Adaptive Video Streaming: a Survey and Case Study

HU, Pili
MobiTeC, IE Department, CUHK
hupili@ie.cuhk.edu.hk
http://personal.ie.cuhk.edu.hk/~hpl011/

December 20, 2011

Abstract

In the past decade, Internet traffic has seen a significant change from web browsing to video viewing. The ongoing trend raises a challenging problem: how to stream data to heterogeous peers?

The designer of such data streaming architecture should bear the following considerations in mind: QoE, server load, network resource efficiency, scalability, etc. The heterogeneous peer network condition makes the design more complicated. The underlying codec ranges from Multi Description Coding to Multi Layer Coding. The data deliver architecture ranges from unicast, multicast, to P2P network. Researchers have focused on different system settings and optimization objectives.

This paper will first sum up several works in the context of adaptive video streaming. At the same time, we do a case study on a commercial adaptive video streaming system, which combines Multilayer Codec and P2P technology. Possible improvements on this system are proposed with reasoning. Some of the conjectures are verified through a corresponding simulation platform based on NS2.

Contents

1 Introduction

- User perceived experience.
- Vendor cost.
- User cost.
- System cost.

2 General Model

3 Problem Scope

3.1 Codec

3.2 Networking

4 Design of Adaptive P2P VoD System

4.1 Codec Choice

4.2 Transimission Protocol Choice

4.3 Overlay Construction

4.4 Peer Selection

4.5 Buffer Management

4.6 Chunck Selection

MMKP, Knapsack
MCMF, Network Flow

4.7 Playback Decision

4.8 User Model

5 A Case Study and Simulation

5.1 Baseline Description

- Blue: Real deployment
- While: Simulation platform
- Green: This work
- Red dash: Equivalency

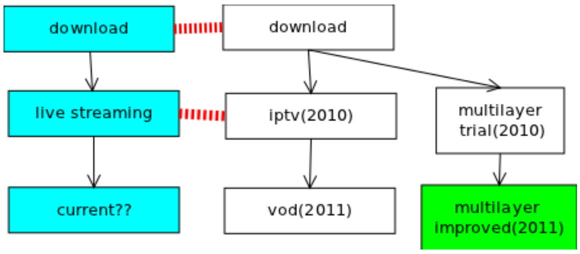
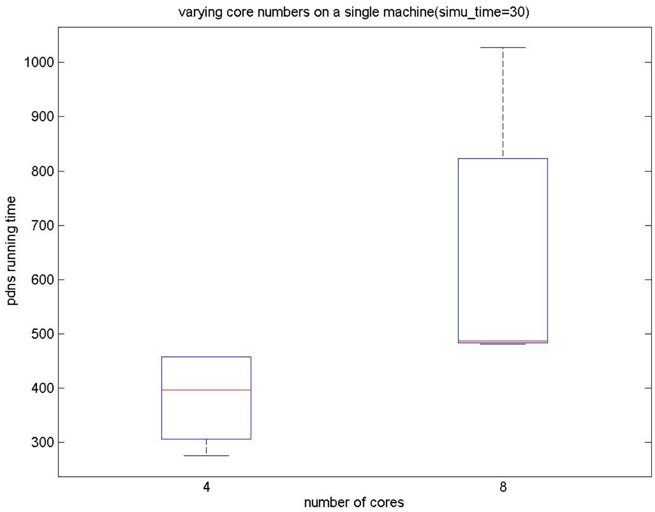


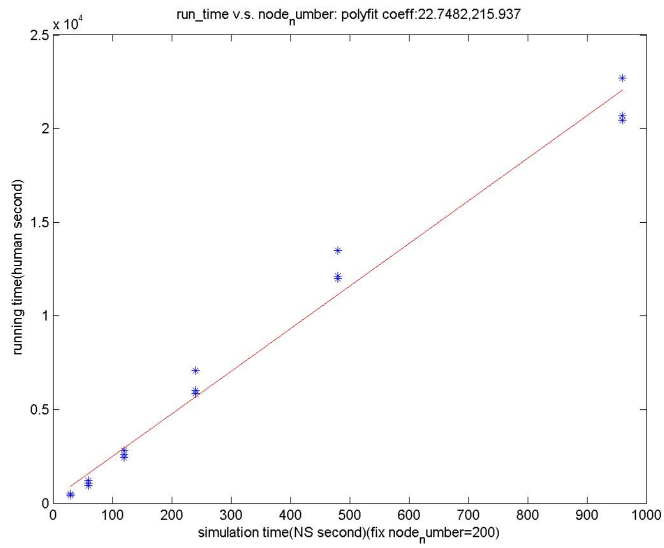
Figure 1: Version Tree of Real and Simulation

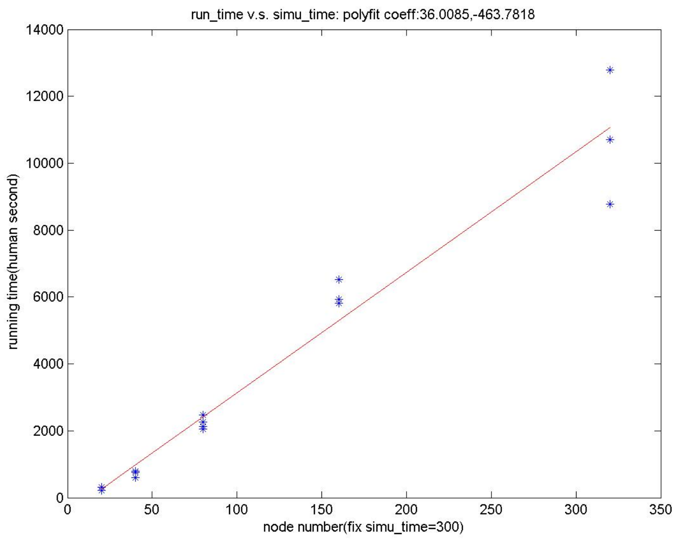


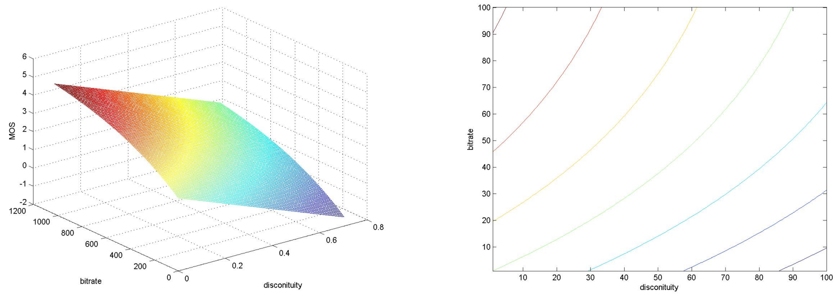
- Number of Nodes: 160
- Simulation Time: 300 (NS seconds)

Wen Zheng's configuration:

- Star like, 4 subnet.
- Subnet parameters:
  - Subnet1: down:10Mb; up:10Mb.
  - Subnet1: down:3Mb; up:0.5Mb.
  - Subnet1: down:3Mb; up:0.5Mb.
  - Subnet1: down:3Mb; up:0.5Mb.
- 3 Layers. Conform to [wang,2011] QoE study.
- Layer parameters:

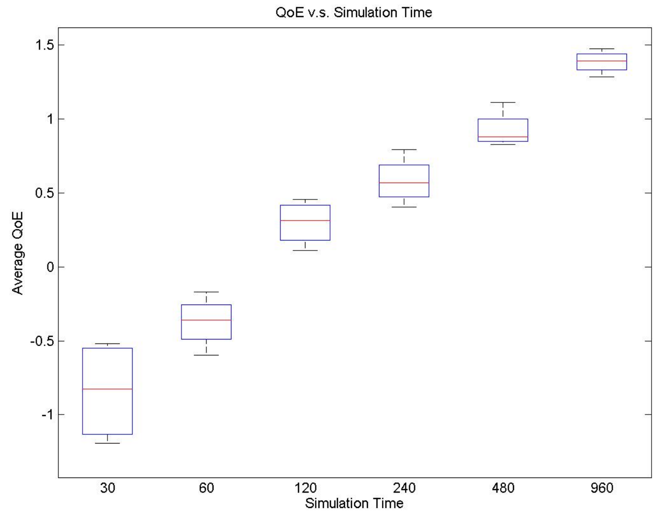

run_time v.s. node_number: polyfit coeff:22.7482,215.937
x 10^4
2.5
2
1.5
1
0.5
0
running time(human second)
0
100
200
300
400
500
600
700
800
900
1000
simulation time(NS second)(fix node_number=200)


run_time v.s. simu_time: polyfit coeff:36.0085,-463.7818
14000
12000
10000
8000
6000
4000
2000
0
running time(human second)
0
50
100
150
200
250
300
350
node number(fix simu_time=300)



(a) 3D (b) contour

Figure 2: Conclusion, QoE and Performance



- Layer1: 256Kbit. (per piece)
- Layer2: 256Kbit. (per piece)
- Layer3: 512Kbit. (per piece)

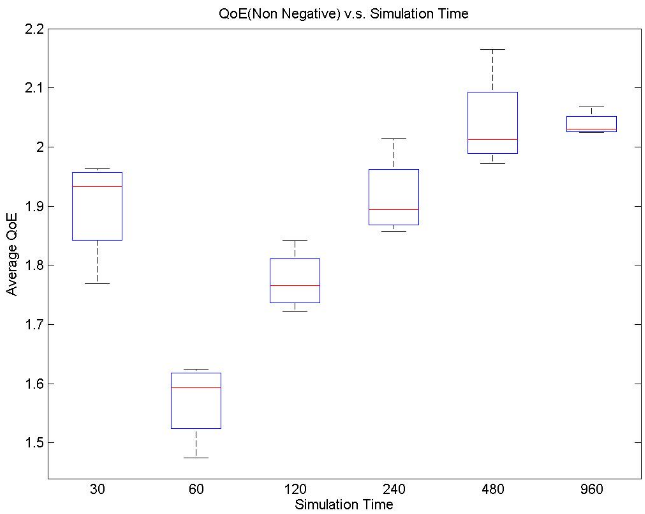
5.2 QoE Model Implementation

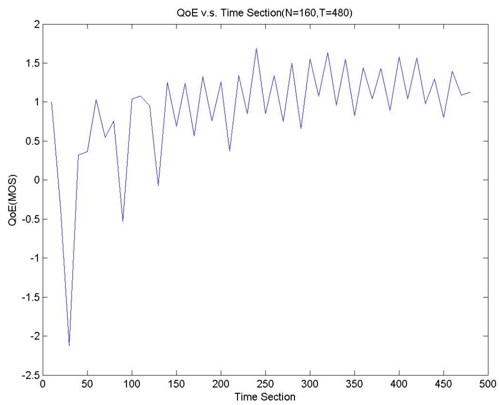
[wang, 2011], B-D tradeoff, continuous version.

$$MOS = c_1 \times d + \alpha \times (1 - e^{-b \times \lambda}) + c_2 \tag{1}$$

5.3 Baseline Test

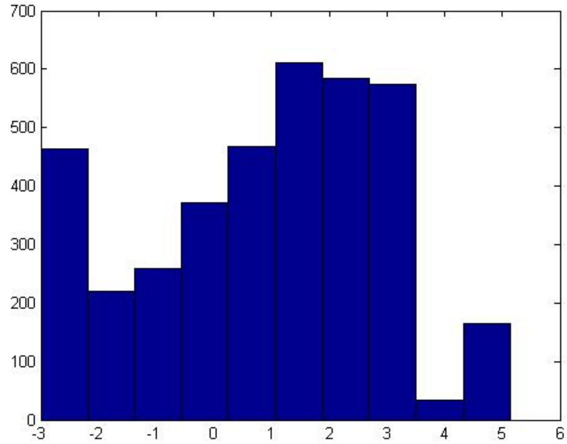
- $'count\_mr' => 3749$





- $'count\_mr\_nonneg' => 2601$
- $'avg\_qoe' =>' 0.871540677419684'$
- $'avg\_qoe\_nonneg' =>' 2.01996612593628'$
- $'node\_num' =>' 160'$
- $'simu\_time' =>' 480'$
- $'core\_num' =>' 4'$
- $'task\_duration' => 8399$
- $'pdns\_duration' => 8302$

Observations:



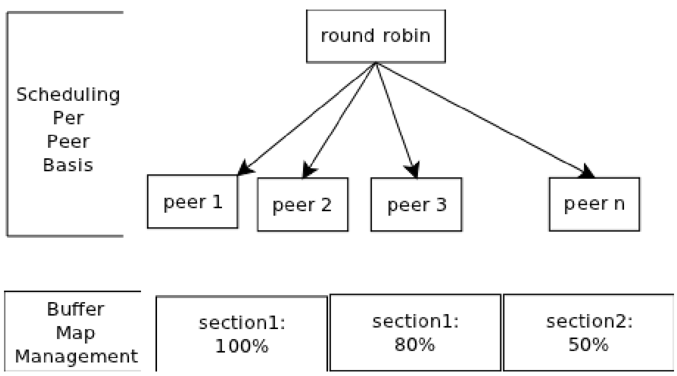


Figure 3: Original Architecture

- System bootstraping stage.
- Steady after about 250 seconds.

5.4 Chunk Selection Architecture Reconstruction

Effect:

- QoE: $0.7 \rightarrow 3.3$

5.5 Priority Based Upgrade

Result:

- QoE: nearly the same.
- Performance: decrease slightly.

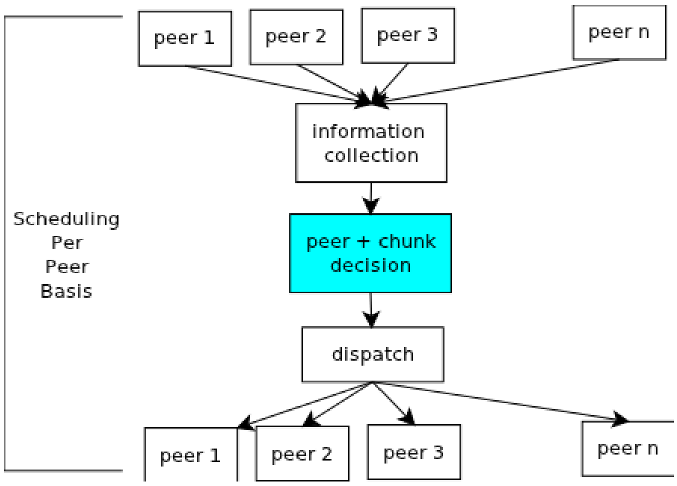


Figure 4: Improved Architecture

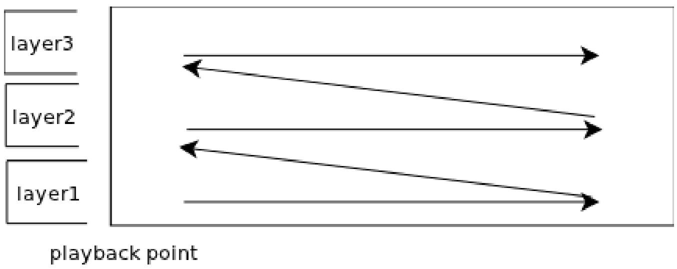


Figure 5: Improved Architecture

Table 1: A Sample Priority Table with Window Size = 5

	t1	t2	t3	t4	t5
layer3	11	12	13	14	15
layer2	6	7	8	9	10
layer1	1	2	3	4	5

Table 2: A Sample Priority Table with Window Size = 5+5

	t1	t2	t3	t4	t5	t6-t10
layer3	11	12	13	14	15	...
layer2	6	7	8	9	10	...
layer1	1	2	3	4	5	...

Table 3: A Sample Priority Table with Window Size = 5+5

	t1	t2	t3	t4	t5	t6-t10
layer3	11	12	13	14	15	...
layer2	6	7	8	9	10	...
layer1	1	2	3	4	5	...

5.6 Scalable Window Size

Reason:

- From trace, many powerful peeers can get full 3 layers in the whole window.
- Give them a chance to download more, and their data far from playback pointer can server others.

5.7 Performance Optimization

Result:

- QoE: nearly the same.
- Performance: 4.5h $\rightarrow$ 3.5h.

5.8 Introduce Randomness in Second Window Section

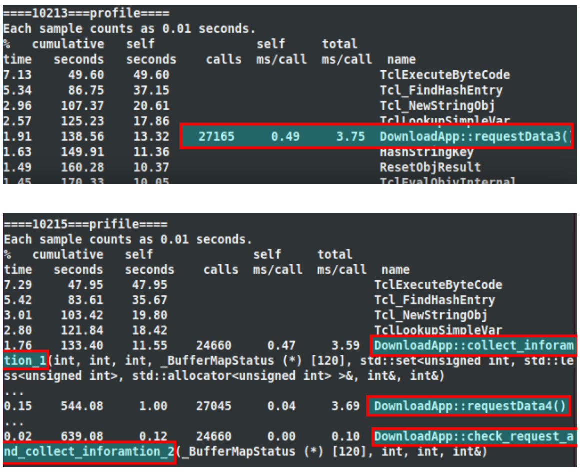
Result:

- QoE: worse.
- Performance: worse.

5.9 Conclusion of the Case Study

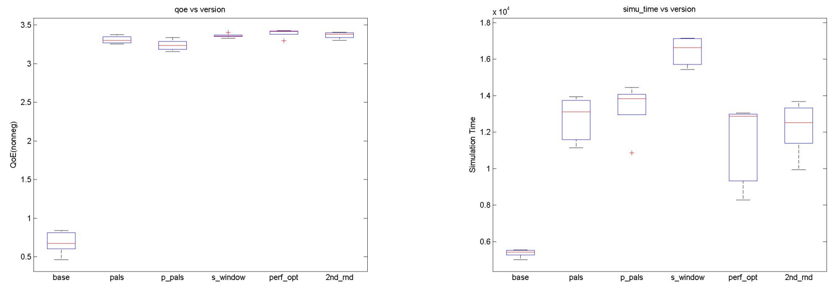
6 Conclusion

- Engineering approach v.s. academic approach:
  where is the biggest cake?
- Time distribution:



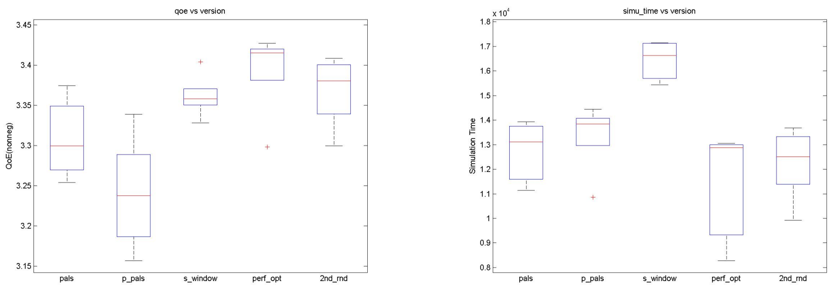
```
====10213===profile====
Each sample counts as 0.01 seconds.
%   cumulative   self              self     total
time   seconds   seconds    calls  ms/call  ms/call  name
7.13     49.60    49.60                             TclExecuteByteCode
5.34     86.75    37.15                             Tcl_FindHashEntry
2.96    107.37    20.61                             Tcl_NewStringObj
2.57    125.23    17.86                             TclLookupSimpleVar
1.91    138.56    13.32    27165     0.49     3.75  DownloadApp::requestData3(
1.63    149.91    11.36                             HashStringKey
1.49    160.28    10.37                             ResetObjResult
1.45    170.33    10.05                             TclEvalObjvInternal

====10215===prifile====
Each sample counts as 0.01 seconds.
%   cumulative   self              self     total
time   seconds   seconds    calls  ms/call  ms/call  name
7.29     47.95    47.95                             TclExecuteByteCode
5.42     83.61    35.67                             Tcl_FindHashEntry
3.01    103.42    19.80                             Tcl_NewStringObj
2.80    121.84    18.42                             TclLookupSimpleVar
1.76    133.40    11.55    24660     0.47     3.59  DownloadApp::collect_inforam
tion_1(int, int, int, _BufferMapStatus (*) [120], std::set<unsigned int, std::le
ss<unsigned int>, std::allocator<unsigned int> >&, int&, int&)
...
0.15    544.08     1.00    27045     0.04     3.69  DownloadApp::requestData4()
...
0.02    639.08     0.12    24660     0.00     0.10  DownloadApp::check_request_a
nd_collect_inforamtion_2(_BufferMapStatus (*) [120], int, int, int&)
```

Figure 6: GNU Profile, Output



(a) qoe (b) time

Figure 7: Conclusion, QoE and Performance



(a) qoe (b) time

Figure 8: Conclusion, QoE and Performance

  - 70%, literature survey.(30+ papers)
  - 15%, bugfix of the platform, environment setup.
  - 5%, first unified version(QoE:0.7→3.3, biggest improvement in this study)
  - 10%, scalable window, performance optimization, random 2nd section. (little outcome)

- 6 big versions / 240 runs.
- Auxilary Scripts:
  - .sh:298 lines
  - .pl:791 lines
  - .m:133 lines
- Simulation Code Difference:
  - download_agent.cc: 1940 lines
  - download_agent.h: 301 lines
  - labtesting.tcl: 84 lines

7 Future Works

Acknowledgements

Appendix

References

[1] J. Apostolopoulos, T. Wong, W. Tan, and S. Wee. On multiple description streaming with content delivery networks. In *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1736–1745. IEEE, 2002.

[2] J.G. Apostolopoulos. Reliable video communication over lossy packet networks using multiple state encoding and path diversity. In *Visual Communications and Image Processing*, volume 1. Citeseer, 2001.

[3] S. Bhattacharyya, J.F. Kurose, D. Towlsey, and R. Nagarajan. Efficient rate-controlled bulk data transfer using multiple multicast groups. In *INFOCOM'98. Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1172–1179. IEEE, 1998.

[4] J. Byers, M. Luby, and M. Mitzenmacher. Fine-grained layered multicast. In *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 1143–1151. IEEE, 2001.

[5] Y. Cui and K. Nahrstedt. Layered peer-to-peer streaming. In *Proceedings of the 13th international workshop on Network and operating systems support for digital audio and video*, pages 162–171. ACM, 2003.

[6] L. Dai, Y. Cui, and Y. Xue. Maximizing throughput in layered peer-to-peer streaming. In *Communications, 2007. ICC'07. IEEE International Conference on*, pages 1734–1739. IEEE, 2007.

[7] F. Dobrian, A. Awan, D. Joseph, A. Ganjam, J. Zhan, V. Sekar, I. Stoica, and H. Zhang. Understanding the impact of video quality on user engagement. In *Proceedings of the ACM SIGCOMM 2011 conference on SIGCOMM*, pages 362–373. ACM, 2011.

[8] M. Eberhard, T. Szkaliczki, H. Hellwagner, L. Szobonya, and C. Timmerer. Knapsack problem-based piece-picking algorithms for layered content in peer-to-peer networks. In *Proceedings of the 2010 ACM workshop on Advanced video streaming techniques for peer-to-peer networks and social networking*, pages 71–76. ACM, 2010.

[9] T.Z.J. Fu, D.M. Chiu, and Z. Lei. Designing qoe experiments to evaluate peer-to-peer streaming applications. In *Visual Communications and Image Processing*, 2010.

[10] T.Z.J. Fu, W. Leung, P. Lam, D.M. Chiu, and Z. Lei. Perceptual quality assessment of p2p assisted streaming video for chunk-level playback controller design. In *Packet Video Workshop (PV), 2010 18th International*, pages 102–109. IEEE, 2010.

[11] A.E.L. Gamal and T. Cover. Achievable rates for multiple descriptions. *Information Theory, IEEE Transactions on*, 28(6):851–857, 1982.

[12] V.K. Goyal. Multiple description coding: Compression meets the network. *Signal Processing Magazine, IEEE*, 18(5):74–93, 2001.

[13] S.K. Kasera, G. Hjalmtusson, D.F. Towsley, and J.F. Kurose. Scalable reliable multicast using multiple multicast channels. *Networking, IEEE/ACM Transactions on*, 8(3):294–310, 2000.

[14] S. Khan, K.F. Li, E.G. Manning, and M.D.M. Akbar. Solving the knapsack problem for adaptive multimedia systems. *Stud. Inform. Univ.*, 2(1):157–178, 2002.

[15] J. Liu, B. Li, and Y.Q. Zhang. Adaptive video multicast over the internet. *Multimedia, IEEE*, 10(1):22–33, 2003.

[16] Z. Liu, Y. Shen, S.S. Panwar, K.W. Ross, and Y. Wang. P2p video live streaming with mdc: Providing incentives for redistribution. In *Multimedia and Expo, 2007 IEEE International Conference on*, pages 48–51. IEEE, 2007.

[17] Z. Liu, Y. Shen, S.S. Panwar, K.W. Ross, and Y. Wang. Using layered video to provide incentives in p2p live streaming. In *Proceedings of the 2007 workshop on Peer-to-peer streaming and IP-TV*, pages 311–316. ACM, 2007.

[18] Z. Liu, Y. Shen, K.W. Ross, S.S. Panwar, and Y. Wang. Layerp2p: using layered video chunks in p2p live streaming. *Multimedia, IEEE Transactions on*, 11(7):1340–1352, 2009.

[19] N. Magharei and R. Rejaie. Adaptive receiver-driven streaming from multiple senders. *Multimedia Systems*, 11(6):550–567, 2006.

[20] S. McCanne, V. Jacobson, and M. Vetterli. Receiver-driven layered multicast. In *Conference proceedings on Applications, technologies, architectures, and protocols for computer communications*, pages 117–130. ACM, 1996.

[21] V.N. Padmanabhan, H.J. Wang, P.A. Chou, and K. Sripanidkulchai. Distributing streaming media content using cooperative networking. In *Proceedings of the 12th international workshop on Network and operating systems support for digital audio and video*, pages 177–186. ACM, 2002.

[22] V. Pai, K. Kumar, K. Tamilmani, V. Sambamurthy, and A. Mohr. Chainsaw: Eliminating trees from overlay multicast. *Peer-to-peer systems IV*, v:127–140, 2005.

[23] M. Qin and R. Zimmermann. Improving mobile ad-hoc streaming performance through adaptive layer selection with scalable video coding. In *Proceedings of the 15th international conference on Multimedia*, pages 717–726. ACM, 2007.

[24] R. Rejaie, M. Handley, and D. Estrin. Quality adaptation for congestion controlled video playback over the internet. In *ACM SIGCOMM Computer Communication Review*, volume 29, pages 189–200. ACM, 1999.

[25] R. Rejaie and A. Ortega. Pals: peer-to-peer adaptive layered streaming. In *Proceedings of the 13th international workshop on Network and operating systems support for digital audio and video*, pages 153–161. ACM, 2003.

[26] H. Schwarz, D. Marpe, and T. Wiegand. Overview of the scalable video coding extension of the h. 264/avc standard. *Circuits and Systems for Video Technology, IEEE Transactions on*, 17(9):1103–1120, 2007.

[27] T. Szkaliczki, M. Eberhard, H. Hellwagner, and L. Szobonya. Piece selection algorithm for layered video streaming in p2p networks. *Electronic Notes in Discrete Mathematics*, 36:1265–1272, 2010.

[28] Jingjing Wang, Z. J. Tom Fu, Dah Ming Chiu, and Zhibin Lei. Perceptual quality assessment on b-d tradeoff of p2p assisted layered video streaming. In *VCIP*, 2011.

[29] Y. Wang, A.R. Reibman, and S. Lin. Multiple description coding for video delivery. *Proceedings of the IEEE*, 93(1):57–70, 2005.

[30] T. Wiegand, G.J. Sullivan, G. Bjontegaard, and A. Luthra. Overview of the h. 264/avc video coding standard. *Circuits and Systems for Video Technology, IEEE Transactions on*, 13(7):560–576, 2003.

[31] M. Wien, H. Schwarz, and T. Oelbaum. Performance analysis of svc. *Circuits and Systems for Video Technology, IEEE Transactions on*, 17(9):1194–1203, 2007.

[32] X. Xiao, Y. Shi, and Y. Gao. On optimal scheduling for layered video streaming in heterogeneous peer-to-peer networks. In *Proceeding of the 16th ACM international conference on Multimedia*, pages 785–788. ACM, 2008.

[33] X. Xiao, Y. Shi, Y. Gao, and Q. Zhang. Layerp2p: A new data scheduling approach for layered streaming in heterogeneous networks. In *INFOCOM 2009, IEEE*, pages 603–611. IEEE, 2009.

[34] X. Xiao, Y. Shi, B. Zhang, and Y. Gao. Ocals: A novel overlay construction approach for layered streaming. In *Communications, 2008. ICC'08. IEEE International Conference on*, pages 1807–1812. IEEE, 2008.

[35] J. Zhao, F. Yang, Q. Zhang, and Z. Zhang. On improving the throughput of media delivery applications in heterogenous overlay network. In *Proc. IEEE Globecom.* Citeseer, 2006.

[36] Youtube, Demo of User Controlled Adaptation, `http://www.youtube.com/`

[37] Pili Hu, 2011, Lightweight Distributing Toolset, `https://github.com/hupili/Lightweight-Distributing-Toolset`