The CSI Journal on
Computer Science and Engineering
Vol. 10, No. 2 & 4 (b), 2012
Pages 20-27
Regular Paper

A Distributed Improvement Mechanism for Mesh-Based Peer-to-Peer Video-on-Demand Streaming

Abdollah Ghaffari Sheshjavani Behzad Akbari

Department of Electrical and Computer Engineering, Tarbiat Modares University, Tehran, Iran

Abstract

In P2P video streaming, the way of organizing peers in an overlay is one of the most important elements to achieve optimum use of system resources and better quality of service. For this purpose, the peers that they contribute more upload bandwidth, should located closer to the servers. This paper presents a distributed improvement mechanism to performmesh-based video-on-demand. This mechanism proposes a new method tolocatepeers withmore upload bandwidth, closer to the servers, with regard to playback points of them. The results of evaluating the proposed mechanism shows that our mechanism efficiently enhanced the quality of service parameters, and it is robustness in dynamic environments, the results also confirmed that our mechanism acts as an incentive mechanism.

Keywords: Video-on-Demand, Peer-to-Peer, Improvement Mechanism, Incentive Mechanism, Video Streaming.

1. Introduction

With the fast deployment of high speed residential access networks, nowadays video data dominates the main part of the Internet traffic. Video streaming applications can be classified into two categories: live and on-demand. In a live streaming session, a live video content is disseminated to all users in real-time. The video playbacks on all users are synchronized. To the contrary, video-on-demand (VoD) users enjoy the flexibility of watching whatever video clips whenever they want. The playbacks of the same video clip by different users are not synchronized and different users might watch the same video with very large differences in viewing point.

The basic solution for video streaming over the Internet is the client-server service model. Some video content providers like YouTube and Netflix use this model to deliver video content to the users. To increase the performance and reliability of content delivery in client-server model, video contents are replicated on a number of servers and the requests of video streaming are balanced among the servers. For this purpose, content delivery networks (CDN) play an important role in large scale video delivery over the Internet based on client-server model. In recent years, increasing the interest of watching online videos and the cost of providing the necessary bandwidth for a large number of users in client-server model has caused an increasing attention to Peer-to-Peer (P2P) and also hybrid CDN-P2P approaches. In P2P approaches, the resources (e.g. bandwidth) of the internet clients are used to deliver video content to the users. In order to improve the quality of the delivered video in P2P networks, using hybrid CDN-P2P approach has gained more attention during recent years.CDN-P2P approach estry to gain the benefits of both CDN and P2P model [1]-[4].

In P2P approaches clients are organized in an overlay network. In P2P video streaming systems, the way of organizing peers in an overlay is one of the most important elements in optimum use of system resources. In P2P streaming systems, overlay structures can be broadly divided into two categories: tree-based and mesh-based. The tree-based systems, have well-organized structures and usually spread video contents by actively pushing data from a peer to its children. Some of these systems are Overcast [5], ESM [6], Split Stream [7] for live and P2Cast [8], P2PVR [9] for VoD. Within a stable streaming tree, the delay is strictly bounded, but the complexity of maintaining a stable tree is high due to the peer churn.

In mesh-based approach, In order to provide robustness against peer churns and to meet the streaming bandwidth requirement, peer joins to the overlay network by selecting some other peers as neighbor. In this method video is divided into some pieces named chunk, and peers transfer chunks to each other usually by requesting *(pull)* missing chunks from its neighbors. Mesh-pull is based on the buffer map exchange, because of which a child knows exactly the packets that each parent has, and can explicitly pull packets from each of them. Some of these systems for live and on demand video streaming are Cool Streaming [10], BASS [11] and PP Live [12]. Mesh-based P2P streaming systems have low complexity in construction and maintenance of the overlay network and also they are more robust against peer churn in contrary to the tree based systems. Therefore, we focus on mesh overlay, which can achieve resilience to peer dynamics and is easier to implement.

Some of systems like PRIME [13] and Hy Stream [14] use a combination of pull and push based approach. In these systems first video chunks are pushed to child neighbors in a tree structure, and then use pull approach to complete missing chunks. To achieve the best of both tree and mesh approaches, also in some works like [15], [16] and [17] hybrid tree-mesh structures are proposed. In [15]at first streaming uses mesh structure, but after it a tree built over the mesh. This tree is built from the peers that they spend bigger than a threshold time in the system.

It can be seen that for better use of system resources in order to enhance the quality of service, various ways of peer organization in overlay are proposed. Some of systems like P2Cast [8] and P2PVR [9] pay attention to playback offsets of peers, and some like [13] pay attention to the peer's spent time in the system.P2PVR [9] sorts all of the peers into a list in accordance with their playback offsets. Specifically, peers with a larger playback offset are placed toward the beginning of the list (i.e. closer to the streaming server) while those with a smaller playback offset are placed toward the end of the list. Authors in [18] also pay attention to the peer's upload ability. Authors in this paper, with theoretical calculation, conclude that to achieve the best quality of service and optimum use of peer’s resources, the peers with better uploading ability, must be located closer to the CDN servers.

In this paper we propose a distributed improvement mechanism for mesh-based P2P VoD streaming. In this mechanism we try to optimize the use of resources by propose a new method to locate peers that they can do more upload closer to the video servers with awareness to their playback point.

The rest of this paper is organized as follows: In section II we present our proposed mechanism. The performance evaluation of the proposed mechanism is provided in section III. Finally, the paper is concluded in section IV.

2. The Proposed Mechanism

2.1. Problem Analysis

In this paper we use this principle: "to improve the quality of service, the peers that have more upload bandwidth (or more contribute to others), should be located closer (lower hops) to servers". Using this principle in addition to improvement in topology by efficient use of the network resources such as peer's upload bandwidth, incentive the peers to contribute more. In VoD streaming in addition to uploading bandwidth and measuring of contribution of peers, we also need attention to playback point of peers. Assume in a mesh structure peer *A* has bigger upload bandwidth than peer *B*, (*A* has more potential for contribution than *B*), and peer *B* is closer to server, but playback point of *B* precedes *A*, if we change location of *A* and *B*, peer *B* might experience discontinuity in watching video, because of changing location it may not be able to receive video chunks on time. Probability of this event is increases with increase of the distance between playback points of *A* and B. However when this distance isn't large, the probability of peers experiencing discontinuity will below. So if we want to locate peers that have more upload bandwidth, closer to the servers, we should tradeoff between the difference upload bandwidth and difference playback point of peers.

2.2. The Improvement Mechanism

In this subsection we explain our improvement mechanism. The following Summarizes improvement mechanism process: Neighbor peers exchange required information among themselves, such as playback point, share upload bandwidth and distance to server (in hop), then each peer compare its own information with its neighbors. Assume peer *A* is a neighbor of peer *B*. if *A* shared more upload bandwidth than *B* (more enough) and *B* was located in lower hops to server, and playback difference of them was not large (small enough), *A* was requested *B* to exchange their places, and if *B* accepted, then *A* and *B* interchanged their neighbors. This process is repeated periodically, so this mechanism doesn't stop and uninterruptedly Strives to locate those peers which have more contribute, close to servers. Also especially when the network has churn, this mechanism continuously repairs the negative effects of churn (by always trying to locate more contributing peers close to servers all the time). Another advantage of this mechanism is that it is distributed, and does not have any overhead on the servers.

The steps of improvement process of our mechanism are: 1) Finding candidate neighbors for exchange places. 2) Selecting the best neighbor from candidate neighbors and sends the improvement request to it. 3) If the neighbor is in agreement with the request (if all condition are right, it agrees with the request), the two peer interchange all their neighbors between each other. (All neighbors of *A* become neighbors of *B* and all neighbors of *B* become neighbors of *A*.)

2.2.1. Finding candidate neighbors to exchange places

Each peer sends this information with buffer-map message to all its neighbors periodically: distance from server (in hop), contributed upload bandwidth and playback point. Each peer compares this information to its neighbors periodically. In our improvement mechanism each peer to find candidate neighbors for exchange places, consider these parameters: 1) the distance between neighbor and server must be lower than the distance between peer and server (in hop). 2) Difference between peer and its neighbor in contributed uploading bandwidth should be greater than a minimum value. (Upload bandwidth of peer must be greater than its neighbor). 3) If the playback point of peer and its neighbor have a difference, the difference between peer and its neighbor in contributed uploading bandwidth, must be greater than the value that the tradeoff line specifies.

For exchange places of peers in mesh structure we should be do suitable tradeoff between the difference between peer and its neighbor in contributed uploading bandwidth on one hand and the difference between playback points of peer and its neighbor on other hand. That is why we define a line and trade off on the slope of this line. This line distinct the minimum difference between peer and its neighbor in contributed uploading bandwidth as function of the difference between play back points of peer and its neighbor. As it can be seen in figure 1, if the playback point of peer is precedes its neighbor, (and upload bandwidth of peer is also greater enough than its neighbor) it can select this neighbor as a candidate. But if the playback point of peer's neighbor is precedes it (that usually it is happened), to select this neighbor as a candidate, the difference between peer and its neighbor in contributed uploading bandwidth (upload bandwidth of peer must be greater than its neighbor) must be greater than the value that the line specifies. In figure 1, the difference between contributed upload bandwidth (Do CUB) is calculated from equation 1, and the difference of playback points (Do PP) is calculated from equation 2. If the point of (Do CUB, Do PP) of peer and its neighbor is located in the gray area of figure 1, then exchange places with this neighbors, is permitted, otherwise would be non-permitted.

$$\text{Do CUB} = \text{CUB}_{peer} - \text{CUB}_{neighbor} \quad \text{(Kbps)} \qquad (1)$$

$$\text{Do PP} = \text{PP}_{neighbor} - \text{PP}_{peer} \quad \text{(second)} \qquad (2)$$

For example in figure 1, both black and white points have same difference between upload bandwidth, but the difference between playback points of black point, is greater than the threshold specified by the line, therefore cannot carry out exchange places for this point.

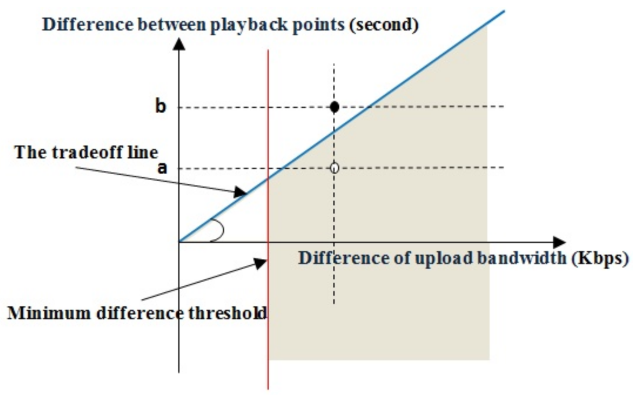


Figure 1. The tradeofflinethat uses in our mechanism

If there are more than one candidate neighbors, then peer must select one of them and then start interchanging neighbors operation. If there is not any candidate neighbor, it means that this peer has not priority to get closer to the servers. Finding candidate neighbors is repeated periodically, but when a neighbor was selected, and then an interchanging neighbors operation started, finding candidate neighbors stopped and it would not run until the end of this operation, and when an interchanging neighbors operation ended, then finding new candidate neighbors started again.

2.2.2. Selecting the Best Neighbor Among Candidate Neighbors

When a peer finds more than one neighbor as candidate neighbor, it must select the best of them for interchanging neighbors operation. For this purpose, each peer uses scoring to its candidate neighbors. This score calculate according to equation 3.

$$\text{Score} = \text{Do CUB} - \text{Do PP} \qquad (3)$$

Finally the neighbor with the highest score can be selected as the best candidate for interchanging neighbors operation.

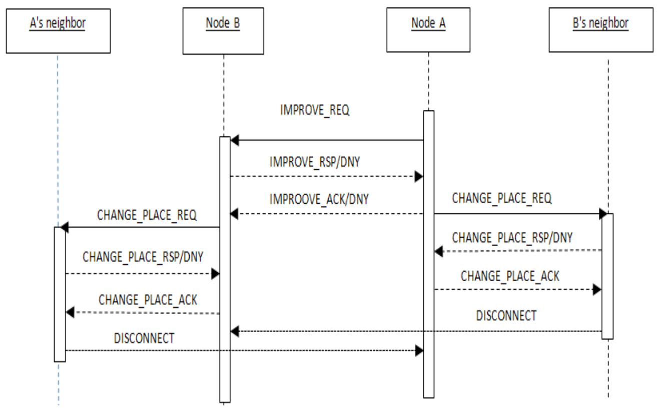


Figure 2. interchanging neighbors operation

2.2.3. Interchanging Neighbors Operation

As it can be seen in figure 2, after peer *a* selecting one neighbor like *B* to exchange places in the mesh, peer *A* sends *improvement request* to *B*. When *B* receives the message from *A*, if there isn't any error with the request message (requester must be neighbor of it and etc.) *B* has to send a positive response to request message, except following conditions:

1- When peer *B* does an improvement process with one of its neighbors, before this process ends, it doesn't start another process with others, and doesn't accept any request from others to start improvement process.

2- When peer *B* interacts with the video such as jump forward, before completing interaction operation it doesn't start and doesn't accept request to start any improvement process. (Especially if use clustering topologies, jump operation may change the cluster of peer).

If Peer *B* agrees to *improvement request*, then it sends improvement response message to *A* and with this message it sends list of all its neighbors. But if peer *B* doesn't have the conditions of accepting request, it sends *improvement deny* message to *A*. If peer a receives *improvement deny* message, then it terminates this interchanging neighbors operation (and improvement process) and starts new improvement process again. If peer *A* receives *improvement response* message, then it sends *acknowledgement* message to *B* and with this message it sends list of all its neighbors, but if there are errors or change conditions (such as *A* jumping forward) it sends *improvement* deny message to *B*. after *A* sends *acknowledgement* message to *B*, it can send *change place request* message to neighbors of *B*. also when *B* receives *acknowledgement* message, from *A*, it can sends *change place request* message to neighbors of *A*, but if it receives *improvement eny* message, it terminates this interchanging neighbors operation.

In our mechanism in order to specify the end of one successful improvement process and manage improvement processes, we define a *period time* parameter. This parameter determines the time between the two improvement processes for a peer. So if a peer wants to start new improvement process, from the beginning of previous process, more time than an improvement period, must be passed. This time starts after receiving *improvement response* message (for peer *A* in above) and receiving g*acknowledgement* message (for peer *B* in above).

When from start of an improvement process more time than an improvement period, passed, peer can start new improvement process, and also can accept to an improvement request. When receive a *change placer quest* message, neighbors of two side of improvement process (for example neighbors of *A*), disconnect from previous neighbor (disconnect from A) by sending *disconnect* message, and connect to other side of improvement process by sending *change place response* message (connect to *B*).

After receiving *change place response* message, each of peers *A* and *B* (two sides of improvement) add sender to the list of their neighbors and send *acknowledgement* message to it. Thus an improvement process reaches the end and after ending their *period time*, peers *A* and *B* can be ready for the next improvement process. We also notice to the following points in our mechanism:

1- In the above steps, peers may interact with video such as jump forward or exit from system, so peers in each of the above steps should consider these problems.

2- Peers that receive change place request message, in two conditions accepted this request: 1) If the other side of improvement process is one of its neighbors 2) Else if it has empty connection and can accept new neighbor.

3. Performance Evaluation

For performance evaluation of the improvement mechanism, we simulate a complete pull-based Vo D system. We evaluate quality of service parameters such as discontinuity and distortion in the case of using and don't using our improvement mechanism at differences condition of system and also study the incentive effect of mechanism. In addition it, we also try to find trade off points for screw parameter such as slope of line (in figure1) and the improvement period time.

3.1. Simulation Methodology

To perform evaluation of our mechanism through network simulation, the OMNeT++ v.4 [19] with INET framework [20] for simulating TCP/IP networks in OMNET++ and Over Sim [21] for simulating P2P systems in OMNeT++ are used. Physical topology was generated using Georgia Tech Internet Topology Model (GT-ITM) [22] tools with 28 AS (backbone router) and 28 access router per AS in top-down mode and with all drop-tail queues. For this simulation we designed a hybrid CDN-P2P structure for streaming Vo D As shown in figure 3, this system uses central tracker to create special mesh structure. In this structure peers are organized in clusters based on their playback point.

Each cluster has 2 direct links to the server. Peers can get neighbors only from one cluster. Since, peers can jump forward or backward on video. Peers arrival process is Poisson distribution with mean inter arrival time of 2 seconds. Each peer selects upload bandwidth with uniform distribution between 350 Kbps to 2 Mbps and downloads bandwidth between 1 to 4 Mbps and delays between 15 ms to 20 ms. For streaming video, neighbor peers exchange their buffer-map to each other (like Cool Streaming).

We use adaptive buffer-map exchange mechanism [23] to decrease bandwidth overhead. By using this mechanism, peers can buffer all video chunks and efficiently reduces bandwidth overhead. To balance the chunks request, each peer requests chunks from their neighbors with awareness to their upload ability. Because of randomness of P2P systems, all simulations are repeated 4 times for the Verbose ARD Talk sample video trace file from [24] and all the peer’s outputs are averaged for each scenario.

Table 1 shows the rest of our simulation parameters. The most important parameters in showing efficiency of our mechanism are quality of service parameters such as *discontinuity* and *distortion.*

3.2. Simulation Results

In this section we first try to find trade off quantities for slope of tradeoff line (in figure 1) and improvement period time. Then evaluate quality of service parameters in the case of using and not using our improvement mechanism at different conditions of system. At the end we study the incentive effect of proposed mechanism.

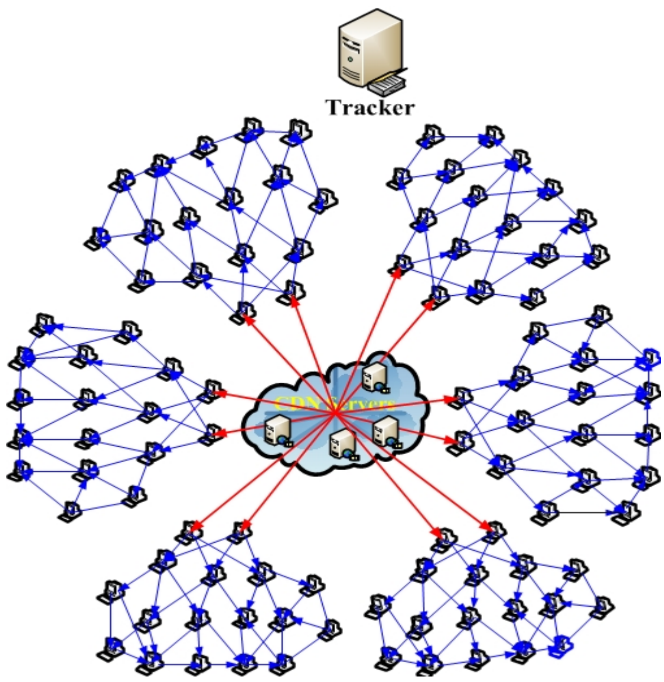
Table 1. Simulation parameters

Parameters	Value
Number of peers	200
Maximum packet size	1000 Bytes
Video codec	MPEG4 Part I
Video FPS	25
Number of frames in GoP	12 frames
Length of films	30 minitues
Sturtup buffering	12 s (300 frame)
Average video bit rate	512 Kbps
Chunk size	1 frame
W (in equation 3)	0.5
Simulation duration	800 second
Number of CDN servers	4
Mean neighbor size of each peer	6
CDN servers upload bandwidth	10 Mbps
Size of each cluster	15 peer

3.2.1. Finding Trade off Values for Parameters

To find tradeoff points, we change values of slope of tradeoff line and improvement period time then compare the results.

The best result has the lowest discontinuity, and distortion. In this set of experiments, peers remained in the system once they had joined, and don't have any interactive action with the video. In figure 4 slope of line is variable, it can be seen that when slope of line increases from zero to 0.25; discontinuity and distortion become lower. But when slope of line increases up to more than 0.5; discontinuity and distortion increases with a gentle slope. It seems from figure 4 that the best values for slope of line are quantities between 0.25 and 0.5.



Figuer 3. Schema of designed system.

In figure 5 Improvement Period Time is variable, it can be seen that when Improvement Period Time increases from 5 second to 35 second, discontinuity and distortion doesn't change much, but when improvement period time increases up to more than 35, discontinuity and distortion increase.

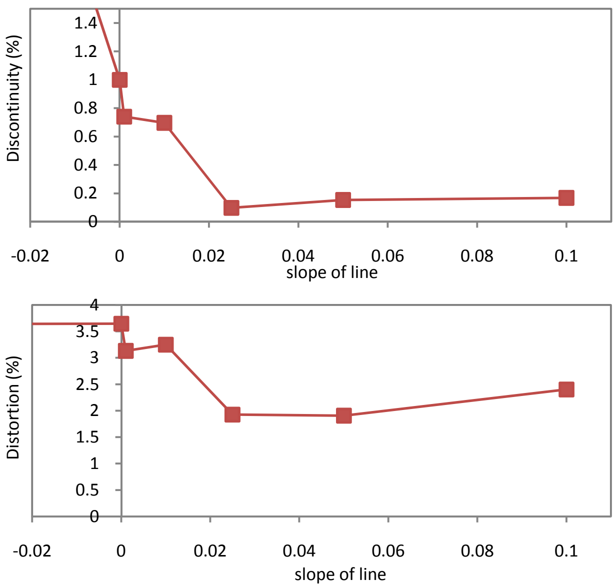


Figure 4. Discontinuity and distortion as function of the slope of tradeoff line

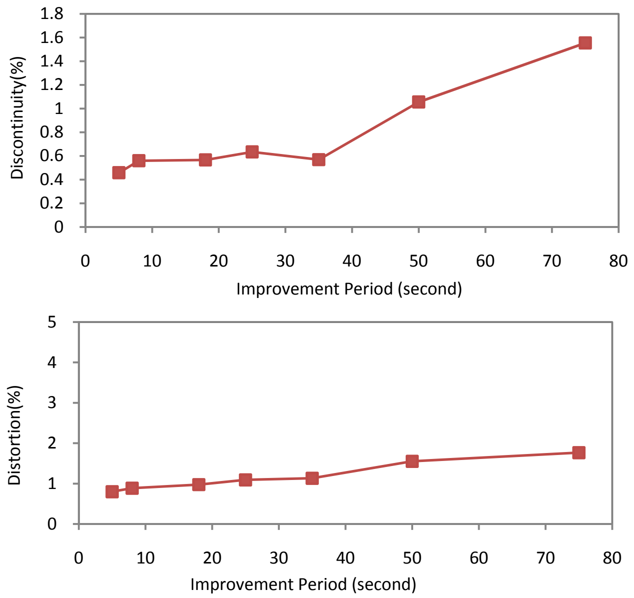


Figure 5. Discontinuity and distortion as function of Improvement Period Time

It seems from figure 5 that the best values for Improvement Period Time are quantities below 35 seconds. Also we know if lower quantities for Improvement Period Time are selected, the overhead of mechanism increases, but figures 5 shows that the overhead of our improvement mechanism doesn't much effect on quality of service parameters, even it runs every 5 seconds. However in selecting the value for this parameter, we must pay attention to round trip delay of messages, and also this depends on the condition of congestion in the network. It should select big enough that when an improvement process start, after it doesn't ended, other process doesn't start.

3.2.2. Performance Evaluation in Stable Environment

In this set of experiments, all of the peers remained in the system once they had joined, and don't have any interactive action such as jump forward and played the video continuously from the beginning to the end. In this section we change the number of peers in the system from 50 to 250 and evaluate quality of service parameters in the case of using and not using the improvement mechanism.

In figure 6, we show the evaluation of quality of service parameters as function of the number of peers. It can be seen that by using our mechanism, we can achieved very significant improvements in quality of service parameters.

3.2.3. Performance Evaluation in Dynamic Environment

A further series of experiments was performed to investigate the robustness of the improvement mechanism to dynamic peer interactions such as jumping forward and exit. In the first 200 second of the experiment, all of the peers performed playback only. From the 200th second to 800th second and with uniform distribution, in the first experiment between 0% and 90% of the nodes were randomly a jump forward, and in the second experiment between 0% and 60% of the nodes were randomly exit. The average discontinuity and distortion of the all peers were measured all along the experiments.

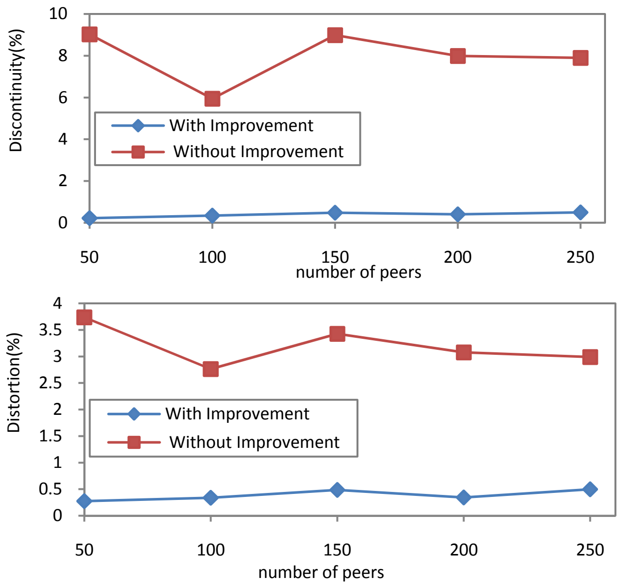


Figure 6. Discontinuity and distortion as function of number of peers

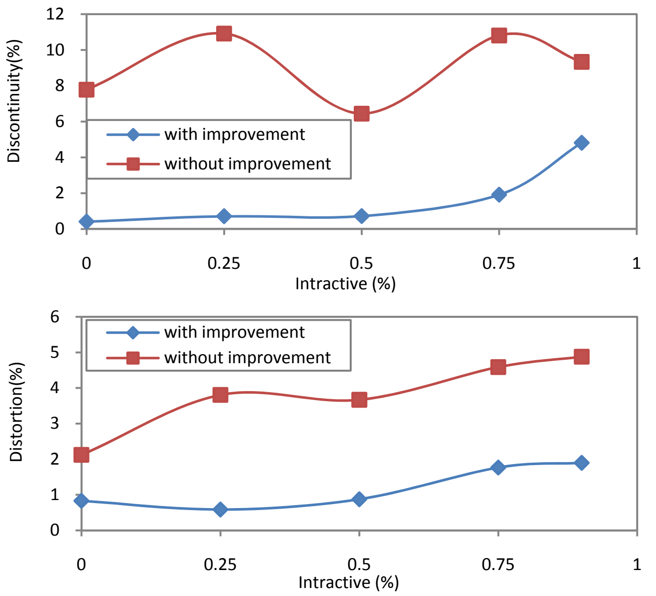


Figure 7. Discontinuity and distortion as function of interactive

In figure 7 and figure 8 the evaluation of quality of service parameters as function of the percent of interactive and exit peers are shown. It can be seen in figure 7 that by using our improvement mechanism, our system is robust. Although when the percentage of interactive peer is bigger than 70, quality of service reduces, but usually this percentage of interactive don’t happened.

It can be seen in figure 8 that our improvement mechanism, against the peers’ exiting is somewhat robust. And also the quality of service is much better than when improvement mechanism is not used.

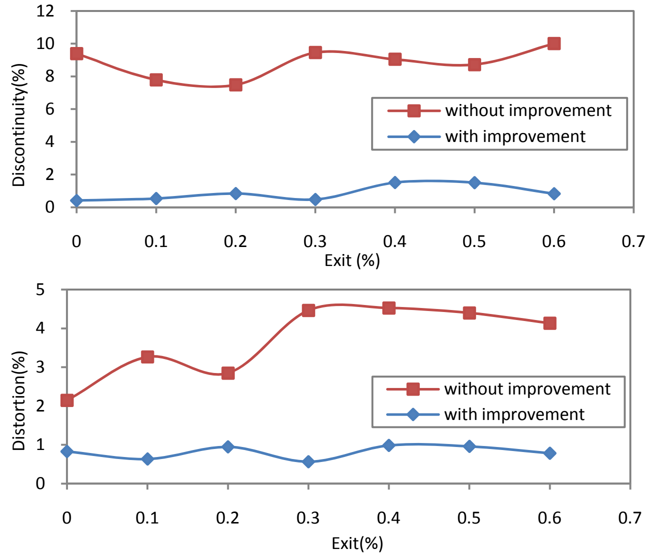


Figure 8. Discontinuity and distortion as function of percent of exit

3.2.4. Evaluation the Incentive Effect of Proposed Mechanism

Our improvement mechanism also can incentive the peers to more contribution in video streaming. To prove this statement it is enough to show that "by using our improvement mechanism, peers that share more upload bandwidth, experience better quality of service". For this purpose we classified results of peer's quality of service parameters based on their upload bandwidth, and then averaged each class and showed in figure 9 and10 as function of classes of upload bandwidth. In these figures shows this results in the case of using and not using the improvement mechanism. It can be seen that with use improvement mechanism, peers that share more upload bandwidth, experience better quality of service. So we can conclude that our mechanism can act as an incentive mechanism too.

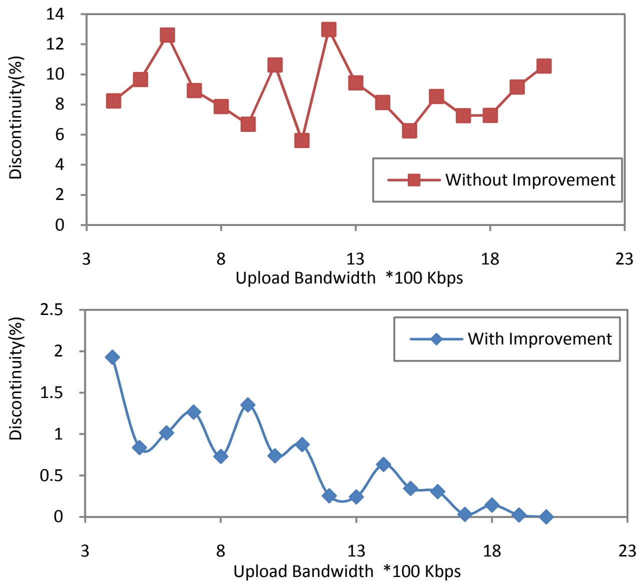


Figure 9. Discontinuity as function of peer’s shared uploads b and width

Not e that the shared upload bandwidth that is used is in this Figures is the maximum band Width that peer shared, and it is different from the bandwidth that consumed.

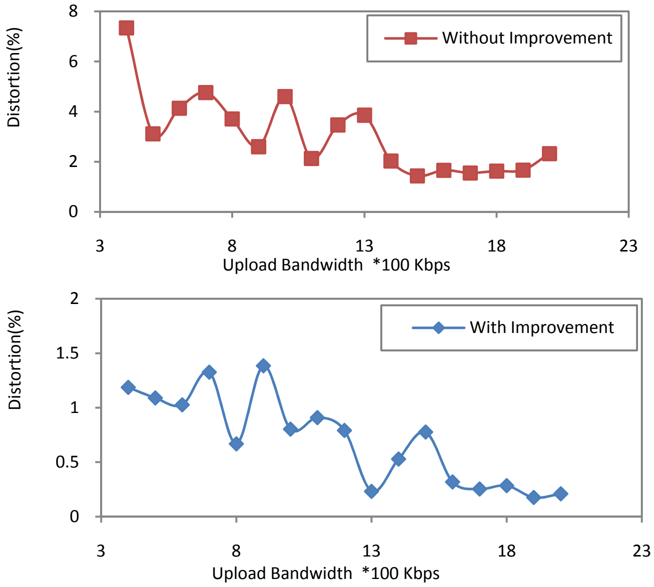


Figure 10. Distortion as function of peer's uploads b and width

4. Conclusion and Future Work

This paper has presented a distributed and continuous improvement mechanism for performing on-demand video. This mechanism which is applied to achieve optimum use of system resources, proposed a new method to locate the peers that they contribute more upload bandwidth, closer to the servers, it do this with regard to the playback point and shared upload bandwidth of peers. The servers are not involved in the mechanism and mechanism run only in peers. The results of performance evaluation of the proposed mechanism show that our mechanism efficiently enhanced the quality of service parameters, and the mechanism is robust in dynamic environments. The results also confirmed that the mechanism can acts as an incentive mechanism.

We are currently extending this work along several dimensions. First, we are evaluating our system performance in scenarios where the distribution of outgoing bandwidth is in the presence of free-riders and we are tuning our mechanism to punish free-riders. Second, we are working to enhance our overlay topology by comparison between performances of several topologies. Third, we also are studying to reduce the server stress of our system by trade-off between quality of service, and server stress.

References

[1] Y. Liu, Y. Geo, and C. Liang, "A Survey on Peer-to-Peer Video Streaming Systems," *Journal of Peer-to-Peer Network Application*, vol. 1, no. 1, pp. 18-28, 2008.

[2] Z. Sheny, J. Luo, R. Zimmermanny, and A. V. Vasilakosz, "Peer-to-Peer Media Streaming: Insights and New Developments," *Journal of Proceedings of the IEEE*, vol. 99, no. 12, pp. 2089-2109, 2011.

[3] N. Ramzan, H. Park, and E. Izquierdo, "Video Streaming over P2P Networks: Challenges and Opportunities," *Journal of Signal Processing and Image Communication*, vol. 27, no. 5, pp. 401-411, 2012.

[4] S. Seyyedi, and B. Akbari, "Hybrid CDN-P2P Architectures for Live Video Streaming: Comparative Study of Connected and Unconnected Meshes," *Proc. IEEE Int'l Symp. Computer Networks and Distributed Systems*, pp. 175-180, 2011.

[5] J. Jannotti, K. David, and et al., "Overcast: Reliable Multicasting with an Overlay Network," *Proc. IEEE Int'l Symp. Operating Systems Design and Implementation*, pp. 80-84, 2000.

[6] Y. Chu, G. Rao, and H. Zhang, "A Case for End System Multicast," *IEEE Trans. Selected Areas in Communications*, vol. 20, no. 8, pp. 1456-1471, 2002.

[7] M. Castro, P. Druschel, A. Kermarrec, M. Nandi, A. Rowstron, and A. Singh, "SplitStream: High-bandwidth Multicast in Cooperative Environments," *Proc. ACM Int'l Symp. Operating Systems Principles*, pp. 298-313, 2003.

[8] Y. Guo, K. Suh, J. Kurose, and D. Towsle, "P2Cast: Peer-to-Peer Patching for Video On-demand Service," *Journal of Multimedia Tools and Applications*, vol. 33, no. 2, pp. 109-129, 2007.

[9] Y. Shuai, C. Shieh, C. Lin, and S. Wang, "P2PVR: A Playback Offset Aware Multicast Tree for On-demand Streaming with VCR Function," *Journal of System Architecture*, vol. 57, no. 4, pp. 392-403, 2011.

[10] X. Zhang, L. Jiangchuan, L. Bo, and P. Yum, "Cool Streaming/DONet: A Data Driven Overlay Network for Live Media Streaming," *Proc. IEEE Int'l Conf. Computer Communications*, pp. 2102-2111, 2005.

[11] C. Dana, D. Li, D. Harrison, and C. N. Chuah, "Bass: Bit Torrent Assisted Streaming System for Video-on-demand," *Proc. IEEE Int'l Workshop on Multimedia Signal Processing*, pp. 1-4, 2005.

[12] X. Hei, C. Liang, J. Liang, Y. Liu, and K. W. Ross, "A Measurement Study of a Large-scale P2P IPTV System," *IEEE Trans. Multimedia*, vol. 9, no. 8, pp. 1672-1687, 2007.

[13] N. Magharei, and R. Rejaie, "PRIME: Peer-to-Peer Receiver-Driven Mesh-Based Streaming," *IEEE/ACM Trans. Networking*, vol. 17, no. 4, pp. 1052-1065, 2009.

[14] C. Y. Liu, K. Wang, and Y. L. Hsieh, "Efficient Push-Pull Based P2P Multi-streaming Using Application Level Multicast," *Proc. IEEE Int'l Symp. Personal Indoor and Mobile Radio Communications*, pp. 2586-2590, 2010.

[15] N. T. Jahromi, B. Akbari, and A. Movaghar, "A Hybrid Mesh-tree Peer-to-Peer Overlay Structure for Layered Video Streaming," *Proc. IEEE Int'l Symp. Telecommunications*, pp. 706-709, 2010.

[16] J. Luo, "Practical Algorithm for Minimum Delay Peer-to-Peer Media Streaming," *Proc. IEEE Int'l Conf. Multimedia and Expo*, pp. 986-991, 2010.

[17] J. Liu, S. G. Rao, B. Li, and H. Zhan, "Opportunities and Challenges of Peer-to-Peer Internet Video Broadcast," *Proc. IEEE Int'l Conf. Multimedia and Expo*, pp. 11-24, 2008.

[18] S. Kang, and H. Yin, "A Hybrid CDN-P2P System for Video-on-demand," *Proc. IEEE Int'l Conf. Future Networks*, pp. 309-313, 2010.

[19] "OMNeT++ Network Simulation Framework," http://www.omnetpp.org/, May 2012.

[20] I. Vari, "INET Framework for OMNeT++," http://inet.omnetpp.org, June 2012.

[21] I. Baumgart, B. Heep, and S. Krause, "Over Sim: A Scalable and Flexible Overlay Framework for Simulation and Real Network Applications," *Proc. IEEE Int'l Conf. Peer-to-Peer Computing*, pp. 87-88, 2009.

[22] "GT-ITM Topologies for the OMNeT++ Simulation Platform and Over Sim Framework," http://mm.aueb.gr/research/GTITM-OMNeT4/, December 2009.

[23] A. Ghaffari, and B. Akbari, "A Novel Buffer-map Exchange Mechanism for Pull-based P2P VoD Streaming Systems," *Proc. IEEE Int'l Symp. Telecommunications*, pp. 62-66, 2014.

[24] "Arizona State University Video Trace Library," http://trace.eas.asu.edu/TRACE/pics/FrameTrace/mp4/Verbose_ARDTalk.dat, October 2010.

Behzad Akbari received his B. S., M. S. and Ph D degree in Computer Engineering from Sharif University of Technology, Tehran, Iran, in 1999, 2002 and 2007, respectively. He joined, as an assistant professor, to department of Electrical and Computer Engineering at Tarbiat Modares University, Tehran, Iran in 2007. His main research interests are Internet QoS, media streaming over the Internet, peer-to-peer networks, peer-to-peer video streaming, wireless video communications, data center networking, software defined networks (SDN) and network performance modeling and evaluation.

E-mail: b.akbari@modares.ac.ir

Abdollah Ghaffari Sheshjavai received his B. S. and M.S. degree in Computer Engineering from Air University of Science and Technology (Shahid Sattari) and Tarbiat Modares University, Tehran, Iran, in 2008and 2013, respectively. His main research interests are P2P networks, P2P video streaming and Internet QoS. He currently serves as a researcher in Multimedia Networking Laboratory (MNL) at Tarbiat Modares University.

E-mail: abdollah.ghaffari@modares.ac.ir

Paper Handling Data:

Submitted: 30.05.2014
Received in revised form: 28.09.2014
Accepted: 14.10.2014
Corresponding author: Dr. Behzad Akbari,
Department of Electrical and Computer Engineering, Tarbiat Modares University, Tehran, Iran.