IJICTR
International Journal of Information & Communication Technology Research

Volume 10- Number 1 – Winter 2018 (13-25)

An Incentive Mechanism to Mitigate the Free-riding in VoD Streaming over Hybrid CDN-P2P Networks

Abdollah Ghaffari Sheshjavani
Department of Electrical and Computer Engineering
Tarbiat Modares University
Tehran, Iran
abdollah.ghaffari@modares.ac.ir

Behzad Akbari*
Department of Electrical and Computer Engineering
Tarbiat Modares University
Tehran, Iran
b.akbari@modares.ac.ir

Hamid Reza Ghaeini
Pillar of ISTD
SUTD, Singapore, 487372
Ghaeini@acm.org

Received: 29 August, 2017 - Accepted: 5 February, 2018

Abstract— **Peer organization is one of the most challenging issues in peer-to-peer (P2P) video streaming systems. Free-riding reduces the overall performance of these systems. The peers placed closer to the video source will have a higher quality of service. Therefore, we can mitigate the free-riding effect and encourage peers to increase uploading participation by putting peers with more upload bandwidth closer to the video sources and setting free riders far from these video sources. In this paper, we propose a distributed incentive mechanism, which extensively enhances mesh-based P2P video-on-demand streaming systems. In our proposed mechanism, peers will organize an overlay based on their playback point and contributed upload bandwidth. This mechanism applies to both P2P, and hybrid CDN-P2P approaches. The obtained results demonstrate the efficiency of our proposal in term of the quality of service, robustness in dynamic environments and resiliency against free riding.**

Keywords- video-on-demand, peer-to-peer, improvement Mechanism, incentive mechanism, Video streaming

I. INTRODUCTION

Nowadays, video data consumes most of the Internet traffic. Video streaming applications can be classified into two categories: live and on-demand. In a live streaming session, video content disseminated to all users in real-time. The video playback on all users is synchronized. On the contrary, video-on-demand (VoD) users enjoy the flexibility of watching whatever video clips whenever they want. The playback of the same video clip on different users is not synchronized. Therefore, the available peer resources should always be more versatile in VoD. The basic solution for streaming video over the Internet is the client-server model and replicating videos on the several servers. This approach is called content distribution network (CDN). However, increasing the interest of watching video-over-IP and the cost of providing the required bandwidth for a large number of users in Client-Server model and also the availability of the resources in

* Corresponding Author

clients increased the attention to P2P and hybrid CDN-P2P approaches. The P2P approach can provide cost-effective large-scale streaming, while CDN approaches require a high cost of the infrastructure to support video streaming with desired performance. The hybrid CDN-P2P approach tries to take advantages of both CDN and P2P models. These systems have attracted much more attention from the researchers due to its ability to improve the quality of the delivered video in P2P networks and provide system scalability in CDNs. In P2P and hybrid CDN-P2P approaches, clients are organized in an overlay network. In these systems, the way of organizing peers in an overlay is one of the most important elements in optimum usage of system resources. There are three main overlay structures [1-3]: Tree-based, Mesh-based and Hybrid.

Tree-based systems [4, 5] have well-organized structures and usually spread video contents by actively pushing data from a peer to its children. Within a stable streaming tree, the delay is strictly bounded, but the complexity of maintaining a stable tree is high. In mesh-based approach, peers join the overlay network by selecting some peers as neighbors. In mesh-based P2P systems [6-8], video (or files) are divided into some pieces named chunk. Overlay peers transfer chunks to each other usually by requesting (pull) missing chunks from their neighbors. The main difference between file sharing and video streaming systems is that chunks in video streaming have a dead time. These systems are very robust against peer churn, due to the randomness embedded in the peering procedure and the high peering degree. However, the main disadvantages of mesh-based topologies are:

1. The overhead (that comes from the periodical exchange of buffer-maps and status messages among the peers).
2. The sub-optimal peer organization and neighbor selection.

The sub-optimal peer organization may waste bandwidth to some extent. To solve the first problem, in [9] we propose adaptive buffer-map exchange mechanism. This mechanism decreases bandwidth overhead via sending only a portion of the buffer-map concerning the playback point of each neighbor. In this paper, we address the second problem and try to improve peer organization.

Various methods of peer organization of each type of P2P streaming systems proposed to enhance the quality of service and better use of the system resources. Some of video-on-demand (VoD) systems like P2Cast [4] and P2PVR [5] consider the playback offsets of peers, and some other systems like [10] consider the peer's spent time in the system. Some works such as [11] try to enhance structure based on establishing neighborhood relationship according to peer's upload capacity. In this manner, the number of children for each node depends on its outgoing link capacity. In live video streaming systems, some works such as mTreebone [12] and GLive [13] try to optimize overlay with better organizing peers in the overlay. Peer's spent time in the system and their upload capacity are considered to optimize overlay in mTreebone. In this system, peers that have more upload capacity have more children too. To reduce delay (by reducing the depth of treebone), peers with more children located closer to the source. In GLive, upload capacity is the main parameter to organize peers in the overlay. In this system, also peers with higher upload bandwidth are located closer to the media source. Authors in [14] also consider the peer's upload capacity. By performing some theoretical studies they conclude that to achieve the best quality of service and optimum usage of peer's resources, the peers with better uploading ability must be located closer to the CDN servers. Furthermore, the authors in [15] studied the effects of free riders with a probabilistic approach. They concluded that closer nodes to the source are less likely to be blocked. Thus, we can get an incentive mechanism, which is to get peers with more contributions closer to the source while putting free riders far from the servers. If free riders can be judged and put in leaves or farther from the source, then the damage would be minimal. Also, the authors of [14], and [15] show that placing peers closer to the streaming servers based on their upload abilities not only can reduce the effects of free riding, but also encourage peers to contribute more resources and improve the overall quality of service. However, these studies did not implement a comprehensive method to solve these issues.

Three main questions that this paper tries to answer are:

- How could we locate peers with better upload capacity closer to the servers by considering the fact that the network conditions and peer capabilities might be changed?
- Does the peer upload capability is the most important factor in choosing the peers in video-on-demand streaming systems to be close to the servers?
- How are the side effects of this method on the overall performance of the overlay?

To address these questions, in this paper, we proposed an entirely distributed incentive mechanism for mesh-based P2P VoD streaming which can be used in both P2P, and hybrid CDN-P2P approaches. This mechanism optimizes the resource consumption by placing peers with better uploading capacity closer to the servers while setting free-riders far away from the sources. Also, this mechanism encourages the peers to contribute more, by providing the better quality of service for peers with better uploading abilities. The results of our simulation demonstrate the soundness and completeness of proposed mechanism in providing the better quality-of-service, usage of peer resources and network bottleneck reduction.

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

II. RELATED WORK

The works on hybrid CDN-P2P in the literature categorized into three main groups:

1. Some works such as [16-18] proposed CDN-P2P architectures.

2. Some works such as [19] and [20] proposed optimization methods to deliver contents appropriately in these architects.
3. The measurement works such as [21] that evaluate CDN-P2P systems.

In [16] authors propose two mesh-based architectures (connected and unconnected) for hybrid CDN-P2P live video streaming. They conclude in this paper that the QoS of CDN-P2P connected mesh architecture has is better than both CDN-P2P unconnected mesh and pure P2P mesh architectures. In [19] authors proposed an economic replica placement mechanism to offer Hybrid CDN-P2P streaming service by using traditional CDNs to optimize the number and places of replicas for P2P service based on the economic model. The result shows this mechanism can yield the maximum net profit.

Authors in [21] conducted a measurement study on a hybrid CDN-P2P VoD streaming service provider in China. According to the results of this work, the video contents stored in peers update quite slowly and the average lifetime of cached videos is longer than one week. The results also show that a large-scale VoD streaming service with a small-scale fixed infrastructure could be provided by utilizing the slow-varying contents cached in peers and deploying some enhancement mechanisms to the CDN.

In P2P approaches and P2P section of CDN-P2P approaches, clients are organized in an overlay network. P2Cast [4] and P2PVR [5] are tree-based systems. P2PVR sorts all of the peers into a list by 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 placed toward the end of the list.

CoolStreaming/DONet [6], UR-aware [7] and PPLive[8] are mesh-based approaches for P2P video streaming. The original CoolStreaming had a BitTorrent-like content discovery mechanism. The quality of Service of mesh-based systems can be improved by using adaptive content and deadline aware chunks scheduling [22]. This method attempts to request frames with the highest priority from peers which can deliver them in a shorter time.

Some of systems such as PRIME [23] use a combination of pull and push based approaches. 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, some works such as [10], [24] and [25] proposed hybrid tree-mesh structures. In general, most proposals since then take a mesh-based pull system as the basis; their common goal is to “distil” push trees out of the mesh structure and to optimize the performance of these trees.

There are many studies about free riders and incentive mechanisms for P2P video streaming. These studies could be classified in two different areas, which are micro-currency and quality of service (QoS). In micro-currency methods such as [26, 27] peers pay virtual money when downloading and get money when uploading. In QoS-based methods [28-30], the system tries to provide peers with different contribution or credits with different quality of service. In these systems, peers can receive better service if they contribute more. Reciprocity-based (direct and indirect) schemes are QoS-based methods too. In these schemes, peers refer to the histories of reciprocal peer behavior during their decision-making processes. A typical example for direct reciprocity is the tit-for-tat incentive method used in BitTorrent-like applications. According to reputation differentiation, a reputable peer rewarded better service than a disreputable peer. Some works such as PRIME [23] and [31] using taxation as a peer-incentive mechanism. Taxation based incentive mechanism offers a flexible framework that allows the tradeoff between individual peer's fairness/welfare and the system-wide social welfare.

III. THE PROPOSED MECHANISM

A. *Problem analysis*

In this paper, the main incentive principle is to improve the quality-of-service of the peers that have more upload capacity and locating these peers closer to the video servers. In VoD streaming in addition to uploading capacity and contribution of peers, we also need attention to playback point of peers. For better understanding the mechanism, assume having two peers named A and B that peer A has higher upload bandwidth than peer B, (A has more potential for contribution than B), and peer B is closer to server (Fig.1), 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. The probability of this event is increasing with an increase of the distance between playback points of A and B. However, when this distance is not significant, the probability that peers experience discontinuity will be lower than the previous scenario. Therefore, if we want to locate peers that have more upload capacity, closer to the servers, we should tradeoff between the upload bandwidth capacity and their playback points.

B. *The improvement mechanism*

In this part, we explain our improvement mechanism. The primary point of this mechanism is that neighbor peers exchange the required information among themselves, such as playback point, share upload bandwidth and hop distance to the server. Hence each peer could compare its information with its neighbors. Assume peer A is a neighbor of peer B. If A shared more upload bandwidth than B (large enough) and B located in lower hops to the server and playback difference of them was not large (small enough), A request B to exchange their neighboring information. Then A and B change their neighbors with prior approval of B. This process is repeating periodically and uninterruptedly until locating the peers with higher upload bandwidth closer to servers. Especially when the network has churned, this mechanism continuously repairs the negative effects of churn (by continuous trying to locate more contributing peers close to servers all the time). This mechanism does not create any overhead for the servers.

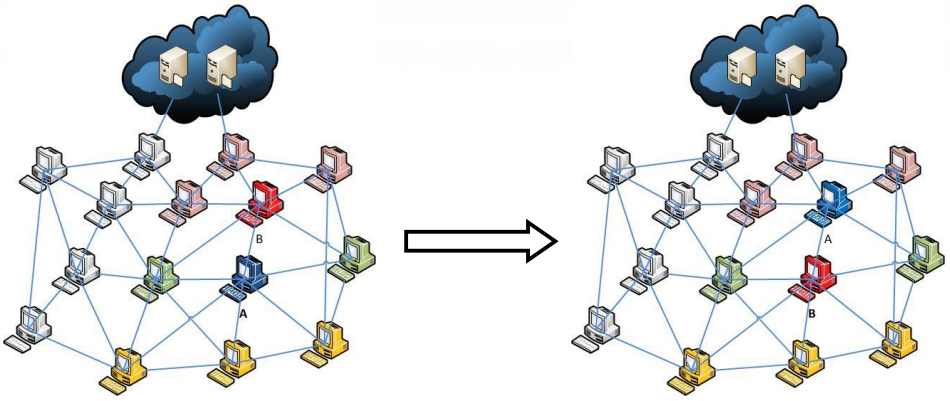


Fig. 1. Improvement process(*A* and *B* changed their neighbors)

The steps of improvement process of our mechanism are:

- measuring the distance from the servers
- Finding candidate neighbors to exchange places.
- Selecting the best neighbor from candidate neighbors.
- Interchanging neighbors operation.
- Churn handling.

1) Measuring distance from the servers

To evaluate distance of peers to the servers we use hop count (*HC*) feature of video chunks (the number of peers that exist in the path to the servers). The distance of peers calculated as following: each peer keeps a variable named *Min Hop Count (MHC) = -1*. When a peer receives a chunk, it increments one to the *HC* of this chunk and calculates *MHC* as follows:
If (*MHC* = -1) **then set** *MHC* = *HC* of received chunk.
Else *MHC* =**minimum** (*MHC*, *HC* of received chunk).

Peers periodically set their distance as *MHC* and reset *MHC* to (-1). When peers do interactive with video, it will reset their distance and *MHC* to (-1). Finally, each peer periodically sends its distance and other required information with buffer-map exchange message to its neighbors.

2) Finding candidate neighbors to exchange neighbors

Each peer periodically sends status information with a buffer-map message to all its neighbors, includes distance from the server (in hop), shared upload bandwidth and playback point. This mechanism designed for mesh-pull-based VoD systems. Hence, peers send information with the buffer-map message to reduce overhead. After it, each peer periodically compares its status information to its neighbors. To find candidate neighbors to exchange places, each peer considers the following parameters:

- The candidate neighbor must be located closer to the server (in hop).
- The difference between peer and its neighbor in uploading bandwidth should be greater than a minimum value.
- If the playback points of peer and its neighbor have a difference, the difference between peer and its neighbor in uploading bandwidth must be greater than the value that the tradeoff line specifies.

To exchange places of peers in mesh structure, we should tradeoff between the difference between peer and its neighbor in shared uploading bandwidth on the one hand and the difference between playback points of peer and its neighbor on the other hand. That is why we define a line and trade off on the slope of this line. This line denotes the minimum difference between peer and its neighbor in shared uploading bandwidth as a function of the difference between playback points of peer and its neighbor. As can be seen in Fig. 2, if the playback point of a peer like A proceeds each of its neighbors like B, (and upload bandwidth of A is also greater enough than B, and B is closer to servers than A), A can select B as a candidate. But if the playback point of B precedes A (that is usually the case), to select B as a candidate, the difference between A and B in shared uploading bandwidth should be greater than the value that the line specifies (upload bandwidth of A must be greater than B).

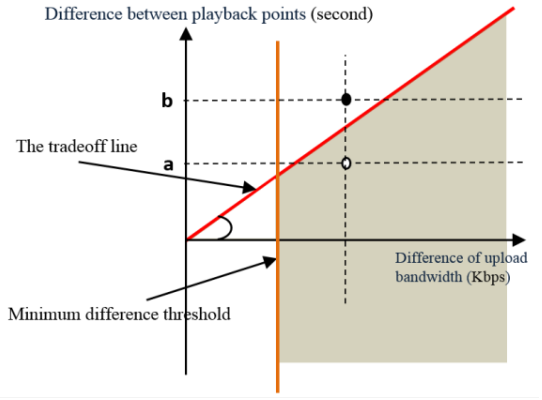


Fig. 2. The tradeoff line that uses in our mechanism

As can be seen in Fig. 2 the tradeoff line distinct the minimum difference between peer and its neighbor in shared uploading bandwidth as a function of the difference between playback points of them. In this figure in the gray area (such as white point) then exchange places with this neighbor permitted, otherwise (such as black point) does not permitted. If there is not any candidate neighbor, it means that this peer has not the priority to get closer to the servers. Finding candidate neighbors is repeating while a neighbor selected and the interchanging neighbors

operation will start and finding candidate neighbors will stop and will not continue to find until the end of this operation. When an interchanging neighbor operation finished, finding new candidate neighbors will start again.

In this work, we use capacity or shared upload bandwidth instead of contribution upload bandwidth because first, it is straightforward and, second, if a peer has a good upload capacity, but located far from the servers; it may not use this capacity because of data bottleneck and distance from servers.

3) Selecting the best neighbor among candidates

When a peer finds more than one candidate neighbor, it must choose the best of them for interchanging neighbors operation. For this purpose, peers use scoring according to equation 1, to sort candidate neighbors in a list:

$$Score = upload\ BW.diff. - playback\ diff \quad (1)$$

If a peer was not a free rider, but it has a neighbor that was a free rider (free rider has low contributed upload bandwidth such as below 100 Kbps), then peer in above step for this neighbor do not pay attentions to the difference of playback points. Finally, the neighbor with the highest score will be selected as the best candidate for interchanging neighbors' operation. If the peer does not start interchanging neighbors' operation with this neighbor, then the next neighbor in the list will be selected and this process will continue.

4) Interchanging neighbors operation

As can be seen in Fig. 3, after selecting one (the best) neighbor like *B* to exchange places, peer *A* sends *improvement request* to *B*. When *B* receives the request message, except the following conditions, it sends a positive response to *A*:

- When *B* does an improvement process with one of its neighbors, it does not start another process with others before the previous process ends.
- When peer *B* interacts with the video such as jump forward, before completing interaction operation it does not start and does not accept any improvement process.

If peer *B* agrees to *improvement request*, then it sends improvement response message to *A*, and with this message it sends a list of all its neighbors. Nevertheless, if peer *B* does not have the conditions of accepting the request, it sends the *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 an *acknowledgment* message to *B*. with this message, it sends a list of all its neighbors. Nevertheless, it sends the *improvement* deny message to *B* if there are errors or changing conditions (such as *A* does jumping forward). After *A* sends an *acknowledgment* message to *B*, it can send a *change place request* message to the neighbors of *B*. Furthermore, when *B* receives an *acknowledgment* message from *A*, it can send *change place request* message to the neighbors of *A*, but if it receives *improvement deny* message, it terminates this interchanging neighbors operation.

When the neighbors of A and B receive the place changing request message, they disconnect from their former neighbor (A or B), and connect to another side of improvement process (B or A) by sending place changing response message. Thus, an improvement process reaches at the end, and peer A and peer B can be ready for the next improvement process. In our mechanism to specify the end of one successful improvement process and manage them, we define improvement period parameter. This parameter determines the time between the two improvement processes for a peer. This time starts after receiving improvement response message (for peer A) and acknowledgment message (for peer B). When from the start of an improvement process more than one improvement period slot passed, the peer can start new improvement process and can accept an improvement request. To reduce failure, we assume that control messages use reliable communication channel. Note that all above steps are performing by the system and we assume that all of the peers are honest. Therefore, users do not interfere with these steps and do not refuse to change places or send some fake information about their upload bandwidth and distance from servers.

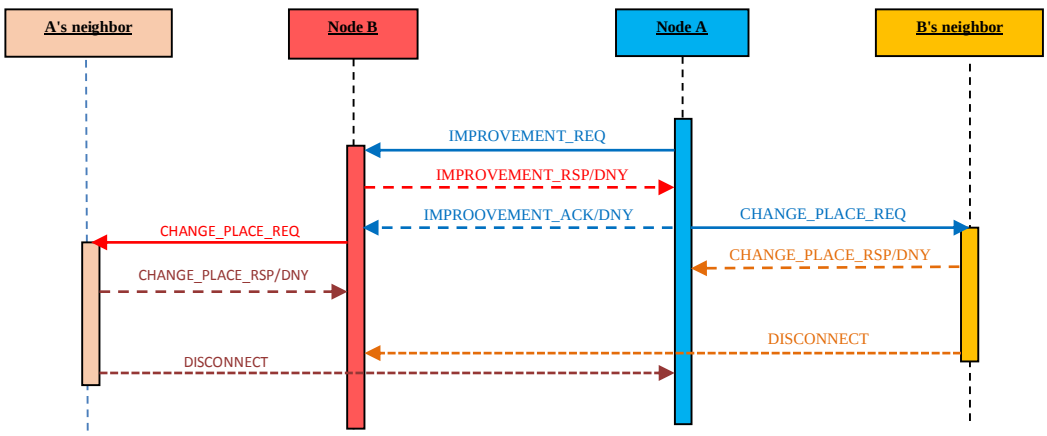


Fig. 3. interchanging neighbors operation

5) Churn handling

In the above steps, peers may interact with video such as jump forward or exit from the system. Therefore, in each of the above steps, we should consider these problems. In some steps, at the end of the replacing process optimal solution founded and we do not need to continue the process. However, in some cases like interchanging neighbor operation, exiting one side of the improvement process, may increase the number of neighbors of the other neighbor. In this condition, rollback operation is not possible. Because if one side of improvement process fails due to power failure, network failure, system failure, or such like, it cannot inform others. Therefore, the other side of improvement process cannot judge the status of it. Besides disconnecting neighborhood relationship to previous neighbors by each side of improvement operation also has some challenges. Therefore, for solving this problem, we use an exit management method that proposed in [32]. This method can prune weak connection in the overlay. If in an improvement process the number of neighbors of a peer becomes very high, then the response time of this peer will increase. Therefore, its neighbors evaluate the peer (and its connection) as a weak (or exited) peer and some of them prune their connections to this peer. The method is as follows: If a peer does not receive any packet from one of its neighbors during a given threshold time, then it disconnects the neighborhood relationship to this neighbor. Note that in pull-base streaming systems, neighbor's peers exchange buffer-map message to each other periodically; so we do not need to send other messages such as alive messages.

IV. Performance Evaluation

For performance evaluation of the improvement mechanism, we simulate a complete pull-based VoD system. We evaluate the quality of service parameters such as discontinuity and distortion in the case of using, and not using our mechanism at the different conditions of the system, and study the incentive effect of mechanism. In addition, we also try to find a trade-off points for mechanism parameters such as the slope of the line (in Fig.2) and the improvement period.

A. Simulation Methodology

To perform the evaluation of our mechanism through network simulation, the OMNeT++ v.4 [33] is used. OMNeT++ is a modular and discrete event simulator used for simulating communication networks. The INET Framework [34] used for simulating TCP/IP networks in OMNET++. INET framework implements UDP, IP, and Data Link Layer in OMNeT++. OverSim [35] is a framework in OMNeT++ for simulating P2P systems. OverSim Uses INET framework to simulate underlay layers. For this simulation, we designed CMPVoD [17] a new hybrid CDN-P2P architecture for streaming VoD. As shown in Fig.4, this system uses the central tracker to create special mesh structure. New joining peer assigned to the last cluster. However, when the number of peers in the last cluster is greater than cluster size parameter, the tracker will create a new cluster. Interactive peers will assign to a cluster based on their new playback point. Each cluster has two direct links to the CDN server. Peers arrival process is Poisson distribution with a mean inter-arrival time of 2 seconds. Each peer selects upload bandwidth with a uniform distribution between 350 Kbps to 2 Mbps and downloads bandwidth between 1 to 4 Mbps. We use adaptive buffer-map exchange mechanism [9] to decrease bandwidth overhead. To balance the chunks request, each peer requests chunks from their neighbors by considering their upload capability. In this system, we try to control network congestion by reducing uselessly consumed bandwidth without reducing the quality of service. To this end when peers have more than 12 seconds of video that are not played in their buffer, they are not allowed to download video more than the bit rate of the video. However, when a peer has not enough chunks of video in its buffer to play, to reduce startup delay, this mechanism allows the peer to exceed download rate up to triple of video bit rate. Because of randomness of P2P systems, all simulations repeated 4 times for the Verbose_ARDTalk sample video trace file from [36] and all the peer's outputs averaged for each scenario. Table 1 shows the rest of our simulation parameters.

The most important parameters of our study are server stress, overhead and QoS parameters, such as discontinuity, distortion, startup delay, and seek delay. We will describe each parameter as follows.

- *Discontinuity:* the total time that peer experiences discontinuity playback is divided into total time that peer has spent to watch the video. When a peer does not have 80 percent of the next second of video, its pause and start buffering.

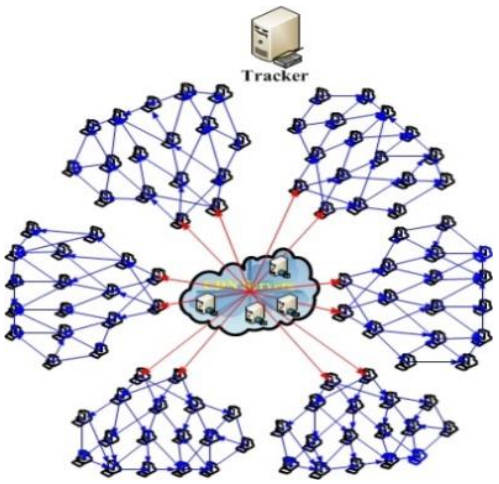


Fig. 4. An Organized view of CMPVoD.

TABLE 1 .Simulation parameters

parameters	Value
Number of peers	200
Video codec	MPEG4
Video FPS	25
Number of frames in GoP	12 frames
Sturtup buffering	12 s (300 frames)
Average video bit rate	512 Kbps
Chunk size	1 frame
Simulation duration	800 second
Number of CDN servers	4
Mean neighbor size of each peer	6
Buffer-map exchange period	1s
Size of each cluster	15 peer

- *Distortion:* sometimes, a peer may do not receive some chunks of video correctly before their playback deadlines. If the number of this chunks is not fair enough for playback discontinuity, peer continues playing video, but the quality of video reduces. In this paper, we evaluate total distortion. Total distortion is the total number of frames that are not played correctly divided by total frames of video that played. Some types of distortion are:

  I) late arrival distortion: when frame does not arrive or arrives after playback deadline.

  II) Bit error on data: if frame arrives, but some data of it is lost, frames cannot be played correctly.

  III) Dependency distortion: in this simulation, video codec is MPEG, in this codec if a frame is lost, other frames that are related to this frame will also be lost.
- Startup delay: Elapsed time from when the peer sends a request to watch a video until when the peer starts to play, which is computed from the sum of joining delay and buffering delay.
- Seek delay: Elapsed time from when the peer sends a request to the tracker for jumping until when the peer starts to watch from that point. It is a summation of cluster change and buffering delay.
- Server stress: The total number of peers that receive video directly from the CDN video servers. In other words, server stress is the number of streams provided by the servers which each stream rate have a bit rate equal to the video bit rate.

B. *Simulation results*

In this section, we first try to find the trade-off quantities for the parameter; then, evaluate the quality of service with and without using our mechanism by performing the evaluation in different conditions of the system. Finally, we study the incentive effect of proposed mechanism.

1) *Finding trade-off values for parameters*

To find tradeoff points, we change values of the slope of tradeoff line and improvement period. The best result has the lowest discontinuity and distortion. In this set of experiments, peers remained in the system once they had joined, and do not have any interactive. In Fig.5 slope of the line is variable, it can be seen when the slope of the line increases from zero to 0.025 discontinuity and distortion become lower. However, when the slope of the line increases up to more than 0.05, discontinuity and distortion increase with a gentle slope. It is obvious in Fig. 5 that the best values for the slope of the line are quantities between 0.25 and 0.5. Fig. 6 also shows when improvement period increases up to more than 35s, quality of service decrease. In this figure, Improvement Period is variable; when Improvement Period increases from 5 to 35 second, the discontinuity and the distortion doesn't change too much. However, when improvement period parameter increases up to more than 35 second, discontinuity and distortion increase.

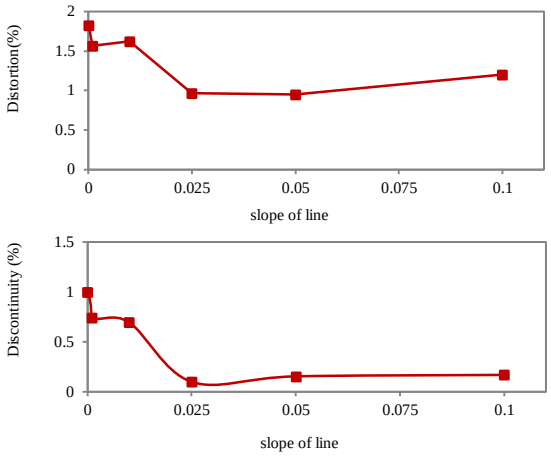


Fig. 5. Quality of service as function of the slope of trade-off line.

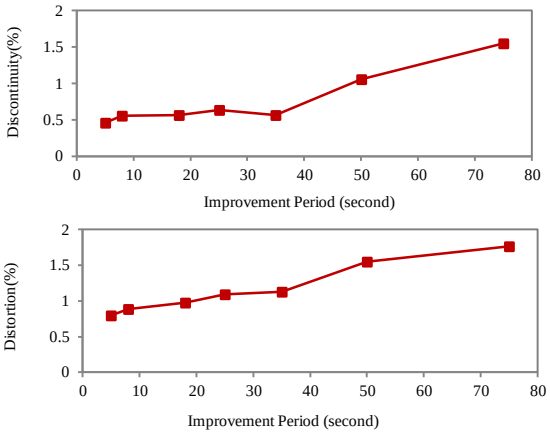


Fig. 6. Quality of service as function of Improvement Period.

As we will also discuss in the overhead analysis section, if lower quantities for Improvement Period are selected, the overhead of mechanism increases, but Fig. 6 shows that the overhead of our improvement mechanism does not much affect the quality of service parameters, even it runs every 5 seconds. However, in selecting the value of this parameter, we must pay attention to round trip delay of messages, and this depends on the condition of congestion in the network. Therefore, it should be selected big enough which, during an improvement process run, other process does not start.

2) *Performance evaluation in stable environment*

In this set of experiments, all of the peers remained in the system once they had joined, and do not have any interactive action such as jump forward and played the video continuously from the beginning to the end. In this section, at the first experiment, we change the number of all peers in the system from 50 to 250. At the second experiment, we use 200 peers in the system and modify the number of peers in each cluster from below 10 to more than 50 (the number of clusters is changed from 4 to more than 20 in Fig. 4). The average discontinuity and distortion of all peers were measured all along the experiments in the case of using and not using the improvement mechanism.

In Figs. 7-10, we show the evaluation of the quality of service parameters as a function of the number of peers and the number of peers in each Cluster (Cluster

size). Fig. 8 also shows cumulative distribution function (CDF) of the discontinuity of the 200 peers with use and without using our improvement mechanism. It can be seen in these figures that by using our improvement mechanism, we can achieve very significant improvements in quality of service parameters. In addition, Fig.10 shows the average server stress as a function of Cluster size. Average server stress is the average number of streams provided by the server. It can be seen in these figures that when the Cluster size increase from below 10 to more than 50, the average server stress for 200 peers reduce from more than 35(more than 17 percent of client-server model) to below 8 (below 4 percent of client-server model). So using our improvement mechanism, we can do a better tradeoff between average server stress and quality of service parameters.

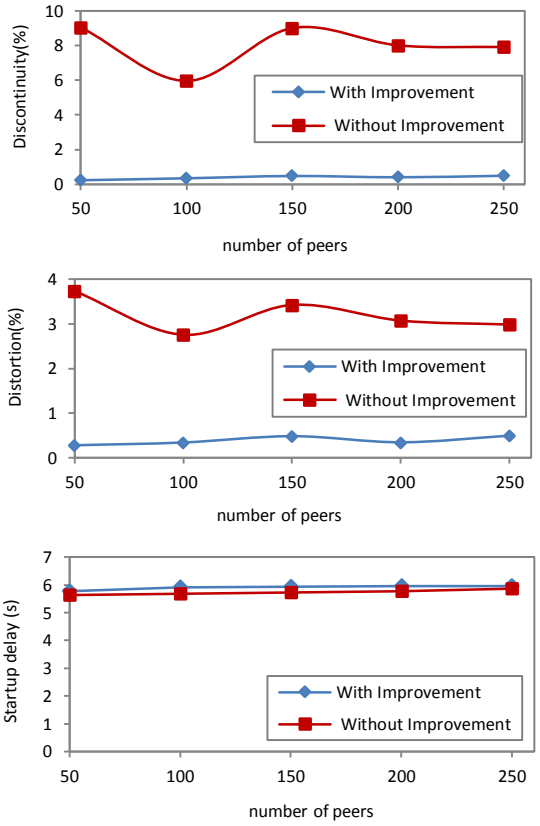


Fig. 7. Discontinuity, distortion and startup delay as function of number of peers

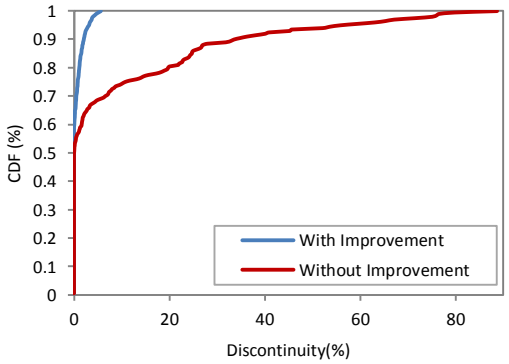


Fig. 8. CDF of Discontinuity for 200 peers

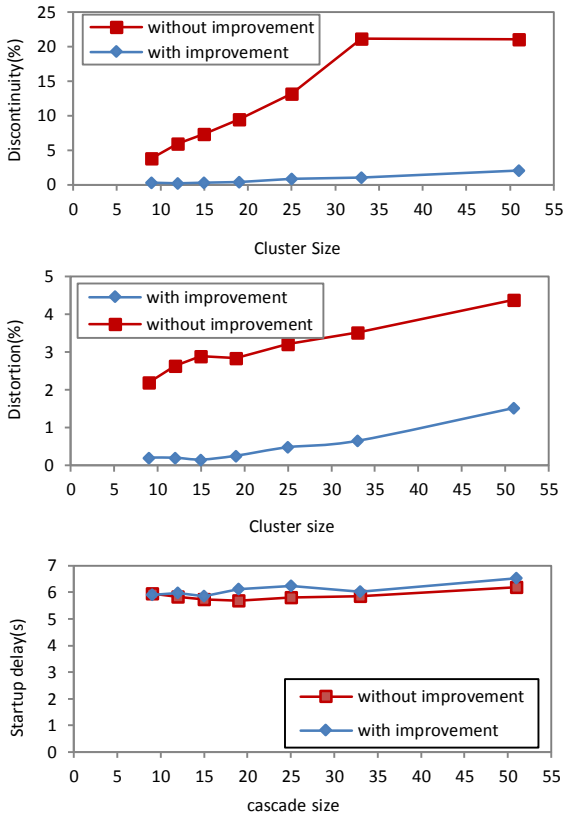


Fig. 9. Discontinuity, distortion and startup delay as function of Cluster size

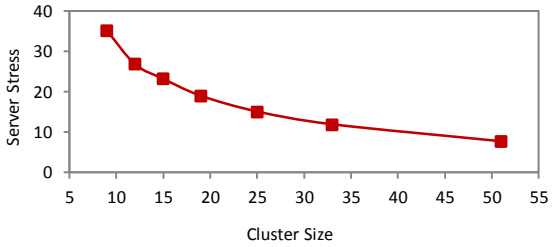


Fig. 10. Average server stress as function of Cluster size

3.2.3 Performance evaluation in dynamic environment

Further series of experiments was performed to investigate the robustness of the improvement mechanism to peer interactions such as jumping forward and exiting from the system. In this set of experiments, at the first 200 seconds of the experiment, all the peers performed playback only. From the 200 to 800 seconds 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. In Figs. 11 and 12 the evaluation of the quality of service parameters as a function of the percent of exit and interactive peers are shown. Figs. 11 and 12 show that our mechanism is robust against churn. Although when the percentage of the interactive peer is bigger than 70, quality of service reduces. Usually, this percentage of interactive does not happen. In Fig. 11, one can see that our improvement mechanism is a bit robust against the peer's churn. Also, the quality of service is much better than when improvement mechanism is not used.

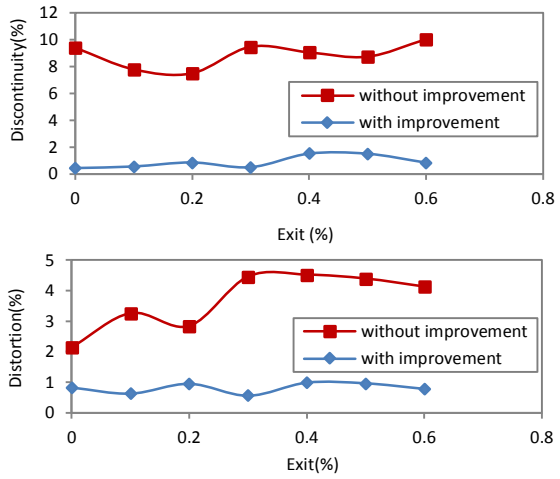


Fig. 11. Discontinuity and distortion as function of percent of exit.

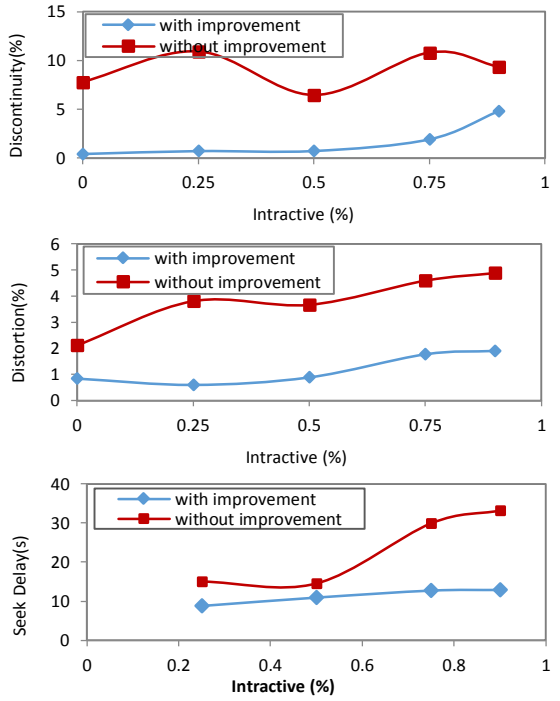


Fig. 12. Discontinuity, distortion and seek delay as function of percent of interactive.

4. Bandwidth overhead and mechanism behavior

As mentioned before our improvement mechanism is fully distributed and do not have any overhead on the servers. So for evaluation bandwidth overhead, it needs to calculation bandwidth overheads of all messages of this mechanism that peers send. To study the mechanism behavior, these messages also must be analyzed and the number of them sending and receiving by each peer as well as the size of them must be measured. In this subsection, we analyze Bandwidth overhead and mechanism behavior in two experiments of above experiments. These experiments are selected because, in these experiments, mechanism behavior change with changing variable parameters, and they had better show the behavior of the system. At the other experiments, mechanism behavior does not change much. In the first experiment, mechanism behavior evaluated as a function of Improvement Period, and at the second experiment, it evaluated as a function of cluster size. In both experiments, the simulation time is 800 seconds and the number of peers is 200, which peer's arrival process is Poisson distribution with the mean inter-arrival time of 2 seconds. So, mean of peers' lifetime is 600 second.

Fig. 13 shows the average improvement Request messages that each peer sends to its neighbors in all of its live time. It can be seen in Fig. 13 part (a) that with Improvement Period increase, the number of improvement request increases, but this increase is because of the increase of improvement deny, so it doesn't lead to increase improvement operations. Fig. 13 part (b) also shows that with Cluster size increase the number of improvement request increases, but in this case, it leads to the increase of the number of improvement operations. Fig. 14 shows the average improvement operations that each peer during its life involves in. The average number of improvement operation is equal to the twice of the average number of improvement response message (in Fig. 13). It can be seen in Fig. 13 and 14 that with increasing Improvement Period the numbers of improvement operation decreases; and with increasing Cluster size the number of improvement operations increased.

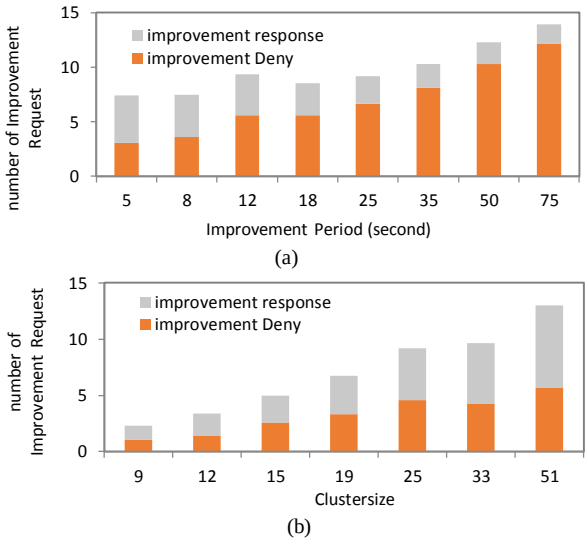


Fig. 13. Improvement messages as function of (a) Improvement Period, (b) Cluster size.

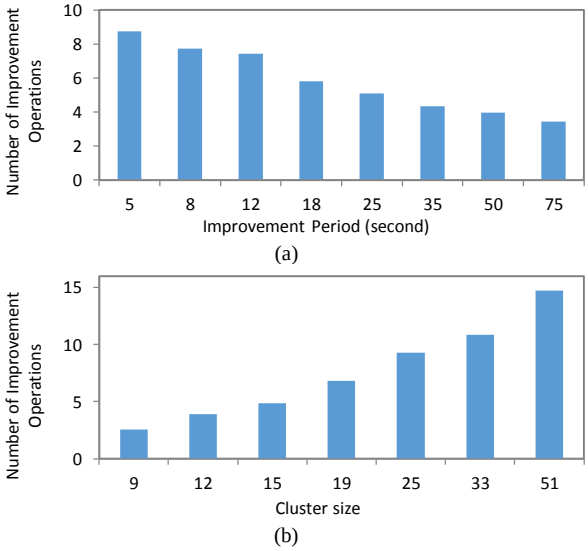
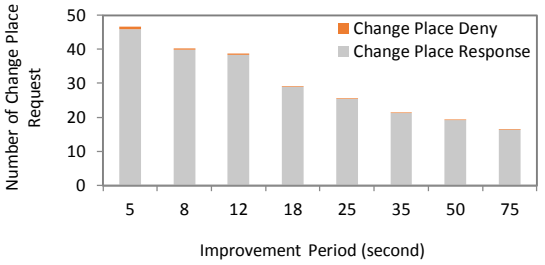
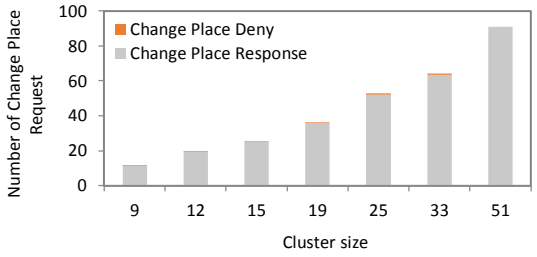


Fig. 14. Improvement operations as function of (a) Improvement Period, (b) Cluster size

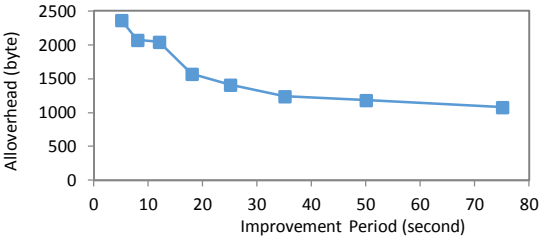


(a)

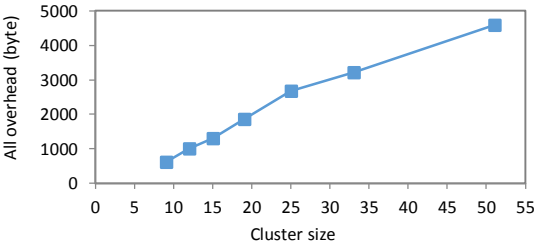


(b)

Fig. 15. The number of Change place messages as function of (a) Improvement Period, (b) Cluster size.

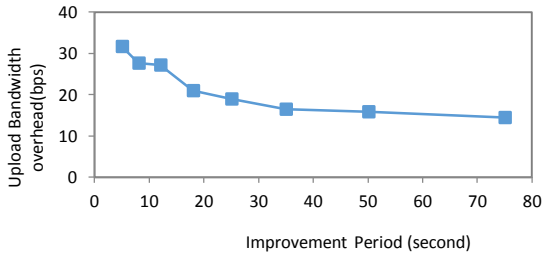


(a)

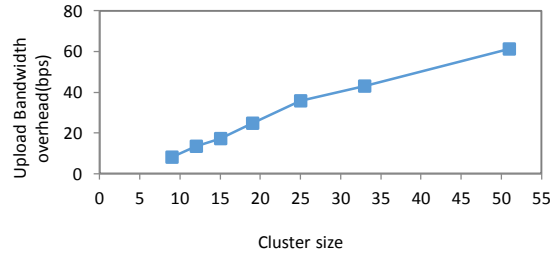


(b)

Fig. 16. Upload overhead as function of (a) Improvement Period, (b) Cluster size.



(a)



(b)

Fig. 17. Average upload bandwidth overhead as function of (a) Improvement Period, (b) Cluster size.

Fig. 15 shows the average number of change place request messages that each peer sends to its neighbors in all its lifetime. It can be seen in Fig. 15 part (a) that with increasing Improvement Period the number of change place request messages decrease. Fig. 15 part (b) shows that with increasing Cluster size, and with increasing the number of peers in each Cluster, the number of change place request messages increase. Fig. 15 also shows the number of change place deny messages are negligible.

Fig. 16 shows the average upload overhead of mechanism in all peers, which this overhead is the total byte of mechanism messages that a peer sends to its neighbors in its all live time. Fig. 17 also shows average upload bandwidth overhead of mechanism that it is average upload overhead of mechanism divide to average peer's lifetime (in bps). It can be seen in Fig. 16 part (a) and Fig. 17 part (a) that with increasing Improvement Period the upload overhead and so upload bandwidth overhead decreases. Fig. 16 part (b) and Fig. 17 part (b) shows that with increasing Cluster size upload overhead of mechanism and so upload bandwidth overhead increases; in other words, bandwidth overhead of mechanism has a direct relation to Cluster size. In these figures, also it can be seen that bandwidth overhead of mechanism is low (below 100 bps), so to find trade-off points of mechanism parameters such as Improvement Period, the bandwidth overhead of mechanism can be disregarded.

3) Evaluation the incentive effect of proposed mechanism and impact of free riders

The other advantage of our mechanism is punishing the free riders and incentive peers to contribute more. To prove incentive effect of our improvement mechanism, 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, as shown in Fig. 18 and 19, we classified the results based on peer's upload bandwidth. These figures show the quality of service parameters based on peers' upload bandwidth in the case of using and not using our mechanism. Can be seen that with using improvement mechanism, peers with more upload bandwidth will experience a better quality of service.

As mentioned before, note that the upload bandwidth that used in our mechanism is the maximum bandwidth that peers shared, and it differs to the contributed bandwidth. For a better illustration of this concept, Figs. 20 and 21 show the peers' contributed upload bandwidth as a function of peers' shared upload bandwidth in the case of using and not using our improvement mechanism. Can be seen in these figures that by using proposed mechanism peers with better upload bandwidth contribute more bandwidth in the system. Table 2 also shows that the average contribution of upload bandwidth of peers and average received video data per peer with using our mechanism is much better than without using this mechanism.

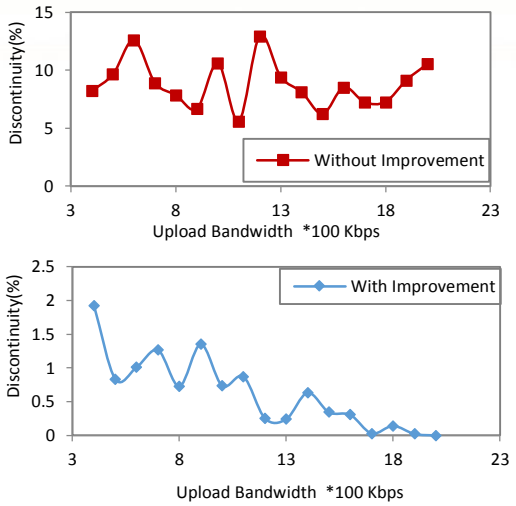


Fig.18. Discontinuity as function of peer's shared uploads bandwidth

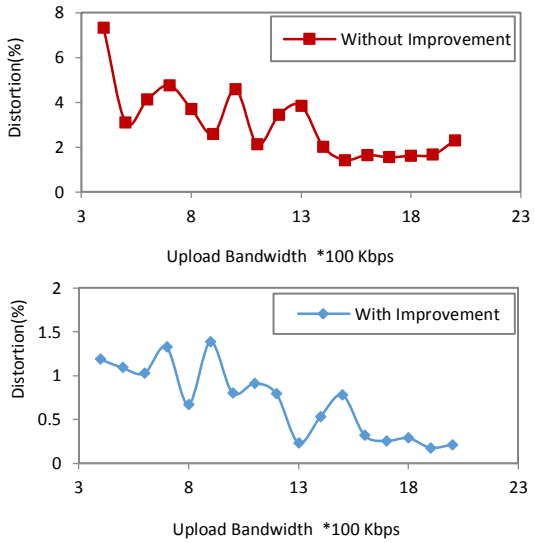


Fig. 19. Distortion as function of peer's uploads bandwidth

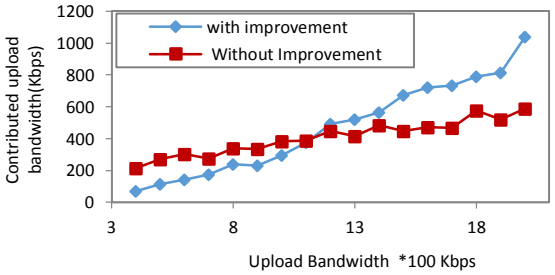


Fig. 20. Contribution bandwidth as function of peer's upload bandwidth

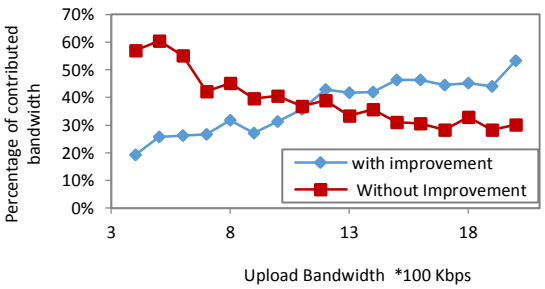


Fig. 21. Percentage of contributed upload bandwidth as function of peer's uploads bandwidth

TABLE 2. ANALYSIS OF CONSUMED UPLOAD BANDWIDTH IN THE EXPERIMENT

parameters	With imp.	Without imp.
Avg conrtibution bandwidth of peers (Kbps)	458.07	395.15
Avg control overhead of peers (Kbps)	5.86	5.17
Avg consumed bandwidth of servers (Kbps)	2464	2645
Avg control overhead of servers (Kbps)	1.075	1.078
All Kbyte sent by peers (200 peer-Kbyte)	6871050	5927250
All Kbyte sent by servers(4 server-Kbyte)	985600	1058000
All Kbyte sent (Kbyte)	7856650	6985250
All Kbyte control overhead sent (Kbyte)	88330	77981
All video Kbyte sent (Kbyte)	7768320	6907269
Avg received video data per peer (Kbps)	518.2	460.5

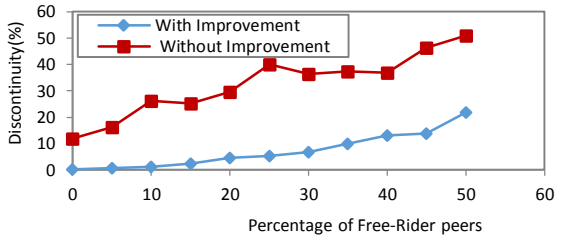
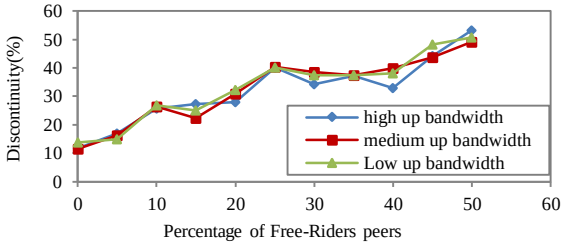
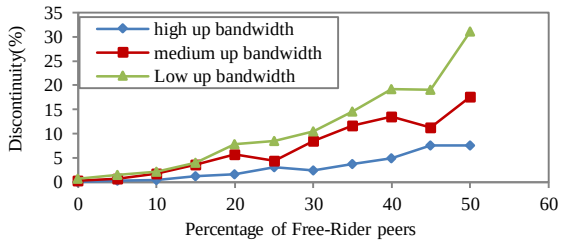


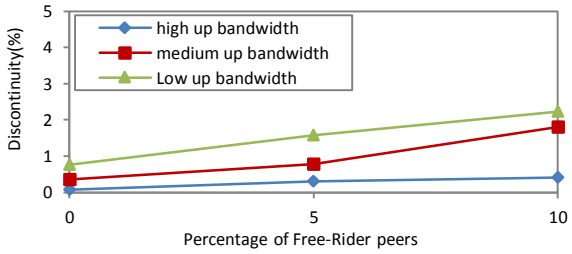
Fig. 22. Discontinuity as function of Free-riders



(a)



(b)



(c)

Fig. 23. Discontinuity as function of free-riders for various contribution levels (a) without Improvement (b) and (c) with improvement

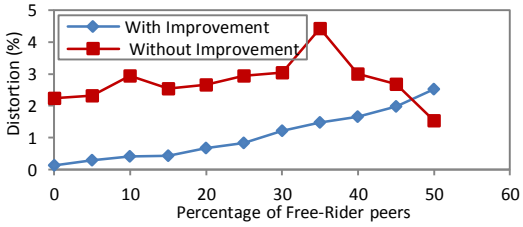
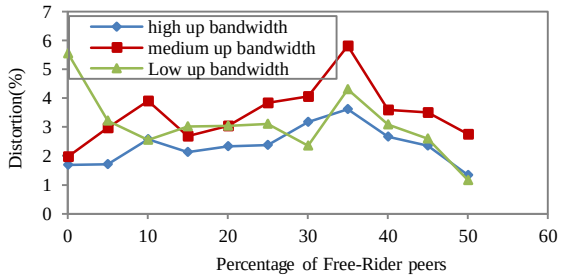


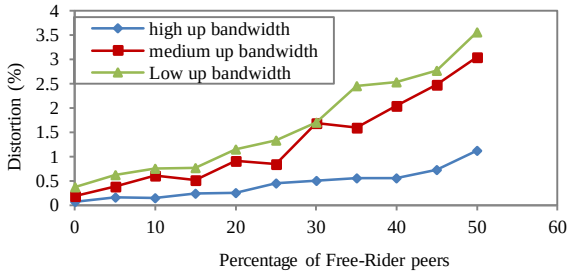
Fig. 24. Distortion as function of Free-riders

To illustrate the incentive effect of our mechanism, when some of the peers are free riders, we defined free-riders as peers that do not contribute to stream any chunk or in other words do not share their upload capacity to stream video. Also, tree type of contribution levels of peers is defined: peers with below 512 Kbps upload bandwidth and free-riders (low upload bandwidth), peers with upload bandwidth between 512 to 1024 Kbps (medium upload bandwidth) and peers with more than 1024 Kbps upload bandwidth (high upload bandwidth). In Figs. 22 to 26, the system is evaluated for various percentages of free-riders.

Figs. 22-25 show discontinuity and distortion as a function of the percentage of free riders. It can be seen in these figures that without the use of our mechanism, all types of peers received the same quality of the video, but with the use of our mechanism, peers that contribute more receive better quality of the video, also the average discontinuity and distortion significantly decrease. Fig. 26 also shows the percentage of peers that do not receive enough chunks of video to start playing. It can be seen in these figures that when the percentage of free-riders is more than 35, without using our mechanism, the percentage of not played peers are increased from 30 to 50 percent, but with use our mechanism this is below 15 percent. In Fig. 24 and 25(a) when percentages of free-riders are more than 35 percent, distortion decreases, due to increasement of not played peers.



(a)



(b)

Fig. 25. Distortion as function of Free-riders for various contribution levels (a) without Improvement (b) with improvement

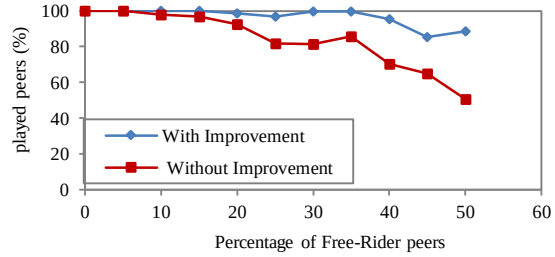


Fig. 26. Percentage of peers that played as function of Free-riders

V. Conclusion

In this paper, we proposed a distributed free-riding resiliency incentive mechanism for on-demand video streaming over peer-to-peer mesh networks. This mechanism places peers with more upload bandwidth closer to the video source while putting free riders far from the video source. Also, the video streaming peers organize an overlay based on their playback point and contributed upload bandwidth. This mechanism can be used in P2P and hybrid CDN-P2P approaches. The performance evaluation results of the proposed mechanism demonstrate the soundness and completeness of our mechanism in term of resiliency against free-riding and quality-of-service provision at peers organizing the peer-to-peer mesh network. Furthermore, the results prove that the proposed mechanism create the incentive for the peers inside the peer-to-peer network to use their upload bandwidth while using the download bandwidth. As a future work, we would like to study the effect of dishonest peers on the peer-to-peer mesh networks. Also, we would like to provide resiliency against these dishonest peers.

References

[1] B. Li, Z. Wang, J. Liu, and W. Zhu, "Two decades of internet video streaming: A retrospective view," ACM transactions on multimedia computing, communications, and applications (TOMM), vol. 9, no. 1s, p. 33, 2013.

[2] N. Ramzan, H. Park, and E. Izquierdo, "Video streaming over P2P networks: Challenges and opportunities," Signal Processing: Image Communication, vol. 27, no. 5, pp. 401–411, 2012.

[3] H. Ghaeini, B. Akbari, B. Barekatain, and A. T. Cabrera, "Adaptive video protection in large scale p2p video streaming over mobile wmns," International Journal of Communication Systems, 2015.

[4] Y. Guo, K. Suh, J. Kurose, and D. Towsley, "P2cast: peer-to-peer patch-ing for video on demand service," Multimedia Tools and Applications, vol. 33, no. 2, pp. 109–129, 2007.

[5] Y.-S. Yu, C.-K. Shieh, C.-H. Lin, and S.-Y. Wang, "P2PVR: A playback offset aware multicast tree for on-demand video streaming with vcr functions," Journal of Systems Architecture, vol. 57, no. 4, pp. 392–403,2011.

[6] B. Li, S. Xie, Y. Qu, G. Y. Keung, C. Lin, J. Liu, and X. Zhang, "In-side the new coolstreaming: Principles, measurements and performance implications," in INFOCOM. The 27th Conference on Computer Communications. IEEE, 2008, pp. 1031–1039.

[7] C. S. Lin and J. W. Lin, "UR-aware: Streaming videos over BitTorrent with balanced playback urgency and rareness distribution," Peer-to-Peer Networking and Applications, vol. 9, no. 6, pp. 1114–1125, 2016.

[8] PPLive, "http://www.pptv.com/"

[9] A.G. Sheshjavani and B. Akbari, "An adaptive buffer-map exchange mechanism for pull-based peer-to-peer video-on-demand streaming systems," Multimedia Tools and Appications, , vol. 76, no. 5, pp. 7535–7561, 2017.

[10] N. T. Jahromi, B. Akbari, and A. Movaghar," A hybrid mesh-tree peer-to-peer overlay structure for layered video streaming," 5th International Symposium on Telecommunications, 2010, pp. 706-709.

[11] L. Favalli1, M. Folli, and M. Lanati, "Improved multicast algorithm for overlay multicast in P2P based video streaming"Consumer Communications and Networking Conference, 2010, pp. 1-5.

[12] F. Wang, Y. Xiong, and J. Liu, "mTreebone: A Collaborative Tree-Mesh Overlay Network f or Multicast Video Streaming," IEEE Transaction on parallel and distributed systems, vol. 21, no. 3,pp. 379-392, 2010.

[13] A. H. Payberah, J. Dowling, and S. Haridi, "Glive: The gradient overlay as a market maker for mesh-based p2p live streaming," in Parallel and Distributed Computing, 2011 10th International Symposium on. IEEE, 2011, pp. 153–162.

[14] S. Kang and H. Yin, "A hybrid CDN-P2P system for Video-on-Demand," in Second International Conference on Future Networks, 2010, pp. 309-313.

[15] S. Yang and X. Wang, "A Probabilistic Approach to Analyze the Effect of Free-riders in P2P Streaming Services" in 2008 IFIP International Conference on Network and Parallel Computing, 2008, pp.387- 391.

[16] S.Seyyedi and B. Akbari, "Hybrid CDN-P2P Architectures for Live Video Streaming: Comparative Study of Connected and Unconnected Meshes," in Computer Networks and Distributed Systems, International Symposium on, 2011, pp. 175-180.

[17] A. G. Sheshjavani, B. Akbari, and H. R. Ghaeini, "CMPVoD: A Cluster Mesh-based Architecture For VoD Streaming over Hybrid CDN-P2P Networks," in Telecommunications (IST), 8th International Symposium on, 2016, pp.783-788.

[18] E. Baccaglini, M. Grangetto, E. Quacchio, and S. Zezza, "A study of an hybrid cdn–p2p system over the planetlab network," Signal processing: image communication, vol. 27, no. 5, pp. 430–437, 2012.

[19] M. Garmehi, M. Analoui, M. Pathan, and R. Buyya, "An economic replica placement mechanism for streaming content distribution in Hybrid CDN-P2P networks," Computer Communications, vol. 52, pp. 60-70, 2014.

[20] C. Hu, M. Chen, C. Xing, and B. Xu, "EUE principle of resource scheduling for live streaming systems underlying CDN-P2P hybrid architecture," Peer-to-Peer Networking and Applications, vol. 5, no. 4, pp. 312-322, 2012.

[21] G. Zhang, W. Liu, X. Hei, and W. Cheng, "Unreeling Xunlei Kankan: understanding hybrid CDN-P2P video-on-demand streaming," Multimedia, IEEE Transactions on, vol.17, no. 2, pp. 229-242, 2015.

[22] M. K. Bideh, B. Akbari, and A. G. Sheshjavani, "Adaptive content-anddeadline aware chunk scheduling in mesh-based p2p video streaming," Peer-to-Peer Networking and Applications, vol. 9, no. 2, pp. 436-448, 2016.

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

[24] J. Luo, "Practical Algorithm for Minimum Delay Peer-to-Peer Media Streaming," in Multimedia and Expo (ICME), 2010 IEEE International Conference on, 2010, pp. 986-991.

[25] F. Wang, J. Liu, and Y. Xiong, "Stalbe Peers: Existence, Importance, and Application in Peer-to-Peer Live Video Streaming," in IEEE INFOCOM-The 27th Conference on Computer Communications, 2008, pp. 1364-1372.

[26] J. Zou and L. Chen, "Joint bandwidth allocation, data scheduling and incentives for scalable video streaming over peer-to-peer networks," Multimedia Tools and Applications, vol. 73, no. 3, pp. 1268-1289, 2014.

[27] K. Eger and U. Killat, "Bandwidth trading in BitTorrent-like P2P networks for content distribution," Computer Communications, vol. 31, no. 2, pp. 201-211, 2008.

[28] C. Liang and Y. Liu, "Incentivized Peer-Assisted Streaming for On-Demand Services" IEEE Transaction on parallel and distributed systems, vol. 21, no. 9, pp. 1354 – 1367, 2010.

[29] A. Montazeri, B. Akbari, and M. Ghanbari, " An incentive scheduling mechanism for peer-to-peer video streaming," Peer-to-Peer Networking and Applications, vol. 5, no. 3, pp. 257-278, 2012.

[30] H. Chih-Lin and K. Tzu-Han, "A hierarchical overlay with cluster-based reputation tree for dynamic peer-to-peer systems," Journal of Network and Computer Applications, vol. 35, no.6, pp. 1990-2002, 2012.

[31] H. Hu, Y. Guo, and Y. Liu, "Peer-to-Peer Streaming of Layered Video: Efficiency, Fairness and Incentive," IEEE Transaction on circuits and system for video tecnology, vol. 21, no. 8, pp. 1013-1026, 2011.

[32] A.G. Sheshjavani and B. Akbari, " A Study on Repeat-Request chunks in pull-based Peer-to-Peer Video-On-Demand Streaming," in Electrical Engineering (ICEE), 2014 22nd Iranian Conference on, 2014, pp. 891-896.

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

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

[35] I. Baumgart, B. Heep, and S. Krause, "OverSim: A scalable and flexible overlay framework for simulation and real network applications," in Peer-to-Peer Computing, IEEE Ninth International Conference on, pp. 87-88, Sep. 2009.

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

Abdollah Ghaffari Sheshjavani is currently working towards his Ph.D. at the Department of Electrical and Computer Engineering at the University of Tehran, Tehran, Iran. He obtained his B.S. and M.S. degree in Computer Engineering from Air University of Science and Technology and Tarbiat Modares University, Tehran, Iran, in 2008 and 2013, respectively. His research interests are Internet QoS, Peer-to-Peer networks, Media streaming over the Internet, Caching in telecommunication networks and network performance modeling and evaluation.

Behzad Akbari received his B.S., M.S. and PhD 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, multimedia networking, peer-to-peer video streaming, data center networking, Software Defined Networking (SDN), Network Function Virtualisation (NFV) and network performance modelling and evaluation.

Hamid Reza Ghaeini is a bi-nationally supervised doctoral student at the Singapore University of Technology & Design, and the TU Darmstadt. He received his M.Sc. in Computer Engineering with the focus on peer-to-peer multimedia streaming from the Tarbiat Modares University. He has been awarded several international awards, among them the DAAD, Kulicke & Soffa, and the IEEE ComSoc. He was served as a reviewer in reputable journals like Springer Journal of Wireless Personal Communications, IEEE China Communications, and International Journal of Information & Communication Technology Research.