

Content Reserve Utility Based Video Segment Transmission Scheduling for Peer-to-Peer Live Video Streaming System

Zhu Li¹, Jianwei Huang², and Aggelos K. Katsaggelos³

^{1,3}Multimedia Research Lab, Motorola Labs, Schaumburg, IL, USA,

²Dept of Info Engineering, Chinese University of Hong Kong, China

³Dept of EECS, Northwestern University, Evanston, IL, USA

Abstract— Peer-to-Peer (P2P) distribution has becoming an popular solution for IPTV applications. In this work, we define an utility function over the play back buffer content reserve, and develop a utility gradient based video segment transmission scheduling algorithm for uploading bandwidth allocation in P2P live streaming system, with the goal of minimizing play back freezes. Simulation results demonstrated the effectiveness of the proposed solution.

I. INTRODUCTION

The advance of internet infrastructure makes Peer to Peer (P2P) live streaming a viable solution for IPTV [Jain05] services with medium quality video. Compared with the traditional client-server system, which does not scale well with large and dynamic viewing clients, a P2P system is cheaper to deploy for content providers and offers more content choices to end consumers. P2P scheme utilizes uploading bandwidth and buffer storage capacity of participating peers in a collaborative fashion. It turns the passive content “consumers” in client-server system into active content “providers” and makes fully use of the individual users’ resources, thus is much more scalable and robust.

Despite the great success of many P2P streaming systems, e.g., PPLive [PPLive, He06], CoolStream [Zhang05], PPStream [PPLive], there does not exist much work analyzing the operation and tradeoff of such systems. Recently, [Kumar07] analyzes the pull-based P2P streaming systems based on fluid model, and constructs a simple 2-hop scheduling algorithm for a buffer-less system. A system with buffer is also studied using extensive simulations. For multicast tree based P2P streaming systems, [Padmanabhan03] considers a simple tree management algorithm that improves the resilience of the system utilizing path diversity in the network. However, none of these work give an analytical model of the P2P streaming system under heterogeneous buffer occupancies.

P2P downloading system, like Bit Torrent [Qiu04] has been extensively studied. However, P2P downloading system does not consider real time playback constraint and thus does not have the playback buffer underflow issues as is

addressed in this paper.

In a P2P streaming system, it is desirable to minimize the probability of “freezes” during the live video playback. In other words, the extra buffered content (i.e., content reserve level) of each peer should be kept positive. Here we use the well known α -fair utility function [Mo00] to model users’ satisfaction level of their current content reserve levels, and we allocate network resources (i.e., transmission rates) to different users to maximize the total network utility.. It can be shown that the optimal solution can balance the buffer occupancy level among users. Finally, we propose a greedy heuristic solution that achieves the desirable result with low complexity.

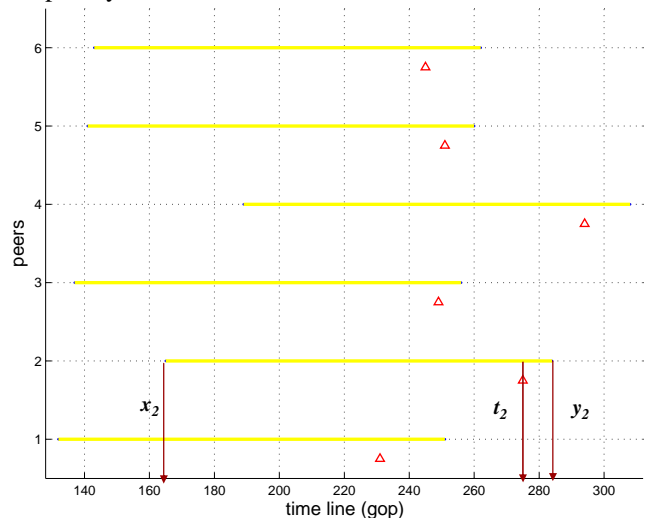


Figure 1. 6 peer playback buffer states

The paper is organized into the following sections. In Section II, we present definitions and formulations of the problem, In Section III, we develop the utility gradient based scheduling algorithm. In Section IV, simulation results are presented along with some discussions, and in Section V, we summarize the results and propose future work.

II. DEFINITIONS AND FORMULATIONS

A. Definitions and Formulations

Consider a P2P System with a video source and n peers, with upload capacities, C_0, C_1, \dots, C_n . Let the content to be

streamed over all peers with a constant bit rate (CBR) R_0 . Each peer k 's ($1 \leq k \leq n$) playback buffer is therefore characterized by the buffer state tuple,

$$S_k = \{x_k, t_k, y_k\} \quad (1)$$

where the buffer content starting point and ending point are x_k and y_k , and the current playback time is t_k . The content reserve level is calculated as $y_k - t_k$.

This is illustrated by an example in Fig. 1, where the playback buffer occupancy states of 6 peers are plotted as horizontal bars. The current playback video segment t_k , or Group of Picture (GoP), is marked for each peer by '^'. Each GoP consists of 0.5 second of video segment coded in IPBP pattern, and for CBR video it consists of $R_0/2$ bits. The playback buffer state, $\{x_2, t_2, y_2\}$ for peer 2 is marked.

An important consideration for P2P live video streaming system is to manage/allocate peer upload bandwidths to prevent peer buffers from underflow, and therefore causing unpleasant "freezes" in the playback. Intuitively, a peer with a playback time stamp t_k close to the end of buffer y_k should be allocated more resources. To characterize this we define an utility function over the content reserve, $U(y_k - t_k)$, where the utility function has the form of,

$$U(x) = \frac{x^{1-\alpha}}{1-\alpha}, 0 < \alpha < 1. \quad (2)$$

Examples of this utility function are plotted in the Fig 2 below. Notice that the parameter α controls the property of the utility function. When α is close to 1, it leads to proportional fair allocation of resource, and when it is close to 0, it leads to total buffer occupancy maximization.

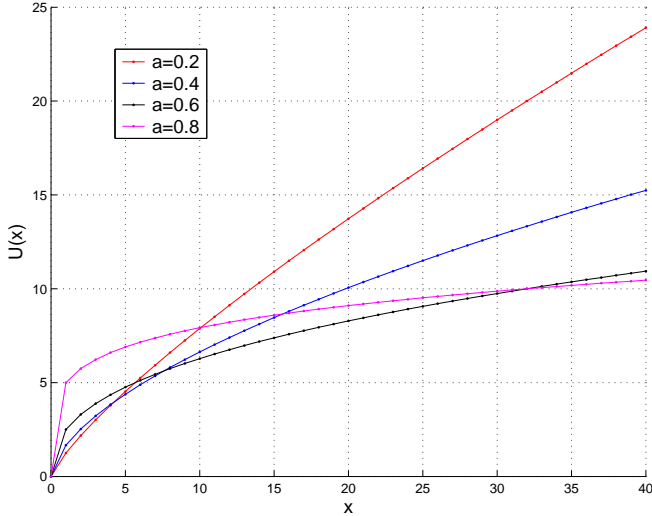


Figure 2 Utility functions

The problem of upload bandwidth allocation is therefore formulated as a constrained optimization problem. For the scheduling period, T , the objective is to allocate upload bandwidth for each peer, R_k , such that the total utility over the playback buffer content reserve is maximized,

$$\max_{R_k} \sum_k U\left(\frac{T}{R_0} R_k + (y_k - t_k)\right), s.t. \sum_{1 \leq k \leq n} R_k \leq \sum_{1 \leq k \leq n} C_k + C_0 \quad (3)$$

In Eq. (3), we assume the upload bandwidth $\{C_k\}$ remains constant for the scheduling period.

This is a typical total resource constrained utility maximization problem. From the radio resource pricing control work for wireless video over multi-access channel [Li06], it is known that at the optimal allocation of resources, each user's utility gradient should be the same. In the context of this work, since all user are using the same utility function, it means all peer's content reserves will converge to the same size at optimal allocation. If all buffer states information are known, then Eq. (3) can be explicitly solved [Li06] for the optimal allocation $\{R_k^*\}$.

$$R_k^* = \left(\frac{T}{R_0} \left(\frac{\sum_{1 \leq k \leq n} C_k + C_0}{n} + \sum_{1 \leq k \leq n} (y_k - t_k) \right) - y_k + t_k \right) \frac{R_0}{T}.$$

Once we have the allocation, the actually transmission of video segments need to be scheduled. For small scheduling intervals, gradient based scheduling algorithms has been shown to maximize the total network utility in various cases, such as hybrid ARQ based TDMA systems [Huang05], CDMA systems [Agr02], and OFDMA systems [Huang07]. The intuitive is to always serve the packet with the largest utility gradient in the queue.

However, the P2P system is different from works in [Li06], [Huang05], [Huang07]. The resource constraints are actually more complicated. The allocated rate R_k is provided by multiple peers,

$$R_k = \sum_j r_{j,k}$$

where for peer k , the additional content reserve is obtained by pulling from serving peers j with rate $r_{j,k}$. The P2P serving rates are subjected to upload capacity,

$$\sum_k r_{j,k} \leq C_j \quad (4)$$

and content availability constraints,

$$r_{j,k} = 0, \text{ if } y_j \leq y_k \quad (5)$$

The constraints in (5) are dependent on the current peer buffer states. For example, $r_{1,k} = 0$ for $k=\{3, 5, 6\}$, for buffer states in Fig. 1.

An optimal rate scheduling will be difficult due to the complexity of constraints in (4) and (5). In a practical pull-based system, the rate allocation and scheduling are discrete, and can be viewed as a video segment transmission scheduling problem.

At each scheduling, each peer k will request a set of video segments, or GoPs starting with time offset $y_k + I$, to fill the playback buffer. For an upload bandwidth of C_j , and scheduling interval T , the number of GoPs can be transmitted by peer j is m_j . Each peer j can allocate $n_{j,k}$ upload capacity to peer k , if they are available. The scheduling formulation is therefore given as,

$$\begin{aligned} \max_{n_{j,k}} \sum_k U(\sum_j n_{j,k} + (y_k - t_k)), \\ \text{s.t. } \sum_k n_{j,k} \leq m_j, m_j = \frac{TC_j}{R_0/2} \\ n_{j,k} = 0, \text{ if } y_j < y_k \end{aligned} \quad (6)$$

This is a combinatorial optimization problem, with total number of possible solution,

$$\binom{n}{m_1} \binom{n-m_1}{m_2} \binom{n-m_1-m_2}{m_3} \dots \binom{n-\sum_{j=1}^{n-1} m_j}{m_n}$$

An exhaustive search will incur prohibitive cost in computation. Instead, we develop a greedy algorithm in the next section.

III. UTILITY GRADIENT DRIVEN SCHEDULING SOLUTION

In a practical solution, we assume a peer with good throughput and computing capabilities is randomly selected to act as a coordinator of a small sub-set of n peers. The transport is the best-effort based UDP scheme. The upload capacities, $\{C_k\}$, stay roughly constant during the playback, and are known at the coordinator. At each scheduling interval, the play back buffer status $\{S_k\}$ are communicated to the coordinator. Based on this information, the coordinator computes a GoP transmission list for each peer that approximates the optimal solution to Eq. (6).

Algorithm 1: GoP Scheduling

```

Collect buffer states  $\{S_k\}$ 
Compute upload capacities  $\{m_k\}$ , and total capacity  $M$ ,
Compute GoP request list  $G$ , sort by utility gradient  $U'_{k,j}$ 
gopCnt =  $\sum_k m_k$ ;  $i=1$ ;
WHILE (gopCnt > 0)
  Find peers  $P = \{k \mid m_k > 0, x_k \leq g_j \leq y_k\}$ 
  IF  $P$  is empty
    % not serviceable,
    Continue.
  ELSE
     $k^* = \arg \min_{k \in P} y_k - g_i$ 
     $m_{k^*} = m_{k^*} - 1$ 
    Add GoP  $g_j$  to peer  $k^*$ 's serving schedule
    gopCnt = gopCnt - 1
     $i = i + 1$ 
  END
END

```

Figure 4. GoP Scheduling Algorithm

The GoP serving priority is based on the requested GoP's location in each peer's buffer. The utility gradient for requested GoP $y_k + j$ of peer k is computed as,

$$U'_{k,j} = U(y_k - t_k + j) - U(y_k - t_k + j - 1) \quad (7)$$

In a greedy scheduling for (6), it means the requested GoP with largest utility gradient should be scheduled first. From the buffer status updates, the coordinator can compute the

following requested GoP number list with its associated utility gradient,

$$G : \{g_i = y_k + j \mid U'_{k,j} \geq U_{\min}\} \quad (8)$$

The size of the GoP request list can be adjusted according to the scheduling interval / total GoP serving capacity, with the utility gradient threshold U_{\min} . The algorithm assigns each GoP in the request list G to a feasible serving peer, in the order of descending utility gradient, until all uploading capacities in the scheduling interval are exhausted.

The serving peer is selected through a closest buffer end first heuristic. For a requested GoP with time offset g_i , only peers with $y_k \geq g_i$ can serve this request. If we sort all serving peers by y_k , the upload bandwidth of the peer with the latest content are the most desirable one, i.e., it can serve all peers with older content. Therefore, their bandwidth should be conserved, and we need to select the peer with y_k closest to the requested QoP g_i .

The greedy algorithm is summarized in Fig. 4.

IV. SIMULATION RESULTS

To verify the performance of the proposed algorithm, we set up P2P video sessions with the following constants in system setting. Let the system average upload bandwidth be,

$$C = \frac{\sum_{k=0}^n C_k}{n}$$

In the first test, we experimented our GoP greedy scheduling algorithm with an $n=6$ peer system. The system settings are summarized in Table 1. The scheduling interval is set as $T=4$ (GoPs). For a total of 30 iterations, or 60 sec playback, the resulting play back buffer content reserve in number of GoPs are plotted.

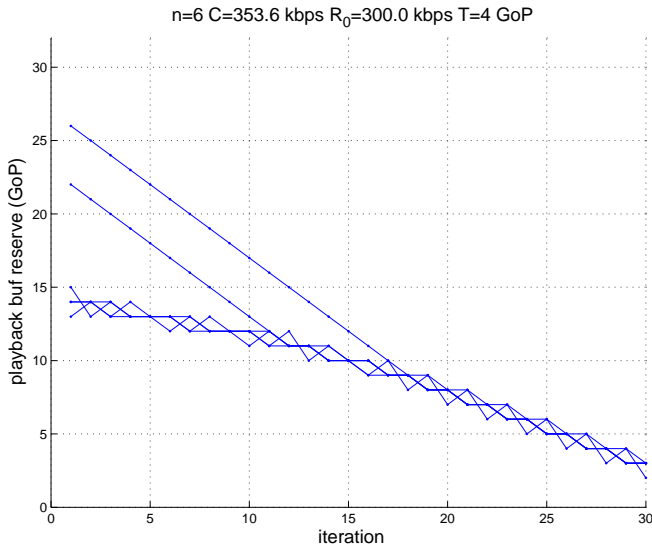
Table 1. P2P system settings

Parameter	Value	Comments
R_0 (kbps)	300	Video playback rate
L (sec)	60	Video content length
C_{std} (kbps)	40	Upload bandwidth variations
W (sec)	30	Content access window size
α	0.6	Utility function parameter

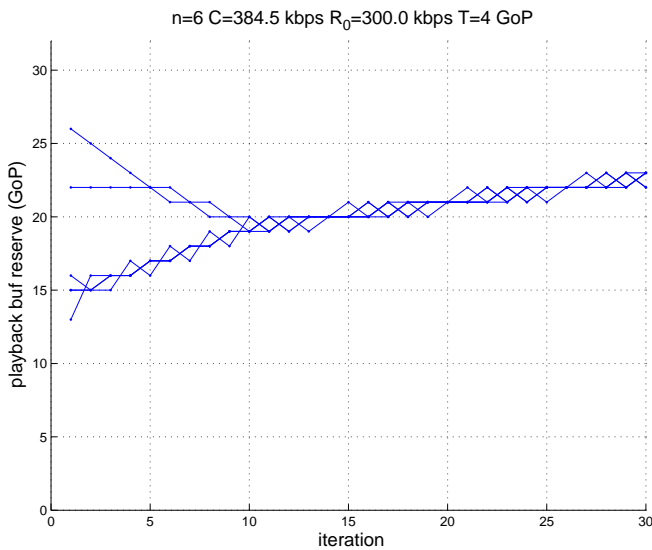
The results for average upload bandwidth $C=322.8$ kbps are plotted in Fig. 5a, and those of $C=384.5$ kbps, or $1.28R_0$, are plotted Fig. 5b. Notice that in both cases, the content reserve sizes are quickly converged, and at later case the content reserve size stabilizes.

To compare we also developed a utility blind scheduling algorithm that just serves GoP request on a first-come-first serve (FCFS) basis. The content reserve sizes are plotted in Fig. 5c. It is clear that the system is not stable and the reserve sizes do not converge, some peers end up with more reserves while others are starved and have numerous freezes in play

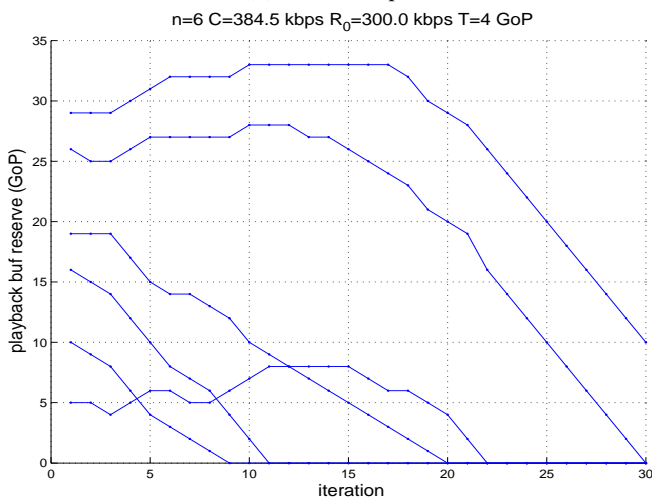
back.



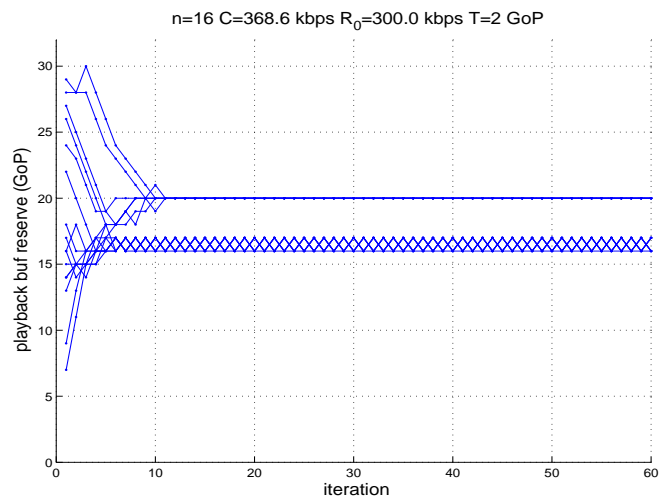
(a) C=353.6



(b) C=384.5kbps



(c) C=384.5kbps, FCFS scheduling

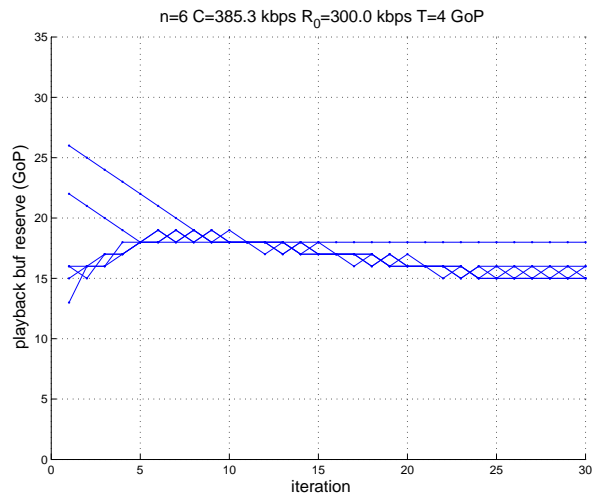


(d) C=368.6kbps

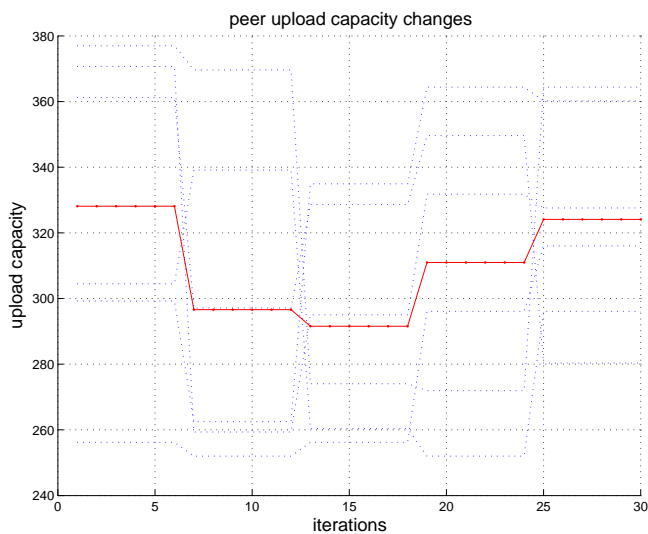
Figure 5. Peer Buffer Content Reserve States

In a second system, we increase the number of peer to $n=16$, source upload rate to $2.5R_0$, and reduces the scheduling interval to $T=2$ (GoP). The system show stable reserve content sizes at an average upload bandwidth of $C=368.6$ kbps. The increase of the size of locally coordinated P2P group does improve the performance.

For time varying upload bandwidth, the system can also adapt to the changes and effectively provide the service. For the same 6 peer set up, the performance of content reserve size is plotted in the Fig. 6a below. The corresponding upload capacity changes and the average capacity C over scheduling iterations are plotted in Fig. 6b.



(a) Performance with time-varying upload bandwidth



(b) upload bandwidth changes

Figure 6. Performance with time-varying upload bandwidth

Notice that the performance is similar to Fig. 5b, and the average upload bandwidth change over time is plotted as solid lines in Fig. 6b, while each peer's upload bandwidth changes are plotted in dotted lines.

V. CONCLUSION AND FUTURE WORK

In this paper we developed a utility gradient based video segment scheduling algorithm for P2P live video streaming solution. The simulation results demonstrated the effectiveness of the proposed solution. In the future we will investigate a combinatorial optimization solution, study the utility function parameter and its impact on convergence speed, and include the end-to-end GoP transmission delays into the system modeling and simulation.

REFERENCES

- [Agr02] R. Agrawal and V. Subramanian, "Optimality of Certain Channel Aware Scheduling Policies," *Proc. of 2002 Allerton Conference on Communication, Control and Computing*, Oct. 2002.
- [Huang05] J. Huang, R. Berry and M. L. Honig, "Wireless Scheduling with Hybrid ARQ", *IEEE Transactions on Wireless Communications*, vol. 4, no. 6, pp. 2801-2810, Nov. 2005.
- [Huang07] J. Huang, V. Subramanian, R. Agrawal, and R. Berry, "Downlink Scheduling and Resource Allocation for OFDM Systems", *Proc. Conference on Information Sciences and Systems (CISS)*, Princeton University, NJ, USA, March 2006.
- [Hei06] Xiaojun Hei, Chao Liang, Jian Liang, Yong Liu and Keith Ross, "Insight into PPLive: A Measurement Study of a Large-Scale P2P IPTV System", *Proc of WWW 2006 workshop of IPTV services over World Wide Web*.
- [Jain05] R. Jain, "I Want My IPTV", *IEEE Multimedia*, vol. 12(3), 2005.
- [Kumar07] R. Kumar, Y. Liu and K. Ross, Stochastic Fluid Theory for P2P Streaming Systems, *IEEE INFOCOM 2007*
- [Li06] Z. Li, J. Huang, and A. K. Katsaggelos, "Pricing Based Collaborative Multi-User Video Streaming over Power Constrained Wireless Down Link", oral paper, *IEEE Int'l Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Toulouse, France, 2006.
- [Mo00] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control", *IEEE/ACM Transactions on Networking*, vol. 8, no. 5, pp. 556-567, 2000

[Padmanabhan03] V.N. Padmanabhan, H.J. Wang, and P.A. Chou, Resilient Peero-to-Peer Streaming, *IEEE International Conference on Network Protocols*, 2003.

[PPLive] PPLive system, URL: www.pplive.com

[PPStream] PPStream, URL: www.ppstream.com

[Qiu04] D. Qiu and R. Srikant. "Modeling and Performance Analysis of BitTorrent-Like Peer-to-Peer Networks," *Proc. ACM SIGCOMM*, Portland, OR, Sept. 2004.

[Zhang05] CoolStreaming/DONet: A Data-driven Overlay Network for Peer-to-Peer Live Media Streaming, X Zhang, J Liu, B Li, TSP Yum - *Proc. IEEE Infocom*, 2005