行政院國家科學委員會補助產學合作研究計畫

iContent: 實現數位優質生活之智慧型個人影音服務

子計畫四:P2P 同儕式視訊傳輸技術

研究工作詳述

In this project we implement different schedulers on a discrete-time event-driven peer-to-peer streaming simulator[1] provided by Zhang. For a fair comparison, all schedulers use the same overlay topology and simulation configuration. Following, we describe simulation configuration and evaluation metrics used in this work. Then, the discussion of impact of different parameter settings on the whole system is provided.

Simulation Configuration

We assume all nodes in our simulation are DSL/Cable link users, that is, upload and download bandwidth are asymmetric, and its distribution is shown in Table. 1. Note that the average outbound capacity is about 345kbps, which is 1.15 times of the average source streaming rate (i.e., bandwidth amplification ratio = 1.15). All nodes participate the session in the initial 5 seconds and each node randomly selects 15 nodes, suggested in [17], as its neighbor to construct a random overlay. All nodes stay until the simulation is terminated. The buffer map exchange interval is set to 500ms and the request period is set to 250ms. The upload capacity of the single source node is set to 2Mbps. Blocks are equal-sized with 1450 bytes. Moreover, sliding window and streaming buffer are set to 12 sec and 62 sec, respectively.

To generate the test video trace for streaming, we concatenate different types of CIF video sequence, which includes high motion (e.g. foreman and football) and low motion (e.g., akiyo and news) video sequences, to form roughly 4-minute test video sequence and encode the test sequence using dyadic encoding structure supported by JSVM 8.0[2] (the H.264/SVC reference software). To obtain a quality-smooth encoded stream, fixed quantization parameter (QP) for all GOP is employed. QP =34 is used to obtain a roughly 300 kbps video stream. We adopt frame-copy supported in JSVM8.0 to conceal errors caused by receiving deadline or other network conditions. The same video trace is repeatedly used in our simulation. These

[1] http://media.cs.tsinghua.edu.cn/~zhangm

[2] cvs –d :pserver:jvtuser:jvt.Amd.2@garcon.ient.rwth-aachen.de:/cvs/jvt login
cvs –d :pserver:jvtuser@garcon.ient.rwth-aachen.de:/cvs/jvt checkout jsvm

configurations are summarized in Table 2.

Evaluation Metrics

On the other hand, to evaluate the performance of each method, we employ (1) *Block-level Delivery Ratio* (2) *Application-level Delivery Ratio* (3) *Peak-Signal-to-Noise-Ratio* (*PSNR*) to evaluate the pure network throughput, the upper application-level perceived throughput and ultimately measured visual quality, respectively. They are defined as follows:

$$\text{Block-Level Delivery Ratio} = \frac{\text{The total size of in-time received blocks}}{\text{The total size of blocks should be received}} \quad (1)$$

$$\text{Application-Level Delivery Ratio} = \frac{\text{The total size of in-time received frame}}{\text{The total size of frames should be received}} \quad (2)$$

$$\text{PSNR} = 10\log_{10}\frac{255^2}{\frac{1}{XY}\sum_{x=1}^{X}\sum_{y=1}^{Y}[s_{ori}(x,y) - s_{recon}(x,y)]^2} \quad (3)$$

Note that due to the high decoding complexity, we merely sample 300 nodes for PSNR computation from 1000 nodes by default.

Performance Comparison

Then we give the comparison of the proposed method with following methods:

Random: Each node independently requests each of its desired blocks from a random selected neighbor who has the block. Chainsaw [6] applies this method.

***Local Rarest First* (*LRF*)**: The blocks of fewer owners from the local view of a node should be requested first. Both BitTorrent [3] and CoolStreaming/DONet [5] employ this method.

Quality-Optimized: The proposed decentralized method aiming to maximize the perceptual visual quality, which is solved via min-cost flow.

Throughput-Optimized: Similar to ***Quality-Optimized.*** Differently, visual quality gain function is set to a fixed value for all blocks. In this case, each block is regarded as equal so that a maximum throughput schedule is figured out.

Rarity-Priority-Optimized: This method aims to maximize the overall throughput using the concept that blocks with rarity should be requested first to make them spread more quickly. Similarly, just replace our visual quality gain function $Q_G(\cdot)$ to their rarity function $P_R(\cdot)$ [7].

Table 1: The network capacity distribution among peers

Peer Inbound (kbps)	Peer Outbound (kbps)	Peer Ratio (%)
3000	1000	10

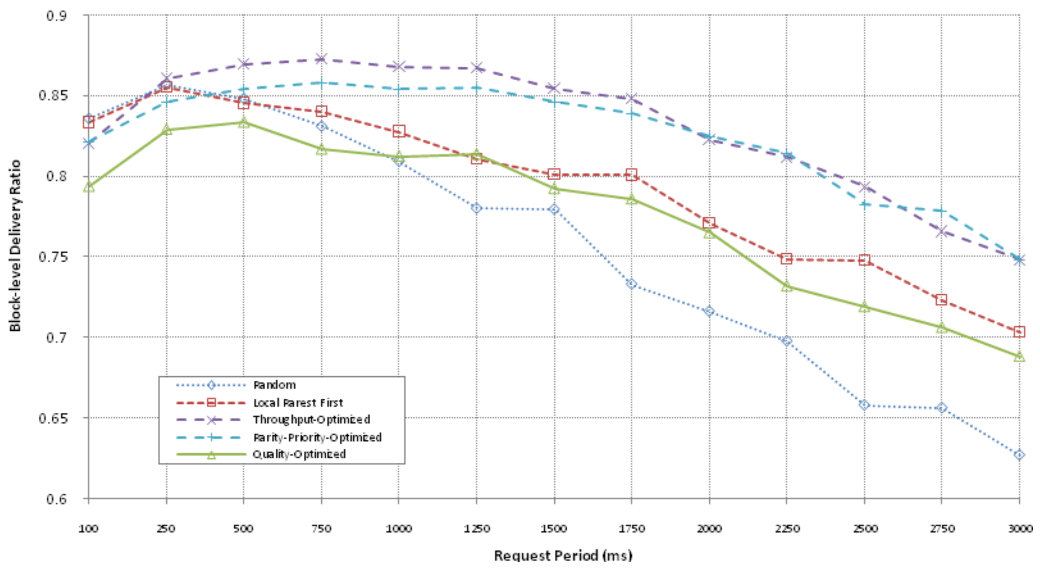
1500	384	50
768	128	40

Table 2: The settings of simulation parameter

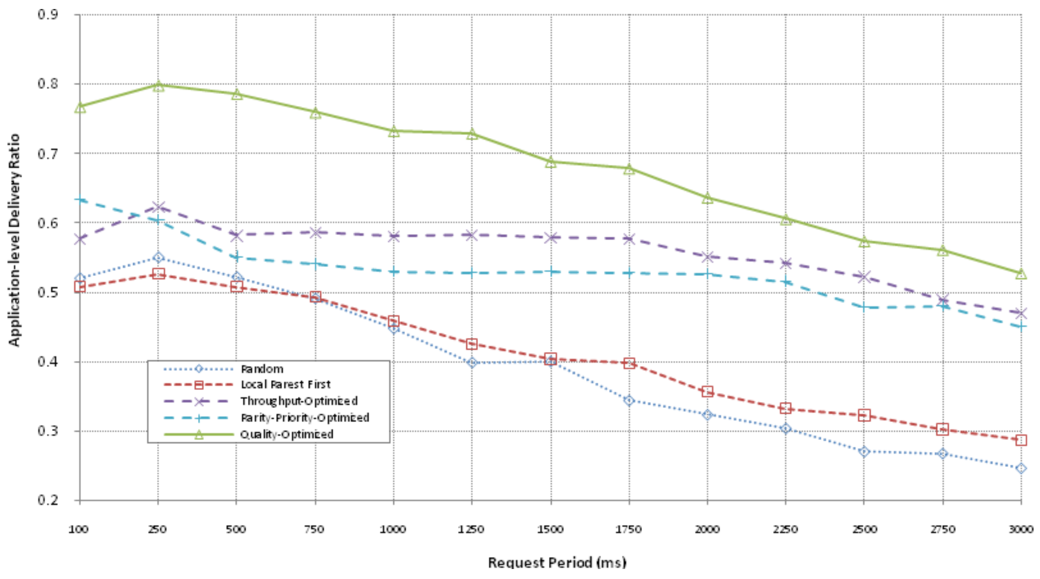
Parameter	**Value**
Simulation Time	1200 s
Request Period	250 ms
Buffer Map Interval	500 ms
Neighbor Count	15
Block	1450 bytes
Video	H.264/SVC, JSVM 8.0, CIF, QP=34, 300 kbps
Error Concealment	Frame-Copy in JSVM 8.0
Source Capacity	2000 kbps
Sliding Window	12 s
Streaming Buffer	62 s

Following, we study the impact of different parameter settings, e.g. request period, the size of sliding window and aggressiveness factor λ, on the performance of the whole system so as to explore proper settings. From Fig. 4 to Fig. 7, we can observe that when delivery ratio is transformed from block-level to application-level, in addition to the proposed scheduler, the delivery ratios of all schedulers are significantly decayed. This is because only some of received blocks can be used to form decodable frames. Most of them are regarded as incomplete as the case in Fig. 1. On the other hand, it is worth noting that since the proposed scheduler does control the distribution of received blocks, the gap between block-level and application-level is reduced. Therefore, compared with other schedulers, the proposed scheduler can provide more useful throughput to upper application.

We first study the block-level and application-level delivery ratio curves for different request period in Fig. 4 (a) and (b), respectively. It can be seen that shorter request period is needed for better performance, especially when a smaller sliding window is preferred. This is because when the request period is smaller, a block earns more chances of being re-requested and can be retried after its request time-out more quickly. Nevertheless, as the request period is below 250ms or even smaller, a peer attempts to send its requests frequently before time-outs of previous requested blocks and merely incurs additional overhead. Hence the overall throughput is decayed. So, we set the request period to 250ms to provide better an application-level throughput.



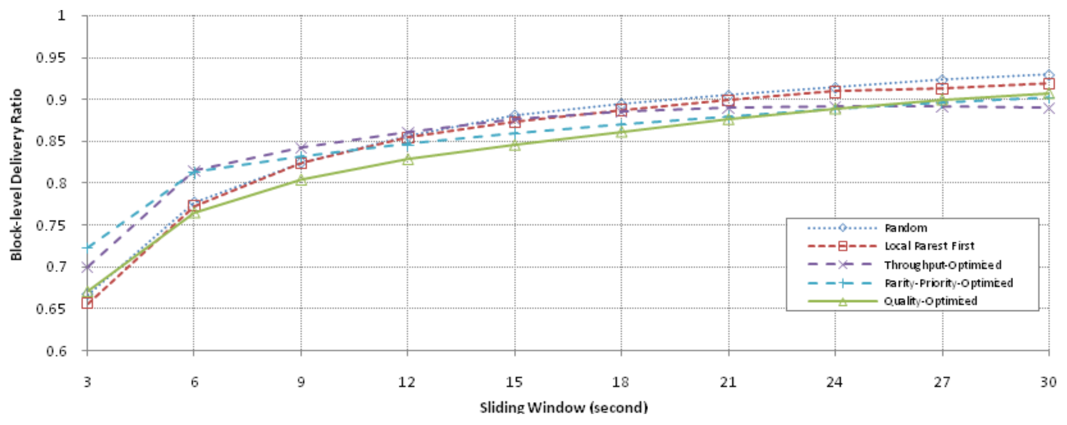
(a)



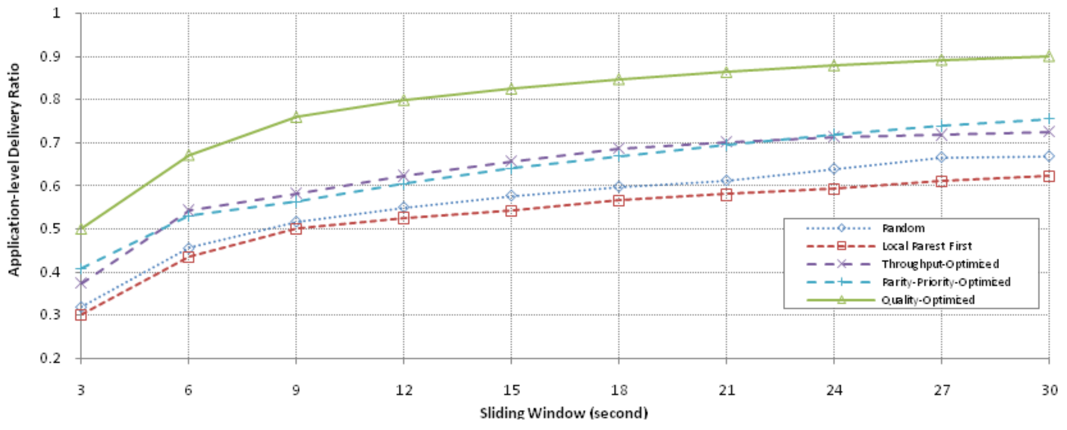
(b)

Fig. 4 (a)(b): Block-level Delivery Ratio and Application-level Delivery Ratio over different Request Period

As illustrated in Fig. 5, a larger sliding window implies that a block stays in it longer; hence a peer has more opportunities of re-requesting it, similar to the condensation of request period. However, a large sliding window means longer startup latency and larger watching delay. Therefore, we set the window size to 12 seconds and condense the request period instead.



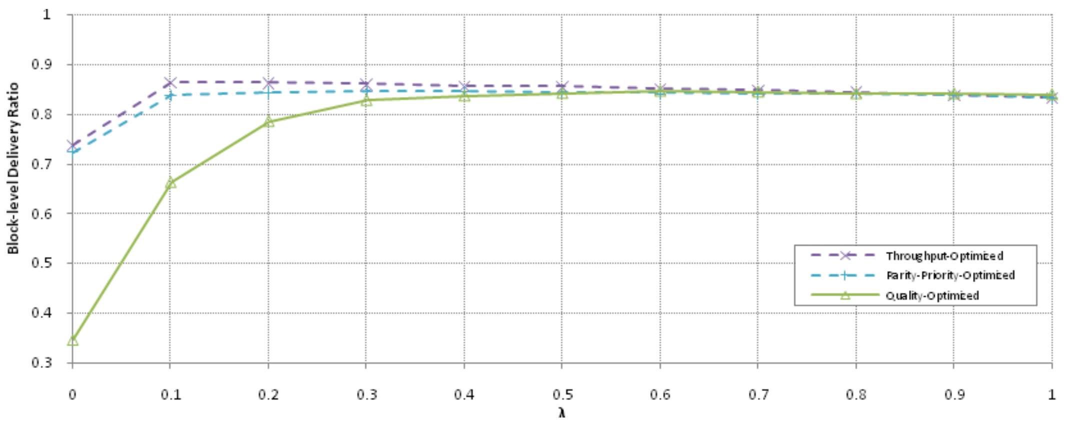
(a)



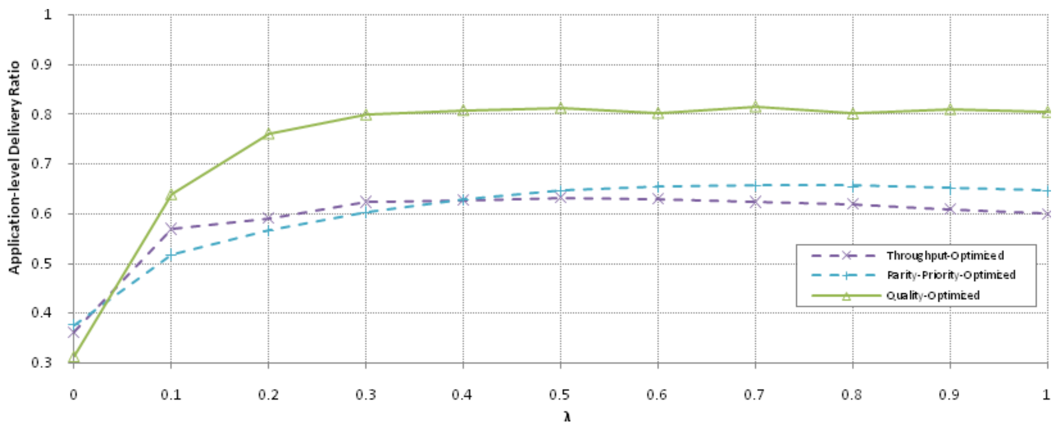
(b)

Fig. 5 (a)(b): Block-level Delivery Ratio and Application-level Delivery Ratio over different Sliding Window

We investigate the impact of different bandwidth allocation aggressiveness factor λ in Fig. 6 by varying the factor λ under different values in [0,1]. It is interesting to see that, due to the variation of source streaming rate, the historical estimator cannot provide good throughput. As the aggressiveness factor is in [0.3,1.0], the historical estimator seems to have little effect on improving the performance of throughput. However, as the bandwidth supply ratio is small, a too aggressive bandwidth allocation strategy can over-estimate and causes peers congested more frequently while a smoother bandwidth allocation benefits overall performance then. Such a phenomenon can be more apparent as the bandwidth supply ratio goes down; therefore we set the aggressiveness factor λ to 0.3 as a more conservative compromise.



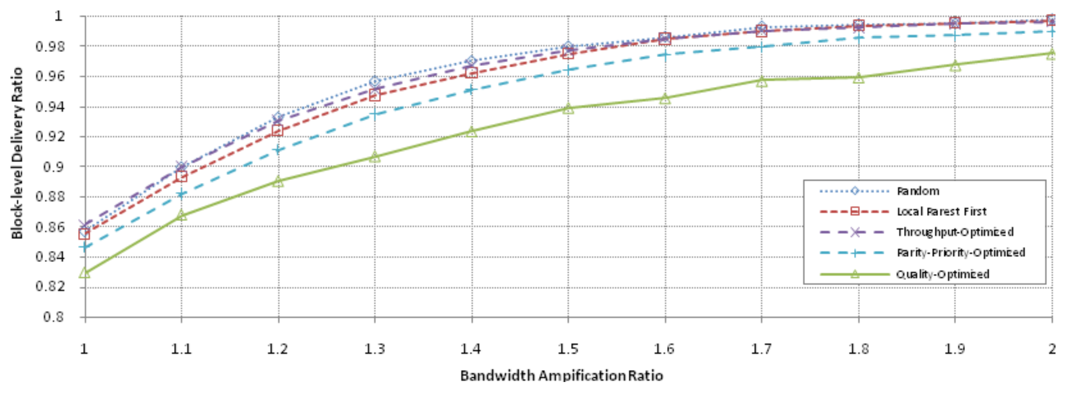
(a)



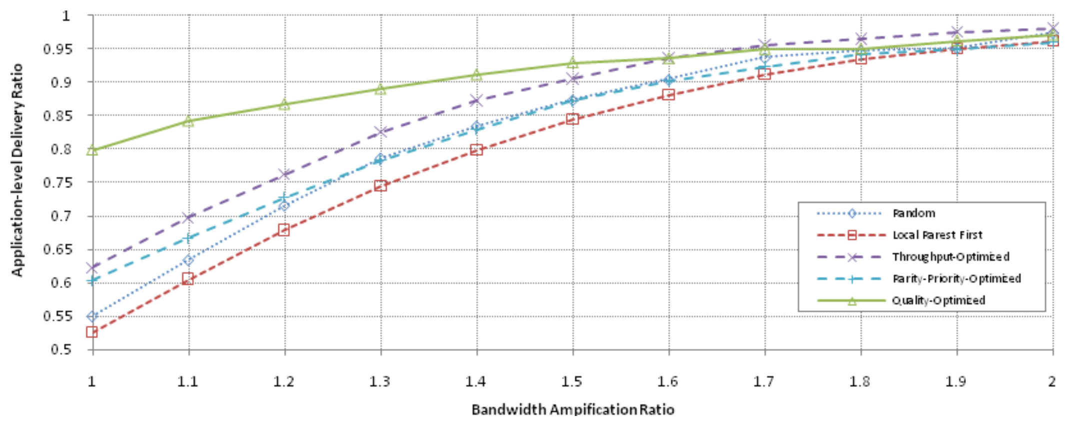
(b)

Fig. 6 (a)(b): Block-level Delivery Ratio and Application-level Delivery Ratio over different Bandwidth Allocation Aggressiveness Factor

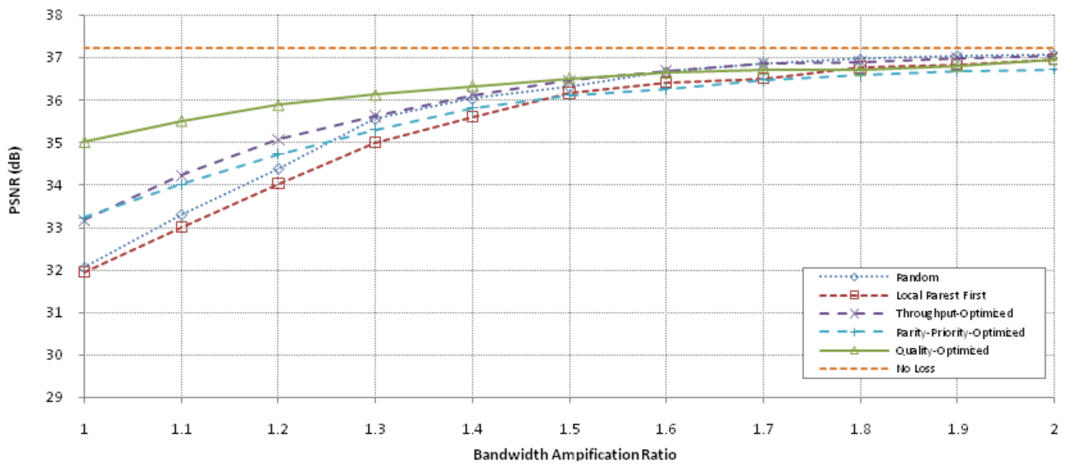
Next, we study the effect of different peer upload capacities on the system. As shown in Fig. 7, we adjust the peer upload capacity according to the amplification ratio, i.e., the outbound bandwidth of each peer i varies from its original value O_i to $2O_i$. We can observe that although our scheduler provides a slightly lower block-level delivery ratio (within 5%), compared with Throughput-Optimized, Rarity-Priority-Optimized, Random, and LRF schedulers, the gain of corresponding application-level delivery ratio are 17%, 19%, 25%, and 27%, and the gain of average PSNR are 1.84dB, 1.79dB, 2.97dB, 3.07dB, respectively. This is because our scheduler is to maximize the rate-distortion benefit instead of throughput. On the other hand, with the increase of bandwidth amplification ratio, the delivery ratios go up and the gaps between the compared approaches get smaller. This is because network resources are too plentiful so that all delivery ratios tend to approach to 1.



(a)



(b)



(c)

Fig. 7 (a)(b)(c): Block-level Delivery Ratio, Application-level Delivery Ratio and Average PSNR over different Bandwidth Amplification Ratio

To represent the impact of variable bit rate, we encode two video sequences which are consisted of several test sequences. Both videos have a common average bit rate of 300kbps but with different variation of source streaming rate. The first one has some rapid scene changes of very high bit rate while the curve of second one is smoother, as depicted in Fig. 8

and Fig. 9, respectively.

The PSNR values provided by different methods for the two videos are shown in Fig. 10 and Fig. 11. It again demonstrates the importance of exploiting content information for the control of distribution of fetched blocks. Our method substantially improves the PSNR up to 1.7 dB on average compared with others. Both the results of the two video sequences show that our scheduler can improve the perceptual visual quality while accommodating to source streaming rate variation and insufficient network capacity. This indicates that content-aware strategies can effectively improve the visual quality under the same limitation of network capacity.

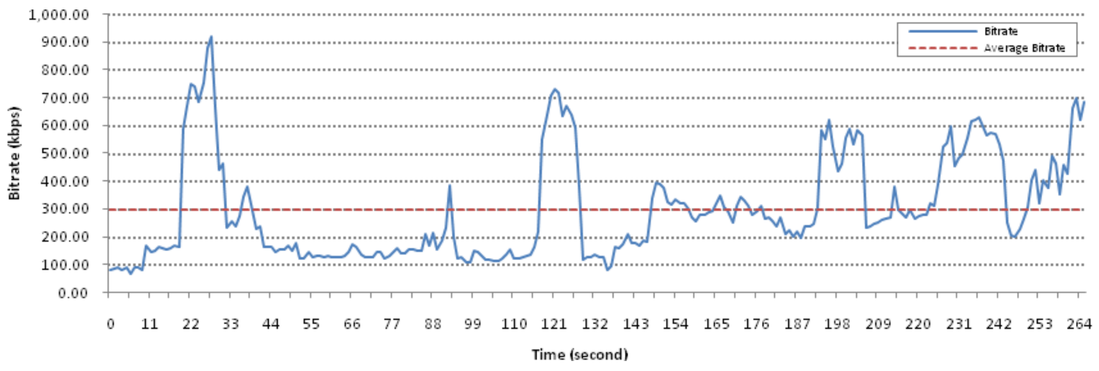


Fig. 8: Bit rate Distribution of the High Variation Video

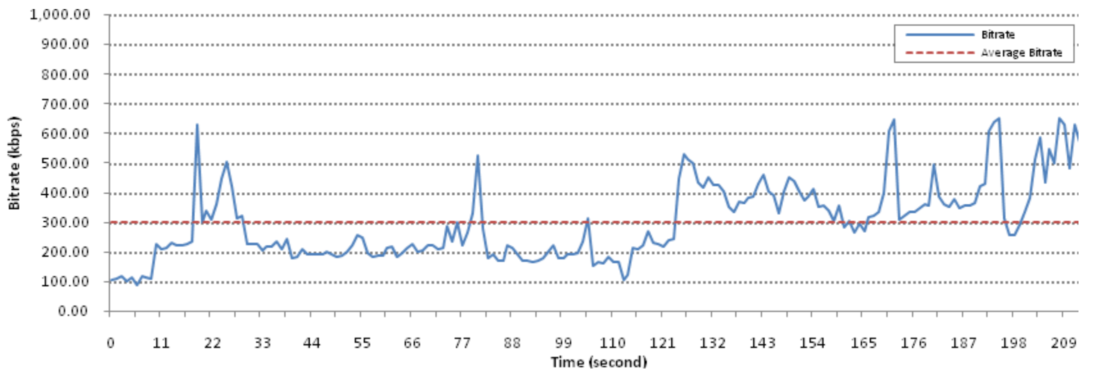


Fig. 9: Bit rate Distribution of the Medium Variation Video

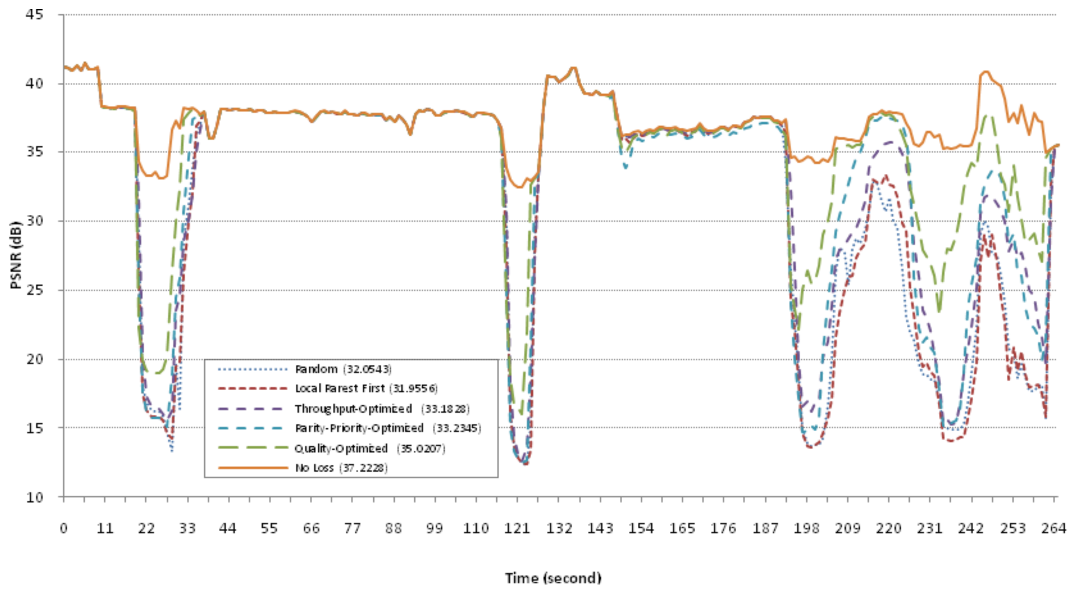


Fig. 10: PSNR over Time (High Variation Video)

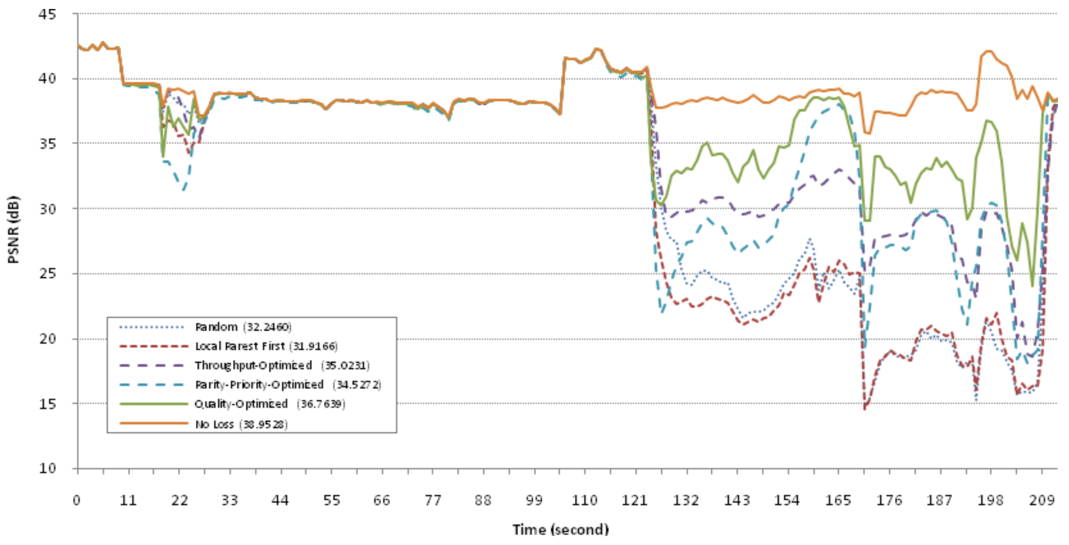


Fig. 11: PSNR over Time (Medium Variation Video)

[1] "PPLive," 2007.[Online]. Available:http://www.pplive.com/

[2] "Gridmedia," 2006[Online]. Available:http://www.gridmedia.com.cn/

[3] Y. Chu, S. G. Rao and H. Zhang, "A Case for End System Multicast," in Proc. of ACM Sigmetrics, June 2000.

[4] J. Jannotti, D. K. Gifford, K. L. Johnson, M. F. Kaashoek and J. W. O. Jr., "Overcast: Reliable Multicasting with an Overlay Network," in Proc. of OSDI, October 2000.

[5] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron and A. Singh, "SplitStream: High-bandwidth Content Distribution in Cooperative Environments," in Proc. of SOSP, October 2003.

[6] V. N. Padmanabhan, H. J. Wang, P. A. Chou and K. Sripanidkulchai, "Distributing

Streaming Media Content Using Cooperative Networking", in Proc. of NOSSDAV, May 2002.

[7] E. Setton and B. Girod, ”Peer-to-Peer Video Streaming”. New York: Springer, 2007.

[8] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum, “Coolstreaming/donet: A data-driven overlay network for efficent media streaming,” in IEEE INFOCOM 2005, Miami, US, Mar. 2005.

[9] V. Pai, K. Kumar, and et al, “Chainsaw: Eliminating trees from overlay multicast,” in IEEE INFOCOM 2005, Conell, US, Feb. 2005.

[10] M. Zhang, Q. Zhang, and S.-Q. Yang, “Understanding the Power of Pull-based Streaming Protocol: Can We Do Better?,” in IEEE Journal on Selected Areas in Communications, special issue on Advances in Peer-to-Peer Streaming Systems, vol. 25, no. 8, Dec. 2007.

[11] M. Zhang, Y.-Q. Xiong, Q. Zhang and S.-Q. Yang. “Optimizing the Throughput of Data-Driven Peer-to-Peer Streaming,” in IEEE Transactions on Parallel and Distributed System, 2008.

[12] N. Magharei and R. Rejaie, “Prime: Peer-to-peer receiver-driven mesh based streaming,” in IEEE INFOCOM 2007, Anchorage, Alaska, USA, May 2007.

[13] A. J. Ganesh, A.-M. Kermarrec and L. Massoulie, “Peer-to-peer membership management for gossip-based protocols,” IEEE Transactions on Computers, vol. 52, no. 2, Feb. 2003.

[14] B. Cohen, “Bittorrent: http://bitconjuer.com.”

[15] P. A. Chou and Z.-R. Miao,”Rate-Distortion Optimized Streaming of Packetized Media,” IEEE Transactions on Multimedia, vol. 8, no. 2, Apr. 2006.

[16] Y. Cui and K. Nahrstedt, “Layered peer-to-peer streaming,” in NOSSDAV, 2003.

[17] R. Rejaie and A. Ortega, “PALS: Peer-to-Peer Adaptive Layered Streaming,” in NOSSDAV, 2003.

[18] V. Venkataraman and P. Francis, “On heterogeneous overlay construction and random node selection in unstructured p2p networks,” in IEEE INFOCOM 2006, Barcelona, Spain, Apr. 2006.

[19] J. Jiang and K. Nahrstedt, “Randpeer: Membership management for Qos sensitive peer-to-peer applications,” in IEEE INFOCOM 2006, Barcelona, Spain, Apr. 2006.

[20] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, Network Flows: Theory, Algorithms, and Applications. Prentice Hall.

[21] C. Y. Chang, C. F. Chou, D. Y. Chan, T.-G. Lin and M. H. Chen, "A q-Domain Characteristic-Based Bit Rate Model for Video Transmission", in IEEE Transactions on Circuits and Systems for Video Technology,2008.

[22] Z. Miao and A. Ortega, “Optimal Scheduling for Streaming of Scalable Media,” in Proc. Asilomar, November 2000.

[25] V. Goyal, “Multiple Description Coding: Compression Meets the Network,” in IEEE Signal Processing Magazine, vol. 18, no. 5, pp.74–93, Sept. 2001.

[26] Nazanin Magharei, Reza Rejaie, and Yang Guo. “Mesh or multiple-tree: A comparative study of live P2P streaming approaches,” in IEEE INFOCOM, Anchorage, May 2007.

[27] M. Dai and D. Loguinov,”Analysis of Rate-Distortion Functions and Congestion Control in Scalable Internet Video Streaming,” in NOSSDAV, 2003.

[28] E. Setton, J. Noh and B. Girod, ”Rate-Distortion Optimized Video Peer-to-Peer Multicast Streaming,” in ACM P2PMMS, 2005.

[29] E. Setton, P. Baccichet and B. Girod, ” Peer-to-Peer Live Multicast: A Video Perspective,” in Proceedings of IEEE, 2008.

[30] P. Baccichet, T. Schierl, T. Wiegand, B. Girod, “Low-Delay Peer-to-Peer Streaming using Scalable Video Coding,” in Proc. International Packet Video Workshop, 2007.

[31] Text of ISO/IEC 14496-10:2005/FDAM 3 Scalable Video Coding, Joint Video Team (JVT) of ISO-IEC MPEG & ITU-T VCEG, Lausanne, N9197, Sep. 2007.

[32] H. Chi, Q. Zhang, J. Jia and X. Shen,” Efficient Search and Scheduling in P2P-based Media-on-Demand Streaming Service,” in IEEE Journal on Selected Areas in Communications, vol. 25, no. 1, Jan. 2007.

[33] H. Schwarz, D. Marpe and T. Wiegand,”Overview of the Scalable Video Coding Extension of the H.264/AVC standard,” in IEEE Transactions on Circuits and Systems for Video Technology, vol. 17, no. 9, Sep. 2007.

[34] *Information Technology-JPEG 2000 Image Coding System: Core Coding System*, ISO/IEC JTC1/SC29/WG1, Tech. Rep. ISO/IEC 15444-1, Jan. 2001.

[A] M. Magharei and R. Rejaie, ”Understanding mesh based peer-to-peer streaming,” in ACM NOSSDAV 2006, Newport, Rhode Island, USA, 2006.