Peer Clustering System for Different Start Video Broadcasting

Hathairat Ketmaneechairat
King Mongkut's University of Technology North Bangkok
Bangkok, Thailand
e-mail: hathairatk@kmutnb.ac.th

Phoemphun Oothongsap
King Mongkut's University of Technology North Bangkok
Bangkok, Thailand
e-mail: phoempn@kmutnb.ac.th

Anirach Mingkhwan
King Mongkut's University of Technology North Bangkok
Bangkok, Thailand
e-mail: anirach@ieee.org

Abstract—**The different start video broadcasting is a new approach to manage video streaming applications. This approach allows unpunctual users to view broadcast programs from the beginning. This paper proposes a cluster model for different start video broadcasting. The model is composed of five processes: peer joining, super node selection, backup-node selection, download paths and leaving node process. The model is simulated and verified by NS-2. The results show that (i) the video server load is diminished (ii) the tracker load is reduced due to the existing of super nodes or clusters, (iii) the play out delay is only to 10 seconds and (iv) the bandwidth utilization is improved as the consequence of the reduced number of control messages. Moreover, the authors have compared the performance of the cluster model with the one of the non-cluster model.**

Keywords-Peer-to-Peer (P2P); live video streaming; video on demand (VoD); different start video broadcating; cluster;

I. INTRODUCTION

Peer-to-Peer applications (P2P) have attracted a large number of Internet users. P2P technologies offer obvious advantages over content delivery network or content distribution network (CDN). P2P technologies improve system scalability with low implementation cost. P2P content delivery is an important technique for commercial systems such as IPTV. There are a lot of popular P2P file-sharing systems that support downloading such as Gnutella [1], Napster [2], Kazaa [3], BitTorrent [4], and eDonkey [5]. The main area of usage is P2P-based file sharing systems, like BitTorrent. Unlike traditional client-server architectures, peers in networks act as clients (leeches) and servers (seeds). A peer is not only downloads file from the network, but also uploads the downloaded file to other users in the network. Parts of the files are exchanged over direct connections between the peers. To enhance the system scalability and reduce the cost, several P2P video streaming applications have been published to use P2P technologies for the streaming of video and audio content. P2P technologies are provided content distribution service for both live video streaming and video-on-demand (VoD). PPLive [6], CoolStreaming [7], UUSee [8], Sopcast [9] and PPStream [10] are demonstrated by the huge popularity of P2P video streaming applications.

These works [6][7][8][9[10] cause unpleasant problems. The first problem is that a far away connection increases network traffic and thus decreases network resource utilization. The second problem is a heavy tracker load. These problems can be eliminated by using a hierarchical architecture as explained in [17]. In [15][16][17][18], the cluster concepts for P2P systems are introduced.

For the live video streaming, a live video content is disseminated to all users in real-time. Hence, all users in the system can watch the same part of the video stream at the same time. If users join the program later on, they will miss the beginning of the stream. For the video-on-demand, the users can watch the video stream anywhere at any time. Multiple users may watch the same movie at the different playback times. Besides these two separated categories, there is a different start video broadcasting [11][19][20] that allows unpunctual viewers to view the stream from the beginning during server broadcast time. By mixing a Peer-to-Peer download concept with a live broadcasting one, a new node can find users who have the needed parts of the stream, and use them as sources for download.

For example, suppose that there is a game of FIFA World Cup which starts at 3 PM and the game will end at 5 PM. A big amount of viewers will connect to the network and select the channel of the game. When the game starts at 3 PM, the viewers can load and view the game in real-time. After the game has started for 15 minutes, a new viewer decides to join the stream. The new viewer will have two choices: (i) view the game as the server broadcasts or (ii) view the game from the beginning. With the first choice, this broadcast will feature live video streaming while the second choice will employ both live streaming and video-on-demand features.

In this paper, a peer clustering system for different start video broadcasting is proposed. This model is created by five steps: peer joining, super node selection, backup-node selection, download paths and leaving node. There are three peer types: super node (SN), backup-node (BN) and normal node (NN). The buffer is used to store existing chunks of each peer in order to offer them to others. This is shown in section III.

The cluster model will be compared with non-cluster model for different start video broadcasting [11][19][20] to check the improved performance of the overall system. The cluster model for different start video broadcasting can be applied for live video streaming and video on demand. In contrast, the cluster model for live video streaming and video on demand will not comply with this approach because the unpunctual viewer has no chance to view the first chunk.

The remainder of this paper is organized as follows: Section II describes related work, Section III explains the system architecture, cluster-based system design and pseudo-code of cluster creation. A comparison of cluster and non-cluster

 ISBN: 978-1-61208-127-4

model system is proposed in Section IV. Section V indicates experimental results. Section VI presents the conclusion and future work.

II. Related Work

This section will briefly explain P2P streaming applications development [6][7][8][12][13][14] and P2P content delivery network (CDN) [15][16][17][18].

Coolstreaming or DOnet [7] is a P2P live video streaming application for only one channel. Every node in the network can be a video-source which produces the content for neighbor nodes. Coolstreaming does not use any tree, mesh or any structure. Additional infrastructure and implementations are necessary to provide more channels via Coolstreaming. Furthermore, there is only one node act as the origin node that stores all of video segments. The departure of an origin node can cause a single point of failure.

Sopcast, UUSee and PPLive are channel-based systems, which provide a lot of different video streams on different channels. So each of the application networks need at least one media encoding server, where the video streams are created and stored, and a well known channel server where the clients can get information about available programs [6][8][12][13][14].

Sopcast [12] has a set of root servers, which maintains the information what peer is available for what channel. Sometimes also peer lists are exchanged between the peers. The most important difference of Sopcast is the usage of UDP as transport protocol [13]. This leads to fast packet transmission but also causes a lot of overhead for control. The usage of an external media player and a second buffer are very inefficient and lead to a huge start-up delay.

UUSee provides the videos by several dedicated streaming servers, so that there is no single point of failure and the video streaming quality especially the playback continuity is improved. The TCP protocol is used to communicate with all peers, exchange the buffer map, measure the round trip time (RTT) and estimate the throughput [8]. If a huge number of peers try to join the same channel in the network at short time duration, a noticeable influence on the network performance has been recognized.

PPLive [14] uses different methods to exchange information about the availability of channels or movies, chunks and pieces. A distributed hash table (DHT) is used to assign dedicated movies to dedicated trackers and to achieve a load balancing [6]. On the other side, PPLive tries to improve its playback quality at the expensive of the network architecture. A locality mechanism, which prefers physically near peers (e.g., of the same ISP) is implemented, but peers with high bandwidth are preferred. This may lead to a bad network performance also for other participants.

Most of these works [6][7][8][12][13][14] have drawbacks related with low bandwidth utilization, high delay and a single point of failure. Thus to improve the performance of content distribution, the peers can be grouped in clusters. Many peers clustering approaches are proposed as the following:

The hierarchical architecture to group peers into clusters called CBT is proposed in [15]. The CBT has two novel algorithms: a peer joining algorithm and a super-peer selection algorithm. The proximity measurements of the RTT value and the TTL value between a pair of peer and super-peer are used. The CBT system improves the performance and scalability, and can be used to build a large-scale BitTorrent-like P2P overlay network.

A novel super node overlay based on information exchange called SOBIE is proposed in [16]. The main contributions are to select the super nodes by considering the aggregation of not only the delay and distance, but also the information exchange frequency, exchange time and query similarity. The SOBIE is guarantees the matching between the physical network and logical network. Moreover, the SOBIE has small-world characteristic to improve the efficiency and robustness.

The super node selection problem for Peer-to-Peer applications is presented in [17]. Three super nodes selection protocols for overlay P2P networks are proposed: SOLE, PoP-Corn and H2O. An integrated approach to the super node selection problem built on strong graph theoretic foundations and guided by realistic applications, can benefit the Peer-to-Peer community through cross-fertilization of ideas and sharing of protocols.

An effective real-time Peer-to-Peer streaming system for the mobile environment is proposed in [18]. The peers are grouped into clusters according to their proximity using RTT values between peers as criteria for the cluster selection. The cluster leaders are using to help a service discovery server. The partial streams help to utilizing the upload capacity with finer granularity than just per one original stream. This is beneficial in mobile environments where bandwidth is scarce.

The non-cluster model for different start video broadcasting is proposed in [11][19][20]. The users can join the real time video stream at different times and view from the beginning or the same part as the server. The proposed model is based on an application layer MESH network. The advantages of the proposed model are a proper buffer size will smooth video quality, the starting delay can be bounded, the video server load is reduced drastically and users with different joining time can view the first frame or beginning of the video stream. The details of this system model are shown in Figure 1.

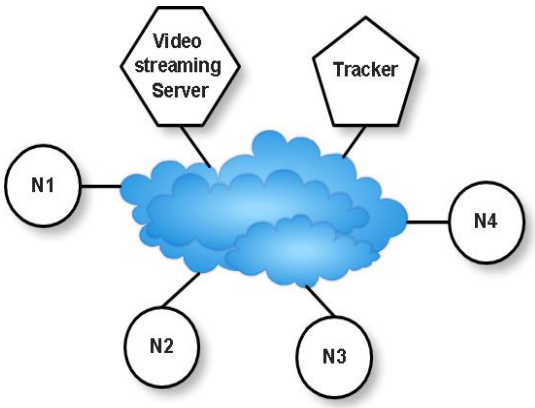

Figure 1. The non-cluster model for different start video broadcasting.

 ISBN: 978-1-61208-127-4

This paper presents a peer clustering system for different start video broadcasting to improve the content distribution for all peers in the network. The peers are grouped into clusters according to the availability of chunks and the join time of each node. A peer can download and upload video content with all peers in the same cluster. The global tracker and local tracker are created. The super node acts as a local tracker. This model can reduce server and tracker load. The control packet will be decreased.

III. CLUSTERING SYSTEM DESIGN

This section introduces the concept of system architecture, cluster-based system design and pseudo-code of cluster creation. When a new node joins and wants to download chunks from the peers in the cluster model for different start video broadcasting, the global tracker has to decide which cluster and super node will be joined.

A. System Architecture

The peer clustering system architecture groups nodes into clusters according to the join time of each node or the available chunks. This peer clustering system is composed of a server, a global tracker (GT), super node (SN) or local tracker (LT), backup-node (BN) and normal nodes (NN, seed and leech). The server is a node that shares all chunks of a live video stream. The global tracker is known by all nodes and maintains the list of all super nodes. The super node acts as a local tracker keeping the list of all nodes in the cluster. All super nodes are connected with global tracker to synchronize the lists of all nodes in the cluster. The super node, normal node and backup-node are all downloader (leech) and uploaders (seed) of chunks.

B. Cluster-Based System Design

For the peer clustering for the different start video broadcasting model, the clustering means the grouping of node partnerships according to their network proximity. The proximity is measured by the joining time of each node and the availability of chunks. In this work, we assume that each peer has enough bandwidth in order to upload and download chunks. Several users join in a short period of time. For the initial work, the number of users is defined as a constant. In the future work, the optimal number of users in the cluster will be calculated as a function of user bandwidth and round trip delay. The clustering is used to control the traffic streams within a P2P system and additionally helps to decrease the load of the server and global tracker. Based on the non-cluster model for the different start video broadcasting presented in [11][19][20], the algorithms of peer exchange information, peer selection, buffer organization, segment scheduling are not changed but extended by a logical clustering mechanism. The clustering is realized by the separation of nodes, super nodes, backup-nodes and the introduction of local trackers.

The cluster based system model consists of a global part and clusters. The global part composes of the server that provides the video stream and the global tracker that is known by all nodes in the network. The clusters consist of a super node (SN), a local tracker (LT), a backup-node (BN) and normal nodes (NN). The local tracker functionality is hosted by the super node itself. Both together are the coordinators for all nodes in their cluster. Whereas the local tracker acts as the source of neighbor lists, the super node is the only node in the cluster that can download from the server and other super nodes of the previous clusters. The normal nodes in a cluster know only other normal nodes within their own cluster and hence can only exchange chunks within the cluster. When the super node leaves from cluster, the backup-node will act as a new super node. The BN maintains a list of all nodes in the cluster. All normal nodes are candidates for the SN and BN. SN and BN selection process is explained in the next section. There are two major advantages of the cluster based system model. First, the clusters are only logical entities that are controlled by different known neighbor lists and the resulting chunk exchange paths. Second, the introduction of 3 levels of chunk sources (server, super nodes, nodes) leads to a controlled distribution of traffic and hence to a decreasing load for dedicated components mainly the server and tracker. An overview of the cluster-based system model is shown in Figure 2. The different components of the cluster based system can be divided into five processes as follows: (i) peer joining process, (ii) super node selection process, (iii) backup-node selection process, (iv) download process and (v) leaving node process.

1) Peer Joining Process

When a new node joins to use a video streaming and wants to participate with neighbor peers in the cluster, it will contact with the global tracker to ask for the cluster and super node. Therefore, the peer joining algorithm has 2 important phases: connect to global tracker and connect to local tracker. For the first phase, all nodes know the address of the global tracker. When a new node contacts with the global tracker and asks for the first chunk. The global tracker will contact SN of each cluster to search for nodes having the first available chunk. The global tracker then selects the cluster which has the maximum number of nodes containing the first chunk. The new node gets the address of the local tracker (SN) and registers there. For the second phase, the new node contacts with the local tracker. The local tracker returns a random list of neighbor peers in the same cluster to the new node. The new node receives a random list of neighbor peers and sends the message to exchange buffer maps with neigh-

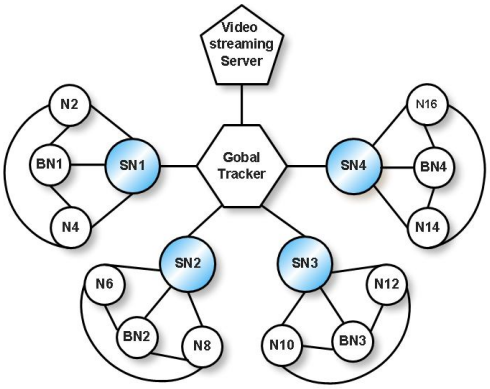

Figure 2. The cluster model for different start video broadcasting.

bor peers. The new node selects neighbor peers to download chunks.

2) Super Node Selection Process

When the first node joins, the global tracker will set the first node to be a super node and local tracker of the first cluster. The cluster size is limit to *m* nodes. After the global tracker received joining message from a new node, it will check the cluster size to select appropriate cluster. The global tracker will verify the member size of the selected cluster. If the size of the selected cluster is less than *m*, the address of SN in that selected cluster will be sent to the joining node. If the size of the selected cluster is full (equals *m*), a new cluster is created. The global tracker will split a node that has first chunk available in the old cluster to be a SN for a new cluster. If there is no cluster in the system, the first cluster is created and the first joining node will be a SN of the first cluster.

3) Backup-Node Selection Process

If the size of cluster is full (equals *m*), the backup-node will be selected. All normal nodes can be selected as backup-node. The backup-node selection is used to keep a list of all peers in the cluster by contact with SN. When the super node leaves from the cluster, a backup-node will be a new SN. The BN will receive the list of all peers in the cluster from SN. The backup node can be selected by three different methods as follow.

- Select the node joining the cluster after the first node. (the second joining node, 2^{nd})
- Select the node joining the middle of the group. (the $\frac{m}{2}^{th}$ node)
- Select the lasted node that joining the group. (the n^{th} node)

For the first method, all nodes in the cluster will have an equal chance to be a SN and BN. The drawback of this approach is a frequent SN and BN selection. The second method selects a new SN and BN not often and works well. The third method selects a new super node not often but may cause packet losses. In this paper, the second method is implemented in the simulation.

4) Download Process

When the new node joins in the cluster and contact with the SN, the local tracker (SN) prepares a random list of peers for a new node. The node knows only peers from the same cluster and can only download from neighbor peers. The buffer of each node is organized into three parts: data buffer, buffer map and sliding window. The data buffer is used to store video frames. The buffer map is a bit vector representing the information of available segments on a node. Each node exchanges its buffer map with its partners periodically. From buffer map information, the peer will decide which partner nodes are used to fetch required segments. If there is more than one partner having the same segments, the peer node will randomly select the partners or select the partners with minimum delay or maximum bandwidth. Besides buffer map, each node needs to have a sliding window which is used to store a number of displaying segments. From this buffer organization, the video segments will be displayed continuously, and the starting delay of each node will be bounded. In this work, the circular buffer is used as buffer management as shown in Figure 3. The buffer data is divided into three parts: playback buffer, displaying buffer, and next available buffer as follows [11][19][20].

- The playback buffer is used to buffer data stream for a certain period of time before playing the stream. The number of frames in playback buffer is calculated from the delay called playback delay between the sending and receiving peer. The playback delay between each peer is random since the mesh-based architecture is used. For simplicity, the playback delay is defined as a maximum delay bound in this group of users in this particular network. Thus, every peer will have the same playback delay.
- The displaying buffer is used to store data that will be viewed by users. This buffer is designed by using a sliding window. The frame in the beginning for buffer is the displaying frame and the next frame is the next frame in the window will be viewed in the next minutes.
- The next available buffer is used to receive new frames. The new frames are received from other peers or partners by using sequential or rarest-first scheduling.

The node will exchange buffer map with the neighbor peers and will select peers to download chunks. The most important concept to download chunks has two ways: sequential download and rarest-first download. The sequential download is used when more chunks are available on the maximum number of neighbor peers. The rarest-first download is used when less chunks is available on the minimum number of neighbor peers. The SNs can download chunks from the server and all other SNs.

5) Leaving-Node Process

If a node leaves from cluster, the local tracker will delete it from list of peers. If the leaving node is a local tracker (SN), the backup-node will be a new local tracker (SN). If the last node leaves the cluster, the cluster is deleted. The local tracker always tells the global tracker about leaving nodes to synchronize the list of SN in the global tracker. The leaving-node process can be divided into three cases: the leaving of super node, back-up node and normal nodes. For the first case, when the super node is leaving from the cluster, it sends flooding message to all nodes. The all nodes

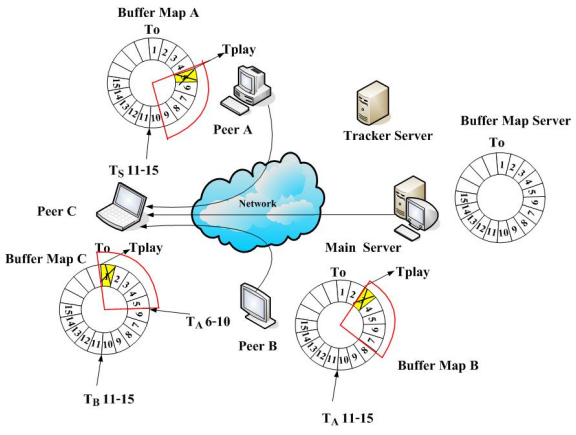

Figure 3. Filling buffer of peer C.

in the cluster will send keep-alive message to their super node. The super node sends keep-alive message to the global tracker. For the second case, the back-up node exchanges information periodically with the super node. If the back-up node leaves, it sends the message to inform the super node. For the last case, the normal node can leave the network at anytime.

C. Pseudo Code of Cluster Creation

The following pseudo code program demonstrates the cluster selection of global tracker, the cluster create and super node selection as shown in Table 1.

TABLE I. PSEUDO CODE OF CLUSTER CREATION

```
Function GetCluster(Node newNode, int neededChunk)
Begin
    Read N_P and Max_Node_In_Cluster
    For each cluster N determine occurrence of neededChunk
    Find cluster X with max occurrence
    If X is not found
        Generate new cluster C
        Set newNode as super node and local tracker of C
    Else
        If number of nodes in X < Max_Node_in_Cluster
            Add newNode to X
            Return X
        Else
            Find best node N for new cluster
            Cut N from cluster X
            Create new cluster C
            Set N to superNode and local tracker of C
            Add newNode to cluster X
            Return X
        End if
    End if
End
```

IV. A COMPARISON OF CLUSTER AND NON-CLUSTER MODEL SYSTEM

In this section, the cluster model will be compared with the non-cluster model for different start video broadcasting. Both system behaviors can be explain as shown in Table 2.

TABLE II. COMPARE STEPS OF NON-CLUSTER AND CLUSTER MODEL FOR DIFFERENT START VIDEO BROADCASTING

Non-cluster Model	Cluster model
• The requesting node searches for Tracker (T). • The requesting node connects with T. • T search for lists of all nodes in its table. • T replies random lists of peers to the requesting node. • The requesting node exchanges buffer map with neighbor peer lists. • The requesting node selects peers to download.	• The requesting node searches for Global Tracker (GT) • The requesting node connects with GT. • GT search for lists of all super nodes in its table. (Local Tracker: LT). • GT replies selected super node and cluster to the requesting node. • The requesting node connects with own super node (LT) and cluster. • LT replied random lists of peers to the requesting node. • The requesting node exchanges buffer map with neighbor peer lists. • The requesting node selects peers to download.

The peer clustering system for different start video broadcasting design can be reduced searching time and server load as the following:

A. Searching Time Reduction

For the non-cluster model, the tracker uses the sequential search to find the list of peers in the peer list table. With the non-cluster model, the searching time of tracker is O(n) where n is the total number of nodes.

For cluster model, the global tracker employs the binary search to seek the proper super node in the peer list table. The local tracker employs the sequential search to seek the list of peers in the peer list table. The global tracker keeps a list of all peers and arrival time of each node. The cluster model reduces the searching time in the global tracker and local tracker. It groups peers into cluster according to joining time (arrival time) of each node. Let server starts broadcast at the time, T = *0* and ends at the time, T = t. The arrival time of each node will be referenced with the server broadcast time and sorted from minimum to maximum value. Each node will be grouped to each cluster according to its arrival time, as shown in Figure 4. Then, the number of clusters is in order of 2^x. The tracker will check the arrival time of each node and then assign the proper cluster to that particular node. Thus, the number of nodes in each cluster is a random

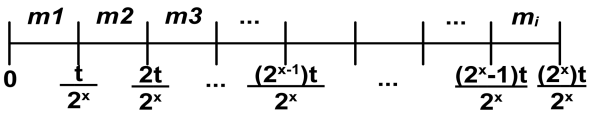

Figure 4. Event time line.

number. Let m_i is the number of nodes in each cluster, n is the total number of nodes in the system and 2^x is the number of clusters. With this structure, the binary search is used to find the proper super node in GT. The searching time of GT is O(log 2^x). The sequential search is used to find the list of peers in LT. The searching time of LT is O(m). Thus, the total searching time of cluster model is equal to is O(log 2^x + m).

$$\text{Non-cluster model} = O(n) \quad (1)$$

$$\text{Cluster Model} = O(\log 2^x + m) \quad (2)$$

B. Server Load Reduction

The peer clustering system for the different start video broadcasting is proposed to reduce the performance of server load. In the non-cluster model for the different start video broadcasting [11][19][20], the server supports all peers. For the cluster model the server supports only a super node in each cluster. Then, the server load can be calculated in Eq. (3), Eq. (4) and Eq. (5).

- Non-cluster model:

$$SL_{NC} = N_P \times N_C \quad (3)$$

- Cluster model with more than one nodes contacting server (Worst Case):

$$SL_C = N_{SP} \times N_C \quad (4)$$

- Cluster model with more only one node contacting server (Best Case):

$$SL_C = 1 \times N_C \quad (5)$$

Note: SL_{NC} denote the server load of non-cluster model.
SL_C denote the server load of cluster model.
N_P is the number of peers.
N_C is the number of chunks.
N_{SP} is the number of super nodes.

From Equations (3), (4) and (5), SL_C is less than or equal to SL_{NC}. The ratio of server load of non-cluster model and cluster model can be calculated as in Eq. (6) and Eq. (7).

$$\frac{SL_C}{SL_{NC}} = \frac{N_{SP} \times N_C}{N_P \times N_C} \quad (6)$$

$$SL_C = \frac{N_{SP}}{N_P} SL_{NC} \quad (7)$$

Since, $N_{SP} \leq N_P$, then $\frac{N_{SP}}{N_P} \leq 1$. Thus, $SL_C \leq SL_{NC}$

V. EXPERIMENTAL RESULTS

This section deals with the experimental performance evaluation of the peer clustering system for different start video broadcasting. The simulation experiments are conducted by the discrete event simulator NS-2 [21].

A. Simulation Setup

The experimental setup will create one video media server. The number of clusters is varied as 1, 2, 4, 6, 8 and 10, respectively. The non-cluster model is denoted by 1. The total number of nodes equals to 300 (limitation of simulation). For simplicity, the number of nodes in each cluster is equal. The bandwidth of all links is set to 2 Mbps, the delay is 10 ms and the joining time varies from 1 sec to 400 sec. The video stream bit rate is 512 Kbps. The video stream length is 32 MB, the size of each chunk is 64 Kb and the number of chunk is 512 chunks. The video play rate is 1 chunk/1 sec. The playback buffer and release buffer size are set to 10 sec and 15 sec, respectively. The buffer size of each node equals 54 sec.

B. Simulation Results

The simulation results are illustrated in Figures 5, 6, and 7. Figure 5 shows the relationship of server load and peer load of non-cluster and cluster model. Figure 5 plots the number of chunks download from server and neighbor peers in the network. The x-axis represents the number of clusters and the y-axis represents the number of downloaded chunks. The number of node is 300 nodes, each node downloads only 512 chunks. Hence, the total number of chunks downloaded equals to 153,600 chunks. The result shows that the number of chunks downloaded from server in the cluster model system is less than in the non-cluster one, and it is a constant. It means that the server serves only one peer. The other peers can download chunk from neighbor peers in the cluster.

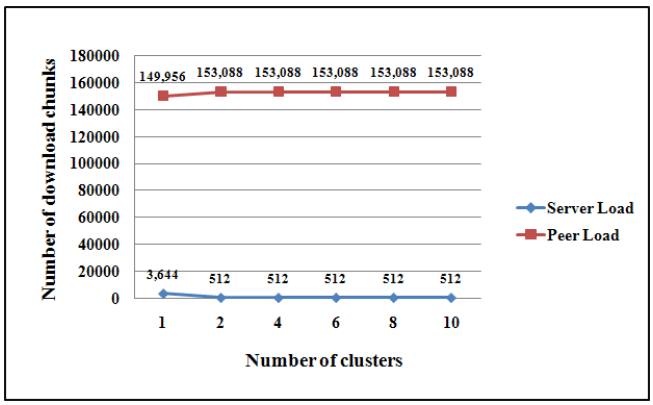

Figure 5. The relationship of server and peer load.

Figure 6 shows the relationship of global tracker connection with peers (tracker load). The x-axis represents the number of clusters (1, 2, 4, 6, 8, and 10) and non-cluster model (1). The y-axis represents the number of peers that has a connection with the tracker. The result shows that the tracker load of the cluster model system is less than the one of the non-cluster model. The tracker of the non-cluster model (1) serves for all peers in the system (300 nodes). The tracker of the cluster model serves for all super nodes or equals to number of clusters (which is equal to 2, 4, 6, 8, and 10 respectively).

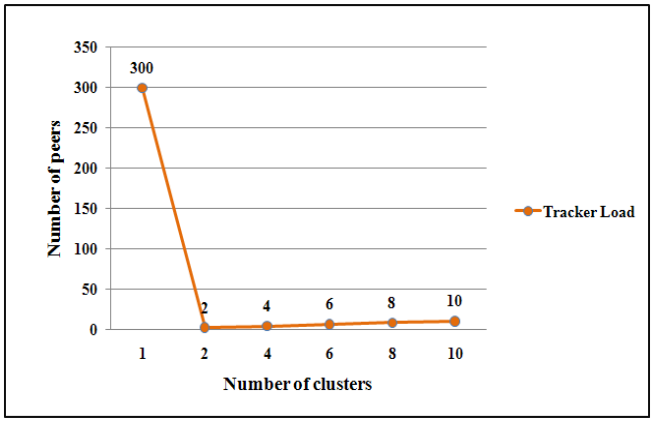

Figure 6. The relationship of tracker connect with peers.

Figure 7 shows the number of TCP control messages of each cluster used to exchange information between peers. The x-axis represents the number of clusters and non-cluster. The y-axis represents the number of control messages that used to connection between the peers. The result

shows that the control messages of the cluster model system are less than that of the non-cluster model.

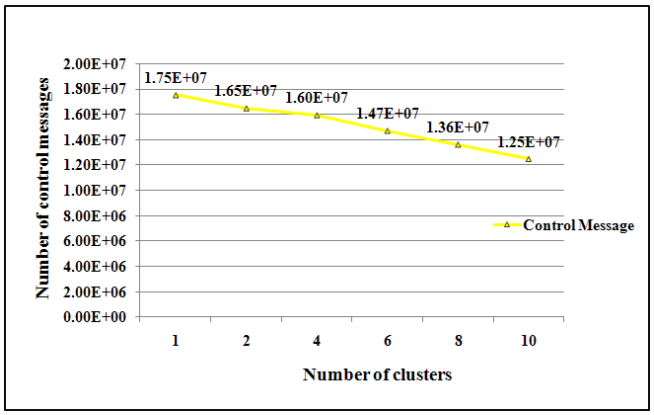

Figure 7. The control message of each cluster.

VI. CONCLUSION AND FUTURE WORK

In this paper, a peer clustering system for different start video broadcasting is proposed. The peers are grouped into cluster according to joining time of each node or availability of chunks. The cluster model is used to control the streaming traffic. The proposed model is created algorithms for peer joining process, super node selection process, backup-node process, download process and leaving node to create the cluster model. The cluster model for the different start video broadcasting will be compared with the non-cluster model. As a result, the proposed model can reduce server load and tracker load. The number of control packet is decreased as the number of cluster or super node increase. The play out delay is only 10 sec. The performance of the cluster model is better than non-cluster model. The unpunctual viewers can join the broadcast video streaming at any point of time and can view any part of the video stream. For further performance evaluation of the different start video broadcasting system, the tracker traffic, and node dynamics are simulated. The backup-node selection method will be reconsidered.

REFERENCES

[1] Gnutella [online], available at http://www.gnutella.com [Accessed: Aug. 10, 2010]

[2] Napster [online], available at http://www.napster.com [Accessed: Aug. 10, 2010]

[3] KaZaA [online], available at http://www.kazaa.com [Accessed: Aug. 10, 2010]

[4] BitTorrent [online], available at http://www.bittorrent.com [Accessed: Apr. 26, 2009]

[5] eDonkey [online], available at http://en.wikipedia.org/wiki/EDonkey 2000 [Accessed: JAN. 14, 2010]

[6] Y. Huang, T. Z. J. Fu, John C. S. Lui, and C. Huang, "Challenges, Design and Analysis of a Large-scale P2P-VoD System," in Proc. ACM SIGCOMM'08 conference on Data communication, pp. 375-388, Aug. 17-22, 2008.

[7] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum, "CoolStreaming/DONet: A data-driven Overlay Network for Efficient Live media streaming," in Proc. IEEE INFOCOM'05, vol. 3, pp. 2102-2111, Mar. 2005.

[8] C. WU and B. LI, "Exploring Large-Scale Peer-to-Peer Live Streaming Topologie," in Proc. ACM Transaction on Multimedia Computing, Communications, and Applications, vol. 4, no. 3, pp. 19-23, pp. 1-25, Aug. 2008.

[9] S. Tang, Y. Lu, J. Martín Hernández, F.A. Kuipers, and P. Van Mieghem, "Topology dynamics in a P2PTV network," in Proc. of the 8th International IFIP-TC 6 Networking conference'09, pp. 326-337, May 11-15, 2009.

[10] X. Su and L.Chang, "A Measurement Study of PPStream," in Proc. of the 3th International conference on Communications and Networking IEEE ChinaCom'08, pp. 1162-1166, 2008.

[11] H. Ketmaneechairat, P. Oothongsap, and A. Mingkhwan, "Smart Buffer Management for Different Start Video Broadcasting," in Proc. ACM ICIS'09, pp. 615-619, Nov. 24-26, 2009.

[12] B. Fallica, Y. Lu, F. Kuipers, R. Kooij, and P. V. Mieghem, "On the Quality of Experience of SopCast," in Proc. of the 1st IEEE International Workshop on Future Multimedia Networking FMN'08, pp. 501-506, Sep. 17-18, 2008.

[13] A. Sentinelli, G. Marfia, M. Gerla, and L. Kleinrock, "Will IPTV ride the peer-to-peer stream?," in Proc. IEEE Communications Magazine, vol. 45, no. 6, pp. 86-92, Jun. 2007.

[14] X. Hei, C. Liang, J. Liang, Y. Liu, and K. W. Ross, "Insights into PPLive: A measurement study of a large-scale P2P IPTV system," in IPTV workshop, International World Wide Web www'06 Conference, May 2006.

[15] J. Yu and M. Li, "CBT: A Proximity-Aware Peer Clustering System in Large-Scale BitTorrent-like Peer-to-Peer Networks," Computer Communication's Special issue on Foundation of Peer-to-Peer Computing, Vol 31/3: pp. 591-602, 2007

[16] Z. Chen, J. Liu, D. Li, H. Liu, and A. Vasilakos, "SOBIE: A novel super-node P2P overlay based on information exchange," Journal of Computers, vol. 4, no. 9, pp. 853-861, Sep. 2009.

[17] V. Lo, D. Zhou, Y.Liu, C. Gauthierdickey, and J. Li, "Scalable supernode selection in peer-to-peer overlay networks," in Proc.of the 2nd International Workshop on Hot Topics in Peer-to-Peer Systems, pp. 18-27, 2005.

[18] J. Peltotalo, J. Harju, L. Väätämöinen, I. Bouazizi, and I. Curcio, "RTSP-based Mobile Peer-to-Peer Streaming System," published in International Journal of Digital Multimedia Broadcasting. Vol. 2010.

[19] H. Ketmaneechairat, P. Oothongsap, and A. Mingkhwan, "Buffer Size Estimation for Different Start Video Broadcasting," in Proc. of the Electrical Engineering/Electronics Computer Telecommunications and Information Technology (IEEE ECTI-CON'10) International Conference, pp. 924-928, May 19-21, 2010.

[20] H. Ketmaneechairat, P. Oothongsap, and A. Mingkhwan, "Communication Size and Load Performance for Different Start Video Broadcasting," in Proc. of the 11th Annual Conference on the Convergence of Telecommunications, Networking & Broadcasting PGNET'10, Jun. 21-22, 2010.

[21] The Network Simulator (NS-2) [online], http://www.isi.edu/nsam/ns/ [Accessed: Feb. 15, 2009]

 ISBN: 978-1-61208-127-4