hops is used to evaluate system maintenance overhead. To compare the maintenance overhead, we set 30% node fail at random, and join in the system at random again. 200, 300, 400, 500, 600, 700, 800, 900, 1000 join in the system at random respectively.

Fig. 3 shows the average lookup hops as a function of node. Fig. 4 shows the average maintenance hops as a function of node. From finger 3 we can see that DRP-DHT has the best lookup performance when the node number is less. With node number increasing, the lookup performance decrease. In addition, from Fig. 4 we can see that the average maintenance hops is maximum. This is because that with the node number increasing, update time of the finger table of the bridge node in DRP-DHT will be longer. This affects the lookup performance. In addition, the maintenance overhead of DRP-DHT is the linear growth. So the scalability is poor. From Fig. 3 and Fig. 4 we can also see that the lookup performance and maintenance of Hier-DHT is the better. This is because routing a lookup request on Hier-DHT is only on the ring formed by layer node, and the finger entry of layer node is also less. The lookup performance of Pseudo-DHT is the worse, and the maintenance overhead is between Hier-DHT and DRP-DHT.

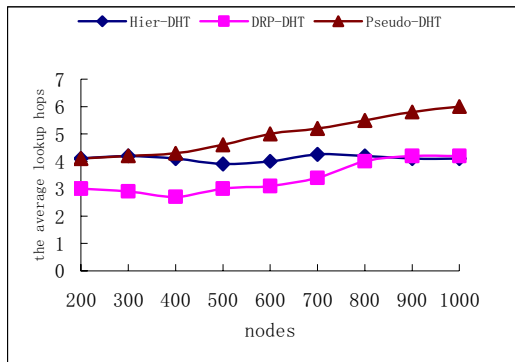


Figure 3. The average lookup hops

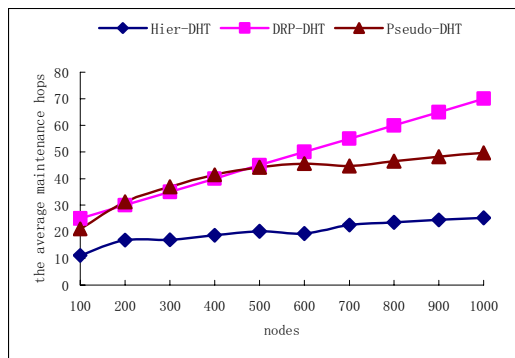


Figure 4. Maintenance overhead

IV. CONCLUSIONS

In the paper, a structured P2P-based hierarchy search model for time-shifted streaming, namely Hier-DHT, is proposed and evaluated. In Hier-DHT, the hierarchy is divided according to the first video block in the buffer of node. The nodes that cache the same first video block form a layer. Nodes are divided into layer nodes and layer-in node. All layer nodes form a ring. Layer node is stronger capacity node in the layer. All register information is registered in layer node. Routing a lookup request is only on the ring formed by lay node. Experimental results show that compared with other model, Hier-DHT has a better cache hit ratio, the better lookup performance and the lower maintenance overhead.

REFERENCES

- [1] S. Deshpande and J. Noh, "P2TSS: Time-Shifted and Live Streaming of Video in Peer-to-Peer Systems", *IEEE International Conference on Multimedia and Expo*, Hanover, Germany, June 2008.
- [2] S. Deshpande and J. Noh, "Pseudo-DHT: Distributed Search Algorithm for P2P Video Streaming", *Tenth IEEE International Symposium on Multimedia*. California, USA. 2008.
- [3] Xiuhong Hou, Ruipeng Ding, "DRP-DHT: Semi-distributed resource sharing model for P2P real-time streaming media", *2010 International Conference on E-Health Networking, Digital Ecosystems and Technologies (EDT)*. April, 2010.
- [4] I. Stoica, R. Morris, D. Liben-Nowell, D.R. Karger, M.F. Kaashoek, F. Dabek, H. Balakrishnan, "Chord: A scalable peer-to-peer lookup protocol for internet applications", *IEEE/ACM Transactions on Networking*, 2003, vol.11(1), 17-32.
- [5] Yunhao Liu, Xiaomei Liu, Li Xiao, Lionel M. Ni, "Location-Aware Topology Matching in P2P Systems", *In:Pro IEEE INFOCOM 2004*, HongKong,China,March 2004.
- [6] Planetsim. <http://projects-deim.urv.cat/trac/planetsim/>. 2011.