2019 11th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)

A P2P Network Framework for Interactive Streaming Media

Hua Yang, Mengdi Liu, Baochuan Li, Zhuqing Dong
School of Computer Science and Technology
Shenyang Aerospace University
Shenyang, China
yangh@sau.edu.cn, {2602489124, 768798356, 978879484}@qq.com

***Abstract*—To overcome the shortcomings of interactive streaming media in real-time transmission, control overhead, stability and scalability in general P2P network, we propose a hybrid P2P network framework called *PPISM* (*P2P Interactive Streaming Media*) for interactive streaming media. The framework adopts a hierarchical structure, which integrates CDN (Content Distribution Network), P2P network and the combine of tree-mesh structure. After building a mechanism of super-node selection and super-nodes tree construction, streaming media data are pushed to the super-nodes tree, which reduces the load of edge servers. Meanwhile, in the aspect of real-time streaming media transmission, the combination of push-pull is adopted. The edge server pushes streaming media data to the requesting node, and the requesting node pulls the missing streaming media data to the super-nodes tree, which improves the real-time transmission of data. The simulation shows that compared with ordinary P2P and simple CDN-P2P architecture, PPISM can significantly decrease the end-to-end delay, streaming media distortion and control overhead.**

Keywords-streaming media; CDN; P2P; tree-mesh

I. INTRODUCTION

In the P2P streaming media system, the P2P streaming media architecture is mainly divided into two categories: tree-based architecture and mesh-based architecture [1]. The results show that the former has higher stability in streaming media quality and has better advantages in terms of playback delay and end-to-end latency. However, in the dynamic churn scenario, more time is needed to compensate for the time consumed by the peer replacement and recovery mechanism. On the contrary, the latter has better adaptability in dynamic churn environments, and peers have more choices for providers of streaming media data and increase the scale of network growth. Therefore, the mesh-based structure has the potential to build large-scale coverage.

In mesh-based architecture [2][3], the node needs to contact at least one node in the overlay network to join the overlay network. In this architecture, streaming media transfers are in chunks, and each chunk consists of a specific portion of the streaming media. Each peer transmits streaming media data chunks to neighbor nodes that request these data chunks. However, the control overhead generated by periodic interactions from buffer mappings between peers also limits the ability of peer nodes in the mesh-based architecture.

Content Delivery Networks (CDN) was originally created to add a new layer of network architecture to the existing Internet. Distribute the content of the website to the edge of the network closest to the user, so that users can get the desired content nearby, reduce Internet network congestion, and improve the response speed and hit rate of users visiting the website. However, the CDN architecture requires a relatively higher cost, is relatively complex to implement, and has weak scalability.

Accordingly, we propose a hybrid system network framework for interactive streaming media that combines CDN and P2P networks. The key mechanism is that streaming will be first pushed by the data source server along the CDN trunk to the edge servers, reducing the load on the data source server. The P2P network uses a combination of tree and mesh. The super-node group stored by the three nodes closest to the streaming ID adopts the tree-based structure. The edge server near the tree pushes the streaming media data to the super-nodes tree, thereby improving the stability and real-time performance of the data transmission. A mesh structure is adopted for ordinary nodes to improve system robustness.

II. OVERVIEW OF PPISM

Based on the Kademlia protocol [4], we design a new P2P network framework PPISM for interactive streaming media. The P2P network architecture design of interactive streaming media is shown in Fig. 1.

The framework consists of three basic components: the Tracker server, the data source servers, and the peer nodes. The peer nodes are classified into super-nodes and normal nodes according to their performance. The function of each component is described as follows:

Tracker server: it is responsible for accepting the join node's network frame request, providing the new peers to the network to provide the boot node in the network, and recording the keyword and ID of the streaming media.

Data source server: it stores the original streaming media files, divides streaming media files into equal-length data chunks, and participate in the streaming media interaction process by using the streaming media data in a push manner.

Peer node: each maintains two routing tables locally, which are the priority transmission routing table and the neighbor routing table. In the process of streaming media transmission, the nodes with strong service capability will be used as the super-nodes, and the information of the super-nodes will be placed in the priority transmission routing table. While the routing table is full, the other nodes will be treated as normal nodes, and the normal nodes will be placed in the neighbor routing table.

978-1-7281-1859-8/19/$31.00 ©2019 IEEE
DOI 10.1109/IHMSC.2019.10162

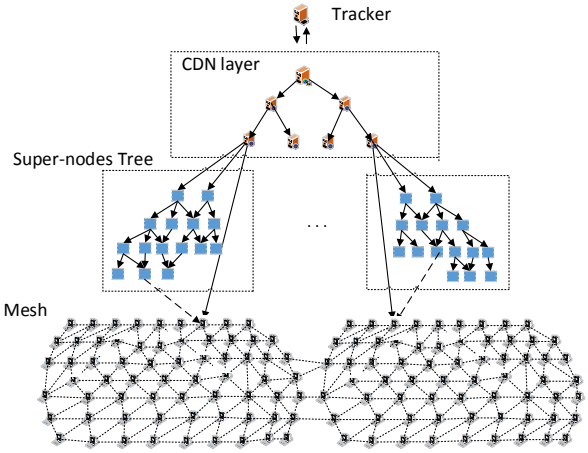

Figure 1. Network architecture

A. Structure of the CDN layer

The structure of the CDN layer is a tree structure. The streaming media data is pushed from the data source server to each child node layer by layer, and finally reaches the edge servers, and the edge servers directly participate in communications with the clients. In the event of an edge server failure, the immediate parent node directly serves the clients. Adopting such a structure can improve the efficiency of data distribution and reduce the server load. Each edge CDN server has two identities: streaming media distribution server and P2P Tracker server. The edge CDN server not only needs to retain certain streaming service capabilities but also maintains a list of customers registered to the streaming multicast group.

B. Structure of the Super-Nodes Layer

Each streaming media has a unique ID. The Tracker server locates the publisher on the edge server responsible for it. The edge server first finds the three nodes closest to the streaming media ID in the region, and then return the super-nodes group information of the three nodes, and build the super-nodes tree with the edge server as the root node. The super-nodes layer structure is shown in Fig. 2(a). Blue squares represent super-nodes, and blank squares represent normal nodes of the mesh layer. Streaming is pushed from the edge server to all super-nodes. When a mesh layer node requests streaming, it searches for the node closest to itself in the super-nodes layer as the source data provider for streaming interaction.

In order to avoid the dynamic joining and exiting of nodes, the data transmission of the entire trunk is greatly affected. Each super-node creates a new contact list locally. The list stores the information of the neighbor nodes in the trunk and its specialized backbone parent node information. When the nodes in the mesh layer leave, it will not affect the data transmission in the trunk. When the node in the trunk leaves, the node information with the strongest capability in the mesh layer's neighbor node routing table is added to the priority transmission routing table, and the appropriate location is found in the trunk for insertion. In the meantime, the node's children will retrieve the lost data from its neighbors.

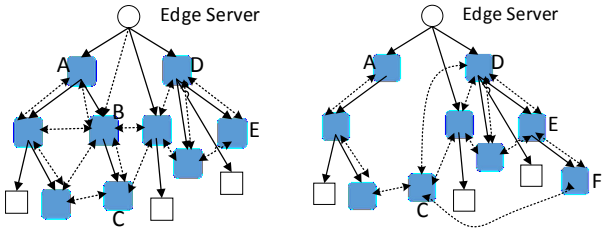

Figure 2. Data transmission of the super-nodes Tree

As shown in Fig. 2(b), when Node *B* exits, Node *C* finds Node *D* through the neighbor route. The node *D* transfers the lost streaming media data to node *C*, and node *F* takes the node *E* as its parent node. Node *F* joins the super-nodes tree and establishes contact with neighboring nodes.

C. Structure of the mesh layer

All nodes except the super-nodes are at this layer. Forming nodes into a structured P2P network through Kademlia routing, and XOR as the standard for measuring node distance. Each node maintains two local neighbor routing tables, one is a priority transmission routing table for storing super-nodes information, and the other is a neighbor routing table for storing normal nodes information. The nodes of the Mesh layer may be in three states: a request state, a forwarding state, and an empty state.

III. DESIGN OF PPISM

A. Routing design

In this framework, when each node joins the network, the system will automatically generate a 128-bit node ID as the unique identifier of the node in the system. In order to make the neighbor nodes closer to each other in physical topology, we add and replace the 32-bit IP address of the node to the first 32-bits of the node ID, and still maintain the node ID at 128-bits. The node ID format is shown in Fig. 3.

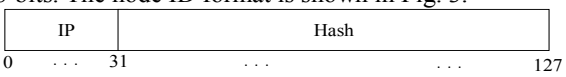

Figure 3. Format of node ID

The distance between nodes is calculated by XOR, as shown in (1). The advantage of this distance calculation method is that all query manipulations for an ID will finally gather in one path. This speeds up the query.

$$\text{Distance}(x, y) = x \oplus y \tag{1}$$

Besides, in order to improve the real-time performance of interactive streaming media data transmission, we introduce the concept of the preferential transmission routing table. In addition to neighbor routing, a new priority routing table is also built locally for each node, due to interactive streaming media to media data. The real-time requirement of the transmission is very high. In order to ensure the efficiency of the query and not waste the bandwidth resources of the local

node, the upper limit of the number of nodes in the priority transmission routing table is 10.

In the Kademlia network, every peer maintains 128 contact lists locally. A K-bucket is a contact list [5], as shown in Fig. 4, the last location of k-bucket stores the latest interactive node information, while the starting location stores the node information that was interacted long ago.

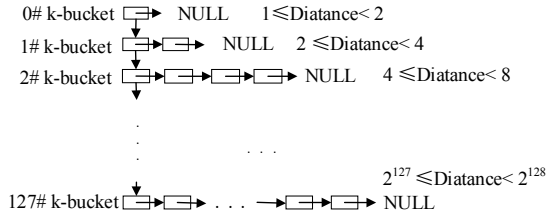

Figure 4. K-bucket structure

B. Super-nodes selection

The choice of super-nodes in this paper mainly considers two aspects: node capability and node online duration.

TABLE I. PARAMETER DEFINITION

parameter	Significance
UB_i	Upload bandwidth of node *i*
T_i	On-line time of node *i*
D_i^j	Relative distance between requester node *j* and source node *i*
S(i,t)	Packet loss rate of sender node *i* at time t
CF(i,t)	The contribution degree of node *i* at time t
C_i	Ability of node *i*
d_i^j	Whether the source node P_j correctly responded to the request of node P_i (0: false, 1: true)
N	Set of nodes that send data block requests to source nodes
R	Data block collection requested by requester node to source node
B	Uplink bandwidth set of each node
P_i	The amount of data uploaded by node *i* during the scheduling cycle

The node capability value is generally defined as a quantized value formed by integrating the evaluation factors such as computing power, storage capacity, bandwidth and online duration of the node *i*, which are referred to as a capability value of the node. Based on the above factors, we give the following formula for calculating the node's ability value:

$$C_i=\frac{T_i^2*UB_i*CF(i,t)}{S(i,t)*D_i^j+1} \tag{2}$$

$$CF(i,t)=(\sum_{i\in N}\sum_{j\in R} d_i^j)*(\frac{P_i}{UB_i}+\frac{UB_i}{\sum B}) \tag{3}$$

The definition of each parameter is shown in Table 1.

Since the super-nodes are the hubs for streaming interaction, the nodes we choose not only require high performance, but also require strong node stability. The longer the online time of the node, the more stable it is, and the more opportunities it provides for the service. The proportion of the online duration is set to be larger when calculating the node capability value, which is T_i^2 in the formula. The farther the relative distance D_i^j of the node is, the more the transmission delay increases and the service capability of the node decreases. The higher the packet loss rate of the node, the longer the waiting time of the requester node. The resending the request message will increase the load of the node and reduce the service capability of the node. The higher the contribution of the node, the greater the chance that other nodes will provide services, and the more powerful the node.

In this paper, the size of the priority transmission routing table is set to 10. The node sorts the node capability of all its neighbor nodes, and preferentially transmits the 10 neighbor nodes with the highest node capacity in the routing table as the super-node.

C. Creation of Streaming Media Multicast Tree Group

When a node publishes a streaming media, it generates a random streaming ID, and the node routes a publish message. The message uses the streaming ID as a key. Kademlia will route this message to the super-nodes in the priority transmission routing table of the 3 nodes closest to the XOR distance of the streaming ID. These super-nodes and edge servers together form a super-nodes tree that pushes streaming media data with the load of the edge server as the root node.

The creation of Streaming Media Multicast Group 1100 is shown in Fig. 5. The node 1001 publishes a streaming media ID of 1100, and uses Kademlia to route the three nodes closest to the distance 1100, which are 1100, 1110, and 1101 respectively. The edge server obtains the super-nodes information from the three nodes to form a super-nodes tree.

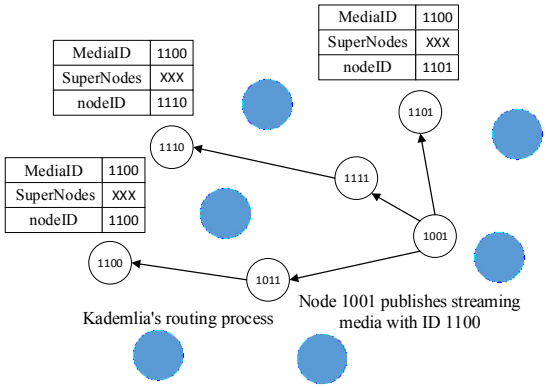

Figure 5. Creation of Streaming Media Multicast Group

D. Distribution of media resources

In PPISM, the media resource distribution flowchart is shown in Fig. 6, where: The node *A* first stores the name and ID of the media resource to the Tracker server. Streaming media data is encoded by a streaming media encoder and sent to the data source server. Data source server pushes streaming media data from top to bottom to edge server layer by layer. When the node *H* searches for the streaming media resource, the tracker server returns the URL information of the streaming media according to the keyword of the query. The

tracker server determines the jurisdiction of the edge server to which the node *H* belongs according to the URL redirection of the CDN. The edge server returns to the node *H* the super-nodes group information where the streaming media is located. Node *H* calculates the hash distance of ID between itself and super-node, i.e., ID_H and ID_{SN}. Taking a node requesting a piece of streaming media data as an example, if the load of the edge server satisfies the requirement of serving node *H*, the edge server pushes streaming media data to node *H*. Otherwise, node *H* pulls the required streaming media data from the closest super-node.

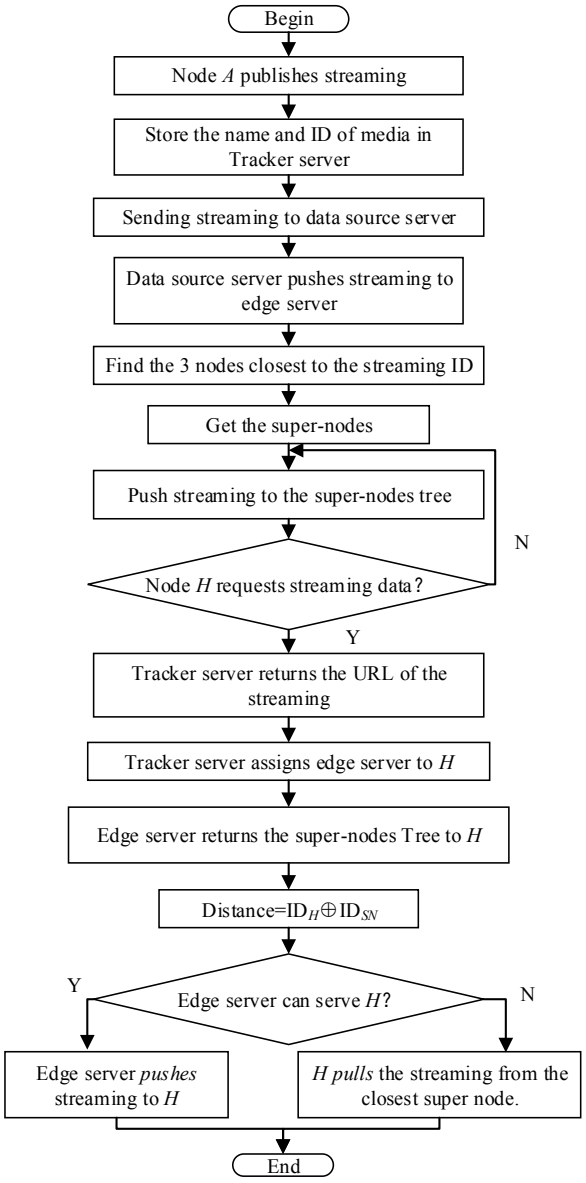

Figure 6. Media Resource Distribution Flow Chart

IV. Experimental Verification

We conducted an experimental simulation in OverSim [6], and tested the performance of the system, including start-up delay, end-to-end delay, streaming media distortion rate and control overhead. Three different architectures are compared, with preset parameters shown in Table 2.

- P2P Mesh-Based: using the common P2P architecture of the Central Tracker.
- CDN-P2P (Simple Mesh): based on mesh hybrid simple P2P-CDN architecture.
- PPISM: the CDN-P2P hybrid architecture we designed.

TABLE II. Experimental Related Parameter

Parameters	Value
Video Codec	MPEG 4
Video Frame Rate	25 frames/sec
Average Video Bit Rate	512 Kbps
Number of frames in GoP	12 frames
Number of nodes	250
No. of CDN Services	5
Video Chunk Size	1 frames
InitPhase Creation Interval	1s
Measurement Time	500s
Transition Time	100s
Lookup Redundant Nodes	16
Lookup Parallel Paths	1
Lookup Parallel RPCs	1

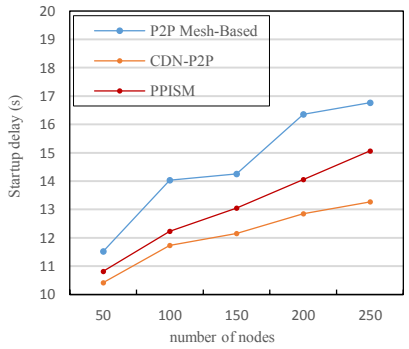

Figure 7. Start-up delay

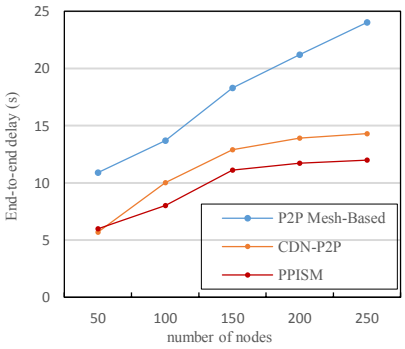

Figure 8. End-to-end delay

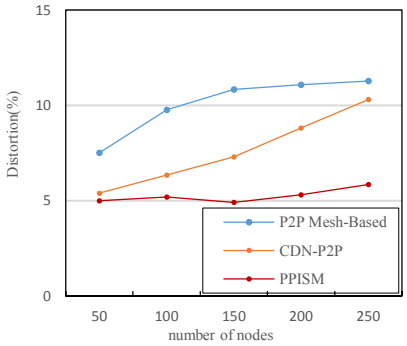

Figure 9. Video distortion ratio

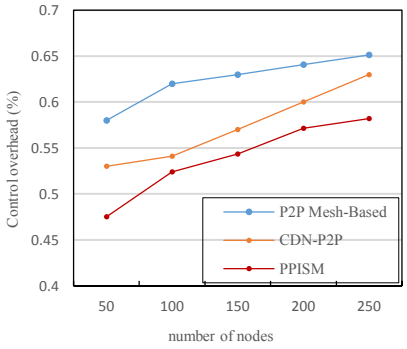

Figure 10. Control overhead

Fig. 7 shows a comparison of start-up delays of these three architectures. With the nodes adding in the network, the time from the user sending the request to the system response playback will gradually increase. The startup delay of PPISM is much smaller than that of the P2P Mesh-Based, but a little higher than CDN-P2P (Simple Mesh), because PPISM needs time to build a super-nodes tree structure.

Fig. 8 shows a comparison of the end-to-end delay times measured for the three architectures. In the end-to-end delay comparison, with the nodes joining the network, the end-to-end delay of PPISM is generally much smaller than the end-to-end delay of the above two architectures, indicating that PPISM is effective in content transmission.

Fig. 9 shows a comparison of the video distortion rate. In the comparison of distortion rate, PPISM is much lower than that of P2P Mesh-Based and CDN-P2P (Simple Mesh). It shows that PPISM can ensure that peers will receive more video block data correctly and improve the quality of video playback.

Fig. 10 shows a comparison of control overhead on the upload bandwidth. PPISM consumes much less bandwidth than P2P Mesh-Based and CDN-P2P (Simple Mesh), indicating that PPISM achieves a lower control overhead.

V. CONCLUSION

We propose a hybrid P2P network framework suitable for interactive streaming media. In this framework, we introduce a super-nodes selection mechanism, construct a super-nodes tree with edge server as its root, use push-pull method to minimize transmission delay, and realize seamless transmission under tree-mesh cooperation in data transmission. Simulation experiments show that the architecture designed in this paper has the lowest end-to-end delay, minimum streaming distortion rate, and the least upload bandwidth consumption. In the future, the selection of super-nodes and network interaction efficiency should be further optimized.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their suggestions. Part of this work is supported by Science and Technology Foundation from the Education Department of Liaoning Province (L201626), in China.

REFERENCES

[1] C. Y. Goh, H. S. Yeo, H. Lim, P. K. Hoong, J. W. Y. Lim, and I. K. T. Tan. "A comparative study of tree-based and mesh-based overlay p2p media streaming," International Journal of Multimedia and Ubiquitous Engineering, vol. 8, Jan. 2013, pp. 97-106.

[2] X. Zhang, J. Liu, B. Li, and T. -S. P. Yum, "CoolStreaming/DONet: a data-driven overlay network for peer-to-peer live media streaming," Proc. of 24th Annual Joint Conference of the IEEE Computer and Communications Societies, Miami, FL, vol. 3, Mar. 2005, pp. 2102-2111, doi: 10.1109/INFCOM.2005.1498486.

[3] A. Vlavianos, M. Iliofotou, and M. Faloutsos, "BiToS: Enhancing BitTorrent for Supporting Streaming Applications," Proc. of 25th IEEE Int'l Conf. on Computer Communications, Barcelona, Apr. 2006, pp. 1-6, doi: 10.1109/INFOCOM.2006.43.

[4] P. Maymounkov and D. Mazieres, "Kademlia: A peer-to-peer information system based on the XOR metric," Proc. of Revised Papers from the First International Workshop on Peer-to-peer Systems, vol. 2429, Oct. 2002, pp.53-65, doi: 10.1007/3-540- 45748-8_5.

[5] Z. -J. Han, R. -C. Wang, and Y. Wang, "PPmcast: a novel P2P-based application-level multicast using Kamulia," The journal of China Universities of Posts and Telecommunications, vol. 16, Sept. 2009, pp. 114-119, doi: 10.1016/S1005-8885(08)60345-3.

[6] I. Baumgart, B. Heep, and S. Krause, "OverSim: A scalable and flexible overlay framework for simulation and real network applications," Proc. of 2009 IEEE Ninth Int'l Conf. on Peer-to-Peer Computing, Seattle, WA, Sept. 2009, pp. 87-88, doi: 10.1109/P2P.2009.5284505.

[7] I. Vari, "INET Framework," 2012, http://inet.omnetpp.org.

[8] H. B. H. Abdallah and W. Louati, "Ftree-CDN: Hybrid CDN and P2P Architecture for Efficient Content Distribution," Proc. of 27th Euromicro Int'l Conf. on Parallel, Distributed and Network-Based Processing (PDP), Pavia, Italy, Feb. 2019, pp. 438-445, doi: 10.1109/EMPDP.2019.8671549.

[9] A. Ghanbari, H. R. Rabiee, M. Khansari and M. Salehi, "PPM - A Hybrid Push-Pull Mesh-Based Peer-to-Peer Live Video Streaming Protocol," Proc. of Int'l Conf. on Computer Communications and Networks (ICCCN), Munich, Aug. 2012, pp. 1-8, doi: 10.1109/ICCCN.2012.6289299.