Expediting P2P Video Delivery through a Hybrid Push-Pull Protocol

Choonhwa LEE[1], Sungho KIM[1], Eunsam KIM[2]
[1]*Division of Computer Science and Engineering, Hanyang University, Seoul 133-791, Rep. of Korea*
[2]*Department of Computer Engineering, Hongik University, Seoul 121-791, Rep. of Korea*
eskim@hongik.ac.kr

***Abstract*—Despite the recent phenomenal success of peer-to-peer video streaming services, their stumbling performance for high-quality videos remains a major obstacle to wider acceptance. This is because high-resolution videos instantly delivered over the Internet are increasingly becoming the norm. This paper presents a novel solution to keep up with ever more challenging QoE expectations. Our proposal of a hybrid push-pull protocol consists of two key components, namely, a new push strategy and an elastic window scheme. The former empowers the hybrid protocol to make an informed push-pull decision based on chunk status and network condition, whereas the latter ensures balance between the two conflicting goals of chunk dissemination and playback deadline. The efficacy of the proposed protocol is validated through a performance study that demonstrates substantial gains compared to existing approaches.**

***Index Terms*—computer networks, content distribution networks, distributed computing, peer to peer computing, streaming media.**

I. Introduction

A wide range of video streaming protocols over the Internet have been explored over the last decade [1]. They can be largely classified into tree-based and mesh-based protocols. According to tree-based schemes, a peer pushes a video stream to its children along delivery trees, whereas mesh-based streaming protocols require no such static topology for video delivery. Instead, a peer pulls video pieces from its neighbors of an unstructured overlay. From periodic chunk map exchanges, a peer knows what chunks its mesh neighbors possess, and acquires its missing chunks from the neighbor peers by making explicit chunk requests. These mesh-pull protocols have been the mainstream for Internet-based P2P streaming research efforts for the past several years [2-5]. Not having a rigid tree of mesh-based protocols means robustness to node churn and failures. Also, there is no need for complicated algorithms to construct and maintain a streaming overlay tree as in tree-push schemes. However, the flipside of the coin is the downsides of high control message overhead and longer streaming delay associated with the chunk map exchange and request-fetch process.

The phenomenal success of peer-to-peer video streaming systems thus far has put us in a position to explore a range of further technical challenges [6-9]. Among them is the fragile performance of current P2P streaming systems for high-quality videos, which is deemed as a major roadblock to their further acceptance. Users are expecting the same quality and viewing experience they used to have with traditional TV systems, and high-resolution videos instantly delivered over the network are becoming a norm today. Therefore, more has to be done for P2P-based streaming services to keep up with such developments.

One compelling way to push the envelope of peer-to-peer video streaming performance is hybrid push-pull approaches by which some chunks are propagated through pushes with the remaining being pulled from the neighborhood [4],[10-11]. It is known that pull-based schemes are robust to unexpected changes such as peer churn and bandwidth fluctuation, while they suffer from longer delivery latency. In contrast, push-based protocols allow for low-delay video delivery, but they are vulnerable to network instability. There have been efforts to explore the possibility of reaping the best of both worlds of push and pull; a hybrid push-pull protocol inherits reduced latency and less control overhead from tree-push mechanisms and protocol simplicity and resiliency to network dynamics from mesh-pull mechanisms.

In this paper, we propose a novel hybrid push-pull protocol that can maximize peer-to-peer streaming performance; by letting the push component expedite video chunk delivery, better streaming performance can be accomplished. The primary ideas of the proposal include a peering strategy based on both delay- and chunk diversity-awareness, and a streaming window adjustment scheme.

The rest of the paper is organized as follows. We start off by reviewing hybrid push-pull swarming protocols in Section 2, which motivates this work. Section 3 proposes a new hybrid push-pull protocol named DP/CP protocol. The proposal is evaluated in our comparative performance study presented in Section 4. Finally, Section 5 summarizes the state of the art in hybrid video streaming technologies, and discusses our future extensions.

II. Hybrid Approach to P2P Streaming

Despite the overhead and delay associated with chunk map exchanges and ensuing chunk fetches, mesh-based protocols have been the mainstream of P2P-based streaming research for the recent years. However, chunk pushing is also possible with mesh streaming overlays. Knowing a neighbor's chunk possession from buffer map exchanges, a peer can voluntarily push missing chunks to it. Chunk

This research was supported by the MSIP (Ministry of Science, ICT&Future Planning), Korea, under the ITRC(Information Technology Research Center) support program (IITP-2015-H8501-15-1006) supervised by the IITP (Institute for Information&communications Technology Promotion) and by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (No. 2013R1A1A2009913).

Digital Object Identifier 10.4316/AECE.2015.04001

delivery delay is consequently reduced from the removal of explicit chunk requests. One notable example of push-based diffusion schemes is DP/LU protocol [12]. According to the protocol, a peer can facilitate network-wide chunk propagation by prioritizing pushing its most recent chunk to a peer owning a set of video chunks least in common with those of the pushing peer. One downside of the scheme is collisions caused by simultaneous pushes of the same chunk from more than one neighbor. More specifically, it is possible for the cost of the chunk collisions to overshadow the gain of decreased delay, especially when chunks are well replicated in the neighborhood.

A. Adaptive Push-Pull Protocols

In order to reduce the redundant chunk transmissions of duplicate pushes, our previous work investigated a hybrid push-pull scheme which switches back and forth between push and pull modes according to chunk replication rate in the network [13]. The protocol aims at achieving the best possible result; video chunks are pushed for a shorter download delay where they are rare in the neighborhood, and chunks are pulled to avoid collisions where they are well replicated. By switching adaptively between push and pull modes, the hybrid protocol is able to strike a balance between chunk propagation speed and network bandwidth efficiency.

The decision of push or pull can be made on the basis either of an individual chunk or a peer's whole chunk set, each of which was named as chunk-wise and peer-wise push-pull protocols. The push-pull decision in the chunk-wise protocol is made on an individual chunk basis; chunk rarity is measured in terms of the number of a particular chunk's occurrences in the neighborhood. It can serve as an indicator that represents the popularity of a chunk with a higher value meaning more common chunks. Chunks with rarity over a certain threshold are transferred via push, because there will be a small chance of bandwidth waste caused by redundant transmissions. Common chunks with low rarity values are propagated via pull mode.

According to the peer-wise push-pull protocol, the push-pull decision is based on a whole chunk set of neighbor peers. The protocol uses an index of buffer map disparity that measures differences among peers' chunk sets. The larger the disparity index, the more diverse chunks among peers, and the less repeated ones. Hence, better likelihood to benefit from chunk pushes due to a less chance of chunk collisions. The buffer map disparity can also be considered in a pair-wise manner instead of for the entire neighbor sets. The full design space of the hybrid push-pull schemes was explored, and their performance was studied in our previous work [13].

III. DP/CP Hybrid Push-Pull Protocol

While the aforementioned push-pull protocols were able to improve video streaming performance to a certain degree, further innovations are needed to support a higher streaming rate. In this section, we present two proposals for P2P-based hybrid streaming: DP/CP (most deprived peer/closest peer first) algorithm and elastic window scheme. The DP/CP algorithm allows our hybrid protocol to make an informed push-pull decision based on the awareness of chunk distribution status and network condition, whereas the elastic window scheme aims at the right balance of chunk dissemination and urgency.

A. DP/CP Push-Pull Protocol

The ability to select proper swarming partners has a decisive impact on P2P-based streaming performance, which has led to the exploration of various peering strategies [11],[14-15]. One attractive approach is to consider underlying network-level parameters such as available bandwidth and network proximity for neighbor peer selection. For instance, RTT can serve as a good indicator of network connection quality that captures distance to a neighbor. It apparently has a non-negligent effect on streaming performance, especially when we consider a communication latency range, 25ms – 500ms, typical of today's Internet environment [11].

RTT-based selection of closer peers likely result in a lesser delay for chunk propagation to its neighbors and throughout the network in the end. Let $\overline{T}_c$ be the average time to transfer a chunk, c, and m the number of swarming neighbors. Then, the minimal time to disseminate the chunk to every peer in the streaming network can be expressed by $\overline{T}_c[\log_m N]$, where N is the peer population in the network. Since m is usually a small fixed number, the other option left for us to reduce the streaming delay is to choose neighbor peers with a smaller round-trip time. However, relying solely on RTT may put the streaming network at risk of developing disconnected islands. RTT ranking-based peer selection would form a cluster of proximal peers somewhat isolated from the rest of the network. This network disconnection adversely affects the streaming performance, causing what is known as chunk starvation or content bottleneck [16].

The precaution we have taken to prevent the bottleneck from occurring is to consider chunk disparity as well as peer latency. Also, the starvation problem is further mitigated by the pull part of our scheduling algorithm as discussed below. Let $DC_i(j)$ represent a chunk set that peer i has and its neighbor peer j does not over the overlap of their download windows, i.e., $C_i(j) - C_j(i)$. Note that the current playback position may vary to a certain extent across streaming peers. Also, $C_i(j)$ indicates a chunk set that peer i owns for the overlapping period of its download window with peer j's. Disparity index d_{ij} measures the difference in chunk sets between peer i and j, i.e., $|DC_i(j)|$. In other words, it indicates the amount of chunks that can be pushed from peer i to j. It is also noted that swarming window size is not the same for all peers; the size is dynamically adjusted according to a function of download buffer status and chunk urgency as described in the next subsection. Then, the disparity index is normalized to a relative disparity D_{ij} in relation to other peers as defined by (1). Again, m indicates the number of peer i's neighbors.

$$D_{ij} = d_{ij} / \Sigma_{k=1}^{m} d_{ik} \quad (1)$$

With the latency between peer i and j defined as $RTT(i, j)$, our latency factor L_{ij} is defined by (2). Based on these definitions, the product of D_i and L_i is used to determine

which peers to push to first. More specifically, peers with a higher value of the product are preferred as push destinations.

$$L_{ij} = \{RTT(i,j)\}^{-1} / \sum_{k=1}^{m} \{RTT(i,k)\}^{-1} \quad (2)$$

Fig. 1 illustrates the basic idea of the proposed DP/CP hybrid scheduling algorithm. Knowing what video chunks its neighbors possess through periodic chunk map exchanges, peer 1 readily determines which chunk set to push to which peer first. According to the hybrid protocol, $D_{12}\times L_{12}$ is computed to 0.23, and $D_{13}\times L_{13}$ to 0.26. Therefore, peer 3 is preferred over peer 2 as the first push destination. Missing chunks are pushed to the selected peer in the order of individual chunk rarity in the neighborhood. In this example, video chunks are propagated in the order of 2, 5, and 7, after which peer 1 pushes chunk 2 and 0 to peer 2.

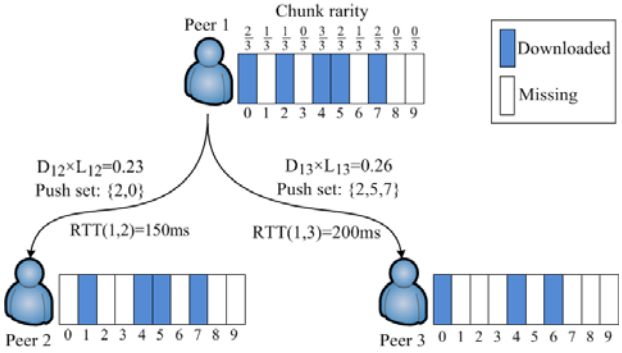

Figure 1. DP/CP hybrid push-pull scheme

Fig. 2 sketches the push and pull threads of the DP/CP hybrid scheduling algorithm. Basically, the protocol prioritizes chunk pushes to expedite their propagation throughout the network by simultaneously taking into account chunk diversity and peer distance. Up to k push candidate chunks are first chosen from the missing chunks of a peer with the highest $D\times L$ with the condition that their rarity is not greater than a certain threshold (β_1). The remaining chunks (i.e., $k - |C|$ chunks) are then filled up on a random basis. This safeguards against excessive duplicate pushes for well replicated chunks in the neighborhood. Chunks to pull are also selected in a similar way; a chunk is chosen from the missing based on its rarity. The selection of a pull chunk is repeated, until its rarity crosses a threshold (β_2). It is considered that those chunks are already sufficiently replicated in the area, so that significant benefit can hardly be expected from an additional copy of them. Rather, it is the time to become concerned about soon-to-play chunks. The rest of the pull set is filled by chunks with the most imminent playback deadlines.

B. Balance of Chunk Dissemination and Chunk Urgency

The idea of the DP/CP protocol is to improve streaming performance by relying on chunk pushes as much as possible without hurting network bandwidth efficiency from the backlash of redundant chunk transfers. As a means to further enhance the performance, we propose an elastic windowing scheme that makes the most of the push-pull hybrid scheme. Basically, a peer's streaming condition can be estimated by keeping track of chunk misses over a certain period of time. Frequent and bursty misses likely indicate that the streaming protocol is struggling to keep up with the video playback rate. Under this circumstance, it would be more desirable to secure urgent video chunks soon needed for playback as quickly as possible than to care about network-wide chunk dissemination. Our elastic window scheme reacts to successive chunk misses by shrinking its window size, which puts more emphasis on imminent chunk download for the node than chunk replication in the network. In contrast, rare misses are viewed as a sign that things go well; the streaming is ahead of where it supposed to be in terms of video chunk playback deadlines. It expands the download window, so that otherwise unused network bandwidth can be used for some chunks that are not in immediate need. By doing so, the protocol shifts itself towards the rarest-first swarming strategy. A peer devotes more resources to fast chunk dissemination for the network, worrying less about bumpy playbacks by missing chunks for itself.

Push thread:
while upload bandwidth available **do**
  $p \leftarrow$ a neighbor with the next highest $D_{ij}\times L_{ij}$
  if $|this.bufferMap - p.bufferMap| > \alpha$ **then**
    sort missing chunks in $p.bufferMap$
      in an ascending order by chunk rarity
    $C \leftarrow$ top k rarest chunks with their rarity $\leq \beta_1$
    while $|C| == k$ **do**
      C += a random chunk out of the missing in $p.bufferMap$
    end while
    push C to p
  else
    break
  end if
end while

Pull thread:
while download bandwidth available **do**
  $p \leftarrow$ a neighbor with the next highest L_{ij}
  if $|this.bufferMap - p.bufferMap| > 0$ **then**
    $C \leftarrow$ top k rarest out of the missing chunks in $this.bufferMap$ with their rarity $\leq \beta_2$
    while $|C| == k$ **do**
      C += the most imminent among the missing
    end while
    send C request message to p
  else
    break
  end if
end while

Figure 2. Push and pull threads of DP/CP hybrid protocol

The proposed scheme grows or shrinks its streaming window dynamically according to video playback continuity. Playback continuity is measured in terms of chunk miss ratio, i.e., the number of missed chunks over the total chunks over a period of time. The window re-adjustment is triggered, when the protocol detects meaningful changes in the miss ratio. More specifically, if a difference of miss ratios between time t and t-1 is greater than a threshold, its new window size becomes $w(t) = w(t\text{-}1)\pm\Delta$. We consider two alternatives for the adjustment as follows.

- *Binary exponential back-off scheme* which reduces the current window size by $2^n \times \Delta$ for a miss event, where n indicates the number of consecutive miss periods. The window grows linearly for no misses.
- *Additive increase/multiplicative decrease (AIMD) scheme* which drastically cuts the window size down to half in the event of a miss. The window also grows gradually for no misses.

In a nutshell, the elastic windowing scheme tries to strike a delicate balance between chunk urgency for the node and chunk dissemination for the network. Along with the DP/CP algorithm that considers chunk dissimilarity as well as network proximity for push-pull decisions, it leads to a synergetic effect in maximizing peer-to-peer video streaming performance.

IV. Performance Simulation Study

In order to prove the efficacy of the proposed hybrid scheme, we performed a simulation study that compares its performance with those of other state-of-the-art protocols. Our streaming protocol simulator is built on PeerSim P2P simulator, being configured with the following setups.

The streaming overlay for the simulation is an unstructured mesh network of about 2,500 nodes on average during the simulation run, each node having 16 randomly-chosen neighbors. A peer starts chunk swarming the moment it joins the overlay. Inter-node delays in the overlay are set to follow the distribution of a known node-to-node latency matrix [17]. Nodes are configured to have a distribution of upload/download bandwidth pairs representative of peer-to-peer streaming networks [18]. 20% of the nodes have 128 kbps and 768 kbps as their upload and download bandwidth, and 40% have 384 kbps and 1,536 kbps. 25% of them are set to have a pair of 1,024 kbps and 3,072 kbps, while the remaining 15% have a pair of 4,096 kbps and 6,144 kbps. Our simulation is driven by a peer behavior model which is based on well-known streaming workload analysis studies. Specifically, a piece-wise-stationary Poisson process is used to model peer arrivals [19], and 90% and 10% of the peers stay in the network by a log-normal distribution and a Pareto distribution, respectively [20]. Consequently, average 76 peers join and leave the streaming overlay every second, keeping the network size close to 2,500 nodes. Video chunks are of 14 Kbyte, and streaming rate is set to 672 kbps, which is translated to 6 chunks per second. Download window has the size of 20 seconds (i.e., 120 chunks) for fixed window cases, and video playback begins 20 seconds after the simulation start.

The performance of the DP/CP protocol is compared with three other protocols: (1) random pull protocol that selects chunks to pull in randomly [2], (2) peer-wise protocol where push-pull decisions are made based on peers chunk diversity [13], and (3) delay-aware protocol which is identical to the DP/CP protocol in Fig. 2, except that peer selections are made solely based on peer latency factor and push chunks are chosen randomly from the selected peer's chunk set. In fact, this delay-aware case is a sub-protocol of DP/CP scheme, and it allows us to estimate contributions from the chunk diversity awareness of DP/CP protocol.

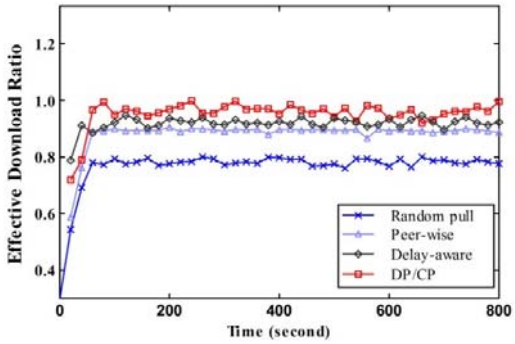

Figure 3. Effective download ratio

We first look at effective download rate that is one of the most important performance metrics for peer-to-peer streaming protocols. It is defined as a ratio of useful download speed (i.e., total downloads minus duplicate and late chunks) to streaming rate. The result showed that our proposal outperforms others. Specifically, the average download rate of DP/CP protocol is measured at 0.97, which is followed by delay-aware push-pull (0.92), peer-wise push-pull (0.89), and random pull protocol (0.8). Poor performance by peer-wise and random pull protocols is attributed to network proximity ignorance and inefficient upload bandwidth utilization, respectively. As presented in Fig. 3, our DP/CP protocol achieves a performance gain of about 9% and 21% over peer-wise and random pull protocols, respectively.

Miss ratio is perhaps the most critical in determining user experiences with video streaming systems. Basically, it indicates the extent of how smoothly the video can be played without freeze or interruption. Miss ratio, i.e., continuity index, is defined as a ratio of the number of chunks that miss their playback deadline to the total chunks. As plotted in Fig. 4, the proposed scheme surpasses other candidates with an average index of 0.043, which is followed by delay-aware hybrid (0.076), peer-wise hybrid (0.107), and random pull protocol (0.212). DP/CP protocol outperforms the peer-wise hybrid and random pull cases by a factor of about 2.5 and 5, respectively. Initial drops at the beginning reflect the startup delay of 20 seconds in the simulation.

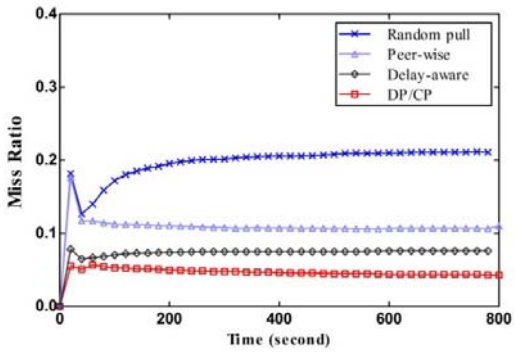

Figure 4. Miss ratio

Fig. 5 plots diffusion rates of the alternatives, which compares chunk propagation rates in terms of the portion of peers that has acquired video chunks since their onset from a video source. The graph shows better performance by our hybrid protocols of delay-aware and DP/CP, converging at

0.94 and 0.96, respectively. Peer-wise and random pull cases are shown to climb to the point of 0.89 and 0.87 at the end. For example, the times to reach the diffusion rate of 60% are 16.3, 17.7, 20.3, and 21.5 seconds, respectively, for DP/CP, delay-aware, peer-wise, and random pull cases. They grow to 20.3, 21.7, 25.9, and 26.4 seconds for 80% diffusion. DP/CP scheme closely trails behind delay-aware case until the overtaking point at 26 second. DP/CP protocol converges at 0.96, higher than 0.94 for delay-aware case, which can be attributed to factoring chunk diversity as well as network proximity into push chunk selection. Also, it is noted that peers come and go all the time during the simulation run, which explains why any of the protocols cannot reach the full rate.

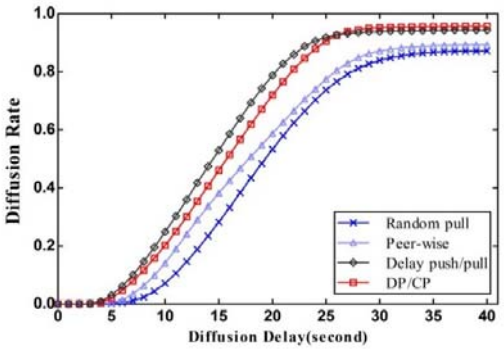

Figure 5. Diffusion rate

In addition, the simulation measures protocol overhead that is made up of control messages and redundant chunk transmissions. Random pull protocol has the least overhead 0.05 on average, which is followed by averages 0.21, 0.25, and 0.25 for peer-wise, delay-aware, and DP/CP protocols, respectively. A breakdown of the results revealed that the overhead is dominated by duplicate chunk transmissions (, which is sometimes an order-of-magnitude larger than control message overhead.) Control message overhead did not show a notable difference among the protocols.

TABLE I. MISS RATIO COMPARISON OF ELASTIC WINDOWS

Window Size	Binary Exponential	AMID
90 (fixed)	12.7	12.7
90 – 120	5.8	6.1
90 – 150	4.3	4.4
90 – 180	4.2	4.0
90 – 210	4.6	4.0
90 – 240	7.2	4.4
240 (fixed)	7.0	7.0

Lastly, Table I compares the chunk miss ratios of binary exponential back-off scheme and AIMD scheme in percentage, as their window size varies. Minimal size of the window is fixed at 90 chunks (, which is worth of 15 seconds playback.) Starting with 120 chunks, the maximum grows by an increment of 30 chunks for each repetition. The binary exponential scheme reaches its lowest at the range of 90 – 180, whereas the AIMD scheme keeps its best performance until one more increment (i.e., until the range of 90 – 210) Lagging behind of the AIMD scheme in reaching the lowest point is expected from the fact that it is more aggressive than the other in reducing its window size in the event of chunk misses. Finally, it is noteworthy that results for the fixed windows of 90 and 240 chunk-wide are also provided for comparison purposes.

V. RELATED WORK

A number of schemes to build hybrid push-pull overlays were investigated [3],[5],[10],[15],[21-22]. The approaches can be classified into mesh-based or tree-based hybrid protocols. A primary idea of mesh-based hybrids is to build an implicit or explicit delivery tree structure on a mesh overlay, so that part of video packets can be pushed over the tree. Without the full complexity to construct a streaming tree from scratch by building on the underlying mesh topology, the hybrid schemes can benefit from low-delay streaming of tree-push delivery.

As a representative example, a video stream can be divided into a set of sub-streams. A small number of initial, successful pulls of sub-streams trigger the push of subsequent chunks of the sub-streams [3],[10]. As a result, the protocols can minimize control overhead associated with pull mechanism. According to tree-based push-pull protocols, streaming chunks are primarily delivered by tree-push method under normal condition, while auxiliary mesh-pulls are used for loss recovery purposes only [21-22]. This way the protocols can minimize adverse effects from the delay and control overhead of pull protocols by having a major portion of video delivered through pushes.

The implication of chunk disparity to streaming performance was reported in the literature [4],[23]. Having a significant impact on playout continuity and startup delay, the disparity index may serve as an indicator to infer peer-to-peer video streaming quality. Furthermore, the index can be utilized as an effective metric to make adaptive push-pull decisions [13]. Also noteworthy development is the research efforts for swarming partnership management that is a key ingredient to P2P streaming protocols. A set of peering strategies has been studied, including network condition-aware schemes [11],[14-15],[24]. However, our DP/CP algorithm considers peer contents as well as network proximity for its swarming partner and push-pull decisions. Being based on both data-driven and network status-driven decisions, the proposal achieves the best outcome; it can maximize peer playback performance, while facilitating faster content dissemination throughout the network.

Our elastic window scheme might be viewed as similar, in spirit, to adaptive scheduling approaches for pull protocols [25-27]. But our scheme uses chunk miss ratio instead of a peer's swarming buffer status to determine a new window size, more importantly, in a hybrid push-pull setting. A larger window likely leads to a greater portion of chunks being pushed, so that the streaming process is expedited without penalizing video playback under way.

Another notable development is that cloud computing research has increasingly been embracing P2P swarming technologies in recent years [28-31]. As a particular example, video chunks can be pushed from the cloud to a small number of selected peers in P2P streaming networks, so that they can serve as a seeder for video swarming to boost the streaming performance [31].

We plan our future research in this direction to explore the synergetic effects of P2P streaming and cloud computing technologies.

VI. Conclusion

This paper presents a new hybrid push-pull protocol for peer-to-peer swarm-based video streaming. The main novelty of our proposal is the DP/CP algorithm that makes push-pull switching decisions based on both video chunk diversity and network proximity. The protocol is further enhanced by incorporating an elastic window scheme that seeks a balance between chunk urgency for the node and chunk replication for the network. The two elements act together to produce a combined synergetic effect that substantially improves peer-to-peer streaming performance. A simulation study demonstrates that the proposed scheme surpasses existing protocols by 9% in terms of effective download ratio and by a factor of 2.5 in terms of chunk miss ratio.

References

[1] J. Liu, S. G. Rao, B. Li, and H. Zhang, "Opportunities and challenges of peer-to-peer Internet video broadcast," Proceedings of the IEEE, vol. 96, no. 1, pp. 11-24, Jan. 2008. [Online]. Available: http://dx.doi.org/10.1109/JPROC.2007.909921

[2] X. Hei, C. Liang, J. Liang, Y. Liu, and K. Ross, "A measurement study of a large-scale P2P IPTV system," IEEE Transactions on Multimedia, vol. 9, no. 8, pp. 1672-1687, Dec. 2007. [Online]. Available: http://dx.doi.org/10.1109/TMM.2007.907451

[3] M. Zhang, Q. Zhang, L. Sun, and S. Yang, "Understanding the power of pull-based streaming protocol: can we do better?," IEEE Journal on Selected Areas in Communications, vol. 25, no. 9, pp. 1678-1694, Dec. 2007. [Online]. Available: http://dx.doi.org/10.1109/JSAC.2007.071207

[4] X. Hei, Y. Liu, and K. Ross, "Inferring network-wide quality in P2P live streaming systems," IEEE Journal on Selected Areas in Communications, vol. 25, no. 9, pp. 1640-1654, Dec. 2007. [Online]. Available: http://dx.doi.org/10.1109/JSAC.2007.071204

[5] D. Kim, E. Kim, and C. Lee, "Efficient peer-to-peer overlay networks for mobile IPTV services," IEEE Transactions on Consumer Electronics, vol. 56, no. 4, pp. 2303-2309, Nov. 2010. [Online]. Available: http://dx.doi.org/10.1109/TCE.2010.5681104

[6] C. Lee, E. Hwang, and D. Pyeon, "A popularity-aware prefetching scheme to support interactive P2P streaming," IEEE Transactions on Consumer Electronics, vol. 58, no. 2, pp. 382-388, May 2012. [Online]. Available: http://dx.doi.org/10.1109/TCE.2012.6227437

[7] L. Yu, L. Gao, J. Zhao, and X. Wang, "SonicVoD: a VCR-supported P2P-VoD system with network coding," IEEE Transactions on Consumer Electronics, vol. 55, no. 2, pp. 576-582, May 2009. [Online]. Available: http://dx.doi.org/10.1109/TCE.2009.5174425

[8] L. Garcia, L. Arnaiz, F. Alvarez, and J. M. Menendez, "Protected seamless content delivery in P2P wireless and wired networks," IEEE Wireless Communications, vol. 16, no. 5, pp. 50-57, Oct. 2009. [Online]. Available: http://dx.doi.org/10.1109/MWC.2009.5300302

[9] N. Ramzan, E. Quacchio, T. Zgaljic, S. Asioli, L. Celetto, E. Izquierdo, and F. Rovati, "Peer-to-peer streaming of scalable video in future Internet applications," IEEE Communication Magazine, vol. 49, no. 3, pp. 128-135, Mar. 2011. [Online]. Available: http://dx.doi.org/10.1109/MCOM.2011.5723810

[10] B. Li, S. Xie, Y. Qu, G. Y. Keung, C. Lin, J. Liu, and X. Zhang, "Inside the new CoolStreaming: principles, measurements, and performance implications," in Proc. the 27th IEEE International Conference on Computer Communications, Phoenix, AZ, USA, Apr. 2008, pp. 1705-1713.

[11] A. Russo and R. L. Cigno, "Delay-aware push/pull protocols for live video streaming in P2P systems," in Proc. IEEE International Conference on Communications, Cape Town, South Africa, May 2010, pp. 1-5.

[12] T. Bonald, L. Massoulie, F. Mathieu, D. Perino, and A. Twigg, "Epidemic live streaming: optimal performance trade-offs," in Proc. ACM International Conference on Measurement and Modeling of Computer Systems, Annapolis, MD, USA, Jun. 2008, pp. 325-336.

[13] D. Jo, S. Helal, E. Kim, W. Lee, and C. Lee, "Adaptive push-pull protocols for P2P-based video streaming," IEICE Transactions on Communications, vol. E94-B, no. 10 pp. 2759-2762, Oct. 2011.

[14] D. Ren, Y. H. Li, and S. G. Chan, "On reducing mesh delay for peer-to-peer live streaming," in Proc. the 27th IEEE International Conference on Computer Communications, Phoenix, AZ, USA, Apr. 2008, pp. 1732-1740.

[15] A. Ghanbari, H. R. Rabiee, M. Khansari, and M. Salehi, "PPM – A hybrid push-pull mesh-based peer-to-peer live video streaming protocol," in Proc. the 21st International Conference on Computer Communications and Networks, Munich, Germany, Jul. 2012, pp. 1-8.

[16] N. Magharei and R. Rejaie, "PRIME: peer-to-peer receiver-driven mesh-based streaming," IEEE/ACM Transactions on Networking, vol. 17, no. 4, pp. 1052-1065, Aug. 2009. [Online]. Available: http://dx.doi.org/10.1109/TNET.2008.2007434

[17] B. Wong, A. Slivkins, and E. Sirer, "Meridian: a lightweight network location service without virtual coordinates," in Proc. ACM SIGCOMM 2005, Philadelphia, PA, USA, Aug. 2005, pp. 85-96.

[18] C. Ashwin, R. Bharambe, and V. Padmanabhan, "Analyzing and improving a BitTorrent network's performance mechanisms," in Proc. the 26th IEEE International Conference on Computer Communications, Barcelona, Spain, Apr. 2006, pp. 1-12.

[19] E. Veloso, V. Almeida, W. Meira, A. Bestavros, and S. Jin, "A hierarchical characterization of a live streaming media workload," IEEE/ACM Transactions on Networking, vol. 14, no. 1, pp. 133-146, Feb. 2006. [Online]. Available: http://dx.doi.org/10.1109/TNET.2005.863709

[20] K. Sripanidkulchai, B. Maggs, and H. Zhang, "An analysis of live streaming workloads on the Internet," in Proc. the 4th ACM Internet Measurement Conference, Taormina, Italy, Oct. 2004, pp. 41-54. [Online]. Available: http://dx.doi.org/10.1145/1028788.1028795

[21] F. Wang, Y. Xiong, and J. Liu, "mTreebone: a hybrid tree/mesh overlay for application-layer live video multicast," in Proc. the 27th IEEE International Conference on Distributed Computing Systems, Toronto, Canada, Jun. 2007, pp. 49-56.

[22] F. Wang, J. Liu, and Y. Xiong, "Stable peers: existence, importance, and application in peer-to-peer live video streaming," in Proc. the 27th IEEE International Conference on Computer Communications, Phoenix, AZ, USA, Apr. 2008, pp. 2038-2046.

[23] C. Wu, B. Li, and S. Zhao, "Diagnosing network-wide P2P live streaming inefficiencies", in Proc. the 28th IEEE International Conference on Computer Communications - Mini-Conference, Rio de Janeiro, Brazil, Apr. 2009, pp. 2731-2735.

[24] X. Zhang and H. Hassanein, "TreeClimer: a network-driven push-pull hybrid scheme for peer-to-peer video live streaming," in Proc. the 35th IEEE Conference on Local Computer Networks, Denver, CO, USA, Oct. 2010, pp. 368-371. [Online]. Available: http://dx.doi.org/10.1109/LCN.2010.5735745

[25] C. G. Gurler, S. S. Savas, and A. M. Tekalp, "Variable chunk size and adaptive scheduling window for P2P streaming of scalable video," in Proc. the 19th IEEE International Conference on Image Processing, Orlando, FL, USA, Sep. 2012, pp. 2253-2256.

[26] C. Lee and E. Kim, "Boosting P2P streaming performance via adaptive chunk selection," IEICE Transactions on Communications, vol. E94-B, no.10, pp. 2755-2758, Oct. 2011.

[27] A. Vlavianos, M. Iliofotou, and M. Faloutsos, "BiToS: Enhancing BitTorrent for supporting streaming applications," in Proc. IEEE INFOCOM Global Internet Symposium 2006, Barcelona, Spain, Apr. 2006, pp. 1-6.

[28] D. Wu, Y. Zeng, J. He, Y. Liang, and Y. Wen, "On P2P mechanisms for VM image distribution in cloud data centers: modeling, analysis, and improvement," in Proc. the 4th IEEE International Conference on Cloud Computing Technology and Science, Taipei, Taiwan, Dec. 2012, pp. 50-57. [Online]. Available: http://dx.doi.org/10.1109/CloudCom.2012.6427568

[29] R. Bruno and P. Ferreira, "SCADAMAR: Scalable and data-efficient Internet MapReduce," in Proc. the 2nd International Workshop on CrossCloud Systems, Bordeaux, France, Dec. 2014. [Online]. Available: http://dx.doi.org/10.1145/2676662.2676673

[30] X. Leon, R. Chaabouni, M. Sanchez-Artigas, and P. Garcia-Lopez, "Smart cloud seeding for BitTorrent in datacenters," IEEE Internet Computing, vol.18, no.4, pp.47-54, Jul.-Aug. 2014. [Online]. Available: http://dx.doi.org/10.1109/MIC.2014.43

[31] A. H. Payberah, H. Kavalionak, V. Kumaresan, A. Montresor, and S. Haridi, "Clive: cloud-assisted P2P live streaming," in Proc. the 12th International Conference on Peer-to-Peer Computing, Tarragona, Spain, Sep. 2012, pp. 79-90.