Chapter 2
Analytical Model of On-Demand Streaming Services Based on Renewal Reward Theory

Hiroshi Toyoizumi

Abstract We propose an analytical model based on renewal reward theory to investigate the dynamics of an on-demand streaming service. At the same time, we also propose a simple method combining a method of multicasts and method of unicasts that can reduce the download rate from the streaming server without causing delay. By modeling the requests as a Poisson arrival and using renewal reward theory, we study the dynamics of this streaming service and derive the optimal combination of unicast and multicast methods. We even show how to estimate the fluctuation of download rates of a streaming service.

2.1 Introduction

Streaming services have become increasingly popular in recent years. However, establishing an efficient large-scale streaming service is still a great challenge because they demand an enormous amount of bandwidth for servers delivering contents. Thus, it is quite important to find an efficient and reliable way to establish a large-scale streaming service over the Internet. There is much research going on to find a better streaming service. For example, [1], [2] proposed a streaming service based on the sophisticated data fragment technique, whereas [3] discusses the possibility of popularity-based delivery and [4] seeks the dynamic structure of a contents delivery network, both aiming to reduce bandwidth. Because there is a wide variety of methods, it is also quite important to evaluate and compare the proposed methods and find the optimal strategy [5]. In most cases, the evaluation is based on the study of arbitrary selected simulations. Only [6], [7] discuss theoretical analysis of reduction of bandwidth of streaming service, but they only succeeded in giving theoretical bounds. In order to understand the dynamics of streaming service, we need an analytically tractable model.

H. Toyoizumi
Graduate School of Accoutancy and Department of Applied Mathematics, Waseda University, Tokyo 169-8050, Japan
e-mail: toyoizumi@waseda.jp

W. Yue et al. (eds.), *Advances in Queueing Theory and Network Applications,*

In this chapter, we propose a simple method combining unicasts and multicasts to reduce the download rate of streaming service. Assuming the arrival of requests is Poisson arrival, we use the technique called renewal reward theory to investigate the dynamics of streaming service. By this analysis, we show we can reduce the download rate by the order of $\sqrt{\rho}$, where ρ is the average download rate required if we use the standard unicast streaming service. Renewal reward theory is one of the fundamental and powerful tools to investigate stochastic processes (see [8], [9], for example). We can derive not only the average overall download rate but also its distribution. This method can be used to design the link speed of the streaming service.

Consider setting up a streaming service (Fig. 2.1). If we use a unicast from the streaming server on each request, users will not experience delay, because the unicast delivers the data on a one-to-one basis. However, using unicasts will result in the waste of bandwidth if users request the same content at the same time. Multicast streaming is realized by copying the data at multicast nodes in a content delivery network so as to reduce the bandwidth. Unlike unicast, multicast is one-to-many, and multicast can deliver the same data to all the users efficiently when sending the same content. However, there is a side effect in multicast streaming. Those who requested later than the start of multicast miss the initial part of the stream. Thus, we propose a simple method using both the unicast and multicast reducing download rate without causing delay. The objective of our method is to reduce the bandwidth required for the streaming server.

Assume there is only one content on the streaming server, for simplicity. We may extend our model to the heterogeneous contents environment, by modeling virtual streaming servers for each content, and treat them separately. A user (or a leaf node of a content delivery network) submits a request for the content to the designated streaming server. The server has two possible options:

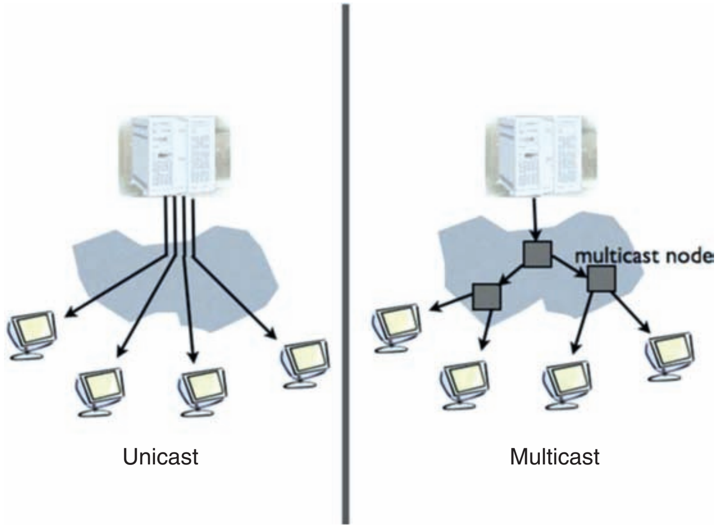

Fig. 2.1 Streaming service on network of unicast and multicast.

(1) Use a multicast so that other users can listen to this content simulataneously, or
(2) Use a unicast that can be listened to by this user exclusively.

Let us see some details on how our method will work. At the time when a user submits a request, if there is no multicast stream, the server has no choice but initiates a new multicast stream. If the server has already started a multicast stream, the server can use a unicast to reduce bandwidth. However, in some cases, the server may save some bandwidth by starting a new multicast even if there is another multicast stream. Figure 2.2 shows an example of how the requests may be handled. Each upward arrow indicates the arrival of requests at the streaming server. The request C1 arrives at the server when there is no stream. Thus, there is no choice, and the server automatically starts a multicast stream (real line). The next request C2, on the other hand, starts listening to the C1 multicast stream, as well as the unicast (dashed line) that corresponds to the top part to which she missed listening (Figure 2.3). The unicast C2 will be terminated when it catches up to the part that has been already stored by listening to the C1 multicast stream. In this way, the request C2 will not see the delay, while saving the bandwidth. At the request C3, the sever selects a new multicast even though there is a C1 multicast. This is because even if C3 started listening to the multicast C1, which has already been started quite some time ago,

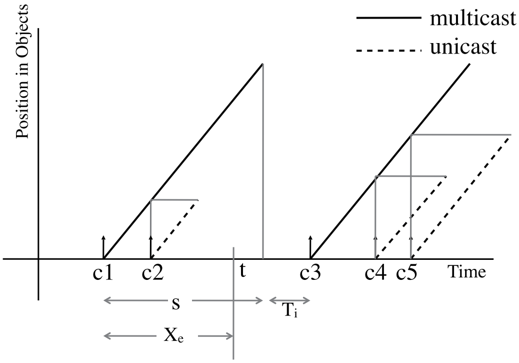

Fig. 2.2 Streaming with unicasts and multicasts. Arrows are the arrivals of request. The vertical line shows the amount the user has listened.

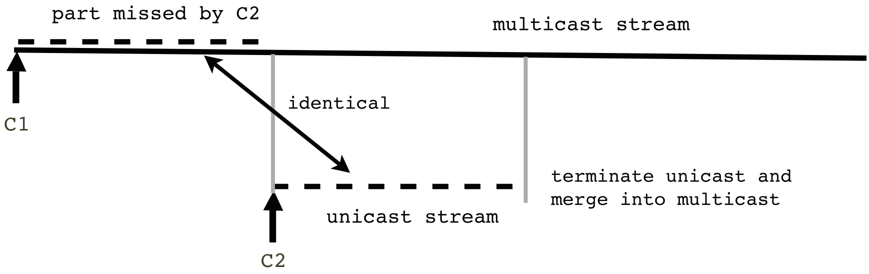

Fig. 2.3 Relationship of C1 multicast and C2 unicast.

C3 has to listen to almost the entire contents by her own unicast. So, there is almost no gain. Thus, instead of listening to the existing multicast, starting a new multicast will reduce the download rate required for future requests, such as C4 and C5. In this chapter, we assume we cannot use the information of future arrival times. So, we need to make a decision under uncertainty.

Note that our method may be applied to a content delivery network on a Peer-to-Peer (P2P) network [4], [10], as well as normal full-scale multicast platforms.

This chapter is organized as follows. In Sect. 2.2, we propose an analytical model to study the optimal strategy for this streaming service, using renewal reward theory. In Sect. 2.3, we present the mean download rate and the optimal strategy. In Sect. 2.4, we derive the download rate distribution. We give some conclusions and remarks in Sect. 2.5.

2.2 Streaming Services and Renewal Model

Assume the server has only one content of the length s, and its download rate of each stream is 1. The arrival of requests is assumed to be a Poisson process with the rate λ. Although this assumption is a mathematical convention, there is research that we can observe a Poisson arrival at the multimedia server in some cases [11].

Suppose that a request arrives at the server at time 0, and the server starts a multicast for this request. Let us assume that all requests arrived during $(0,x]$ are regarded as children of the parent multicast, and the server starts a unicast for each child request. Obviously, x should be no more than the contents length s. Those designated as child requests should listen to the parent multicast and her own unicast simultaneously. The first request arrived after x becomes a new parent and the server starts a new parent multicast. We call x the merging limit time. Our primary goal is to find the optimal x minimizing the total download rate required, using the renewal reward process argument.

Let $N(t)$ be the number of requests arrived during $(0,t]$, and T_n be the arrival time of the nth request ($T_0 = 0$). We evaluate R which is the volume downloaded from the server for the parent and his $N(x)$ child requests; that is,

$$R = \sum_{i=1}^{N(x)} T_i + s, \tag{2.1}$$

because the server has to send the part T_i, which the child request C_i missed listening to in the parent multicast. By conditioning on $N(x)$, we have the expectation of R as

$$\begin{aligned} \mathrm{E}[R] &= s + \mathrm{E}\left[\sum_{i=1}^{N(x)} T_i\right] \\ &= s + \mathrm{E}\left[\mathrm{E}\left[\sum_{i=1}^{N(x)} T_i \middle| N(x)\right]\right]. \end{aligned} \tag{2.2}$$

The arrival is a Poisson process, conditioning on $N(x) = n$, thus the sequence of arrivals $T_1, T_2, \dots, T_n$ is known to be equivalent to the ordered statistics of $U_1, U_2, \dots, U_n$ which is a series of independent and identical random variables uniformly distributed on $(0, x]$ (e.g., see [8, Theorem 2.3.1]). Thus,

$$\begin{aligned} \mathrm{E}\left[\sum_{i=1}^{N(x)} T_i \middle| N(x) = n\right] &= \mathrm{E}\left[\sum_{i=1}^{n} U_i\right] \\ &= \frac{nx}{2}. \end{aligned}$$

Using this in (2.2), we have

$$\mathrm{E}[R] = s + \mathrm{E}\left[\frac{N(x)x}{2}\right] = s + \frac{\lambda x^2}{2}. \tag{2.3}$$

Now, let X_m be the interarrival time of the mth parent multicast. Because the interarrival time of the Poisson process is exponentially distributed and memoryless, the time length to the next request after the merging time limit is again exponentially distributed with its mean $1/\lambda$. Hence, X_m are independent and have the form of

$$X_m = x + T_i, \tag{2.4}$$

where T_i is an exponential random variable with the mean $1/\lambda$. Also, let R_m be the volume downloaded by the mth parent multicast and its child unicasts. Because the arrival is a Poisson process, the sequence of the pair of random variables $(X_m, R_m)_{m=1,2,\dots}$ is independent and identically distributed. Let $S(t)$ be the total accumulated volume demanded by requests whose parent arrived before the time t; in other words,

$$S(t) = \sum_{m=1}^{M(t)} R_m, \tag{2.5}$$

where $M(t)$ is the number of parent multicasts in $[0, t)$. Taking R_m as the reward, the process $S(t)$ is a renewal reward process (see e.g., [8], [9]). This renewal reward representation is used in the following section to derive the average download rate.

2.3 Mean Download Rate and Optimal Strategy

We now find the optimal merging limit time x_0 that minimizes the average download rate from the streaming server. Let $b(x)$ be the average download rate given the merging limit time x, or

$$b(x) = \lim_{t \to \infty} \frac{S(t)}{t}. \tag{2.6}$$

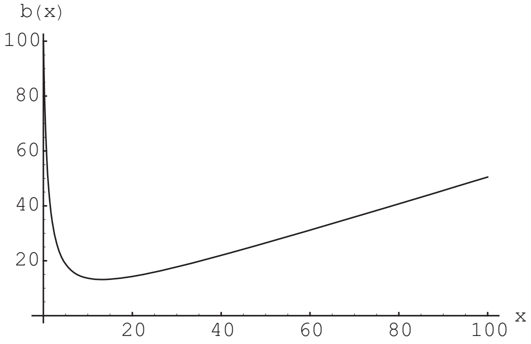

Fig. 2.4 The download rate $b(x)$ and the merging limit time x: the request arrival rate $\lambda = 1$, and the content size $s = 100$.

Theorem 2.1 (Optimal Merging Limit Time). *Assume the requests to the content of size s is a Poisson process with the rate λ. Given the merging limit time x, the average download rate is obtained by*

$$b(x) = \frac{2\lambda s + \lambda^2 x^2}{2(\lambda x + 1)}. \tag{2.7}$$

The function $b(x)$ is indeed a convex function (see Fig. 2.4), so we have x_0 which minimizes $b(x)$ as

$$x_0 = \frac{(1 + 2\lambda s)^{1/2} - 1}{\lambda}. \tag{2.8}$$

Furthermore, we can substitute (2.8) into (2.7); then we have the optimal download rate,

$$b(x_0) = (1 + 2\rho)^{1/2} - 1, \tag{2.9}$$

where $\rho = \lambda s$ corresponds to the scale of this streaming service.

Proof. We know that $S(t)$ is a renewal reward process from Sect. 2.2. Renewal reward theory [8, Theorem 3.6.1] is an extension of the strong law of large numbers to the renewal process. By the strong law of large numbers and (2.5), with probability 1, we have

$$\frac{S(t)}{t} = \frac{\sum_{m=1}^{M(t)} R_m}{M(t)} \frac{M(t)}{t} \to \frac{\mathrm{E}[R]}{\mathrm{E}[X]} = \frac{\mathrm{E}[R]}{x + 1/\lambda} \quad \text{as } t \to \infty, \tag{2.10}$$

where X is the interarrival time of the parent multicasts. It is easy to get (2.7) by substituting (2.3) in (2.10).

Figure 2.4 shows the graph of the average download rate $b(x)$. For a smaller merging limit time x, more requests are treated as multicast, which results in a waste of download rate. On the contrary, for a larger x, we may miss the opportunity of saving download rate by merging the future streams. Thus, we can see a fine balance here. In this case the optimal merging limit time is $x_0 = 9.04988$, well below the content size $s = 100$.

Let us study in some detail the optimal merging limit. Take the optimal merging limit time as a function of the request arrival rate λ in (2.8). Letting $\lambda \to 0$, we have

$$x_0(\lambda) \to s,$$

which means for a smaller request rate we cannot count on the following requests, so "be a child whenever you can" is the best strategy. On the contrary, for large λ, we have

$$x_0(\lambda) \to 0, \quad \text{as } \lambda \to \infty.$$

For a larger request rate, you can always expect the following requests. In this case your strategy would be "be a parent and help the following children."

If we use unicast only, instead of the combination of unicast and multicast, the average streaming rate is ρ. The download rate (2.9) obtained by our method has the order of $\sqrt{\rho}$, which gives us a significant saving of download rate, especially when the size of the streaming service is large (see Fig. 2.5). Theoretically, we could improve (2.9) when we exploit the information of future requests. The theoretical lower bound of the download rate given future information was obtained by [6] as

$$b_0 = \log(1+\rho), \tag{2.11}$$

which is also shown in Fig. 2.5. We see that our method cannot achieve this theoretical limit but still it achieves significant saving.

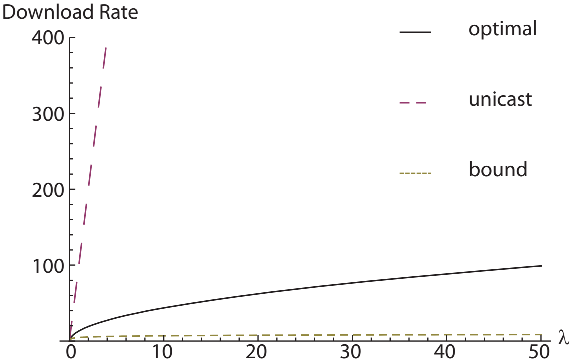

Fig. 2.5 Comparison of download rate: The line unicast is the scheme that uses only unicasts, and the bound is the theoretical lower bound [6]. The contents size is set to be $s = 100$.

2.4 Download Rate Distribution

Using the renewal argument further, we can evaluate the download rate distribution of our merging method. For simplicity, in this section we set the merging limit time x to be the size of the content s. In this case, all requests are treated as child streams whenever they can.

Because we set the download rate of each stream to be 1, the only thing we need to know is the number of active streams. Let L be the number of active streams including both parent and child streams in the steady state.

Theorem 2.2. *The z-transform of the number of active streams L is obtained as*

$$\mathrm{E}\left[z^L\right] = \frac{1}{1+\rho}\left[ze^{(\rho-1)(z-1)/2}\int_{e^{-\rho}}^{1} e^{(z-1)y/2}\frac{dy}{y} + \frac{2}{z+1}\left\{e^{-\rho(1-z)/2} - e^{-\rho}\right\} + e^{-\rho}\right], \tag{2.12}$$

where we set $\rho = \lambda s$.

Proof. Let $L(t)$ be the number of active streams at the time t, and let Y_e be the length to the arrival of the previous parent request from an arbitrary time t (Fig. 2.6). Because Y_e is the forward recurrent time of the renewal interval $s + T_i$, where T_i is exponentially distributed with the mean $1/\lambda$, we have

$$P\{Y_e \leq u\} = \frac{1}{s+1/\lambda}\int_0^u (1 - P\{s+T \leq y\})dy. \tag{2.13}$$

Thus, we obtain the probability distribution of Y_e as

$$(1+\rho)P\{Y_e \leq u\} = \begin{cases} \lambda u & \text{if } u \leq s \\ 1+\rho - e^{-\lambda(u-s)} & \text{if } u > s, \end{cases} \tag{2.14}$$

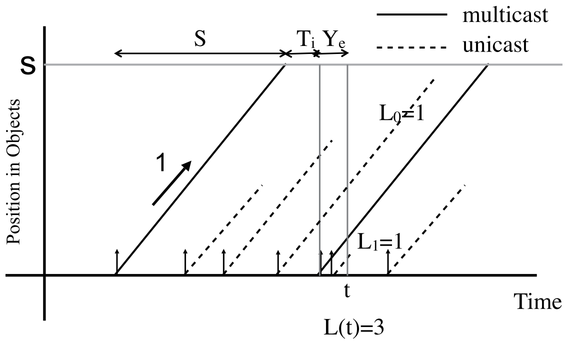

Fig. 2.6 Sample path of streaming service: The fourth child unicast from the previous renewal interval remains active at the time t.

and its density as

$$(1+\rho)\frac{dP\{Y_e \le u\}}{du} = \begin{cases} \lambda & \text{if } u \le s \\ \lambda e^{-\lambda(u-s)} & \text{if } u > s. \end{cases} \tag{2.15}$$

Let L_1 be the number of active child streams at time t that arrived in the renewal interval containing the time t. Furthermore, there is a chance that the child streams started prior to the renewal time still exist after time t (see Fig. 2.6). Let L_0 be the number of those active streams that arrived in the previous renewal interval.

Then, taking into account the parent multicast of this renewal interval, we have

$$L(t) = 1_{(Y_e \le s)} + L_0 + L_1. \tag{2.16}$$

Consider conditioning on $Y_e = u$. In the case when $u \le s$, L_0 and L_1 are independent, and both are Poisson random variables with the mean $\lambda s P_0(u)$ and $\lambda u/2$, respectively, where $P_0(u)$ is the probability that a child stream started in the previous interval still exists at time t. Indeed, the arrival of child streams is Poisson with rate λ, and given the number of arrivals, the survival of each child stream is independent of other streams.

Suppose a child stream arrives U_1 later than the parent multicast that started the current renewal interval. The child stream should exist to cover the missing part of the length U_1, and it is alive up to $2U_1$ from the start of the parent multicast. The child stream exists at time t only when $2U_1 > u$. Because U is uniformly distributed on $[0,u]$, the probability that a child stream exists at time t is $P\{U_1 \ge u/2\} = 1/2$. Thus, L_1, the number of active child streams at time t that arrived in $[t-u,t]$, is a Poisson random variable with the mean $\lambda u/2$. Similarly, suppose a child stream in the previous renewal interval arrives U_0 after the previous parent multicast. Then, the child stream remains active at time t only when $2U_0 > s+T+u$. Thus,

$$\begin{aligned} P_0(u) &= P\{2U_0 > s+T+u\} \\ &= \{\lambda(s-u) - (1-e^{-\lambda(s-u)})\}/(2\rho), \end{aligned} \tag{2.17}$$

because U_0 is a uniform random variable on $[0,s]$. Thus L_0 is a Poisson random variable with the mean $\lambda s P_0(u)$. Using this information we have

$$\begin{aligned} \int_0^s \mathrm{E}\left[z^{L(t)} \middle| Y_e = u \right] dP\{Y_e \le u\} &= \int_0^s z e^{\lambda s P_0(u)(z-1)} e^{\lambda u(z-1)/2} dP\{Y_e \le u\} \\ &= \frac{\lambda z e^{(\rho-1)(z-1)/2}}{1+\rho} \int_0^s e^{(z-1)e^{-\lambda(s-u)}/2} du \\ &= \frac{z e^{(\rho-1)(z-1)/2}}{1+\rho} \int_{e^{-\rho}}^1 e^{(z-1)y/2} \frac{dy}{y}. \end{aligned} \tag{2.18}$$

On the other hand, when $s < u \le 2s$, it is easy to see that no child streams from the previous interval exist at time t. Thus, $L_0 = 0$ and L_1 is a Poisson random variable with its mean $\lambda(s-u/2)$. Hence we have

$$\int_s^{2s} \mathrm{E}\Big[z^{L(t)}\Big|Y_e = u\Big]\,dP\{Y_e \le u\} = \int_s^{2s} e^{\lambda(s-u/2)(z-1)}\,dP\{Y_e \le u\}$$
$$= \frac{2}{(1+\rho)(z+1)}\left\{e^{-\rho(1-z)/2} - e^{-\rho}\right\}. \tag{2.19}$$

Lastly, when $u > 2s$, $L(t) = 0$. Hence, we have

$$\int_{2s}^{\infty} \mathrm{E}\Big[z^{L(t)}\Big|Y_e = u\Big]\,dP\{Y_e \le u\} = \int_{2s}^{\infty} dP\{Y_e \le u\}$$
$$= \frac{1}{(1+\rho)}e^{-\rho}. \tag{2.20}$$

By using all these results and by separating integral intervals appropriately, we can get (2.12).

Corollary 2.1. *The mean and variance of L are given by*

$$\mathrm{E}[L] = \frac{2\rho+\rho^2}{2(1+\rho)} < \rho, \tag{2.21}$$
$$V[L] = \{4\rho^3 - 4\rho^2 + 11\rho + 9 - 4(\rho^2+3\rho+2)e^{-\rho}$$
$$-(\rho+1)e^{-2\rho}\}/\{8(1+\rho)^2\}. \tag{2.22}$$

Here we give a numerical example. If we use only unicasts for requests, L is nothing but a simple M/D/∞ queueing system. Thus, L is a Poisson random variable with its mean $\rho = \lambda s$. In Fig. 2.7, we compare the variance of L of the proposed merging method with the M/D/∞ queue. We already know that we can save the average download rate using our proposed method. In Fig. 2.7, we also see the reduction of the download rate fluctuation, which is another superiority of our method.

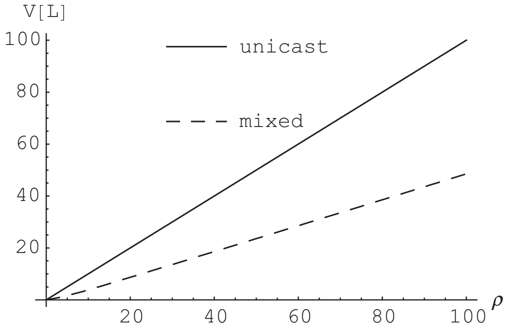

Fig. 2.7 Variance of L in the unicast scheme and the proposed method (mixed).

2.5 Conclusions

In this chapter, we proposed a simple method realizing bandwidth reduction without delay. By using renewal reward theory, we succeed in estimating the download rate, not only the average but also the variance. By using the evaluation, we find that in an optimal case we can reduce the download rate of streaming service by $\sqrt{\rho}$, the squareroot of the streaming service size. Furthermore, we see that our proposed method can also reduce the fluctuation of the download rate. The technique used in this chapter can be adopted to design the bandwidth requirement for general streaming services.

References

1. K. A. Hua and S. Sheu, Skyscraper broadcasting: A new broadcasting scheme for metropolitan video-on-demand systems, in *Proc. SIGCOMM*, pp. 89–100, 1997. [Online]. Available: citeseer.nj.nec.com/hua97skyscraper.html.
2. J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege, A digital fountain approach to reliable distribution of bulk data, in *Proc. SIGCOMM*, pp. 56–67, 1998. [Online]. Available: citeseer.nj.nec.com/byers98digital.html.
3. K. Thirumalai, J. F. Paris, and D. D. E. Long, Tabbycat: an inexpensive scalable server for video-on-demand, in *Proc. IEEE International Conference on Communications, ICC'03*, vol. 2, pp. 896–900, 2003.
4. M. Tran and W. Tavanapong, Peers-assisted dynamic content distribution networks, in *Proc. The IEEE Conference on Local Computer Networks*, pp. 123–131, 2005.
5. A. Mahanti, D. Eager, M. Vernon, and D. Sundaram-Stukel, Scalable on-demand media streaming with packet loss recovery, in *Proc. SIGCOMM'2001*, p. 12, 2001. [Online]. Available: citeseer.nj.nec.com/mahanti01scalable.html.
6. D. L. Eager, M. K. Vernon, and J. Zahorjan, Minimizing bandwidth requirements for on-demand data delivery, *Knowledge and Data Engineering*, vol. 13, no. 5, pp. 742–757, 2001. [Online]. Available: http://citeseer.nj.nec.com/eager99minimizing.html.
7. D. L. Eager, M. K. Vernon, and J. Zahorjan, Optimal and efficient merging schedules for video-on-demand servers, in *Proc. ACM Multimedia (1)*, pp. 199–202, 1999. [Online]. Available: citeseer.nj.nec.com/eager99optimal.html.
8. S. M. Ross, *Stochastic Processes*. New York: John Wiley and Sons, 1996.
9. R. Wolff, *Stochastic Modeling and the Theory of Queues*. New Jersey: Prentice Hall, 1989.
10. Y. Guo, K. Suh, J. Kurose, and D. Towsley, A peer-to-peer on-demand streaming service and its performance evaluation, in *Proc. International Conference on Multimedia and Expo*, vol. 2, pp. II-649–52, 2003.
11. J. M. Almeida, J. Krueger, D. L. Eager, and M. K. Vernon, Analysis of educational media server workloads, in *Proc. 11th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV '01)*, pp. 21–30, 2001.