IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 3, Ver. III (May-Jun. 2014), PP 67-73
www.iosrjournals.org

An Efficient Hybrid Push-Pull Based Protocol for VOD In Peer-to-Peer network

Mohamed Ghettas[1], Putra Sumari [2]
[1]*(computer science, Universiti Sains Malaysia USM, Malaysia)*

Abstract : *Video-on-Demand is a service where movies are delivered to distributed users with low delay and free interactivity. The traditional client-server architecture experiences scalability issues to provide video streaming services. Peer-to-Peer (P2P) techniques for video on demand (VOD) have been shown to be a good enhancement to the traditional client/server methods when trying to reduce costs, increase robustness and solve the scalability issues. Various challenges arise when building a resilient P2P video on demand systems, such as long waiting time to receive the video segments and the quality of service. A new proposed data exchange schema is called Efficient Hybrid Push-Pull based Protocol (EHP3) can reduce the initial playback delay and improve the quality of service. The downstream peer has to accelerate the downloading process of the urgent segments, i.e. those missing segments which are near their playback time. The range of the urgent segments is referred to as the priority region. The upstream peers give a strict priority to segments within the priority region. If the downstream peer has sufficient download bandwidth, it should be able to completely receive these urgent segments before playback time. If there are no missing segments in the priority region, the downstream peer uses the pull protocol to download the missing segments which are outside the priority region. The simulation result shows that the proposed data exchange schema (EHP3) can reduce the initial play back delay and reduce the burden on the server side by increasing the number of caching segments in the playback buffer. Moreover, the proposed schema improves the quality of service by reducing the number of skipped segments during the playback.*

Keywords: *Video-On-Demand; Peer-to-Peer Network, Interactive videos; Video Streaming, P2P Streaming*

I. Introduction

The wide spread of the Internet and deployment broadband access, there has been an urgent need to develop new media services. As a result, the researchers are motivated to design media streaming services such as video over IP applications, video broadcasting and video on demand (VOD). Before the advent of these techniques, user has to download and store the whole movie in his machine in order to enjoy them. Furthermore, the user has to wait a period of time before he is able to watch the movie. Recently with the emerging of media streaming services, the user can watch movie on the fly while the movie is being downloaded with short startup delay and smoothly playback. Moreover, the user is able to perform most of control function (VCR) such as fast forward, fast search, reverse search and rewind at the client side without consume the server bandwidth.

In traditional Video-On-Demand (VOD) such as client-server architecture, the user has more flexibility to choose and download the video with short waiting time and high quality. The user sends a request for the video server and starts watching the video as soon as it receives the response from the video server. In this approach, the server dedicates a portion of its bandwidth to each connected user thus this approach is known as users centered. Consequently as the number of users increases, the server bandwidth turns out to be a serious constraint and the server will be unable to serve any additional users.

A peer-to-peer (P2P) network has been introduced to overcome the server bandwidth bottleneck in traditional VOD and provides VOD services with high quality with negligible waiting time. P2P systems are collection of computers which is connected via network, allowing access to a variety of resources such as files, sensors and others peripherals devices without the need for a coordinator. Peer-to-peer (P2P) model provides an alternative to the traditional client-server architecture where each machine is referred to as a peer. In this architecture a peer can play the role of server and client at the same time. That is, the peer can send request message to another neighbor, and reply to incoming requests from its neighbors. With peer-to-peer networks (P2P), overall performance has been improved as new peer is added to the systems. In P2P system, the peers which are interested to download a certain movie clip are grouped together, communicate and share bandwidth with each other to achieve a common target. Each peer contributes its upload bandwidth capacity to mitigate server bandwidth cost and serves other peers. In order to guarantee playback quality, local caching is used to store different segment of media contents to increase their ability to help others.

Another aspect of a P2P system is its capability to conceal and mask the failure. When a client fails or leaves the network, the P2P system continues to work correctly by using other clients. Bit Torrent is an example where the clients who have cached some video segments can serve other clients to download these segments. As soon as the destination client detects the source client failure, it searches for other source clients and continues downloading the file. On the other side, in the client-server model, all the system stops working if the server goes down. In P2P systems the streaming content is spilt into a sequence segments and send them one by one to the receiver. The receiver is able to play the video as soon as it receives the packets. Video playback with high quality is achieved through the receiving of the packets before playback deadlines. Each client caches a limited number of packets and exchanges them with its neighbors.

II. Related Work

To mitigate the required bandwidth at the dedicated video streaming server, tree-based topologies have been proposed in [1] [2] as shown in Fig. 1 where the nodes are organized into multicast tree(s) and the root of the tree is being the video source. The nodes in the n+1 level receive the video content from nodes in the n level. Moreover, the parent nodes push the received video content to their child nodes. The tree-based overlay suffers from high instability due to peers joining and leaving the tree in an arbitrary time. In addition, the child nodes downloading capacity is limited by the uploading bandwidth of parent node so that the tree height should be minimized to reduce the end to end delay. The multicast tree-based topology can be built by either distribution algorithm [3] [4] or a centralized algorithm [5] [6] can be used to build multicast tree.

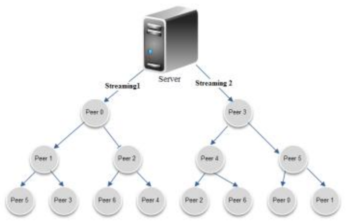

Fig. 1 Application layer multicast tree video streaming.

To address the problem of limited downloading capacity of uploading parent nodes in tree-based topologies mesh-based topologies is proposed. Fig. 2 organizes peers in mesh-based topologies and the peer dynamically establishes connections with neighbors [7] [8]. The authors proposed using the gossip protocol to enable peer to discover the streaming content at the neighbors. In [7] each peer maintains a list of neighbors and available streaming content in their caches. Based on such information, a peer pulls the missing segments from its neighbors. Despite of the mesh-based topologies is robust to dynamic peer churn; it incurs a considerable time delay to exchange segments availability among peers. Hybrid pull-based and push-based has been introduced in [9] to take advantage of better resilience to dynamics with pull-based protocol and short delay with push-based protocol.

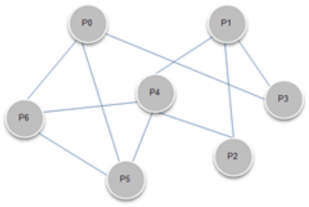

Fig. 2 Mesh-based overlay

Network coding and network information flow are proposed in [10] to increase communication efficiency over a network. In network coding, the intermediate node combines some of packets it has previously received and generates one outgoing data. On the receiving side, the node decodes the received data and gets the original information. If the combination is linear, it is called linear network coding (LNC) [11]. Network coding avoids coordination between nodes in decentralized system and limits the end to end delay so that the overhead is decreased. As a result, the system performance is improved. In contrast, the computation complexity at the receiver side is the main drawback of using network coding. It adds additional delay time but compared to the advantage of using network coding in video streaming, this delay can be ignored. In [12] the network coding segment scheduling algorithm is proposed to improve the throughput and latency performance of block distribution across P2P video on demand systems. With a similar objective, a scheduling algorithm based on network is proposed to fully exploit network resources for efficient deadline-aware VOD service. In [13] provides analysis and effect of passive caching with and without network coding and illustrates how the network coding can reduce the server bandwidth cost.

III. Propsed Work

As the traditional pull streaming protocol, the streaming content is divided into number of segments and further, these segments are divided into number of blocks with fixed size. Contrary, the proposed protocol uses large segment size in order to reduce the size of the segments map which exchanged among peers, consequently the overhead results from periodical exchanging segments map is significantly reduced. The operation of the proposed protocol has two phases: pull-based and push-based respectively. Moreover, the movie segments are classified according to its closeness to the current playback point into two types: urgent segments which are close to current playback point and non-urgent segments which are outside the current playback region. In both case, the peer uses the neighbors list and segments map to determine from whom it can download the missing segments. Our proposed protocol is different from the previous work in that it is just use the push-based algorithm to download the urgent segments and switch to the pull-based protocol to download non-urgent segment in order to reduce the overhead result from unnecessary duplication of sending data segments during push-based phase and limiting the using of push-based protocol to download only the urgent segments.

In push-based phase, instead the pushing process is initiated by the sender; it is initiated by the receiver as it needs to download the missing segments which are near to its current playback point. The peer looks up the segments map to gather information about the neighbors that have replicated these segments in their caches and hence it sends request message (urgent message) to these peers. Therefore, the sending peers start creating random blocks within each same segment and give high priority to the segments depending on their closeness to the current playback point. As long as the missing segments are served by multiple peers simultaneously so that if the receiving peer has sufficient downloading bandwidth, the missing segments are completely downloaded before their playback deadline. Consequently, the playback buffer is saturated and provides smooth playback. In addition, there is no need for central coordinator due to random selection of the creating blocks at the sending peers.

In Pull-based phase, after the urgent segments are completely downloaded, the receiving peer does not notify up the sending peers to stop sending data segments which in turns waste P2P bandwidth and represents more overhead. In order to reduce this overhead, the receiving peer switch to the pull-based phase immediately after it has finished downloading the urgent segments. In the pull-based phase, the downloading peer checks the segments map and selects random peer which holds the missing non-urgent segment. The selections of the sending streaming peer depends on its upload bandwidth, cache size and distance from the receiving peer. Hence, the receiving peer sends a request to the selected peer for downloading the missing non-urgent segment. The sending streaming peer serves the request and replays to the request by sending the missing segments. It is worth mentioning that the receiving peer can concurrently send request to multiple peers to serve different segments.

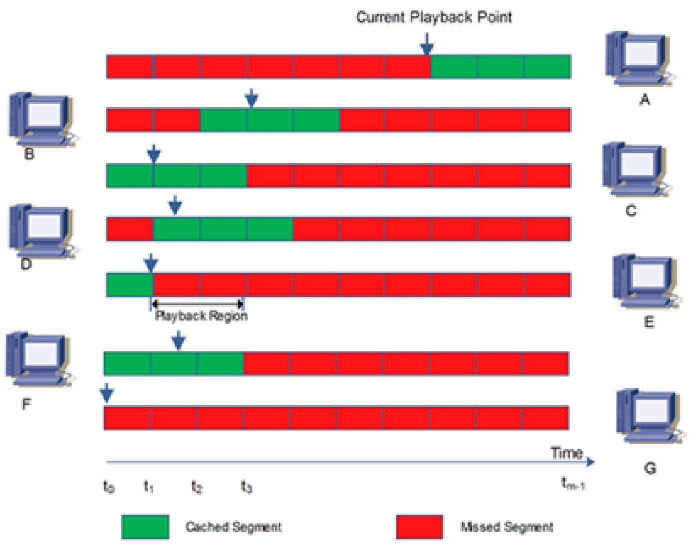

Fig. 3 downloading missed segments using proposed algorithm

Fig. 3 makes the idea more clear where the P2P system consists of seven clients which are downloading the same movie clip. Each client has cache size able to cache three segments of the movie at any given time. Moreover, the playback region is defined as the two segments (urgent segments) next to the current playback point. Hence the last cache segment is considered non-urgent segment. Consequently, each client has to cache the urgent the segments using push-based protocol and download non-urgent segments using pull-based protocol. To make things more clear, the initial position of the current playback point of the client E is at time slice t1. The client E has to download two urgent segments and only one non-urgent segment for playback at time t1, t2 and t3 respectively.

In order to download the urgent segments (in this case two segments), the client E initiates the push-based process by sending a request to peers C, D and F to concurrently download the first segment next to current playback point. In similar fashion, the client E uses the push-based protocol to concurrently download second the segment next to the current playback point from clients B, C, D and F. In order to reduce the overhead result from redundancy segments transmission from multiple clients in case of push-based protocol, the client switches to pull-based protocol to download non-urgent segments. In this case the client E pulls the third segment either from client D or B according to selection criteria such as bandwidth, reliability and segment availability. The download process repeated as the current playback point is shifted in either direction forward or back.

IV. Simulation And Performance Analysis

OPNET Modeler is used to generate the simulation scenarios. For simulation, each streaming session lasts for 10 minutes, the parameter of the server and peers are given below.

1. The server upload capacity is 1 MB per second.
2. Peers' upload capacities are uniform distribution between 80 to 100 KB per seconds.
3. All peers are connected to DSL.
4. The streaming rate is 64 KB per second.

The video is divided into segments. Each segment represents 4 seconds of playback. The buffer size of client/peer is 32 seconds. The initial buffering delay is set to be 16 seconds. The priority region is set to be 8 seconds. The following performance metrics are used to evaluate the proposed Efficient Hybrid Push-Pull based Protocol (EHP3).

1.1. The Playback Buffering Percentage Metric:

It measures the number of segments in the playback buffer of the client. Fig. 4 shows the percentage of segments in playback buffer for Efficient Hybrid Push-Pull based Protocol (EHP3), Network Coding and Vanilla algorithm respectively in case of network size is 88 peers. The numbers of segments in the playback buffer linearly increase with the simulation time. The increase in the playback buffering reduces the server loading. The observation from the comparison indicates that, EHP3 goes faster than other two algorithms. The result indicates that, the percentage buffering in case of EHP3 is 55% but in case of Network Coding is 51% and in case of Vanilla is 43%. Consequently, the numbers of segments in the playback buffer in case of EHP3 are greater than number of segments in playback buffer in case of Network Coding and Vanilla algorithms.

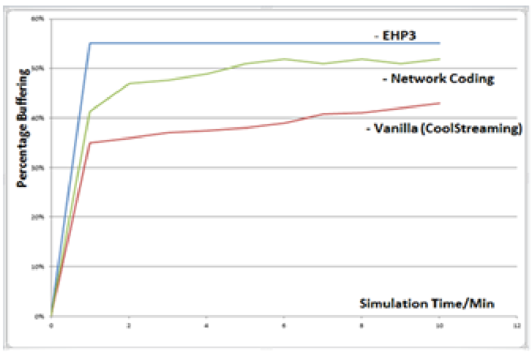

Fig. 4 Average Level of Buffering with Network Size 88 Peers.

Fig. 5 shows the average percentage of buffering segments for Efficient Hybrid Push-Pull based Protocol (EHP3), Network coding and Vanilla algorithm in case of network size 792 peers. The increase of the network size has more significant effect on reducing the server burden by increasing the percentage buffering in case of Efficient Hybrid Push-Pull based Protocol (EHP3). The number of percentage buffering increase from 55% in case of network size 88 peers to 65% in case of network size 792 peers. On contrast, the increase of network size from 88 to 792 peers has a little effect on reducing server loading in case of Network Coding and

Vanilla algorithm. Moreover, the Efficient Hybrid Push-Pull based Protocol (EHP3) supports system scalability more than two other algorithms. In addition, the percentage buffering in case of EHP3 reaches the steady state faster than two other algorithms.

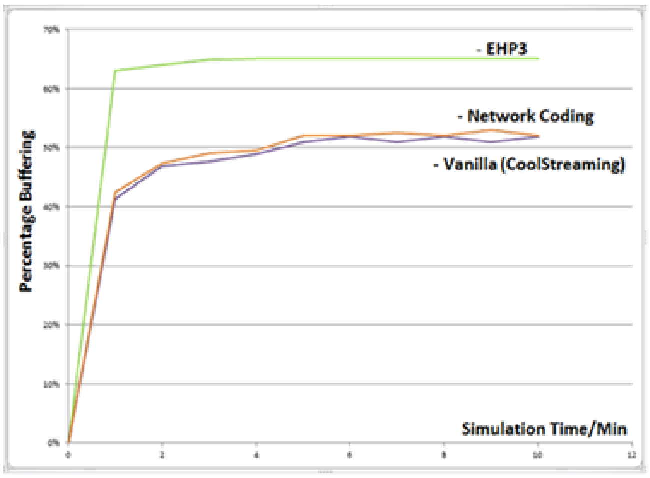

Fig. 5 Average Level of Buffering with Network Size 792 Peers.

From this discussion, the new data exchange scheme (EHP3) reduces the server loading by increasing the number of segments in the playback buffer. In addition, the Efficient Hybrid Push-Pull based Protocol (EHP3) system is more stable than Network Coding and Vanilla algorithms.

1.2. Percentage of Skipped Segments Metric:

It measures the number of segments which are not available at their playback time. Fig. 6 shows the percentage of skipped segments as tuning the length of the initial buffering delay. When the length of the initial playback delay is 8 seconds, the percentage of skipped segments in case of the Efficient Hybrid Push-Pull based Protocol (EHP3) is 2%. On Contrast, the percentage increases to 72.3% and 80.70 % in case of Network Coding and Vanilla algorithm.

The result indicates that, EHP3 improves the quality of service as a result of reducing the percentage of the skipped segments to 2% when it is compared with Network Coding and Vanilla algorithm (72% in case of Network Coding and 80.70% in case of Vanilla). The increase of the initial playback delay length from 8 seconds to 16 seconds reduces the percentage of skipped segments for EHP3, Network Coding and Vanilla. The result shows that the three algorithms have approximately the same percentage of skipped segments when initial playback delay is greater than 8 seconds. It is result from the increase of initial playback delay length in case of Network Coding and Vanilla allows the received client to download and store more segments in playback buffer. Consequently, the percentage of the skipped segments has decreased as in the case of EHP3.

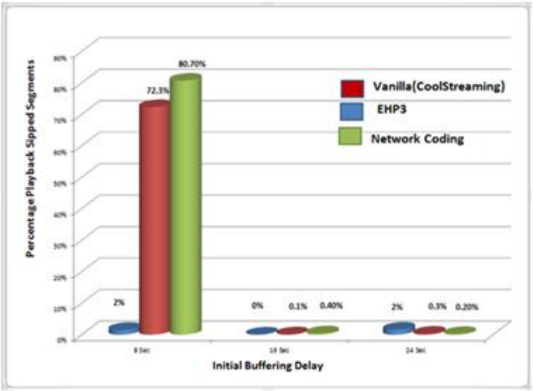

Fig. 6 Average Percentage Skipped Playback Segments as Tuning the Length of Initial Buffering

From the previous discussion, the reduction of the initial playback delay leads to increase the percentage of skipped segments which decreases the quality of service. On contrast, the Efficient Hybrid Push-Pull based Protocol (EHP3) has more effect on improving the quality of service in case of short initial playback delay.

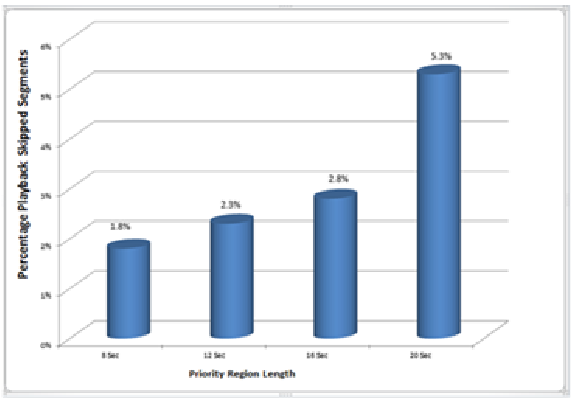

Fig.7 Average Percentage Skipped Playback Segments as Tuning Length of Priority Region

The priority region refers to playback segments which are downloaded from multiple upstream peers simultaneously. Fig. 7 illustrates the percentage of skipped segments when tuning the length of the priority region in case of EHP3. The result indicates the increasing of the priority region leads to increase of the percentage of the skipped segments. On Contrast, the increasing of the priority region accelerates the downloading process of the missing segments. Consequently, the quality of the playback is improved as long as the segments are cached before their playback time. On the other side, the Network Coding and Vanilla algorithm cannot classify the playback segments according to their playback time; hence, there is no priority.
Fig. 8 illustrates the average percentage of buffering level when tuning the length of the initial buffering delay. The average percentage of buffering is 78% in case of initial buffering delay equal to 2 seconds. The increase of the initial buffering delay from 2 to 8 seconds increases the average percentage of buffering from 78% to 92%. This linear relation between the percentage of buffering and initial buffering delay can be explained in terms of EHP3 mechanism behavior. The initial playback delay allows the downstream peer to download more segments from other neighbors so that the playback buffer receives and stores more segments. The increase of the initial buffering delay leads to download more segments before the playback process starts to consume the playback buffer segments.

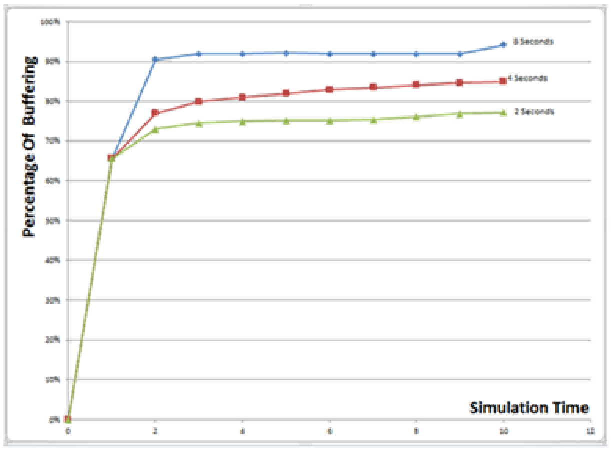

Fig. 8 Average Percentage Skipped Playback Segments as Tuning Length of Priority Region

From the analysis and discussion of the simulation results, the research objectives are achieved. The Efficient Hybrid Push-Pull based Protocol (EHP3) improves the quality of service and reduces both the server loading and initial playback delay when it is compared with Network Coding and Vanilla algorithms. This is can be interpreted in terms of the percentage buffering, percentage of skipped segments and initial buffering delay.

V. Conclusion

The initial playback delay, server loading and quality of service are challenges in VOD in P2P network. This research addresses these issues through the implementation and evaluation of a new data exchange scheme called Efficient Hybrid Push-Pull based Protocol (EHP3). This algorithm increases the average level buffering in order to reduce the server loading. Moreover, it decreases the average skipped segment. As a result, EHP3 improves the quality of service. The proposed algorithm reduces the required time to fill the priority region. Consequently, EHP3 reduces the initial playback delay. The accuracy of the algorithm as well as the implementation is confirmed through number of simulation results. The simulation result shows that, the quality of service is improved. Both initial playback and server loading are reduced comparing with other two algorithms, Network Coding and Vanilla.

References

[1]. Padmanabhan, V.N., H.J. Wang, and P.A. Chou, Supporting heterogeneity and congestion control in peer-to-peer multicast streaming, in Peer-to-Peer Systems III2005, Springer. p. 54-63.

[2]. Castro, M., et al. SplitStream: high-bandwidth multicast in cooperative environments. in ACM SIGOPS Operating Systems Review. 2003. ACM.

[3]. Tran, D.A., K.A. Hua, and T. Do. Zigzag: An efficient peer-to-peer scheme for media streaming. in INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies. 2003. IEEE.

[4]. Banerjee, S., B. Bhattacharjee, and C. Kommareddy, Scalable application layer multicast. Vol. 32. 2002: ACM.

[5]. Deshpande, H., M. Bawa, and H. Garcia-Molina, Streaming live media over a peer-to-peer network. Technical Report, 2001.

[6]. Padmanabhan, V.N., H.J. Wang, and P.A. Chou. Resilient peer-to-peer streaming. in Network Protocols, 2003. Proceedings. 11th IEEE International Conference on. 2003. IEEE.

[7]. Zhang, X., et al. CoolStreaming/DONet: a data-driven overlay network for peer-to-peer live media streaming. in INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE. 2005. IEEE.

[8]. Magharei, N. and R. Rejaie, Prime: Peer-to-peer receiver-driven mesh-based streaming. IEEE/ACM Transactions on Networking (TON), 2009. **17**(4): p. 1052-1065.

[9]. Zhang, M., et al. A peer-to-peer network for live media streaming using a push-pull approach. in Proceedings of the 13th annual ACM international conference on Multimedia. 2005. ACM.

[10]. Ahlswede, R., et al., Network information flow. Information Theory, IEEE Transactions on, 2000. **46**(4): p. 1204-1216 %@ 0018-9448.

[11]. Li, S.Y., R.W. Yeung, and N. Cai, Linear network coding. Information Theory, IEEE Transactions on, 2003. **49**(2): p. 371-381 %@ 0018-9448.

[12]. Annapureddy, S., et al. Is high-quality VoD feasible using P2P swarming? 2007. ACM.

[13]. Wang, H., et al. The benefits of network coding in distributed caching in large-scale P2P-VoD systems. in Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE. 2010. IEEE.