Combinatorial Optimization in Communication Networks
Maggie Cheng, Yingshu Li and Ding-Zhu Du (Eds.) pp. – - –

Dynamic Simulcasting: Design and Optimization

Jiangchuan Liu
School of Computing Science
Simon Fraser University, Burnaby, BC, Canada
E-mail: jcliu@cs.sfu.ca

Bo Li
Department of Computer Science
The Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong
E-mail: bli@ust.hk

Alan T.S. Ip
Department of Computer Science and Engineering
The Chinese University of Hong Kong, Shatin, N.T., Hong Kong
E-mail: tsip@cse.cuhk.edu.hk

Ya-Qin Zhang
Microsoft Corporation, One Microsoft Way, Redmond, WA 98052-6399, USA
E-mail: yzhang@microsoft.com

Contents

Abstract

Video simulcasting enables a sender to generate replicated streams of different rates for a video program, targeting for a set of heterogeneous receivers. Each stream can then be distributed to the target receivers via a broadcast or multicast channel. One of the critical issues in video simulcasting is how to strike a balance between bandwidth consumption and user satisfaction: first, replication introduces redundancy that can potentially lead to an excessive use of bandwidth; second, there is usually a mismatch between each individual receiver's bandwidth requirement and the limited choices among the stream rates. In this chapter, we present *dynamical simulcasting*, which adaptively determines the optimal number of streams that should be generated, and the optimal bandwidth that should be allocated to each stream. Its performance is studied under a variety of system configurations, and the results conclusively demonstrate that dynamic simulcasting can significantly improve user satisfaction, while the computation overhead is kept at a low level.

1 Introduction

With the rapid development and deployment of broadband networks, real-time video distribution is emerging as one of the most important networked applications. The multi-user nature of video programs makes broadcast or multicast an efficient method for delivering video content to a large population of receivers. Such systems have also been effectively supported by existing network infrastructures, *e.g.*, by wireless networks and the Internet using IP multicast or application-layer multicast [27]. A key challenge, however, is how to handle user heterogeneity, as receivers with different platforms, such as PDAs, laptops, and PCs, or with different connection speeds, such as 1.5 Mbps ADSL or 10 Mbps Ethernet, all expect to access video services nowadays. Clearly, a single-rate transmission, though simple, is difficult to satisfy these receivers given their diverse bandwidth requirements and the only one choice of streaming rate.

Simulcasting has been introduced as a viable vehicle to mitigate such mismatches and hence improve user satisfaction [3, 8, 31]. A simulcast server maintains some replicated streams for the same video content but with different rates, and delivers each stream to a specific set of receivers via a multicast channel. Two questions arise for video simulcasting naturally: how to choose an appropriate number of streams, and how to allocate bandwidth to the streams to reduce bandwidth mismatches?

In this chapter, we for the first time present a formal study on the above issues, to which we refer as the *stream replication problem*. The key objective here is to strike a balance between bandwidth economy and user satisfaction; in other words, given some bandwidth constraint, a configuration of replication should minimize the expected bandwidth mismatch for all the receivers. Clearly, the optimal replication must be adaptively configured in a non-static network environment, leading to a *dynamic simulcasting* paradigm.

We first formulate the stream replication problem for a given number of streams, and derive an optimal algorithm. This optimization follows the design nature of most existing simulcasting systems, *i.e.*, the number of streams is predetermined by system operators or service providers [8].

Through theoretical analysis and experimental results, we then demonstrate that the number of replicated streams is a critical factor in the overall system optimization. It is necessary to choose an *optimal*, not an *ad hoc*, number of streams based on the available resources as well as receivers' requirements. In addition, recent advances in video coding have shown that a stream can be replicated or encoded in real-time by fast compression domain transcoding [16]. The fast and low-cost operations for stream setup and termination have also been supported in advanced video streaming standards, such as the MPEG-4 Delivery Multimedia Integration Framework (DIMF) [1]. Given such flexibility, it is possible to adaptively regulate the number of streams to accommodate the receivers' requirements. Therefore, we further consider the use of a flexible number of video streams, and derive an efficient algorithm to jointly optimize the number as well as the bandwidth allocated to each stream.

We also investigate the bandwidth allocation among different video programs (sessions). We note that, for the optimal replication, the expected mismatch of the receivers is a stepwise function of the session bandwidth (the total bandwidth of the replicated streams). As a result, an equal bandwidth allocation often leads to a waste of session bandwidth. We therefore introduce a novel mismatch-aware allocation scheme. It intelligently distributes the unused bandwidth to other sessions, yet preserving general fairness properties.

The performance of dynamic simulcasting is examined under a variety of settings. The results conclusively demonstrate that it significantly outperforms static non-optimal schemes, and the use of the joint optimization for stream number and bandwidth further reduces the bandwidth mismatch. We also study the impact from a number of key factors, including the available bandwidth and the receivers' bandwidth distribution. The results offer some new insights into video simulcasting, and quantitatively demonstrate the various trade-offs between bandwidth efficiency and user satisfaction under different conditions; thus provide a general guideline for capacity planning and bandwidth allocation.

The rest of the chapter is organized as follows. Section 2 presents the background and some related work. The system model for dynamic simulcasting is described in Section 3. Section 4 formulates the problem of optimal steam replication for a single session, and presents efficient algorithms. The inter-session bandwidth allocation is studied in Section 5. Some implementation and computation issues are discussed in Section 6. We then present the performance results in Section 7, and conclude the chapter in Section 8.

2 Related Work

2.1 Simulcasting Protocols

There has been a significant amount of work on simulcasting in the literature. A representative is the Destination Set Grouping (DSG) protocol [3], which targets the best-effort Internet with IP multicast. In DSG, a source maintains a small number of video streams (say 3) at different rates. A receiver subscribes to a stream that best matches its bandwidth. It also estimates the network status according to the packet loss ratios, and reports its estimation to the sender through a scalable feedback protocol. If the percentile of the congested receivers for a stream is above a certain threshold, the bandwidth of the stream is reduced by the sender. If all the receivers experience no packets loss, the stream bandwidth is increased. The choice of the threshold is experience-based, which is not necessarily optimal. In [26], some heuristics are proposed for stream bandwidth adjustment. The objective is to reduce the overall bandwidth originated from the server as well as the aggregate bandwidth in local regions, while not to minimize bandwidth mismatches. An optimal bandwidth allocation algorithm that minimizes bandwidth mismatches for a 2-stream case is proposed in [7]. The algorithm performs an exhaustive search on the receivers' expected bandwidths, based on an observation that the optimal value of the source bandwidth must be one of them. Due to the high complexity of exhaustive search, it is not easy to directly extend this algorithm with more layers.

Simulcasting has also been introduced in many commercial video streaming systems. For example, RelaNetworks' RealSystem G2 supports simulcasting under the name of *SureStream* [8], which generates a fixed number of streams at prescribed rates, and a receiver can dynamically choose a stream commensurate with its bandwidth. Nevertheless, the use of dynamical allocation on the sender's side has not been addressed in these commercial systems.

2.2 Simulcasting versus Layered Transmission

Simulcasting is one of the representatives of multi-rate multicasting for heterogeneous receivers. Another is *layered transmission* [9, 10, 12], in which a sender generates multiple streams (called *layers*) that can progressively refine the video quality. A receiver thus can subscribe to a subset of layers commensurate with its bandwidth.

Layered transmission also suffers from bandwidth mismatches, because adaptation on the receiver's side is at coarse-grained layer level. To minimize this mismatch, protocols using dynamic layer rate allocation on the sender's side have been proposed for cumulative layering, where the layers are subscribed cumulatively staring from a base layer [36, 10, 11, 12]. However, the constraints and hence the optimization strategies for layer rate allocation and stream replication are quite different. For illustration, consider a single video session (program) with a given total bandwidth N and total number of replicated streams (or layers) K. As will be explained later, we assume bandwidth allocation is discrete. For simulcasting, the problem of stream rate allocation is thus to find an optimal K-partition for integer N; for cumulative layering, it is to find an optimal enumeration of K numbers with the maximum one being less than or equal to N. In addition, it can be proved that, the more layers a layered transmission protocol uses, the smaller the mismatch that a receiver would experience (ignoring the layering overheads) [12]. Thus, the use of 'thin layers' has been advocated in some existing protocols [6]. For simulcasting, this is not true, because stream replication would introduce very high redundancy with a large number of streams. As such, it is necessary to find an optimal number of streams.

Different from cumulative layering, noncumulative layering allows a receiver to subscribe any subset of layers that is commensurate with its bandwidth, and hence is more flexible. In this scenario, Byers *et al.* [21] suggest the use of a Fibonacci-like layer bandwidth allocation, which not only enables a fine-grained receiver adaptation, but also minimizes the layer join or leave actions for receivers in a dynamic environment. Their work does not specifically target video distribution; the constraints of session bandwidth and layer number for existing non-cumulative layered video coders, such as the multiple description coders [1], have yet to be addressed.

The above adaptation algorithms generally rely on end-to-end services. There are also distributed rate adaptation algorithms involving operations at both end-nodes and intermediate switches. Fei *et al.* [14] study the construction of multicast trees for rate-adaptive replicated servers. Kar *et al.* [13] study the problem of maximizing the total utility for layered multicast. They assume that some junction nodes are deployed inside the network, and propose two distributed algorithms that converge to the optimal allocation using coordination among the sender, receivers, and junction nodes. In this chapter, we focus on the end-to-end adaptation only, which does not rely on special assistances from intermediate nodes, offering solutions readily applicable to the current best-effort Internet.

Finally, it is worth noting that simulcasting and layering have been compared in many different contexts, such as Internet multicasting [4], TCP-friendly streaming [29], proxy cache assisted streaming [30]. Though it is often believed that layering achieves higher bandwidth utilization as there is no overlapping among the layers, it suffers from the high complexities as well as the structure constraints for both encoding and decoding [18, 28]. As a result, a layered video stream usually has a lower quality than a single-layer stream of the same rate, and, more importantly, it is not compatible to many existing video formats and decoding algorithms. To the contrary, simulcasting produces independent streams, which can serve receivers with simpler and even heterogeneous decoding algorithms. Therefore, simulcasting remains a promising technique to address user heterogeneity and, as mentioned before, has been supported in many commercial streaming systems.

3 System Model and Definitions

3.1 System Model

In our system, a video server distributes a set of video programs using the simulcasting technique: each video program has several replicated streams of different rates. The descriptions of the video programs are advertised to the receivers via a dedicated multicast channel. A receiver interested in a particular video program can thus subscribe to one of the streams to receive the video. We refer to a program and its receivers as a *simulcast session* (or *session*), and the bandwidth allocated to the session as *session bandwidth*, which imposes an upper bound for the total bandwidth of the replicated streams.

The status of the system can be characterized by a 3-tuple, $(C, P, M_{s,t})$, where C is the total bandwidth of the server; P is the total number of sessions, and each session has an index in $[1, \dots, P]$; $M_{s,t}$ is the ratio that a receiver in session s has expected bandwidth t; that is, assume in session s, the total number of receivers is n and the number of receivers with expected bandwidth t is n_t, we have $M_{s,t} = n_t/n$. We stress that this model captures the essentials of many existing video multicasting or broadcasting systems, in which the expected bandwidths of the receivers are heterogeneous and limited by their processing capabilities or access links; the video server, though having a higher output bandwidth, has to accommodate many simultaneous sessions (each with several replicated streams). Therefore, the bandwidth to the sessions as well as to the streams should be carefully allocated. To this end, an end-to-end adaptation framework is adopted in our system, in which both the sender and the receivers perform adaptation to maximize the overall user satisfaction. Specifically, a receiver always tries to subscribe to a stream that best matches its expected bandwidth. It also periodically reports its expectation to the server. The server thus dynamically allocates bandwidth among the sessions as well as the streams within a session according to these expectations. In a system that allows a flexible setting for the number of streams, the server also adaptively estimates the optimal number for each session and then regulates the setting accordingly.

In this chapter, we mainly focus on developing the optimization framework for dynamic simulcasting. The mechanisms for receiver bandwidth estimation and report are out of its scope. These two issues have been extensively studied in the literature, and many of the algorithms can be applied in our system, *e.g.*, [10] (receiver-based bandwidth estimation) and [2, 10] (scalable reporting).

3.2 Measurement of Bandwidth Mismatch

Note that a receiver cannot subscribe to a fraction of a video stream. Assume a receiver's expected bandwidth is t, and the bandwidth of the stream is r, we measure the bandwidth mismatch as:

$$RM(t,r) = \begin{cases} (t-r)/t, & 0 < r \leq t \\ 1, & r > t \quad or \quad r = 0 \end{cases} \tag{1}$$

We use such a relative measure (RM) instead of an absolute mismatch measure because RM will not enlarge the impact of the mismatches perceived by wideband receivers. For example, consider a 1 Mbps receiver that subscribes to a stream of 768 Kbps. The absolute mismatch is 256 Kbps, which is even larger than the maximum absolute mismatch that a 128 Kbps receiver could experience. However, for the 1 Mbps receiver, the degradation of its satisfaction is actually not that severe, and, obviously, the RM measure of 0.25 more fairly reflects this degradation. Furthermore, the RM measure has the following properties: 1) it is monotonically decreasing with the increase of r, provided the receiver successfully subscribes to the video stream, *i.e.*, in the case of $r \leq t$; 2) by assigning a very large output, it penalizes the practically undesirable cases, *e.g.*, $r = 0$, the receiver does not receive any stream, or $r > t$, the receiver cannot subscribe to the stream completely.

For a simulcast session, our objective for both sender and receiver adaptations is to minimize the expected RM for all the receivers. There could be other mismatch or user satisfaction measures, as well as mappings from the mismatch to some application-level performance degradation. For example, different fairness measures, such as the Inter-Receiver Fairness (IRF) function [7], or subjective/objective video quality measures, such as the Peak Signal-to-Noise Ratio (PSNR) [1]. Nevertheless, the optimization algorithms presented in this chapter are general enough, which does not impose strict constraints on the measurement function, and can accommodate other mismatch or fairness measures as well.

For convenience, we list the major notations used in this chapter as follows:

C : the total bandwidth of the server;

P : the total number of video programs (also the number of sessions);

$M_{s,t}$: the ratio of the receivers that have expected bandwidth t in session s; $\sum_{t>0}^{\infty} M_{s,t} = 1$; $\sum_{t>0}^{\infty} M_{s,t} = 1$ for each $s \in [1, \ldots, P]$;

l_s : the total number of the replicated streams for session s;

$r_{s,i}$: the bandwidth of stream i for session s;

$\vec{R}_s$: the bandwidth allocation vector for the streams in session s, $\vec{R}_s = r_{s,1}, r_{s,2}, \ldots, r_{s,l_s}$;

$\phi(t, \vec{R}_s)$: the rate of the best-matching stream for a receiver of bandwidth t, $\phi(t, \vec{R}_s) = \max_{r \leq t, r \in \vec{R}_s} r$;

N_s : the session bandwidth for session s;

T_s : the maximum bandwidth of the receivers in session s; $T_s = \max_{M_{s,t}>0} t$;

$ERM(s, N_s)$: Expected Relative Mismatch (ERM) for session s with session bandwidth N_s;

K : the predetermined number of streams for the fixed stream number case (OptFN or ExpFN).

4 Intra-Session Optimization

In this section, we study the problem of intra-session optimization at the sender's end; that is, given the bandwidth of a session, what is the optimal setting of the replicated streams for this session (video program)? This include optimally allocating the bandwidth of the session to its streams and, if needed, determining the optimal number of streams. The method for inter-session allocation and some practical issues are discussed in the next two sections.

4.1 Problem Formulation

Let $\vec{R}_s$ denote the *bandwidth allocation vector* for session s, $\vec{R}_s = (r_{s,1}, r_{s,2}, \ldots, r_{s,l_s})$, where l_s is the total number of the replicated streams for session s, and $r_{s,i}$ is the rate of stream i. Without loss of generality, we assume that $r_{s,1} < r_{s,2} <, ..., < r_{s,l_s}$. In practice, such rates take only discrete values, for two reasons: First, given a finite number of quantizers, the output rate of a video compressor is always discrete [1]; Second, bandwidth allocation is channelized in many multicast or broadcast networks. Therefore, for convenience, we assume the bandwidths (rates) discussed in this chapter are all integer multiples of a basic unit.

For a given $\vec{R}_s$, a receiver with bandwidth t should subscribe to the stream with the best-matching bandwidth $\phi(t, \vec{R}_s) = \max_{r \leq t, r \in \vec{R}_s} r$. This is a relatively simple operation. The challenging problem is how to determine $\vec{R}_s$ on the server's side, to which we refer as *intra-session allocation.*

For session s, the input for intra-session allocation includes the session bandwidth N_s and the receivers' bandwidth distribution $M_{s,t}$. The output is the minimum Expected Relative Mismatch (ERM) for all the receivers in the session, together with the corresponding bandwidth allocation vector $\vec{R}_s$. Assume T_s is the maximum bandwidth of the receivers in session s. Clearly, an optimal allocation should satisfy that $r_{s,1} > 0$ and $r_{s,l_s} \leq T_s$. The optimization problem thus can be formally described as follows:

$$\begin{aligned} &Minimize \quad ERM(s, N_s) = \textstyle\sum_{t=1}^{T_s} M_{s,t} RM[t, \phi(t, \vec{R}_s)], \\ &Subject\ to \quad 0 < r_{s,1} < r_{s,2} <, \ldots, < r_{s,l_s} \leq T_s, \ \textstyle\sum_{i=1}^{l_s} r_{s,i} \leq N_s \end{aligned} \tag{2}$$

As discussed before, we consider two versions of the above optimization problem: 1) Optimal bandwidth allocation for a given (fixed) number of streams (OptFN), and 2) Joint optimization for the number of streams and their respective bandwidths (OptNB). The latter not only provides a general tool for the choice of stream number in designing a simulcast system, but also serves as the foundation for the system employing a flexible setting of number.

4.2 Optimization for Fixed Number of Streams (OptFN)

In this scenario, we assume the total number of streams is fixed to a given K, or $l_s = K$. Hence, only the bandwidth of each stream ($r_{s,k}$ for $k = 1, 2, \ldots, K$) is to be determined.

Lemma 4.1 *There exists an optimal bandwidth allocation vector for problem OptFN.*

Proof. The number of valid allocations is finite because $r_{s,k} \in Z^+$ and $r_{s,K} \leq T_s$, $k = 1, 2, \ldots, K$. Moreover, the RM measure is well defined for each valid allocation. Hence, there exists an optimal vector. □

We now show an efficient algorithm to solve this problem. Define $\alpha(n, m, k)$ as

$$\min_{l_s=k, r_{s,k}=m, \sum_{i=1}^{k} r_{s,i}=n} \sum\nolimits_{t=1}^{T_s} M_{s,t} RM[t, \phi(t, \vec{R}_s)],$$

that is, the minimum ERM when a total number of k streams are generated with a total bandwidth n, and the bandwidth of stream k is m. The solution to problem OptFN is clearly given by $\min\limits_{1 \leq n \leq N_s, 1 \leq m \leq T_s} \alpha(n, m, K)$. We have the following recurrence relation for $\alpha(n, m, k)$,

$$\alpha(n,m,k) = \begin{cases} \sum\limits_{t=0}^{m-1} M_{s,t} RM(t,0) + \sum\limits_{t=m}^{T_s} M_{s,t} RM(t,m), & if\, m = n > 0, k = 1 \\ \min\limits_{1 \leq j < m} \left\{ \alpha(n-m, j, k-1) - DIFF(m,j) \right\}, & \\ \qquad\qquad if\, m \leq n \leq N_s, 1 < k \leq K, k \leq m \leq min\{n, T_s\} & \\ \infty, \quad otherwise & \end{cases} \tag{3}$$

where $DIFF(m, j) = \sum_{t=m}^{T_s} M_{s,t}[RM(t, j) - RM(t, m)]$. The first equation in (3) stands for a boundary case with only one stream ($k = 1$), which occupies all the session bandwidth. For $k > 1$, one more stream is to be added based on a case of k-1. Without loss of generality, assume this stream is stream k, the highest stream. The difference of ERM, when this stream is added, depends only on the bandwidth of itself and that of stream k-1, because only the receivers that originally subscribe to stream k-1 have the potential of subscribing to stream k. Therefore, given bandwidth m of stream k, the difference of ERM is $\sum_{t=m}^{T_s} M_{s,t}[RM(t, j) - RM(t, m)]$, or $DIFF(m, j)$. The minimum ERM for this k-stream case thus can be obtained by checking each possible bandwidth allocated to stream k-1, i.e., bandwidth of $1, 2, \ldots, m-1$, as shown in the second equation in (3).

As a result, the OptFN problem can be solved by dynamic programming. For the RM function, we have $DIFF(m, j) = \sum_{t=m}^{T_s} M_{s,t}\ [RM(t, j) - RM(t, m)] = \sum_{t=m}^{T_s} M_{s,t}(m-j)/t = (m-j)\sum_{t=m}^{T_s} M_{s,t}t^{-1}$. For each given m, $\sum_{t=m}^{T_s} M_{s,t}t^{-1}$ does not change in the execution of the algorithm. Hence, the values of $\sum_{t=m}^{T_s} M_{s,t}t^{-1}$ for $m = 1, 2, \ldots, T_s$ can be pre-calculated and stored in space $O(T_s)$. Since the size of array $\alpha(n, m, k)$ is $N_s \cdot min\{N_s, T_s\} \cdot K$, and, for each entry of α, we need $O(N_s)$ iterations to find the bandwidth for stream $k-1$ (assume $T_s \leq N_s$), the complexity of the optimal allocation algorithm is bounded by $O(N_s^3 K)$. The corresponding allocation vector can easily found by backtracking relation (3) [25].

Here, the optimization structure for $\alpha(n, m, k)$ depends on the intrinsic property of the adaptation scheme at the receiver's end; that is, the utility (degree of satisfactory) increases when the bandwidth mismatch decreases, and a receiver always subscribes to the best-matching stream. This holds not only with the specific RM function, but also with most other mismatch or fairness measures. Therefore, by using appropriate expressions for $DIFF(m, j)$, the above algorithm can accommodate these utility functions.

4.3 Joint Optimization for Stream Number and Bandwidths (OptNB)

In this scenario, both the number of streams (l_s) and their bandwidth ($r_{s,i}$) are to be optimized. For discrete bandwidth allocation, however, there is an upper bound of l_s, given by $l_s^{\max} = \lfloor\sqrt{1+8N_s}/2 - 1/2\rfloor$. This corresponds to stream bandwidth allocation $(1, 2, 3, \ldots, l_s)$ subject to $\sum_{t=1}^{l_s} t \leq N_s$. Thus, from Lemma 4.1, there also exists an optimal solution to problem OptNB. A *naive* method to find the solution is to try l_s from 1 to $l_s^{\max}$, and call the algorithm for OptFN for each l_s. The complexity of this exhaustive search is $O(N_s^{3\frac{1}{2}})$. Nevertheless, a more efficient algorithm can be designed as follows:

Let $\beta(n, m) = \min\limits_{r_{s,l_s}=m, \sum_{k=1}^{l_s} r_{s,k}=n} \sum_{t=1}^{T_s} M_{s,t}RM[t, \phi(t, \vec{R}_s)]$, that is, the minimum ERM when the session bandwidth is n, and the bandwidth of stream l_s is m. Since there is no constraint on l_s, the solution to problem OptNB is simply given by $\min\limits_{1 \leq n \leq N_s, 1 \leq m \leq T_s} \beta(n, m)$. We also have a recurrence relation for $\beta(n, m)$, as follows,

$$\beta(n,m) = \begin{cases} \sum\limits_{t=0}^{m-1} M_{s,t}RM(t,0) + \sum\limits_{t=m}^{T_s} M_{s,t}RM(t,m), \quad if\, m = n > 0 \\ \min\limits_{1 \le j < m} \left\{ \beta(n-m,j) - DIFF(m,j) \right\}, \\ \qquad\qquad\qquad\qquad if\, m \le n \le N_s, 1 \le m \le min\{n, T_s\} \\ \infty, \quad otherwise \end{cases} \tag{4}$$

The explanation to this relation is similar to that to (3). The first equation represents the same boundary as in (3), except that the existence of only one stream is not explicitly stated, but implied by $m = n$. In the second equation of (4), since there is no limit of the number of streams, the index of k is omitted. As a result, calculating $\beta(n,m)$ and obtaining the optimal allocation for OptNB needs only $O(N_s^3)$ time, which is much lower than the exhaustive search algorithm and, interestingly, even lower than OptFN.

4.4 Remarks on the Lower Bound of ERM

Intuitively, ERM can be reduced if more session bandwidth is allocated. However, our observation from the solutions for OptFN (fixed number of streams) is that ERM cannot be further reduced after a certain N_s. A trivial bound is $N_s^0 = \sum_{k=1}^{K}(T_s - k + 1) = K(2T_s - K + 1)/2$, because the allocation of the highest total bandwidth is $(T_s - K + 1, T_s - K + 2, \ldots, T_s)$. The tight bound N_s^{bound} can thus be represented by $\min\{n' : \min\limits_{1 \le m \le T_s} \alpha(n', m, K) = \min\limits_{1 \le m \le T_s} \alpha(N_s^0, m, K)$. An effective method to find this N_s^{bound} is based on the optimal bandwidth allocation for cumulative layered multicast with K layers: If there is no constraint of session bandwidth, we can build a mapping from the cumulative layer bandwidth to the stream bandwidth: $r_{s,k} = \sum_{i=1}^{k} r'_{s,i}$, where $r'_{s,i}$ is the bandwidth of layer i [4]. This makes the two schemes achieve the same session ERM. Specifically, when the layer bandwidth allocation is optimal, this mapping gives the optimal stream bandwidth allocation with no bandwidth constraint. As a result, N_s^{bound} is given by $\sum_{k=1}^{K}\sum_{i=1}^{k} r'_{s,i}$, which can be calculated in time $O(T_s^2 K)$ [12]. This mapping is illustrated in Figure 1.

In a bandwidth-limited case, however, the optimization structure for stream replication is different from that for cumulative layering, and the choice of K becomes critical. As will be shown in our numerical results, the use of a flexible number of streams can remarkably reduce ERM for session bandwidths beyond N_s^{bound}.

5 Inter-Session Bandwidth Allocation

We now consider the issue of bandwidth allocation for different sessions, or *inter-session bandwidth allocation*. Our basic goal is to achieve a fair yet efficient allocation. There are various notions of fairness for session bandwidth allocation, especially in a broadcast or multicast scenario [19, 20]. In a centralized network, like a cellular network, fairness is also related to administrative issues as well as charging policies. Hence, rather than define a new inter-session allocation framework and claim its fairness, we try to identify the unique properties of our application, and enhance the system performance within existing frameworks. Specifically, we observe that the session ERM for our optimal replication algorithm is a stepwise function of session bandwidth, and the ERM of OptFN even becomes flat after a certain session bandwidth (see Figure 1 and the numerical results in Section 7 for illustration). Thus, some bandwidth allocated to a session can be wasted; it would be beneficial to distribute such bandwidth to other sessions.

We refer to such an enhancement as *ERM-Aware Allocation* (EAA). For illustration, we use an *Equal Share based Allocation* (ESA) as the basic allocation framework, which allocates the bandwidth uniformly among the sessions. Denote $mERM(s, n)$ as the minimum ERM when bandwidth n is allocated to session s, and $\tau(s, d)$ as$[mERM(s, n_s) - mERM(s, n_s + d)]/d$, *i.e.*, the reduction of $mERM$ per bandwidth unit when d units are added to session s. The following heuristic algorithm provides a simple EAA implementation:

```
1 :  $N_s \leftarrow \lfloor C/P \rfloor$, $s = 1, 2, \ldots, P$;
2 :  While $mERM(s, N_s) = mERM(s, N_s - 1)$ do $N_s \leftarrow N_s - 1$;
     $s = 1, 2, \ldots, P$;
3 :  $\sigma \leftarrow C - \sum_{s=1}^{P} N_s$;
4 :  Repeat
5 :      $\varphi \leftarrow \arg_d \max_{s=1,2,\ldots,P, \text{ and } d \le \sigma} \tau(s, d)$,
         $s' \leftarrow \arg_s \max_{s=1,2,\ldots,P, \text{ and } d \le \sigma} \tau(s, d)$;
6 :      if $\tau(s', \varphi) > 0$, then $N_{s'} \leftarrow N_{s'} + \varphi$, $\sigma \leftarrow \sigma - \varphi$;
7 :  Until $\tau(s', \varphi) = 0$ or $\sigma = 0$.
```

Given an equal allocation (Step 1), the above algorithm first reduces

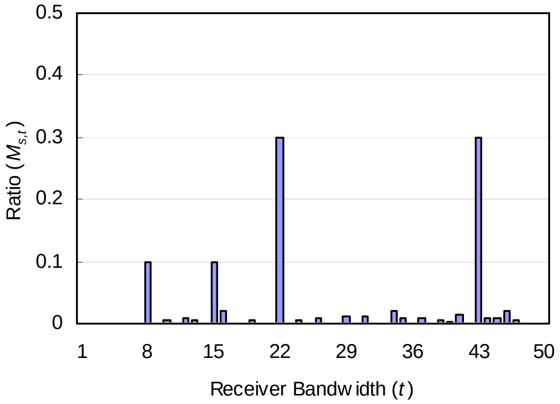


Figure 1: In this simple example, most receivers' bandwidths are distributed at four points: 8 (10%), 15 (10%), 22 (30%), and 43 (30%). Assume the number of streams is fixed to 3. In the case of cumulative layered multicasting, the cumulative layer bandwidth allocation (8, 22, 43) minimizes the expected mismatch, as long as the session bandwidth is no less than 43. Note that this session bandwidth is even lower than the maximum receiver bandwidth (about 47). If there is no constraint of session bandwidth, this is also gives the optimal stream bandwidth allocation for simulcasting, but the total bandwidth of the streams now is 73 (=8+22+43). The mismatch cannot be further reduced by adjusting (either increasing or decreasing) the bandwidth of any stream, even if there is extra session bandwidth. On the other hand, for a limited session bandwidth, say 70 (< 73), the allocation (8,15,43) becomes the optimal choice for simulcasting. Actually, this is the optimal allocation for any session bandwidth between 66 (=8+15+43) and 72. In other words, the session ERM for OptFN is a stepwise function of the session bandwidth.

the bandwidth of each session as much as possible without increasing the session's current $mERM$ (Step 2). It will then re-allocate the unused bandwidth to the sessions (Steps 3 to 7); each time a session that has the maximum ERM reduction per bandwidth unit is selected (Step 5).

Assume N'_s, $s = 1, 2, \ldots, P$, are the session bandwidths allocated by ESA. It can be easily proved that $\sum_{s=1}^{P} mERM(s, N_s) \leq \sum_{s=1}^{P} mERM(s, N'_s)$ and $mERM(s, N_s) \leq mERM(s, N'_s)$, $s = 1, 2, \ldots, P$. Hence, EAA can reduce not only the average ERM for all the sessions, but also the ERM of each session. In the worst case, EAA yields the same ERM as ESA.

6 Implementation Issues and Computation Overhead

The optimal stream replication algorithms can be implemented offline, assuming the distributions of the receivers' expectations are available and stationary for a large population [23]. On the other hand, an online implementation, or dynamic simulcasting, would achieve higher bandwidth utilization in a non-static environments. Nevertheless, there is a potential issue of computational overheads associated with online adaptation. The choice of a video codec that is compatible to such a dynamic allocation algorithm is also very important. In this section, we discuss these practical issues.

6.1 Computation Overhead and Algorithm Optimization

We have applied a set of techniques to speed up the optimization algorithms. First, when calculating $mERM(s, n)$ for $n = N_s$, the values of $\alpha(n, m, k)$ for $n < N_s$ are all available in intermediate stages; we can thus obtain $mERM(s, n)$for any $n \leq N_s$ in time $O(N_s^3 K)$ only. For EAA allocation, if necessary we can incrementally calculate the session ERM for each $n > N_s$ in time $O(N^2 K)$. Similar techniques can be used in OptNB as well. Second, our algorithms are based on the bandwidth distribution of all the receivers, while not the bandwidth of an individual receiver. In a typical multicast environment, the bandwidths of the receivers usually follow some clustered distribution. For instance, they often use standard access technologies, such as a 128 Kbps ISDN line, a 1.5 Mbps ADSL, or a 10 Mbps shared Ethernet; or those in a local region may share upstream links, and hence experience the same bottleneck bandwidth. Although individual receivers may experience short-term bandwidth fluctuations, or dynamically join or leave a session

Setting	Execution Time (ms)					
(C, P, T_s)	OptFN*	EAA	Total	OptNB*	EAA	Total
(256,10,15)	1.5	0.5	15.5	1.1	0.5	11.6
(1024,20,30)	2.6	1.2	53.2	1.9	1.1	39.1
(1200,15,50)	4.2	1.4	64.4	3.9	1.3	59.8

* Execution time for one session.

Table 1: Execution times for the allocation algorithms

provided that the system support these operations (*e.g.*, in the IP multicast environment), such clustered distributions could persist for a relatively long time. As such, the adaptation algorithm can be executed infrequently, only when the distribution has substantially changed. This can be identified by statistical methods, such as the Pearson's χ^2-test or the Kolmogorov-Smirnov (K-S) test [24]. Finally, the curve of the minimum ERM for a particular session is independent of the receiver status of other sessions. When the status of a session changes, only its own $mERM$ curve need to be re-calculated, together with an execution of inter-session allocation.

We have implemented the optimization algorithms using C++ on an Intel Pentium III 900MHz PC with 256MB memory. The execution times of different settings are listed in Table 1. It can be seen that the computation overhead is not significant; the results can be obtained in a relatively short time that is suitable for real-time adaptation.

6.2 Video Stream Replicating

In practice, replicated video streams can be obtained through rate control at the source coding stage, or through media scaling mechanisms, *e.g.*, *transcoding*, which converts an existing video stream to a stream with a different bit-rate or format [16]. Our optimization algorithm does not specify any particular video coding scheme in the application layer. It can cooperate with different media scaling schemes. Nevertheless, a scheme with a wide dynamic range, fast responsiveness, and fine granularity in terms of rate adaptation is of particular interest. This has been demonstrated by advanced video transcoders using motion vector replication or frequency-domain manipulation [16]. Furthermore, it is worth pointing out that the emerging MPEG-7 standard defines *transcoding hints*, a set of meta-data that effectively help the transcoding procedure meet the specific speed or bandwidth requirements yet preserve high video quality [17].

7 Performance Evaluation

In this section, we evaluate the performance of dynamic simulcasting, and try to identify the key factors that influence the performance. For the sake of comparison, we also implement a non-optimal scheme that is often cited in the literature: the exponential (also called multiplicative [5] allocation with a fixed number of streams (ExpFN) [9, 12]. In ExpFN, the stream bandwidths form a geometric progression, *i.e.*, $r_i = \lfloor \rho^{i-1} r_1 \rfloor$, $i = 2, 3, \ldots, K$. Such exponential setting can cover a broad dynamic range with a limited number streams, so as to meet the diverse bandwidth demands from receivers. To achieve a fair comparison, we assume that both the minimum and the maximum receiver bandwidths are known, and r_1 is set to the minimum. Given constraints $\sum_{i=1}^{K} r_i \leq N_s$ and $r_K \leq \eta T_s$, the spanning factor ρ can be simply determined by a bisection search. Here, $\eta < 1$ is a damping factor, which ensures a reasonable portion of receivers can subscribe to stream K. Without this factor, the bandwidth of stream K will be set to T_s, and thus only the receivers of the maximum bandwidth can subscribe to stream K. Such receivers are very few, possibly only one, resulting in a waste for the high bandwidth setting of stream K. In our experiments, η is set to 0.85, the same as that in [10].

7.1 Numerical Results

7.1.1 Intra-session Bandwidth Distribution

We first study the performance of the stream replication schemes in a single session. To reflect the heterogeneous nature of the receivers, we model the expected bandwidths of the receivers in the session by a multi-modal distribution. Specifically, we observe that most access and video decoding components on the receiver's side follow some specific standards, yet some use customized software or hardware [1]. Therefore, a mixture Gaussian model [15] is used to represent the bandwidth distribution for a large population. This model consists of w clusters, each following a Gaussian distribution. In our simulation, the minimum and maximum receiver bandwidths are 2 and 50, respectively. Assume each bandwidth unit is 32 Kbps, this range covers the bandwidths of many available network access techniques. The standard deviation of a cluster is set to 10% of the cluster mean. Thus most bandwidth differences are within ±10%, yet a few reach about ±40% or more, which reflects the flexibility in device design. By using different w, this model can be viewed as a generalization of those models used in

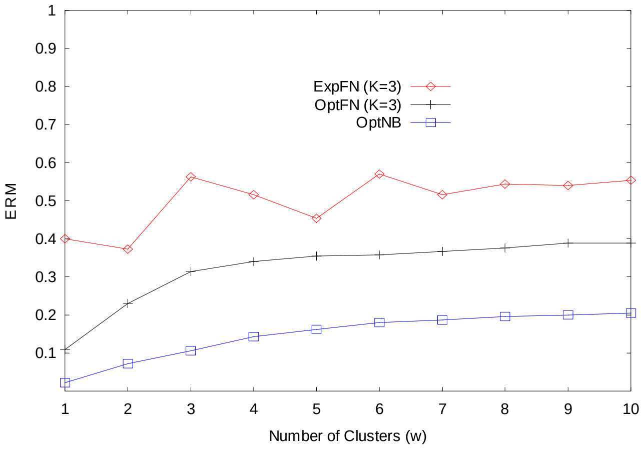


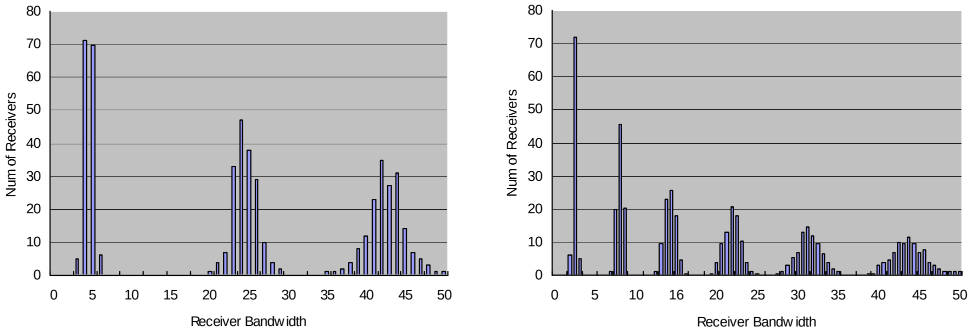
Figure 2: Session ERM for different bandwidth distributions. $N_s = 75$, K=3 for both OptFN and ExpFN. Note that an ERM reflects the bandwidth mismatch, which is the lower the better.

previous studies [5, 7].

We assume the session has 500 receivers and draw 500 samples from the model to obtain a bandwidth distribution instance of the receivers. All the results presented are averages over ten instances.

In Figure 2, we show the impact of the bandwidth distribution for the single session. It can be seen that the session ERMs of the optimal replication schemes for dynamic simulcasting (OptFN and OptNB) are relatively small when there are only a small number of clusters (w). With the increase of w, their session ERMs also become larger. Note that w can be viewed as the *degree of heterogeneity* of the receivers: the higher the value of w, the more heterogeneous the session is, as the receivers' bandwidths are distributed in more clusters. Consequently, it is more difficult for the streams to match the demands of the receivers, especially when the number of streams is predetermined. On the contrary, since ExpFN is actually not aware of the distribution, its ERM does not evidently increase with w, and, for all w, the performance of this non-adaptive is much lower than the two optimal schemes.

In the following studies, we use two distributions of w=3 and w=6 as representatives (see Figure 3 for their instances).



(a)

(b)

Figure 3: Bandwidth distribution of the receivers in a session. The total number of receivers is 500, uniformly distributed in the clusters, *i.e.*, each cluster has around $500/w$ reeivers. (a) w=3; (b) w=6.

7.1.2 Effect of Session Bandwidth

In this set of experiments, we study the effect of the bandwidth allocated to a session. Figure 4 shows the session ERM as a function of the session bandwidth under different settings. It is clear that both optimal allocation schemes significantly outperform ExpFN; at a medium to high bandwidth, the improvement of ERM is often over 0.2. For example, in Figure 4 (a), the ERM of OptNB is reduced to 0.15 with a medium session bandwidth (75). The ERM of ExpFN, however, remains higher than 0.5, which translates into an average bandwidth utility under 50%.

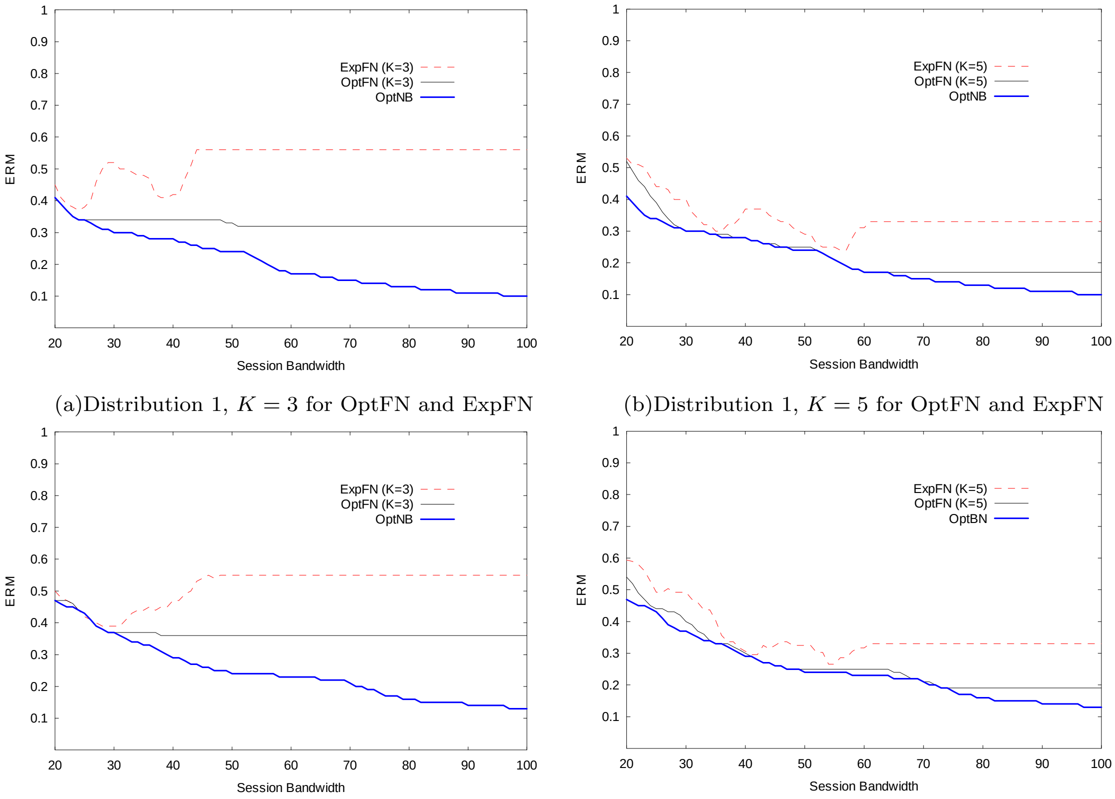
An interesting phenomenon of ExpFN is that its performance is not necessarily improved by allocating more bandwidth to the session. In Figure 4(b), though the ERM of ExpFN at the session bandwidth of 60 is lower than that at 40, it is noticeably higher than that at only 50. In Figures 4(a) and (c), the performance is even the worst for all the bandwidths greater than 50. This is because the receivers' bandwidth distribution is not taken into account in this allocation scheme, and hence some unreasonable stream bandwidth settings could occur. To the contrary, as shown in Figure 4, the ERM of OptFN or OptNB is non-increasing with the increase of the session bandwidth. This can also be formally proved from recurrence relations (3) and (4). Considering the performance gap and the unpredictable behavior of the non-optimal scheme, we believe that the optimal replication effectively complements the expansion of session bandwidth.

The number of streams also influences the performance for the replication schemes. In Figure 4(b), the ERM of OptFN is close to that of the joint optimization scheme (OptBN) for bandwidths between 30 and 65, which basically means that 5-stream is the best choice in this interval. However, with lower or higher session bandwidths, it is no longer optimal, and the gaps are about 0.5 or more at some points. In Figure 4 (a), OptFN is close to OptNB only for session bandwidths between 20 and 25, and the gaps are usually more than 0.15 for other bandwidths. Moreover, the ERM of OptFN becomes flat for bandwidth over 52, because the optimal allocation for this 3-stream setting has been reached (As illustrated in Figure 1, this allocation corresponds to the optimal allocation for cumulative layered multicast). On the contrary, OptNB can generate more streams to use the extra bandwidth and hence further reduce the bandwidth mismatch. As a result, with large session bandwidths (> 95), the ERM of OptNB is reduced to less than 0.1, which is much lower than that of OptFN. Similar observations can be made from Figures 4 (c) and (d) as well.

7.1.3 Impact of the Number of Streams

To further study the impact of the number of streams (K) used in OptFN and ExpFN, we let K vary from 1 to 9. Figure 5 shows the results when bandwidths 55 and 75 are allocated to the session, respectively. It can be seen that the variations of the ERMs with different numbers of streams are as large as 0.3 (excluding the single stream case, K=1) for both OptFN and ExpFN. For OptFN, obviously there is an optimal setting for K, at which the session ERM is minimized. Intuitively speaking, if K is very small, the receivers' choice is limited and the adaptation is not flexible; an extreme case is K=1, the single-rate transmission, where all the receivers have to subscribe to this single stream. On the other hand, if K is large, the redundancy of replication will contradict the improvements given by the flexible choices.

The exact optimal setting of K can be found by the algorithm for OptBN. Note that this optimal setting in Figure 5(a) (distribution 1) is different from that in Figure 5 (b) (distribution 2) for the same session bandwidth. Moreover, as shown in Figure 4, when the session bandwidths are different, the optimal settings are also different, even for the same distribution. In other words, there is no a universal choice for the optimal number of streams, suggesting a dynamic setting.



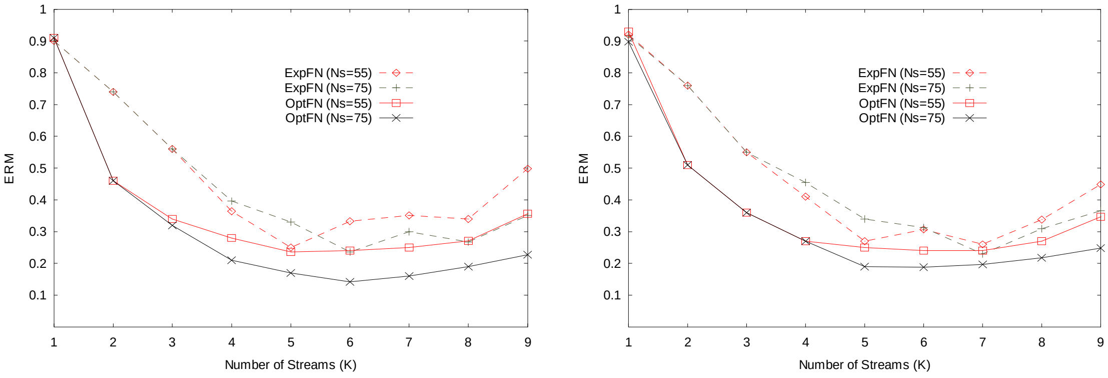
(a)Distribution 1, $K = 3$ for OptFN and ExpFN

(b)Distribution 1, $K = 5$ for OptFN and ExpFN

(c)Distribution 2, K=3 for OptFN and ExpFN

(d)Distribution 2, K=5 for OptFN and ExpFN

Figure 4: ERM as a function of session bandwidth for different allocation schemes.



(a)

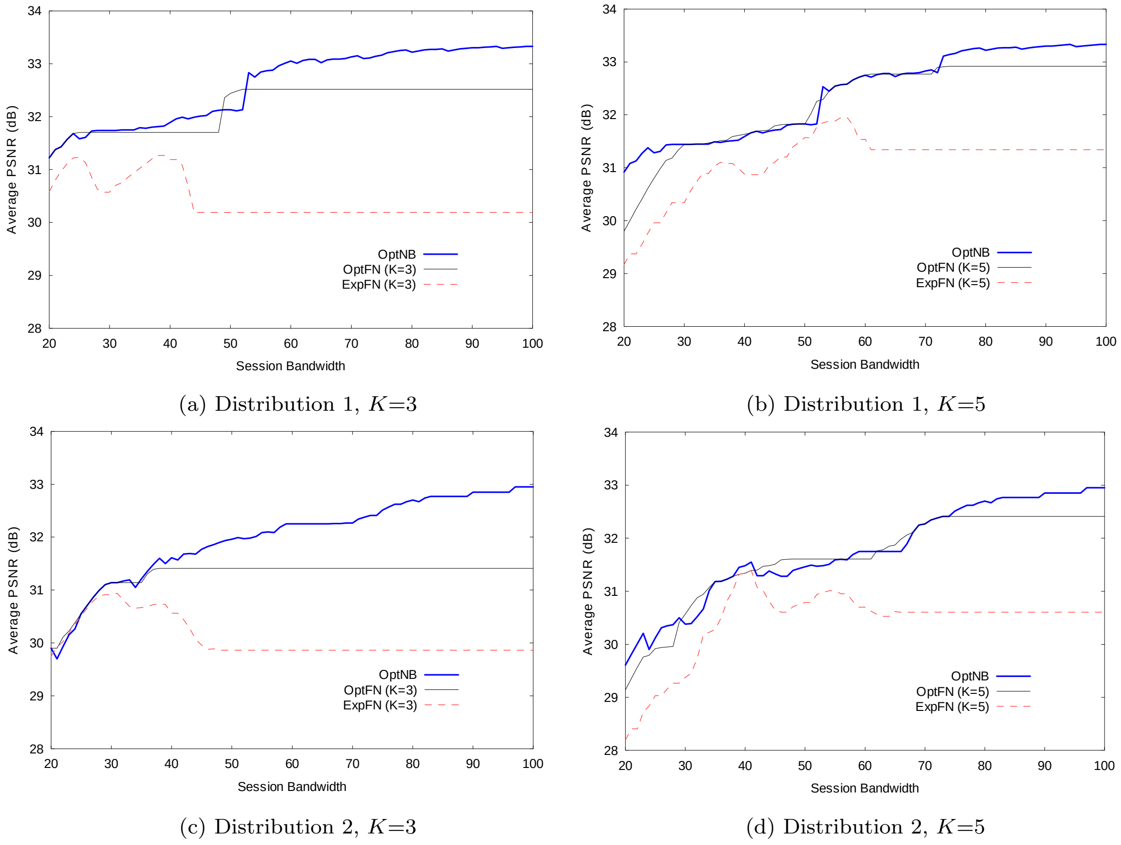
(b)

Figure 5: ERM as a function of the number of streams for OptFN and ExpFN. (a) Distribution 1. (b) Distribution 2.

For the ExpFN allocation, the ERM is also reduced by using a proper number of streams, but remains much higher than OptFN for most settings.

7.1.4 Perceived Video Quality

Since our target application is video distribution, we also examine the video quality achieved by different replication schemes. We use the standard MPEG-4 video encoder with TM-5 rate control to generate replicated video streams at different rates. The average video quality of all the receivers for a standard test sequence ”Foreman (CIF)” is presented in Figure 6, where the quality is measured by the Peak Signal-to-Noise Ratio (PSNR) of the Y channel [1]. It can be seen that the optimal replication algorithms for dynamic simulcasting generally improve the perceptual video quality. With medium and high session bandwidths, the gaps between OptNB and OptFN are about 0.5 to 1 dB, and the gaps between OptNB and ExpFN are usually larger than 2 dB; both are noticeable from the video coding point of view. This is consistent with our observations on the relationship between ERM and session bandwidth in Section 7. Since PSNR often has a linear relationship with the transmission bandwidth, Figure 6 is not simply an inverse and rescaled version of Figure 4. In particular, for OptNB, although its ERM is non-increasing with session bandwidth, we find that PSNR is not necessarily non-decreasing; see for example, Figure 6(d), session bandwidth 20 through 50.



(a) Distribution 1, K=3

(b) Distribution 1, K=5

(c) Distribution 2, K=3

(d) Distribution 2, K=5

Figure 6: Average PSNR as a function of session bandwidth for different allocation schemes. Each point is calculated at a certain session bandwidth, and those for the same allocation scheme are then connected with a line.

7.1.5 Effect of Inter-Session Bandwidth Allocation

Finally, we study the effects of inter-session bandwidth allocation. We assume that the demand probabilities for different video programs follow a Zipf distribution with a skew factor of 0.271, as suggested by movie rental statistics [22, 23]. The number of clusters for each session is uniformly distributed in between 2 and 7. In the experiments, we assume that there are 2500 receivers belonging to 15 sessions, and draw 2500 samples from the above model to obtain a receivers' status distribution for the whole system.

The performances of different combinations of the intra- and inter-session allocation schemes are compared in Figure 7. The results are consistent with our previous observations in intra-session allocation; that is, OptNB generally outperforms OptFN if the same inter-session allocation scheme is employed. However, the impact of different inter-session allocation schemes is also non-negligible. It can be seen that, the ERM-Aware Allocation (EAA) consistently outperforms the Equal Share Allocation (ESA), both with OptNB and with OptFN. At low or medium bandwidths, the performance gaps can be larger than 0.1. Although the contribution of EAA is not so significant as that of intra-session optimization, in view of its relatively low computation overhead, we believe that it is still worth consideration in practice. More interestingly, for bandwidth around 500, the ERM of OptFN plus EAA is quite close to that of OptNB plus EAA, and is better than that of OptNB plus ESA. This is because the preset number of streams (5 in our study) is likely to be the optimal choice for medium bandwidths (see Figure 4), and the choice of inter-session allocation thus has more influences on the ERM. To conclude, EAA is particularly suitable for the cases where the stream number is fixed and the bandwidth resource is relatively scarce.

7.2 Simulation Results

In this set of experiments, we simulate the simulcasting algorithms using the LBNL network simulator *ns-2* [35]. To be compatible with the current Internet where TCP is the dominant traffic, we advocate the TCP-friendly bandwidth adaptation paradigm in this simulation [34]. We stress however that our optimal stream replication algorithms can be used with other adaptation paradigms. In the TCP-friendly paradigm, the expected bandwidth of a video receiver is estimated as the long-term throughput of a TCP connection as if the connection is running over the same path. A receiver thus can control the subscription bandwidth to avoid starving the background

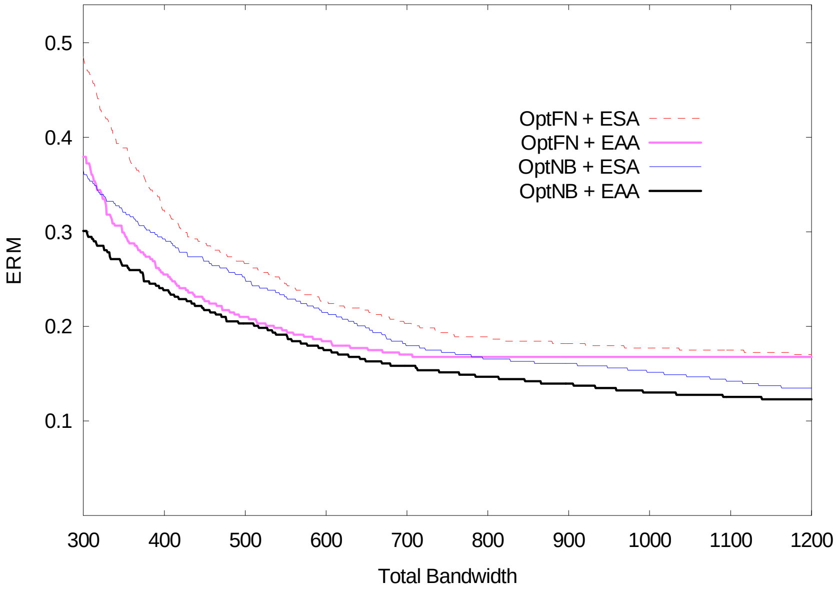


Figure 7: ERM as a function of the total bandwidth for all the sessions for different combinations of the intra-session and inter-session allocation schemes.

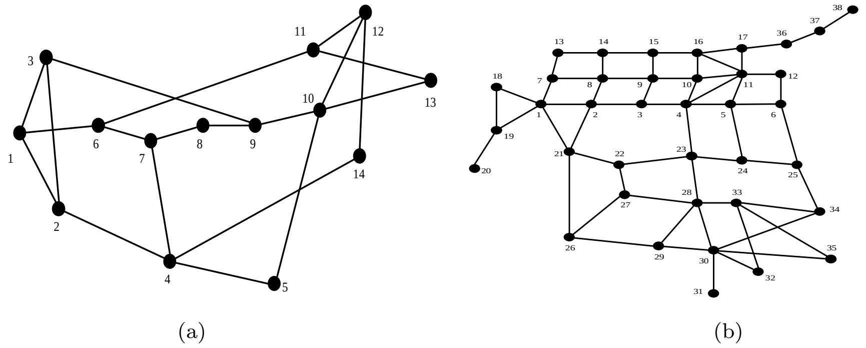


Figure 8: Simulation topologies. (a) 14-node NSFnet network; (b) 38-node CTnet network.

TCP flows and, meanwhile, try to achieve a fair share with them. We assume that the sender performs re-allocation every 10 seconds, which is also the adaptation period for each receiver. In addition, when severe congestion occurs (over 10% packet loss in our simulation), a receiver can instantly switches to a stream of lower rate.

Regarding the estimation of equivalent TCP throughput, there have been many efforts in the literature [34]; a general conclusion is that an estimation model relies on round-trip time (RTT), packet size, and loss event rate. The latter two can be easily estimated on the receiver's side; the estimation of RTT, however, involves a feedback loop between the sender and the receiver. Since the topologies for our simulation is relatively small, we assume that each receiver reports its expected bandwidth to the sender every two seconds. Such report also serves a request for RTT estimation. In a large multicast network, to avoid the well-known feedback implosion problem, some feedback mergers can also be deployed inside the network [36].

Two typical network topologies are used in our simulation: the NSFnet and the China Telecom Network (CTnet), as depicted in Figure 8. In each topology, a node represents a FIFO drop-tail router with a queue size of 25 packets, and each edge represents a link of bandwidth 2 Mbps. Given a topology, we use the following method to produce an instance for simulation:

Placement of end-nodes: A video server is attached to a randomly selected node, and an average of 5 video receivers are attached to each of the remaining nodes.

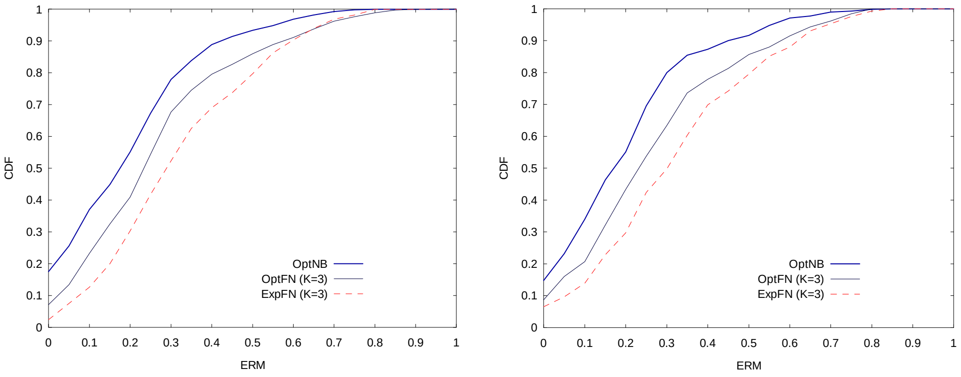
Cross-traffic: TCP Reno connections are randomly placed between node pairs (except for the node attached the server), such that there is on average

3 TCP connections running over a link, while the minimum and maximum numbers are 0 and 9, respectively. As a result, the expected available bandwidth for a link is on average 500 Kbps; yet the minimum and maximum are 200 Kbps and 2 Mbps, respectively. The packet size is 500 bytes for both TCP and video traffic.

To mitigate the effect of randomness, we generated 10 instances for each topology using the above method. All instances were simulated for 1000 seconds, which is long enough for observing steady-state behaviors. We sample an RM (Relative Mismatch) value for each receiver per five seconds. Figure 9 shows the cumulative distributions of all sampled RM values for the three replication algorithms. We can see that for our optimal stream replication algorithms, more receivers have an RM value close to 0, the optimal value, and the probability of the RMs greater than 0.30 is relatively small (the probability is less than 0.2 for OptNB, and 0.25 for OptFN). On the contrary, for ExpNB, about 30% of the receivers would experience an RM that is even higher than 0.5, *i.e.*, a bandwidth mismatch more than half of one's expected bandwidth. Consequently, the ERM values for the NSFnet are 0.21 (OptNB), 0.27 (OptFN), and 0.34 (ExpFN), and that for the CT-net are 0.24 (OptNB), 0.29 (OptFN), and 0.37 (ExpFN). Such performance gaps are consistent with our observations in the numerical studies. They also imply that, under the TCP-friendly adaptation paradigm, a simulcast system employing OptFN is "fairer" in bandwidth sharing than that employing ExpFN, and the use of OptNB can further improve fairness.

8 Conclusions and Future Work

This chapter presented a formal study on the problem of adaptive stream replication for video simulcasting. Our main objective is to minimize the expected mismatch for all the receivers in a session. We formulated the optimization problems for stream bandwidth allocations, both with a flexible number of streams and with a fixed number of streams. We then presented efficient solutions, which leading to the design of a novel simulcasting paradigm, called *dynamic simulcasting*. We also discussed some of the key implementation issues for dynamic simulcasting, including the computation overhead and the choice of video encoders. Its performance was studied under a variety of configurations. The results conclusively demonstrated that the optimal replication schemes in dynamic simulcasting can significantly improve the overall system performance in terms of both bandwidth



(a)

(b)

Figure 9: Cumulative distribution function (CDF) of the RM samples. (a) 14-node NSFnet; (b) 38-node CTnet.

utilization and perceptual video quality.

The algorithms we discussed and analyzed in this chapter involve end-to-end adaptation only, which is actually independent of the underlying physical medium. Therefore, it is applicable to diverse broadcast- or multicast-capable networks. We are currently developing a comprehensive video simulcasting system using the optimal replication schemes. This is non-trivial from both theoretical and practical points of view, as it involves not only the optimization of individual components, but also the integration of the whole system. One of the important issues is stream switching. Given that a video stream generally has a complex syntax and its content is highly dependent, quality drift would occur after switching, and such drift could propagate to many following frames. There have been some preliminary work on seamless switching [32, 33]; however, it remains a difficult undertaking worth further investigation. Other issues include real-time video replication and dynamic stream creation or termination with low overheads. The impacts from channel errors and background traffic as well as the stability of the system are also possible avenues for further research.

References

[1] Y. Wang, J. Ostermann, and Y.-Q. Zhang, *Video Processing and Communications*, (Prentice Hall, September 2001).

[2] H. Schulzrinne, S. Casner, R. Frederick and V. Jacobson, *RTP: A transport protocol for real-time applications*, (RFC 1889, January 1996).

[3] S. Cheung, M. H. Ammar, and X. Li, On the use of destination set grouping to improve fairness in multicast video distribution, in *Proceedings of IEEE INFOCOM'96*, (March 1996) pp. 553-560.

[4] T. Kim and M. H. Ammar, A comparison of layering and stream replication video multicast schemes, in *Proceedings of NOSSDAV'01*, (June 2001).

[5] S. Gorinsky and H. Vin, The utility of feedback in layered multicast congestion control, in *Proceedings of NOSSDAV'01*, (June 2001).

[6] L. Wu, R. Sharma, and B. Smith, Thinstreams: An architecture for multicast layered video, in *Proceedings of NOSSDAV'97*, (May 1997).

[7] T. Jiang, E. W. Zegura, and M. H. Ammar, Inter-receiver fair multicast communication over the Internet, in *Proceedings of NOSSDAV'99*, (June 1999).

[8] SureStream - Delivering superior quality and reliability, *RealNetworks White Paper*, (March 2002) http://www.realnetworks.com/products/servers/wp_surestream.html.

[9] S. McCanne, V. Jacobson, and M. Vetterli, Receiver-driven layered multicast, in *Proceedings of ACM SIGCOMM'96*, (August 1996) pp.117-130.

[10] D. Sisalem and A. Wolisz, MLDA: A TCP-friendly congestion control framework for heterogeneous multicast environments, in *Proceedings of IEEE/IFIP IWQoS'00*, (June 2000).

[11] Y. Yang, M. Kim, and S. Lam, Optimal partitioning of multicast receivers, in *Proceedings of IEEE ICNP'00*, (November 2000).

[12] J. Liu, B. Li, and Y.-Q. Zhang, A hybrid adaptation protocol for TCP-friendly layered multicast and its optimal rate allocation, in *Proceedings of IEEE INFOCOM'02*, (June 2002).

[13] K. Kar, S. Sarkar, and L. Tassiulas, Optimization based rate control for multirate multicast sessions, in *Proceedings of IEEE INFOCOM'01*, (April 2001).

[14] Z. Fei, M. H. Ammar, and E. Zegura, Multicast Server Selection: Problems, Complexity and Solutions, *IEEE Journal on Selected Areas in Communication*, (2002).

[15] D. Titterington, A. Smith, and U. Makov, *Statistical Analysis of Finite Mixture Distributions*, (Wiley Publishers, New York, 1985).

[16] J. Youn, J. Xin, and M.-T. Sun, Fast video transcoding architectures for networked multimedia applications, in *Proeedings of IEEE International Symposium of Circuits and Systems (ISCAS'00)*, (May 2000).

[17] P. Kuhn, T. Suzuki, and A. Vetro, MPEG-7 transcoding hints for reduced complexity and improved quality, in *Proceeding of PacketVideo'01*, (April 2001).

[18] W. Li, Overview of the fine granularity scalability in MPEG-4 video standard, *IEEE Transactions on Circuits and Systems for Video Technology*, (March 2001) vol. 11, no. 3, pp. 301-317.

[19] H. Wang and M. Schwartz, Achieving bounded fairness for multicast and TCP traffic in the Internet, in *Proceedings of ACM SIGCOMM 98*, (September 1998).

[20] A. Legout, J. Nonnenmacher, and E. W. Biersack, Bandwidth allocation policies for unicast and multicast flows, *IEEE/ACM Transactions on Networking*, (August 2001) vol. 9, no. 4.

[21] J. Byers, M. Luby, and M. Mitzenmacher, Fine-grained layered multicast, in *Proceedings of IEEE INFOCOM'01*, (April 2001).

[22] G. Zipf, *Human Behavior and the Principle of Least Effort, Reading*, (MA: Addison-Wesley, 1949).

[23] A. Dan, D. Sitaram, and P. Shahabuddin, Scheduling policies for an on-demand video server with batching, in *Proceedings of ACM Multimedia'94*, (October 1994).

[24] A. Alan, *Introduction to Categorical Data Analysis*, (NY: John Wiley and Sons, 1996).

[25] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, (2nd Edition, MIT Press, Cambridge, MA, 2001).

[26] X. Li and M. H. Ammar, Bandwidth control for replicated-stream multicast video distribution, in *Proceedings of HPDC'96*, (August 1996).

[27] S. Banerjee and B. Bhattacharjee, A Comparative Study of Application Layer Multicast Protocols, *Technical Report*, Univeristy of Maryland, College Park, (2002).

[28] J. D. Lameillieure and S. Pallavicini, A comparative study of simulcast and hierarchical coding, *Technical Report*, (European RACE project, subgroup WG2, February 1996).

[29] P. de Cuetos, D. Saparilla, and K. W. Ross, Adaptive streaming of stored video in a TCP-friendly context: multiple versions or multiple layers, in *Proceedings of Packet Video Workshop*, (April 2001).

[30] F. Hartanto, J. Kangasharju, M. Reisslein, and K. W. Ross, Caching video objects: layers vs versions?, in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME'02)*, (August 2002).

[31] J. Liu, B. Li, and Y.-Q. Zhang, Adaptive Video Multicast over the Internet, *IEEE Multimedia*, (January/February 2003) vol. 10, no. 1, pp. 22-31.

[32] J. Macher and G. Anderson, Multi-program transport stream switching, White Paper, *Thales Broadcast & Multimedia*, available at http://www.broadcastpapers.com/sigdis/ThalesStreamSwitching01.htm.

[33] Y.-K. Chou, L.-C. Jian, and C.-W. Lin, MPEG-4 video streaming with drift-compensated bit-stream switching, in *Proceedings of IEEE Pacific-Rim Conference on Multimedia (PCM'02)*, (December 2002).

[34] S. Floyd and K. Fall, Promoting the use of end-to-end congestion control in the Internet, *IEEE/ACM Transactions on Networking*, (August 1999) vol. 1, no. 4, pp. 458-471.

[35] S. McCanne and S. Floyd, *The LBNL Network Simulator, ns-2*, http://www.isi.edu/nsnam/ns/.

[36] B. Vickers, C. Albuquerque, and T. Suda, Source adaptive multi-layered multicast algorithms for real-time video distribution, *IEEE/ACM Transaction on Networking*, (December 2000) vol. 8, no. 6, pp. 720-733.