Journal of Computer Science 8 (7): 1134-1142, 2012
ISSN 1549-3636

Peer-to-Peer Video Conferencing Using Hybrid Content Distribution Model

Alhamza Munther, Salah Noori, Azlan Osman,
Ayad Hussain, Imad Jasim and Ali Shanoon
National Advanced IPv6 Centre of Excellence (NAv6),
Universiti Sains Malaysia, 11800 Penang, Malaysia

Abstract: Problem statement: Multimedia Conferencing System (MCS) is a server-based video conferencing system. One of the most critical limitations faced by this approach is the scalability issue. In the MCS, the outgoing bandwidth of the server is shared among all concurrent clients. Thus, the more clients there are, the lesser the bandwidth each client can have. The performance of this approach therefore deteriorates rapidly as the number of simultaneous clients increases. In addition, a pure server-based solution is expensive. **Approach:** In this research, the server-based infrastructure is modified into a peer-to-peer video conferencing system while preserving the same functionality and features of the existing MCS. This modification can be achieved using a hybrid content distribution model, which is a combination of fluid and chunk content distribution models to distribute parts of the video stream fairly among participants. The hybrid content distribution model offers a better way of handling heterogeneous networks because it can distinguish between a fast peer and a slow peer, dealing with each one according to its capabilities. **Results:** In our proposed system, the function server will not be used for video distribution. Instead, it will only be used for monitoring and controlling the peers to reduce the burden on the servers. Experimental results conducted in the nation advanced IPv6 center as a real environment and live conferencing. **Conclusion:** This will lead to overcome the problem of scalability and a bandwidth bottleneck on the main server and achieve good way to distribute video chunks.

Key words: Peer-to-peer, content distribution, heterogeneous network, widespread applications, video conferencing system

INTRODUCTION

Video conferencing applications are scalability issue limitation. Therefore, the overlay network is used to overcome all other limitations through the utilization of peer resources. An overlay network is a computer network built on top of another network. Nodes in the overlay are considered connected by virtual or logical links each corresponding to a path, perhaps through many physical links, in the underlying network (Andersen *et al*., 2001). Three kinds of overlay networks transfer the content parts: the Application Layer Multicast (ALM) network similar to (Pendarakis *et al*., 2001), the Peer-to-Peer (P2P) network such as Napster (Saroiu *et al*., 2003) and the Content Distribution Network (CDN) such as. Implementing multicast functionality at the application layer of the ALM network is feasible. CDN deploys servers in multiple, geographically diverse locations distributed over several Internet service providers. The client is requesting for the content directly from the nearest available server. P2P is an extremely popular method in which nodes in the network, called peers, offer resources such as bandwidth, processor and storing capacity to other nodes. Consequently, as the number of users increases, the global resources of the network also grow. Peers that serve another peer can also be chosen using proximity network criteria to avoid bottlenecks.

On the other hand, there are generally two types of content distribution models as classified in (Saleh *et al*., 2011): Fluid model and Chunk Content Distribution model. Fluid model provides continuous transferring of the content from the source to the multiple receivers (Liu *et al*., 2003; Hossain *et al*., 2009). This model has a tightly coupled connection (directly distributed bit by bit continuously from source to destination) between adjacent peers; therefore, it is considered as an optimal distribution model to utilize bandwidth for fast peers while causing congestion for slow peers. The second type, Chunk content distribution model, chops the content into equally sized pieces (called chunks) and subsequently distributes each chunk. A peer not

Corresponding Author: Alhamza Munther, National Advanced IPv6 centre of Excellence (NAv6), University Sains Malaysia, 11800 Penang, Malaysia

distributed piece until it has fully received that piece. Chunk model is considered a loosely coupled connection (stores the chunks prior to their distribution) (Kwon and Byers, 2004). Consequently, this model is considered an optimal distribution model for slow peers. We therefore suggest using a hybrid content distribution model that uses a fluid model to distribute video parts to a fast peer while employing a chunk model to distribute the video chunk to a slow peer.

The main contribution of this work is preserving the same functionality and features of the existing Multimedia Conferencing System (MCS) while modifying the server-based infrastructure to become a P2P video conferencing system. This modification is achieved using the hybrid content distribution model to attain a scalable, real-time, video conferencing solution. In our system, the function server will not be used for video distribution. Instead, it will only be used for monitoring and controlling the peers to reduce the burden on servers and to overcome the problem of a scalability and bandwidth bottleneck. Distinguishing between slow and fast peers will depend on the mechanism of the hybrid content distribution model, with the fast peers directly distributing content as in a fluid model while the chunk model is used with a slow peer.

Related work: Providing multipoint video conferencing service is challenging because of its high bandwidth demand and strict streaming quality requirement. Compared with traditional server-based solutions as in the Ramadass (2010), a server-based video conferencing system, one of the most critical limitations facing this approach is a scalability bottleneck, whereby the outgoing bandwidth of the server is shared among all concurrent clients. Specifically, the more clients there are, the lesser the bandwidth each client can have. Hence, the performance of this approach deteriorates rapidly as the number of simultaneous clients increases. In addition, this system is considered to be expensive compared with the P2P network.

Generally, there are two types of systems used to implement video conferencing applications. The first type is the centralized video conferencing system. This type is further classified into two systems: the Multipoint control unit MCU device system. MCU is a central device with a larger bandwidth for Internet connection than a regular participant (ITU, 1997). MCU has the capability to serve an N number of participants in a multipoint conference. The number of participants relies on the type of device and the server-based system; these systems use a centralized server to distribute the video signal among the participants. Each participant requires software such as WebEx 2003 or Ramadass (2010) to be able to log in to the conference and communicate directly with other participants. Generally, one of the biggest drawbacks of the centralized system is that it is not scalable. In addition, it requires a higher bandwidth to disseminate a single video signal among participants. The second type is the overlay network video conferencing system. As what happens in ALM, the concept of the overlay network video conferencing system is the possibility of implementing a multicast functionality at the application layer (Suman *et al*., 2002). The ALM approach has the ability to disseminate video signals faster, but the main disadvantage of this system is that it copes badly with a heterogeneous network. The slow node cannot bear the burden of the flow; thus, the video stream loses packets. P2P is another overlay network type used to transfer video stream chunks. The main advantage of this approach is scalability; each peer offers resources such as bandwidth, processor and memory to other nodes. The P2P overlay network used in different applications, for example, file sharing (Cohen, 2003), video streaming (Xin *et al*., 2005) and video conferencing (Akkus *et al*., 2006). On the other hand, most P2P video conferencing systems use fluid encoding to distribute the video stream among participants as in Vanets (Hossain *et al*., 2009). Vanets is a P2P video conferencing system that distinguishes between active and passive participants (active participants are producers of video stream, whereas passive participants represent viewers only). Vanets takes advantage of transcoding sees (Vetro *et al*., 2003) to allocate streaming rates optimally for all participating peers in the conference. In other words, transcoding can change the bit rate to meet the requirements of peers, as explained by (Xin *et al*., 2005). In transcoding, the video signal is changed by the relaying peer to meet a lower encoding rate through either re-encoding or changing key parameters such as the quantization values of Perlman (2004). P2P multipoint video conferencing using layered video (Akkus *et al*., 2006) employs a layered video with two layers: the base and the enhancement. The two layers have almost equal bandwidths. Hence, sending a base layer and an enhancement layer is not different from sending two different base layers. All participants that receive a base layer and with an enhancement layer added will enhance the video quality. The multi-rate and multiply P2P video conferencing system (Ponec *et al*., 2009) proposes that different receivers in the same group can receive a video at different video rates. In this system, an optimal set of tree structures is determined for routing multi-rate content using scalable video coding (Schwarz *et al*., 2007). This system divides peers into many groups. Each group can be represented by a tree

and each peer in the tree can receive different video rates rather than a single rate (Chen *et al*., 2008). A small amount of P2P video conferencing systems uses the chunk distribution model (Afergan, 2006) because this model causes a delay in the system. However, this model is used widely in P2P video streaming (Bertinat *et al*., 2009) and in P2P file sharing (Cohen, 2003). The CAllCMe is one of the systems that follow the chunk distribution model, which aggregates video streams from multiple sources. In other words, all video streams are aggregated and combined into one stream. This operation is conducted using video composition operations (Afergan, 2006). The final resulting stream containing all the source images, referred to as the Panorama, is received by the conference participants. The advantage of this model is that it does not use network coding, where coding/encoding increases the processing time and computational complexity at each host. The disadvantage is that it entails a too much delay and traffic. This drawback in the model is caused by all video chunks being aggregated into a large buffer that leads to end-host delay. This problem is called the stream diffusion metric (Bianchi *et al*., 2009). In the current study, we proposed the conversion of the MCS server-based system to a P2P video conferencing system called the P2PMCS using a hybrid content distribution model (a combination of Fluid and Chunk content distribution model) without a coding network. In this way, we overcome the disadvantages of both models and get more nearly optimal behavior to distribute the content in our system.

MATERILAS AND METHODS

The proposed system P2PMCS is modifying infrastructure for the existing server-based Ramadass (2010) to peer-to-peer network. It preserves the same features of the existing MCS, whereby the function server will not be used for video distribution. Instead, it will only be used for monitoring and controlling peers to reduce the burden on servers and to overcome the problem of scalability and bandwidth bottlenecks. The hybrid content distribution model is responsible for distributing video chunks among peers. Below we give a description of the framework of the proposed system. Generally, the framework consists of three main units: P2P control message, Neighbor selection unit and distribution unit.

P2P control messages: This unit exchanges messages between the central server and the peers to transfer the responsibility of distributing the video chunks from the server and entrusting these to the peers. Server-based MCS used the RSW control criteria (Sureswaran and Subramaniam, 1995) which define all messages to manage a conference. In our proposed P2PMCS, some of the messages need to be added to the RSW control criteria for the P2P architecture. The additional messages can be classified into two types: Peer-to-Peer messages and Peer-to-server messages.

P2P message: These messages occur only between peers. They provide information on how making the process of distributing the video chunks smooth. This type consists of three messages “Flooding Segment,” “SplitValue,” and “Start Swarm.” Each message involves more than one packet.

Flooding segment: After the peer (i) (peer I is any peer that is already logged on to the conference) logs into the system and exchanges a set of messages to the main server, he gets a list of all peers with their IPs in the conference. Peer (i) sends a small packet, starts a timer and waits for an acknowledgment from other peers. The time between sending the packet and the receiving acknowledgment is measured. This process is repeated with all other peers in the conference. Afterward, peer i stores all the peers’ acknowledgments in a table and sends a copy of this table to the server. To achieve this process four packets are needed: FloodingSeg, ACKFlooding, Select You and ThxACK.

Flooding seg: This packet is sent by the peer (i) to all other peers that received IPs for them from the server. This packet is considered the first step in calculating the Round Trip Time (RTT) value. This packet consists of one field of up to 100 bytes filled with a set of 1’s.

Ack flooding: This packet is sent to reply to “FloodingSeg”. Thus, the new peer will stop the timer to calculate the RTT value. This packet consists of one field which indicates this peer is ready to become a neighbor for peer (i).

Select you: This packet is sent by the peer (i) to inform another peer that is selected to be a neighbor to him. This packet consists of two fields, one containing the peer ID and another for status (selected or not).

Acceptack: This packet is sent to reply to the previous packet “Select You” to inform the new peer he will accept the offer.

Split value: This message is sent between peers to agree on the value of splitting the video stream. In other words, each video stream will split into many sub-streams, with each sub-stream containing C number of chunks. Therefore, the sender peer will send the splitting value to inform receiver peers about this value.

This message consists of three packets: split value, valueack and update value.

Split value: This packet is sent by the peer (i) to his neighbors to inform them about a value of the split video stream. In other word, the size of a video chunk is sent.

Value ACK: This packet is sent by peer I neighbors of peer I to reply to the previous packet "SplitValue" and inform him the split value was received.

Update value: This packet is sent by peer I update the split value occasionally. This packet consists of one field containing the new value of splitting.

Start swarm: This message is considered to be the first step to take before sending chunks of video to avoid sending duplicate video chunks. In addition, this message informs neighbors of peer I to exchange video chunks among them. It consists of three packets: CHKSEQ, CHKSEQACK and StartSwarm.

Chkseq: In peer source (peer that sends video stream chunks), the main video stream is split into many chunks, with each chunk having a unique sequence number. The initial step before sending any chunk is to send this packet in order to check the unique sequence number for the specific chunk. It consists of one field containing the sequence number for the chunk.

Chkseqack: This packet is sent by the neighbor peer to reply to the previous packet "CHKSEQ" and tell the peer source if this chunk was received before or not.

Start swarm: This packet is sent by a peer source after being accepted by a neighbor peer in order to start exchanging video chunks with all his neighbors. After the neighbors of peer source receive this packet, sending the video chunks can be started immediately.

Peer-to-server messages*:* This type of message sent by a peer to the server aims to update the server with all information happening among the peers. The server will then store the message so that other peers can benefit from it. This type contains two messages: ConfigNbrTable, RecoverYPeer.

ConfigNbrTable: This message is sent by a new peer to the server to ask him to join the conference after neighbors are selected. In P2PMCS, we need to determine the neighbors for each peer who would exchange parts of the video stream with them based on RTT values to identify the nearest peers in the underlying network. This message consists of two packets: Negtable and ACKReceiV.

Nrbtable: This packet is sent by a peer to inform the server that he selected these peers as neighbors. This packet consists of 2 K (i) fields, where K (i) is the first field referring to the number of neighbors to peer (i) Packet size is 5 k (I) bytes (4 bytes for IP address and 1 byte for RTT value). The number of neighbors must not exceed Max.

ACK Receiv: The server sent this packet as a reply peer to the "Nrbtable" packet. Upon acceptance of the list, the neighbors begin the swarming operation. This packet consists of one field which contains the status indicating whether the previous packet was received or not.

Recovery Peer: This message sent by the peer to the server informs the peer about one or more of the neighbors left in the neighbor list. Thus, some peers must be added to compensate for the remaining peers. This message consists of two packets: PeerLeaV and recovery.

PeerLeaV: Peer sends this packet to the server to inform him he needs a new peer because one of his neighbors left.

RecoverY: This packet is sent by the server to reply to Peerleav which either contains or does not contain a new IP to recover the missing neighbors for peer (i). If no new IP is contained in PeerLeav, it means the server does not have a new IP.

Neighbor selection unit: This unit is used to select neighbors to any new peer logged on to the conference. The main purpose of this unit is to select the nearest peers for a new peer to exchange video chunks with them (Ardizzone *et al*., 2007). In P2PMCS, these neighbors are chosen by measuring the distance between a new peer and the rest of the peers in the conference. This distance is determined by calculating the RTT value. In other words, the duration it takes to send the Flooding Seg packet from the new peer to all other peers until receives the ACKF loading packet from the receipt side. In the meantime, a new peer starts to create a list of neighbors by recording the name of the neighbor with the value of the RTT and the IP address for each peer. In this list, peers are arranged according to the values of the RTT from least to most (the least RTT is located at the top of the list).

The number of neighbors does not exceed MAX, which is the largest number of neighbors for each new peer.

Distribution unit: This unit is concerned with the distribution of video chunks. It is represented by the Hybrid content distribution model which is considered the core of this unit. Generally, the hybrid content distribution model consists of 4 components: data receive, flow distribution, traffic monitoring and binding buffer.

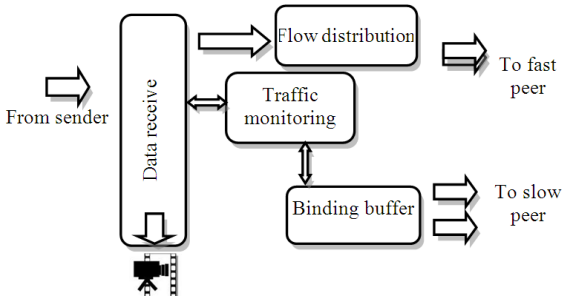

Fig. 1: Hybrid content distribution model units

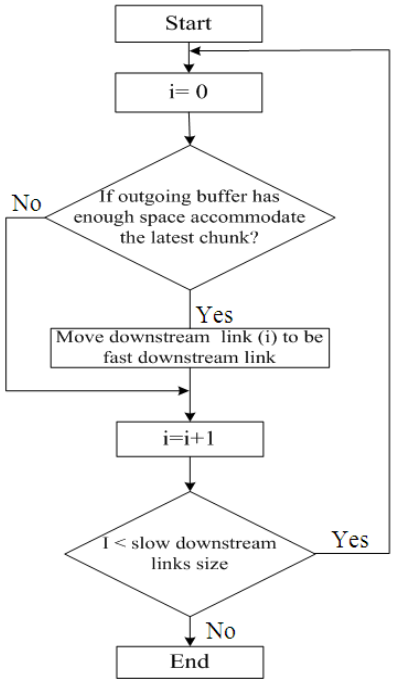

Fig. 2: Moving fast downstream link form binding buffer to flow distribution

As shown in Fig. 1, the data receive section handles all the communication of the sending peer. Data Receive also receives the chunk's packets from the incoming TCP buffer and stores them in the hard disk as part of the data content. Data Receive then receives a message from the traffic monitoring section to pass these chunks to the flow distribution or to the binding buffer section where flow distribution represents the Fluid model part without any network coding. The binding buffer represents the chunk model part. In this way, hybrid content distribution model is considered a combination of both content distribution models, fluid and chunk.

Flow distribution: As mentioned previously, the Flow Distribution section represents the fluid model but without network coding, where the upstream link is tightly coupled with all other fast downstream links (continuous transfer of content from upstream link to downstream link). Flow distribution stores the arriving chunks from the Data Receive section in a small incoming buffer and transfers the chunks directly and continuously to the application layer buffer for each downstream link (receiver). This section works well with fast peers because each peer has the capability to transfer the chunks of the content quickly. At the same time, it sends these chunks directly to the Binding Buffer unit to make these chunks available for slow peers.

Binding buffer: The binding buffer section represents the chunk model wherein it binds the upstream link to slow downstream links in a loosely coupled way. Hence, content is transferred indirectly because it is stored in a temporary buffer before being transferred to the slower downstream links. This component receives the chunks from the Flow Distribution and builds a group of adjacent chunks in a buffer to be used by slow peers.

Traffic monitoring: The traffic monitoring section is the key to controlling the entire hybrid content distribution model. Traffic monitoring has two main functions. First, it collects information from all other sections inside the model per session and processes this information to make appropriate decisions. It also ensures that the flow of content is seamless and free from congestion. Second, moving peers from flow distribution to binding buffer and vice versa are as follows: Traffic monitoring tracks all downstream links and monitors the outgoing buffer overflow. If this buffer is always full and has no sufficient buffer space for more chunks, this downstream link is moved to the Binding Buffer unit because this peer will not have enough bandwidth to transfer content directly. The same procedure happens inside the Binding Buffer for any downstream link requesting the latest building chunks or when a peer has enough bandwidth to accommodate several chunks.
The traffic monitoring makes a decision to move this peer to the Flow Distribution so that it can transfer content quickly, as shown in Fig. 2.

The whole scenario for the distribution of video chunks is illustrated in Fig. 3. After the stream is divided into several chunks in the splitter, the hybrid content distribution model receives chunks through the Data Receive section. Each chunk is distributed separately using different establishments for the content distribution model (e.g., chunk1 through establishment1, chunk2 through establishment2 and so on). These chunks usually have the same size. Each establishment of the hybrid content distribution model forwards the received chunks to fast peers that can immediately be written into the downstream link across the flow distribution section.

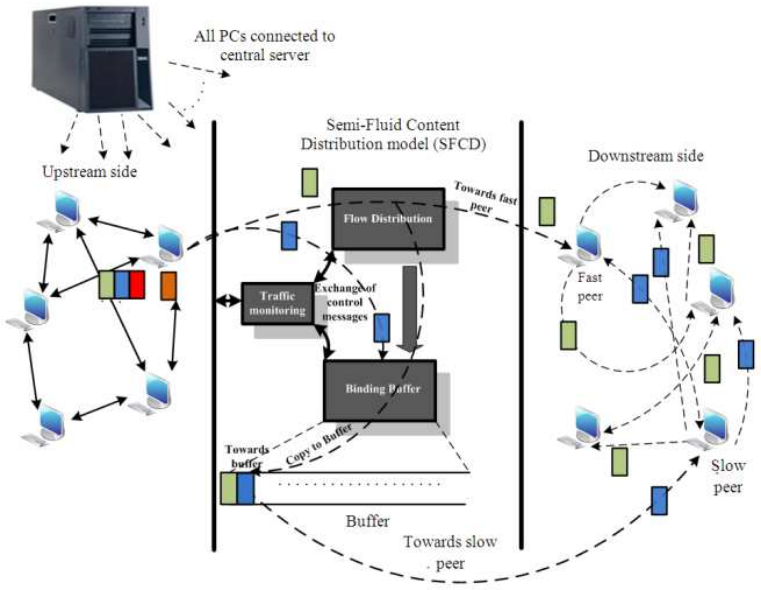

Fig. 3: Distribution for two different video chunks

At the same time, it sends these chunks directly to the Binding Buffer section and builds a group of adjacent chunks in a buffer for transfer later to slow peers.

The load on the main server in a server-based MCS and P2PMCS: Assume the number of users N, the number of active users (source of video) S and the video bit rate R. Depending on the server-based system, where video stream is transferred from the active user to the server, the server redistributes the video stream to other users. Therefore, the total upload bandwidth that must be contributed by the central server can be formulated as Eq. 1:

$$T_{us} = [(N \times S) - S] \times R \quad (1)$$

In P2P networks, the scenario is different; here, the central server is entrusted with the distribution process for peers to disseminate video chunks. Active peer distributes the video chunks to his neighbors. For example, assume active peer p has G neighbors and the size of video chunk is C. As a result, total upload bandwidth to each peer can be calculated by collecting the chunks C distributed to neighbors G where each neighbor peer will be responsible for distributing parts of the main video stream.

RESULTS

A software prototype of P2PMCS has been implemented in the laboratory of the National Advanced IPv6 Centre (NAv6). Two different scenarios, server-based and P2P network, were tested. Ten PCs were used as the client with different specifications. One server of Intel Core 2 Duo CPU E6750 2.66 GHz with 3.00 GB memory and 250 GB Hitachi Hard Drive was set up in a mesh topology connected to a P2P scenario Fig. 3. The CPU usage of the central server is tested in both architectural server-based system and P2P. It is measured by a windows task manager.

Our result shows a steady state for CPU usage. Results for both systems are shown in Fig. 4. In the server-based system, the overhead on the server Our result shows a steady state for CPU usage. Results for both systems are shown in Fig. 4. In the server based MCS CPU usage was 1% for one user, which then increased to 43% with 10 users. In contrast, CPU usage for P2PMCS decreased to 14% with 10 users. The second vital factor tested was the upload bandwidth of the central server for both systems. A NetLimiter tool was used to measure the uplink bandwidth and the video bit rate at 384 kbps. Figure 5 reveals the result of the comparison of upload bandwidth in the server for both server-based and P2P MCS system. In server based the upload bandwidth was about 400 kbps with two users This value increased gradually when the number of users increased, with the bandwidth even reaching 3900 kbps with 10 users. Meanwhile, the values of upload bandwidth in the P2PMCS ranging between 3.5 kbps and 52 kbps depend on the number of participating in the conference.

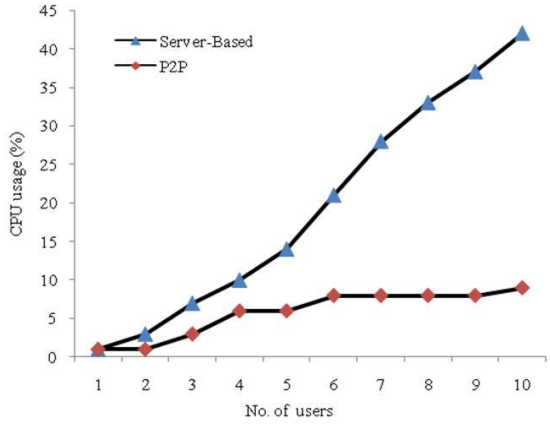

Fig. 4: CPU usage for the central server in a server-based MCS and P2PMCS

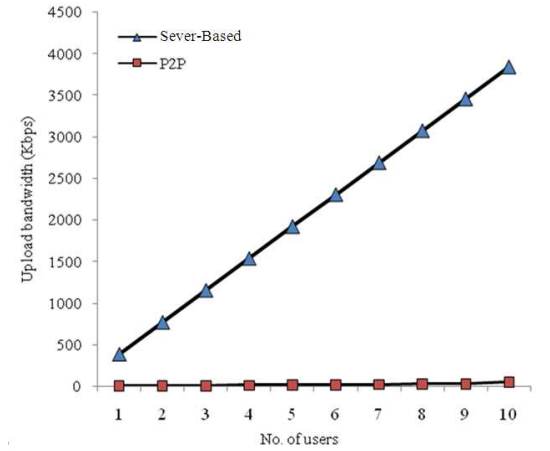

Fig. 5: Upload bandwidth for the server with server-based MCS and P2PMCS

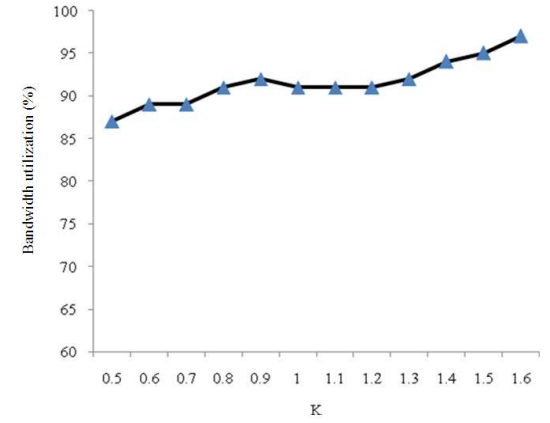

Fig. 6: Utilization bandwidth for peers in P2PMCS

In addition, we calculate utilization uplink over the entire P2PMCS, where each peer is responsible for uploading each video chunk already received. Simply defined, the average per-connection bandwidth is equal to K × bw, where K K≤0, is a tunable parameter. In case (K≤1), then the connection bandwidth is less than or equal to the description band width. Figure 6 shows the utilization bandwidth for the mesh topology in Fig. 3, peers are able to efficiently utilize the upload bandwidth (>%95).

DISCUSSION

According to the analyzed results, the performance of P2P MCS is noticeably better than that of server-based MCS. In the server-based system, the overhead of CPU on the server clearly increases when the number of users increases. Meanwhile, in the P2P system, there is a simple load on the CPU generated by the monitoring process of the peers and the exchange information with peers. Furthermore, the larger upload bandwidth of the central server in the server-based MCS compared to in the P2PMCS, because in the server-based MCS, the server is responsible for disseminating video stream to all users. In comparison, in P2PMCS, the functionality of the server is limited to monitoring peers providing information about the conference. In addition, the mesh always allows full leverage of the upload bandwidth that peers make available to the system. The upload capacity utilization is a little short of 100% because of the startup phase when peers are unable to utilize their uplinks effectively during the startup phase.

CONCLUSION

In this study, the existing server-based MCS is modified into a P2PMCS to reduce the burden on the central server. In the P2PMCS, the distribution process of video stream chunks is entrusted to peers. The hybrid content distribution model is a combination of Fluid and Chunk content distribution models used to distribute chunks of the video stream. This model is distinguished by its fast and slow peers, which it handles separately according to its capabilities. In comparison, the fastest peer fluid model is used to distribute chunks while the chunk model is used to distribute video chunks.

REFRENCES

Afergan, M., 2006. Experience with some principles for building an internet-scale reliable system. Proceedings of the 5th IEEE International Symposium on Network Computing and Applications, Jul. 24-26, IEEE Xplore Press, Cambridge, MA., pp: 3-3. DOI: 10.1109/NCA.2006.24

Akkus, I.E., M.R. Civanlar and O. Ozkasap, 2006. Peer-to-peer multipoint video conferencing using layered video. Proceedings of the IEEE International Conference on Image Processing, Oct. 8-11, IEEE Xplore Press, Atlanta, GA., 3053-3056.

Andersen, D., H. Balakrishnan, F. Kaashoek and R. Morris, 2001. Resilient overlay networks. SIGOPS Proceedings of the 18th ACM Symposium on Operating Systems Principles (SOSP), (SOSP' 01), Banff, Canada, pp: 131 145.

Ardizzone, E., L. Gatani, M.L. Cassias, G.L. Re and M. Ortolani, 2007. Enhanced p2p services providing multimedia content. Adv. Multimedia, 2007: 637-646. DOI: 10.1155/2007/26070

Bertinat, M.E., D.D. Vera, D. Padula, F.R. Amoza and P. Rodriguez-Bocca *et al*., 2009. GoalBit: The first free and open source peer-to-peer streaming network. Proceedings of the 5th International Latin American Networking Conference, Sept. 24-25, ACM Press, Pelotas, Brazil, pp: 49-59. DOI: 10.1145/1636682.1636691

Bianchi, G., N.B. Melazzi, L. Bracciale, F.L. Piccolo and S. Salsano, 2009. Fundamental delay bounds in peer-to-peer chunk-based real-time streaming systems. Proceedings of the 21st International Teletraffic Congress, Sept. 15-17, IEEE Xplore Press, Paris, pp: 1-8.

Chen, M., M. Ponec, S. Sengupta, J. Li and P.A. Chou, 2008. Utility maximization in peer-to-peer systems. Proceedings of the International Conference on Measurement and Modeling of Computer Systems, Jun. 2-6, ACM Press, Annapolis, MD, USA., pp: 169-180. DOI: 10.1145/1375457.1375477

Cohen, B., 2003. Incentives build robustness in bittorrent. The Pennsylvania State University.

Hossain, T., Y. Cui and Y. Xue, 2009. Vanets: Case study of a peer-to-peer video conferencing system. Proceedings of the 6th IEEE Consumer Communications and Networking Conference, Jan. 10-13. IEEE Xplore Press, Las Vagas NV, pp: 1-2. DOI: 10.1109/CCNC.2009.4785023

ITU, 1997. Multipoint control units for audiovisual systems using digital. International Telecommunication Union.

Kwon, G.I. and J.W. Byers, 2004. Roma: Reliable overlay multicast with loosely coupled tcp connections. Proceedings of the Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies, Mar. 7-11, IEEE Xplore Press. DOI: 10.1109/INFCOM.2004.1354511

Liu, J., B. Li and Y.Q. Zhang, 2003. Adaptive video multicast over the internet. IEEE Multimedia, 10: 22-33. DOI: 10.1109/MMUL.2003.1167919

Pendarakis, D., S. Shi, D. Verma and M. Waldvogel, 2001. Almi: An application level multicast infrastructure. Proceedings of the 3rd USENIX Symposium on Internet Technologies and Systems, Mar. 26-28, San Francisco, California, USA., pp: 49-60.

Perlman, R., 2004. Models for ip multicast. Proceddigns of the 12th IEEE International Conference on Networks, Nov. 16-19, IEEE Xplore Press, pp: 678-682. DOI: 10.1109/ICON.2004.1409261

Ponec, M., S. Sengupta, M. Chen, L. Jin and P.A. Chou, 2009. Multi-rate peer-to-peer video conferencing: A distributed approach using scalable coding. Proceedings of the IEEE International Conference on Multimedia and Expo, Ju. 28-Jul. 3, IEEE Xplore Press, New York, pp: 1406-1413. DOI: 10.1109/ICME.2009.5202767

Ramadass, S., 2010. Multimedia Conferencing System. 1st Edn., VDM Verlag, ISBN-10: 3639238524, pp: 188.

Saleh, S.N., M. Feily, S. Ramadass and A. Hannan, 2011. Performance study of fluid content distribution model for peer-to-peer overlay networks. Adv. Wireless, Mobile Netwo. Appli., 154: 181-190. DOI: 10.1007/978-3-642-21153-9_17

Saroiu, S., K.P. Gummadi and S.D. Gribble, 2003. Measuring and analyzing the characteristics of napster and Gnutella hosts. Multimedia Syst., 9: 170-184. DOI: 10.1007/s00530-003-0088-1

Schwarz, H., D. Marpe and T. Wiegand, 2007. Overview of the scalable video coding extension of the H.264/AVC standard. IEEE Trans. Circ. Syst. Video Technol., 17: 1103-1120. DOI: 10.1109/TCSVT.2007.905532

Suman, B., B. Bobby and K. Christopher, 2002. Scalable application layer multicast. Proceedings of the 2002 Conference on Applications, Technologies, Architectures and Protocols for Computer Communications, Aug. 19-23, ACM Press, Pittsburgh, PA, USA., pp: 205-217. DOI: 10.1145/633025.633045

Sureswaran, R. and R.K. Subramaniam, 1995. A control criteria to optimize collaborative document and multimedia conferencing bandwidth requirements. Proceedings of the IEEE Singapore International Conference on Network, Jul. 3-7, IEEE Xplore Press, pp: 555-559. DOI: 10.1109/SICON.1995.526349

Vetro, A., C. Christopoulos and H. Sun, 2003. Video transcoding architectures and techniques: An overview. IEEE Signal Process. Mag., 20: 18-29. DOI: 10.1109/MSP.2003.1184336

Xin, J., C.W. Lin and M.T. Sun, 2005. Digital video transcoding. Proc. IEEE, 93: 84-97. DOI: 10.1109/JPROC.2004.839620