A First Look at the Properties of Many-to-One Data Flows

V. T. Sam, P. Y. Ho, and Jack Y. B. Lee
Department of Information Engineering
The Chinese University of Hong Kong
Hong Kong
{vtsam5, pyho5, yblee}@ie.cuhk.edu.hk

Abstract

With the rapid emergence of peer-to-peer applications, more and more applications will generate many-to-one rather than one-to-one traffic flows. While a number of previous works have reported the measured characteristics of available network bandwidth, these measurements are primarily conducted for one-to-one flows or for a physical network link. This paper addresses this void by reporting the characteristics of aggregate available bandwidth in many-to-one data flows over TCP. Based on extensive measurements conducted in the PlanetLab and public FTP servers, we analyze the statistical properties of aggregate data flows originating from multiple senders destined to the same receiver. Our results show that many-to-one data flows exhibit substantially more consistent statistical properties over a long time scale, and thus could open a new way to achieve probabilistic performance guarantees in multimedia applications.

1. Introduction

Conventional network applications typically deliver data in a one-to-one manner, usually over the Transmission Control Protocol (TCP). This model works well in client-server type applications where there is only a single sender, i.e., the server, transmitting data to a single receiver. The recent rapid developments in peer-to-peer (P2P) applications however, have resulted in significantly more many-to-one data flows where multiple senders (i.e., peers) simultaneously transmit different portions of the data stream to a receiver. Apart from P2P applications, recent efforts in multi-source and multi-path video streaming [1-6] also belong to this many-to-one data flow model.

This shift in the data delivery model could open up a new dimension to achieving one of the grand challenges in Internet applications – quality of service. Specifically, given the best effort nature of the Internet, the behavior of data flows in the Internet can vary substantially, and is often difficult to predict. Numerous previous works [7-13] have investigated the measured characteristics of available network bandwidth in one-to-one data flows and for a physical network link but the results only confirm the inherent difficulty in predicting bandwidth availability in the Internet. Moreover, none of these previous studies considered many-to-one data flow where multiple senders transmit data to the same receiver simultaneously.

This study addresses this void by reporting results from measurement experiments conducted to capture the characteristics of the *aggregate* available bandwidth in many-to-one data flows over TCP. Based on extensive measurements conducted in the PlanetLab [14] and using public FTP servers, we analyze the statistical properties of aggregate data flows originating from multiple senders destined to the same receiver. Our results show that many-to-one data flows exhibit substantially more consistent statistical properties over a longer time scale when compared to one-to-one flows, and thus could open a new way to achieve probabilistic performance guarantees in multimedia applications.

In the rest of the paper we first review in Section 2 various measurement methodologies and summarize the previous work on network bandwidth estimation and traffic analysis. In Section 3 we present our measurement methodology and then analyze the measurement results in Section 4. In Section 5 we discuss some potential applications of the many-to-one data flow model and then summarize the paper in Section 6.

2. Background and Related Work

Internet traffic has been studied extensively in the literature. It is generally accepted that Internet traffic cannot be adequately modeled by a Poisson process [15]. A number of studies showed that network traffic is in fact self-similar exhibiting long-range dependency. On the other hand, the *complementary* problem of modeling available bandwidth, e.g., link capacity minus the bandwidth consumed by network traffic, has received little attention.

2.1. One-to-One Data Flow

Recently researchers have taken the first step by developing software tools for estimating available bandwidth for both individual network links as well as end-to-end network path. Note that the term 'bandwidth' can assume one of three different meanings, depending on the particular measurement tool and methodology adopted:

Link capacity: The capacity of a link is the maximum data rate a flow can utilize when there are no other traffic flows sharing the link. Note that this may or may not be the same as the link's physical bandwidth, depending on whether the router implements any rate-limiting control over the data flows. Well-known tools for estimating the *per-hop* link capacity include *pathchar* [9], *clink* [10] and *pchar* [11]. If a data flow traverses N links from the sender to the receiver and C_i is the capacity of link i, then $C=\min\{C_1,C_2,\ldots C_N\}$ will be the end-to-end capacity of the data flow. Tools for estimating the end-to-end capacity include *bprobe* [12] and *pathrate* [13].

Unutilized bandwidth: A second definition of available bandwidth is the *unutilized* capacity of a link [7]. Let $u_i(t)$ be the utilization of link i at time t. Then the average link utilization (in normalized unit from 0 to 1) in the interval $[t, t+\tau)$, denoted by $u_i(t, t+\tau)$, can be computed from

$$u_i(t,t+\tau) = \frac{1}{\tau}\int_t^{t+\tau} u_i(t)\,dt \qquad (1)$$

Given C_i as the capacity of link i, then the unutilized capacity of link i during the interval $(t, t+\tau)$, denoted by $U_i(t, t+\tau)$, can be computed from

$$U_i(t,t+\tau) = C_i[1-u_i(t,t+\tau)] \qquad (2)$$

Thus the end-to-end unutilized bandwidth is simply equal to the minimum unutilized bandwidth of the N links:

$$U(t,t+\tau) = \min_{i=1\ldots N}\{U_i(t,t+\tau)\} \qquad (3)$$

Well-known tools for estimating the end-to-end unutilized bandwidth include *pathload* [7] and *IGI* [8].

Achievable bandwidth: A third definition of available is the end-to-end throughput achievable by a TCP (or TCP-friendly) flow in passing through a network path. In contrast to unutilized bandwidth, a TCP flow will always be able to obtain its fair share of bandwidth even if the network path is already fully utilized, assuming that the competing flows also share bandwidth fairly (e.g., TCP and TCP-friendly flows). It is easy to see that achievable bandwidth will be more realistic for applications transporting their data over TCP or TCP-friendly protocols, and so will be the definition we adopt in the rest of this paper.

Let $d(t,t+\tau)$ be the amount of data received in the interval $[t, t+\tau)$ by the receiver using TCP. Then the achievable bandwidth for the interval is then given by

$$r(t,t+\tau) = \frac{1}{\tau}d(t,t+\tau) \qquad (4)$$

2.2. Many-to-One Data Flow

As discussed in Section 1 the emerging peer-to-peer applications increasingly make use of parallel transport connections to deliver data from multiple senders to a receiver. Thus the data throughput achievable will be the aggregate of the throughput of all the senders.

Let there be N senders, with $d_i(t,t+\tau)$ denoting the amount of data received in the interval $[t, t+\tau)$ by the receiver from sender i using TCP. Then the aggregate available bandwidth for the interval is then given by

$$A(t,t+\tau) = \frac{1}{\tau}\sum_{i=0}^{N-1} d_i(t,t+\tau) \qquad (5)$$

Note that as the number of senders increase the aggregate bandwidth will increase, and eventually it may exceed the receiver's own access link capacity. In that case the aggregate available bandwidth will be limited by the receiver's link capacity rather than the senders' available bandwidth. For this reason we have built into our experiments mechanisms to detect and isolate these cases.

3. Experiments Design

To study the many-to-one achievable bandwidth by TCP, we conducted two sets of experiments, one in the Internet using FTP servers and the other in the PlanetLab using our own custom-developed measurement software. We choose to use TCP because its congestion control algorithm will automatically probe for available bandwidth as well as react to network congestions (and other competing flows). Thus the achievable data throughput over TCP is a good representation of the available bandwidth between the sender and the receiver.

The first experiment is aimed at obtaining real-world measurements of aggregate available bandwidth from multiple senders. This poses a significant challenge as real-world public servers around the Internet are clearly not opened to such experimentations. To work around this problem we made use of public FTP servers that have mirrored a common large software distribution (FC4 Disc 1 of size 635MB) and then setup our client software to download the large file from multiple FTP servers simultaneously to execute a many-to-one data transfer process. Note that we are only interested in the achievable data throughput and thus the actual data being transferred are irrelevant. We chose FC4 Disc 1 simply because it has a large file size and is widely mirrored in many public FTP servers.

In each measurement, N (from 1 to 10) FTP servers are randomly drawn from a pool of 54 Fedora mirror sites in the Internet. The local receiver resided in The Chinese University of Hong Kong (CUHK) downloads separate FC4 Disc 1 of size 635MB from each server simultaneously. Thus, there are N concurrent TCP connections sending data to the single receiver. When a whole copy of the disc is downloaded from the server, the receiver would download the same file again from the same server. Each run lasted for 2 hours, after which the receiver will continue to draw FTP servers from the pool until 10 measurements runs with N=1 to 10 are completed. These 10 runs count as one set of measurement.

The second experiment was conducted in PlanetLab - a global test-bed with hundreds of hosts residing in many different countries around the world connected through the Internet. A custom-developed measurement program was installed in the PlanetLab nodes (total 286 nodes) for performing this experiment. This program acts either as a server to send dummy data to the receiver, or acts as a receiver to make a data transfer request to a server. A central control server residing in CUHK randomly selects one receiver from a pool of PlanetLab Nodes and changes its state from WAIT to RECEIVE to request data from N servers. It will also configure the receiver with the required number of senders (N) and an active node list.

To begin a measurement run the receiver randomly chooses N (1 to 10) senders from the pool of nodes as servers, and then initiates a TCP connection to each server. Upon receiving the request, the servers will then send dummy data to the receiver simultaneously.

Note that before the actual measurement run, a pre-measurement, which lasts for 20 seconds, is performed to ensure that the end-to-end bandwidth from the senders to the receiver will not be too large. This is done for two purposes. First, high-bandwidth senders will easily use up all the capacity of the receiver's access link, thereby distorting the measurement results. Second, sending too high a data rate in the PlanetLab may slow down other services running in the same PlanetLab nodes. Therefore, the system will replace a high-bandwidth sender with another one from the pool of senders until the aggregate available bandwidth does not exceed 1Mbps. Each measurement run lasted for 2 hours. The receiver will vary the number of senders from 1 to 10 in subsequent runs. These 10 runs are counted as one set of measurement.

4. Measurement Results

In this section we present the measurement results obtained from the experiments described in Section 3.

4.1. Correlation Between Flows

First, we examine the correlation coefficient of the throughputs of the FTP servers in the Internet and the senders in PlanetLab. The correlation coefficient measures the degree of correlation between the throughputs of two senders. If two flows from different senders have a high correlation coefficient, they may share a common bottleneck in the network.

Fig. 1 plots the cumulative distribution function (CDF) of the correlation coefficient of the throughputs of the FTP servers and the PlanetLab senders respectively. The result shows that 80% of the sender pairs have a correlation coefficient less than 0.2. Since more senders to a common receiver may have a higher probability for two of the flows to go through a common bottleneck, we examine the correlation coefficient for the experiments with three, five and eight senders and plot the results in Fig. 2. The results are very similar to the above ones. 80% of the sender pairs still have a correlation coefficient less than 0.2 even with different number of senders, thus confirming that the senders are not correlated.

4.2. Aggregate Available Bandwidth

To compare the properties of one-to-one and many-to-one data transfers, we compute the mean and coefficient of variation (CoV) values for different combinations of senders in an experiment. Our purpose is to examine how the number of senders affects the above parameters during the same experiment.

Fig. 3 (FTP) and 4 (PlanetLab) plot the mean and CoV of the aggregate available bandwidth $A(0,x)$ versus the length of the time interval, i.e., x, used in computing them. For example, a time interval of 200 seconds in the x-axis means that the mean is computed using measurement data in the interval [0, 200] seconds, i.e., E[A(0,200)] (c.f. Eq. (5)). A remarkable observation in Fig. 3a and Fig. 4a is that the mean throughput is relatively stable across the measurement duration, with only small fluctuations.

The CoV in Fig. 3b and Fig. 4b, on the other hand, shows significantly higher variations over the measurement period, implying long-range variations over a time scale of thousands of seconds. The FTP case also shows significantly higher variations than the PlanetLab case. In analyzing the trace data we found that some FTP connections will be disconnected after some time for unknown reasons. Our client software in such cases will attempt to reconnect to the FTP server to resume the disconnected flow but this process takes some time, of which the throughput of the disconnected sender will drop to zero. This may be partially responsible for the higher variations than the PlanetLab case.

Comparing the CoV of different number of senders we can easily see that the more the senders the less the variations in the CoV. This is because the senders are relatively uncorrelated and so random variations will be partially cancelled out in multi-sender flows but not in single-sender flows.

4.3. Predictability of Bandwidth Properties

The previous results show that properties of the aggregate available bandwidth become more stable and consistent when the number of senders is large. We explore in this section whether it is possible to estimate the properties of the aggregate flow over a long period of time, using just a short measurement period at the beginning. If this is possible then it will be very useful for admission control as well as quality-of-service control in bandwidth-sensitive applications.

To investigate the predictability of the aggregate flow's properties, we compute the mean and standard deviation (StD) of the aggregate available bandwidth for the initial T seconds, i.e., $[0, T]$. Then we compute the mean and StD in each subsequent 100 seconds and compare them with the initial estimation to see how far the subsequent mean and StD deviate from the initial estimation.

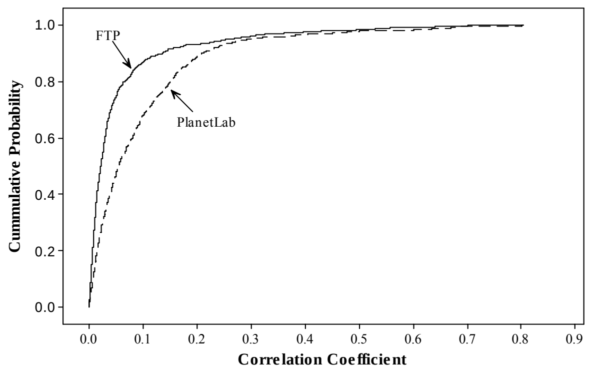
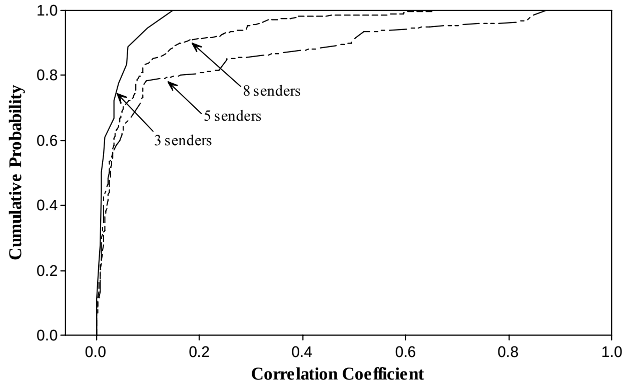
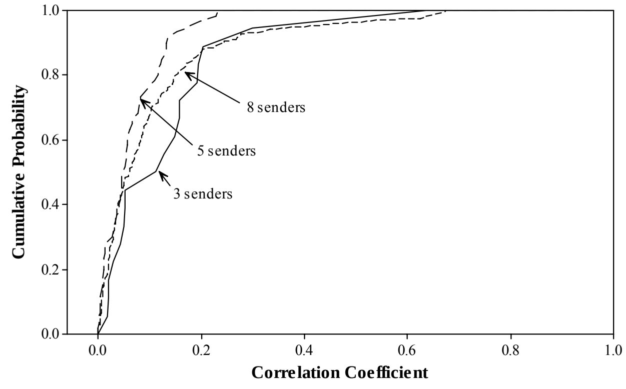


Figure 1. The cumulative distribution of the correlation coefficient of the throughputs of the FTP servers and PlanetLab senders.

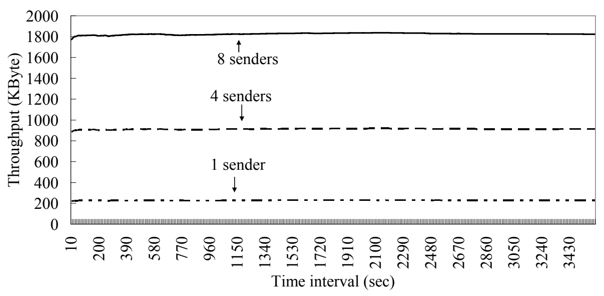


(a)

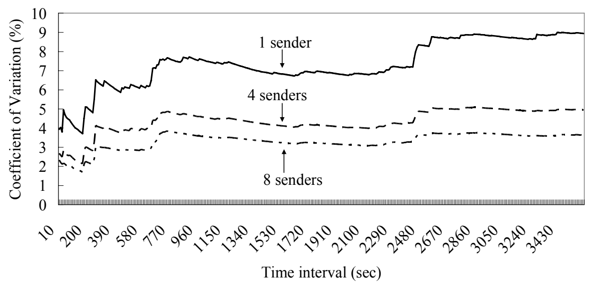


(b)

Figure 2. The cumulative distribution of the correlation coefficient of the 3, 5 and 8 flows from (a) FTP servers and (b) PlanetLab senders.

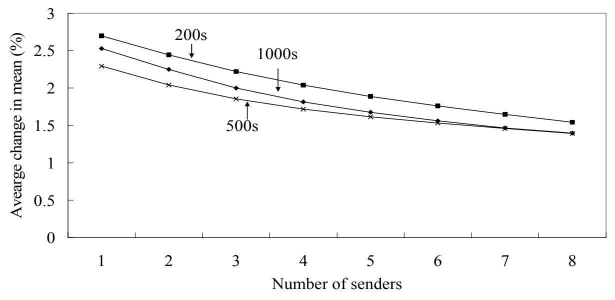


(a)

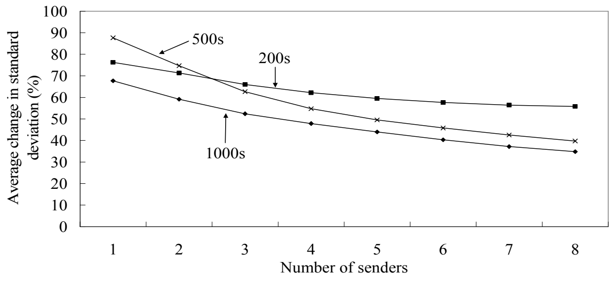


(b)

Figure 3. The (a) mean throughputs and (b) CoVs of the FTP servers in different time intervals.

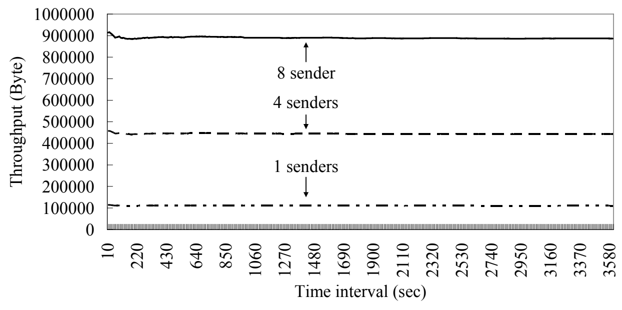


(a)

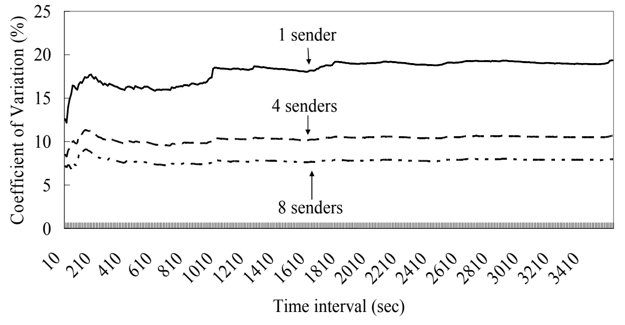


(b)

Figure 5. The deviations of (a) mean and (b) StD from different initial estimation periods for the FTP servers.

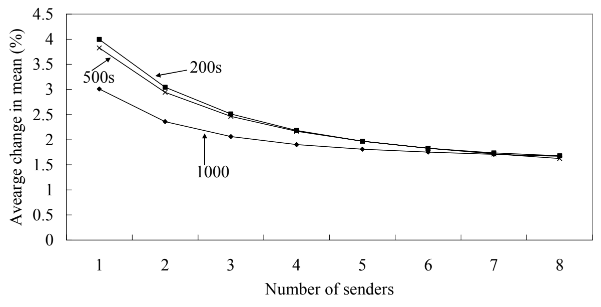


(a)

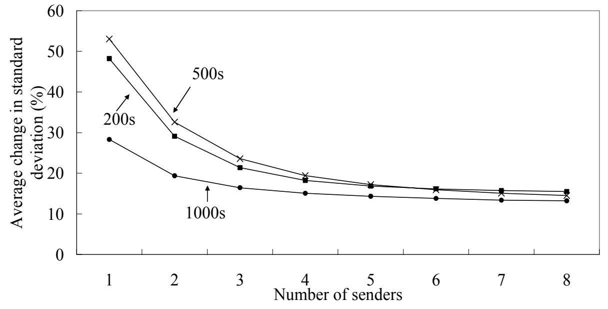


(b)

Figure 4. The (a) mean throughputs and (b) CoVs of the PlanetLab senders in different time intervals.



(a)



(b)

Figure 6. The deviations of (a) mean and (b) StD from different initial estimation periods for the PlanetLab senders.

The results are plotted in Fig. 5 and 6, with T={200, 500, 1000} seconds, versus number of senders ranging from 1 to 8. Interestingly the results show a longer initial estimation period does not necessarily lead to more accurate mean and StD estimation. By contrast, when the number of senders is increased from 1 to 8, the deviations in both mean and StD consistently decrease. These results confirm the observations that the bandwidth availability of a single sender is very difficult to predict, even if we allow for an initial measurement duration of 1000 seconds. By contrast, with multiple senders the aggregate available bandwidth will become far more consistent and predictable, even when given a shorter initial measurement duration.

5. Applications

The many-to-one data transfer model could open up a new way to perform network resource allocation, traffic engineering, as well as transmission scheduling in bandwidth-sensitive applications. We outline some possible applications in this section.

Beginning with the simplest application, we could apply the many-to-one data transfer model to downloading multimedia data streams for playback [5]. In current media playback software such as RealVideo or Windows Media Player, the player will prefetch a buffer with media data before commencing actual playback. The process may take a few seconds if bandwidth is sufficient, up to many minutes if either network bandwidth is insufficient or the media stream is encoded in a very high playback data rate.

In any case once playback begins the player will continue to receiver data from the server and maintain a continuous playback as long as data arrives sufficiently fast to keep the buffer non-empty. In practice, however, it is not uncommon for the media player to run into buffer underflow (e.g., due to fluctuations in network bandwidth), causing playback interruptions. This problem could be alleviated if we know the statistical properties of the available bandwidth so that sufficient data can be buffered before commencing playback to ensure continuous playback. This is exactly where we can apply the many-to-one data transfer model.

Specifically, let C_i be the aggregate data received in time interval i after download process begins; R be the video bit-rate and w be the time to start playback. To ensure continuous playback the amount of data received at any time must not be less than the amount of data consumed, i.e.,

$$\sum_{j=1}^{i} C_j \geq R(i-w), \forall i > w \qquad (6)$$

or else buffer underflow will occur, causing playback interruptions.

Let n be the length of the video session in number of time intervals. Assuming that C_j is normally distributed (based on the bandwidth model) with its CDF denoted by F(x), then the left hand side of (6) can be computed from the n-time auto-convolution of F(x), denoted by $F^{(n)}$(x). Therefore to guarantee continuous playback with a probability of Δ, we simply need to ensure that the following condition holds:

$$F^{(n)}(R(n-w)) \leq (1-\Delta) \qquad (7)$$

and the smallest value of w that satisfies (7) will be the earliest time to start playback. In practice, the video player can perform the measurements of F(x) (i.e., mean and standard deviation) in parallel with the initial download period, and then begin playback once the condition in (7) is satisfied.

Taking the example one step further, the media streaming application, even after playback has begun, can continue to measure the statistical properties of the aggregate data flow and then perform adaptive playback controls such as slightly slowing down or speeding up media playback rate to compensate for the bandwidth fluctuations. Note that in this case the playback rate adaptation can be performed far less frequently as the statistical properties of the many-to-one data flow is far more consistent than one-to-one data flow [6].

On the other hand, if the media stream is encoded using scalable codec such as MPEG4 FGS or multiple description coding (MDC), the (distributed) streaming servers, upon receiving the statistical properties of the many-to-one data flow from the client, can potentially apply the knowledge to adaptively fine-tune the media bit-rate to transmit to the client.

The above examples focus only on a single media streaming session. In a content distribution network or a peer-to-peer network, there will be many such many-to-one streaming sessions operating simultaneously. By allowing the senders and receivers to exchange the statistical properties of the on-going data flows, it is possible to enable the system to perform admission control and resource allocation for new streaming sessions.

6. Summary and Future Work

In this work we have taken a first look at the properties of aggregate available bandwidth in a many-to-one data transfer model. Our results consistently show that by increasing the number of senders in the aggregate flow it is possible to obtain reasonably accurate estimations of the aggregate flow's parameters such as mean and standard deviations of the aggregate throughput using a short initial measurement period.

This finding could open up a new way to perform network resource allocation, traffic engineering, as well as transmission scheduling in bandwidth-sensitive applications, such as voice conference or video streaming. The authors are in the process of expanding the experiments to obtain a wide range of measurement data to verify the effectiveness and applicability of this approach; exploring and designing practical protocols/algorithms to apply the many-to-one data transfer model in real multimedia applications; and developing the theoretical framework to characterize the general properties of the many-to-one data transfer model.

7. Acknowledgements

The authors wish to express their gratitude to the anonymous reviewers for their constructive comments in improving this paper. This work was funded in part by an Earmarked Grant (CUHK4211/03E) from the HKSAR Research Grant Council, and the UGC Area of Excellence in Information Technology Scheme (AoE/E-01/99). This work is affiliated with the Microsoft-CUHK Joint Laboratory for Human-centric Computing and Interface Technologies.

8. References

[1] V. N. Padmanabhan, H. J. Wang, P. A. Chou, and K. Sripanidkulchai, "Distributed streaming media content using cooperative networking," in *ACM NOSSDAV*, Miami, FL, May 2002.

[2] T. Nguyen and A. Zakhor, "Multiple Sender Distributed Video Streaming," *IEEE Trans. on Multimedia*, vol. 3, pp.315-326, Apr. 2004.

[3] D.Y. Xu, M. Hefeeda, S. Hambrusch and B. Bhargava, "On Peer-to-Peer Media Streaming," *International Conference on Distributed Computing Systems 2002*, Vienna, Austria, pp.363-371, July 2002.

[4] Jin B. Kwon and Heon Y. Yeom, "Distributed Multimedia Streaming over Peer-to-Peer Network" *Euro-Par 2003, 9th International Conference on Parallel and Distributed Computing*, Klagenfurt, Austria, August 2003.

[5] S. C. Hui and Jack Y. B. Lee, "Modeling of Aggregate Available Bandwidth in Many-to-One Data Transfer," *Proc. of the Fourth International Conference on Intelligent Multimedia Computing and Networking*, July 21-26, 2005, Utah, USA.

[6] S. C. Hui and Jack Y. B. Lee, "Playback-Adaptive Multi-Source Video Streaming," *Proc. of the Fourth International Conference on Intelligent Multimedia Computing and Networking*, July 21-26, 2005, Utah, USA.

[7] M. Jain and C. Dovrolis, "End-to-End Available Bandwidth: Measurement Methodology, Dynamics, and Relation with TCP Throughput," *IEEE/ACM Trans. on Networking*, vol.11, no. 4, pp. 537-549, Aug. 2003

[8] N. Hu and P. Steenkiste, "Evaluation and Characterization of Available Bandwidth Probing Techniques," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 6, pp. 879-894, Aug. 2003.

[9] V. Jocobson. "Pathchar: A Tool to Infer Characteristics of Internet Paths," ftp://ftp.ee.lbl.gov/pathchar/, Apr. 1997

[10] A. Downey, "Using Pathchar to Estimate Internet Link Characteristics," in *Proc. of ACM SIGCOMM*, Sep, 1999

[11] B. A. Mah, "pchar: a Tool for Measuring Internet Path Characteristics," http://www.kitchenlab.org/www/bmah/Software/pchar/ , Feb. 2005

[12] R. L. Carter and M. E. Crovella, "Measuring Bottleneck Link Speed in Packet-Switched Networks," *Performance Evaluation*, 27,28:297 – 318, 1996.

[13] C. Dovrolis, P. Ramanathan, and D. Moore, "What do Packet Dispersion Techniques Measure?" *In Proceedings of IEEE INFOCOM*, pp. 905 – 914, Apr. 2001.

[14] Planetlab Homepage : http://www.planet-lab.org/

[15] V. Paxson and S. Floyd, "Wide-area traffic: The failure of poisson modeling," *IEEE/ACM Trans. on Networking*, vol.2, issue 3, pp. 226-244, Aug. 1994