ABSTRACT

Recent advance of real-time audio/video transcoding technology enables streaming servers to manage the constrained network-I/O bandwidth available to the Internet at a fine-grained level. In this article, we propose a bandwidth allocation mechanism for differentiated services (DiffServ) on the streaming servers. It aims to deliver high bit rate streams to high priority requests without over-compromising low priority requests. We formulate the bandwidth allocation problem as optimization of a harmonic utility function of the stream quality factors and derive the optimal streaming bit rates for requests of different classes under various server load conditions. We prove that the optimal allocation scheme, referred to harmonic proportional allocation, not only maximizes the system utility function, but also guarantees proportional sharing between classes with different pre-specified differentiation weights. We evaluate the harmonic allocation scheme via extensive simulations and compare it with an absolute DiffServ strategy and a proportional-share strategy tailored from relative DiffServ in networking and operating systems. Simulation results show that the optimal allocation can meet the objective of relative DiffServ in both short and long timescales and greatly enhance the service availability and maintain low queuing delay when the streaming server load is high.

Keywords

Service Differentiation, Harmonic Proportional Allocation

1. INTRODUCTION

To support ubiquitous access to the proliferating continuous media on the WWW, scalable streaming servers must be able to provide different levels of quality of service (QoS) to various client requests. It is because clients are different in their visiting interests, access patterns, service charges, and receiving devices. They can connect to a streaming server using a wide variety of devices, ranging from set-top boxes to PDAs. Their capabilities to receive, process, store and display continuous media (i.e., video and audio) vary greatly. Given the diversity of client devices and their needs, the server has to adapt the content and provide different QoS levels accordingly. From the perspective of the servers, the request arrival rates are non-stationary. It is desirable to provide different levels of QoS to different requests during various access periods. To the end, there is a growing demand for replacing the current same-service-to-all paradigm with a model that treats client requests differently based on the client access patterns and the server resource capacities [9]. Service differentiation aims to provide predictable and controllable per-class QoS levels to requests of different classes. It can also enhance the service availability because a request may be admitted and processed with a negotiated and degraded service quality by a heavily loaded server rather than being simply rejected.

Service differentiation has been an active research topic in the arena of networking since its architecture was formulated in 1998 [9, 12, 20]. Its goal is to define configurable types of packet forwarding so as to provide per-hop differentiated services for large aggregates of network traffic. Network alone is not sufficient to support end-to-end service differentiation. There are recent studies on service differentiation provisioning in Web applications at the server side. In Web applications, response time is a primary QoS metric. Current response-time differentiation strategies mostly rely on admission control and priority-based resource scheduling on individual servers [1, 3, 7, 8, 11, 14, 19, 21, 31], and node partitioning in server clusters [6, 33]. It is known that streaming services have very different characteristics and requirements from conventional Web services. Streaming services are usually constrained by disk-I/O and especially network-I/O bandwidth at the server side [2, 13, 32]. The service quality is measured not only by startup latency, but more importantly by allocated stream bandwidth – streaming bit rate. The response-time oriented strategies are not sufficient for service differentiation on streaming servers.

There were studies on QoS-aware bandwidth management for rich media web services; see [10, 16] for examples. Their focuses were on transformation of the format, color depth and sizes of images as well as rich-texts to make a good tradeoff between user-perceived document quality and transmission time. This paper shares the objective of these work on content adaptation, but focuses on server-side bandwidth management for continuous streaming applications. There are two encoding schemes with streaming media: constant bit rate (CBR) and variable bit rate (VBR). CBR scheme maintains a constant streaming bit rate by varying media quality. It generates predictable media file sizes and simplifies the allocation of server and network resources, as shown in [2, 13, 32]. In contrast, VBR scheme ensures constant media quality by varying streaming bit rate. However, VBR streams exhibit high variations with their resource requirements and may lead to low resource utilization at the server side [5, 15]. In this article, we address the problem of providing differentiated CBR-encoded streaming services and consider video as an example of streaming media.

The idea of performing bandwidth management through adaptation of video frame rate and color depth was demonstrated by early experimental video gateway systems; see [4] for an example. Recent advance of stream transcoding technology makes it possible to dynamically transform a video stream from its original encoding bit rate to degraded ones at a fine-grained level [25, 30]. In other words, current transcoding techniques can adjust the bit rate of a
CBR-encoded video stream on the fly according to the allocated network-I/O bandwidth and make the bandwidth usages controllable by the server’s request scheduler at application level. For example, Roy et al. at HP Labs exploited the multimedia instruction set of the new Itanium processor to save computational power such that one transcoding unit can support many concurrent sessions in real-time [25]. There are other layered video coding techniques that represent videos in a quality enhancement layer hierarchy so as to support different levels of video quality. Compared to such scalable video coding techniques, video transcoding techniques can support differentiated streaming services at a much finer grained level. This paper focuses on the development of bandwidth allocation strategies, which complements the existing content adaptation prototyping work on streaming servers or proxies.

As a major constrained resource class, network-I/O bandwidth is a focus of general purpose resource management systems, as well. Although general resource management models, such as Q-RAM [23] and FARA [24], can be applied to the control of bandwidth allocation on streaming servers, their objectives are mostly on maximizing overall system performance by content adaptation. Adaptability aside, service differentiation schemes also demand QoS predictability and fairness. There are recent studies on proportional share scheduling for resources like CPU, memory, and network bandwidth [22, 29, 28]. However, few derived schemes are able to achieve the best quality of services system wide.

In this article, we propose a QoS-aware network-I/O bandwidth allocation mechanism for service differentiation provisioning on streaming servers. We formulate the bandwidth allocation problem as the optimization of a harmonic utility function of the weighted stream quality factors and derive optimal allocations for request classes with different priorities under various server load conditions. We prove that the optimal allocation not only maximizes the utility function, but also guarantees proportional bandwidth sharing between the classes with respect to their pre-defined differentiation weights. We conduct extensive simulations of the allocation scheme and compare it with an absolute service differentiation strategy and a proportional-share strategy tailored from relative service differentiation in the arenas of networking and operating systems. Simulation results show the optimal allocations achieve the objective of relative service differentiation in both short and long timescales and enhance the service availability to a great extent when the server load is high.

The rest of the article is organized as follows. Section 2 formulates the bandwidth allocation problem for service differentiation. Section 3 presents two allocation schemes: proportional-share scheduling and harmonic proportional scheduling. Section 4 discusses a number of implementation issues of the schemes. Section 5 shows the simulation results of the schemes. Related work is given in Section 6. Section 7 concludes the article.

2. BANDWIDTH ALLOCATION PROBLEM

Figure 1 shows the framework with N client request classes in a streaming cluster with M servers. It assumes a dispatcher to make request classification and admission decisions. The classification can be done based on clients’ profile, fee, and device, etc. For each admitted and classified request by the dispatcher, two key quality metrics are streaming bit rate and delay in its listen queue. A QoS-aware request scheduler needs to determine: (1) how much network-I/O bandwidth (streaming bit rate) should be allocated to the requested stream; and (2) when the stream should be delivered. This paper focuses on the optimization of bandwidth allocation. Its interaction with different request scheduling schemes will also be discussed.

The objective of the network-I/O bandwidth allocation problem is to determine stream bandwidth of each request class in such a way that the overall quality of service is optimized and meanwhile the stream qualities are guaranteed to be proportional to their pre-specified differentiation weights. The weights can be determined by clients’ priorities, receiving devices, payments, etc. Divide the scheduling process into a sequence of short intervals of bandwidth allocation and request scheduling. The allocation decision needs to be carried out in each interval, based on the measured bandwidth release rate and the predicted arrival rate of request classes.

There are two types of service differentiation architectures [9]. One is absolute service differentiation, in which each request class receives an absolute share of resource usages. A primary concern with this scheme is its weak ability of adaptation to fluctuating arrival rates from various clients. In Section V, we will show that without a priori knowledge about the clients’ access patterns, the absolute service differentiation scheme could lead to low resource utilization.

The second one is relative service differentiation. In this scheme, service quality of class i is better or at least no worse than that of class i + 1 for 1 ≤ i ≤ N − 1. The term "or no worse" is necessary, since under heavy-load conditions all request classes will tend to receive the minimum QoS level. Although applications and clients do not get an absolute QoS assurance, this differentiation scheme assures that the class with a higher desired QoS level (referred to as a higher class) will receive relatively better service quality than a lower class. So it is up to the applications and clients to select appropriate QoS levels that best meet their requirements, cost, and constraints. In order for a relative service differentiation scheme to be effective, the scheme must satisfy two basic properties: predictability and controllability. Predictability requires that higher classes should receive better or no worse services than lower classes, independent of the class load distribution and that the differentiation be consistent. Controllability requires that the scheduler contains a number of controllable parameters that are adjustable for the control of quality spacings between classes.

In streaming services, a unique QoS metric of streams is the streaming bit rate. It has lower bound and upper bound. For example, 1 Mbps could be referred to as the lower bound of the streaming bit rate for an MPEG-1 movie. The upper bound is the video initial encoding bit rate because today’s transcoding can only dynamically degrade the streaming bit rate. We argue that an effective relative service differentiation scheme on streaming servers should meet the following additional requirements:

1. Upper and lower bounds: Quality guarantees should be provided for all requests. Admission control is needed to prevent the system from being overloaded.
2. Availability: One goal of offering different levels of QoS on a streaming server is to serve as many requests as possible at acceptable QoS levels to retain the business. If the available network-I/O bandwidth at server side is sufficient to provide the lower bound of QoS level to all requests, rejection rate could be minimized.

3. Fairness: Requests from lower classes should not be over-compromised for requests of higher classes.

We define a channel as the bandwidth unit allocated to a stream. It is the lower bound of the bit rate for a stream. Assume the requests in different classes have independent arrival processes. Therefore, the aggregate traffic of the server is determined by the superposition of the $N$ traffic flows. Let $\lambda_i$ be the arrival rate of requests in class $i$ in an interval. Then, the aggregate arrival rate $\lambda = \sum_{i=1}^{N} \lambda_i$. Let $\mu_i$ be the rate of channel allocation to requests in class $i$. We define a quality factor $q_i$ of the requests in class $i$ as

$$q_i = \frac{\mu_i}{\lambda_i}.$$  

(1)

It represents the quality of request class $i$ in the current bandwidth allocation interval. For example, if the rate of channel allocation to class $i$ is $8$ per time unit and the request arrival rate of class $i$ is $4$ per time unit, the quality for requests in class $i$ in the current interval is $2$. Note that in general, the relationship between the quality of a request and its allocated resources is not necessarily linear. We define the quality factor $q_i$ of a stream request as a linear utility function of its allocated bandwidth because the user-perceived quality of a CBR-encoded video is proportional to its streaming bit rate.

Let $B$ denote the bound of the aggregate channel allocation rate during the current bandwidth allocation interval. It is the number of channels to be released in the current interval, plus the unused ones from previous intervals. We have the resource constraint of

$$\sum_{i=1}^{N} \mu_i \leq B.$$  

(2)

From the system’s perspective, service availability is an essential objective. Suppose the request arrival rate of class $i$ in the current allocation interval is $4$ per time unit and the rate of channel allocation to class $i$ is $3$ per time unit because of heavy system load. Although the calculated quality factor for the class $i$ is $0.75$, due to the existence of lower bound of streaming bit rate (one channel), one request from this class may be rejected and other three requests receive the lower bound of streaming bit rate.

Assume all videos have the same initial encoding bit rate $K$. It represents the upper bound of bandwidth allocation of each stream. We consider the network-I/O bandwidth allocation for service differentiation when $\lambda \leq B < \lambda K$. That is, the total number of available channels is enough to guarantee the minimum QoS level but not enough to support the maximum QoS level for all contending streams. Otherwise, the problem becomes either trivial or infeasible. For example, when $B \geq \lambda K$, the request scheduler can simply give each class the maximum bit rate $K$. When $B < \lambda$, the admission control must do rejection. Hence, for service availability, we have an additional constraint:

$$1 \leq q_i \leq K.$$  

(3)

3. HARMONIC PROPORTIONAL SHARE ALLOCATION ALGORITHM

In this section, we present a harmonic proportional share allocation scheme that not only optimizes an overall system utility function, but also ensures proportional bandwidth sharing between request classes. Consider $N$ contending request classes with quality differentiation weights $\delta_1, \delta_2, \ldots, \delta_N$. Since relative service differentiation requires class $i$ would receive better or no worse services than class $i+1$, without loss of generality, weights $\delta_i$ are sorted in a non-increasing order as $\delta_1 \geq \delta_2 \geq \cdots \geq \delta_N$.

### 3.1 Proportional Share Allocation

We first give an application-level proportional-share allocation. It borrows its idea from proportional-share scheduling in the arenas of networking and operating systems. At the network level, Dovrolis et al. proposed a proportional model to ensure the per-hop queueing delay of the packets in different classes to be proportional to their pre-defined differentiation weights [12]. At the OS level, there has also been a renewal of interest in proportional-share scheduling [22, 28, 29]. For example, Waldspurger and Weihl proposed lottery scheduling for fair share resource management for CPU utilization [29]. Lu et al. designed control-theoretical approaches for cache space allocation in proxies to provide proportional hit rates to different request classes [22].

Our proportional-share bandwidth allocation assigns quality factors to request classes in proportion to their quality differentiation weights $\delta_i$. Recall that each request needs to maintain the lower bound of quality factor. For service availability, in each bandwidth allocation interval, the proportional-share scheme states that for any two classes $i$ and $j$, $1 \leq i, j \leq N$,

$$\frac{q_i - 1}{q_j - 1} = \frac{\delta_i}{\delta_j},$$  

(4)

subject to constraints of (2) and (3).

According to constraint (2), the objective of (4) leads to a proportional channel allocation rate

$$\mu_i^* = \lambda_i + \frac{(B - \lambda)\lambda_i \delta_i}{\sum_{i=1}^{N} \lambda_i \delta_i}.$$  

(5)

Correspondingly, the proportional quality factor of class $i$ is calculated as:

$$q_i^* = 1 + \frac{(B - \lambda)\delta_i}{\sum_{i=1}^{N} \delta_i \delta_i}.$$  

(6)

The quality factor of (6) reveals that the proportional-share allocation scheme generates consistent and predictable schedules for requests of different classes on streaming servers. The classes with higher differentiation weights $\delta_i$ receive better or no worse services than the classes with lower $\delta_i$, independent of variations of the class loads. The quality factor of each request class $i$ is controlled by its own channel allocation rate $\mu_i$.

### 3.2 Harmonic Proportional Allocation

Note that the proportional-share allocation scheme that aims to control the inter-class quality spacings does not necessarily yield best overall QoS. To optimize the overall QoS and meanwhile ensuring quality spacings between the classes, we define a weighted harmonic function of the quality factors of all the classes, which ensures the lower bound of streaming bit rate, as the optimization function. Specifically, we formulate the bandwidth allocation for service differentiation as the following optimization problem:

$$\text{Minimize} \sum_{i=1}^{N} \frac{\delta_i}{q_i - 1}$$  

(7)

Subject to constraints (2) and (3).
The minimization of the harmonic objective function \( \hat{q} \) requires that higher weighted stream classes be allocated more bandwidth but this biased allocation should not-compromise the share of lower weighted classes. Note that when \( \lambda = B \), the quality factor \( q_i \) of each class \( i \) is equal to its lower bound and there is no need for optimization.

The optimization above is essentially a continuous convex separable resource allocation problem. According to theories with the general resource allocation problems \( 18 \), its optimal solution occurs only if the first order derivatives of each component function of \( \hat{q} \) over variables \( \mu_1, \mu_2, \ldots, \mu_N \) are equal. Specifically, the optimal solution of \( \hat{q} \) occurs when

\[
\frac{\delta_i \lambda_i}{(\mu_i - \hat{\lambda}_i)^2} = \frac{\delta_j \lambda_j}{(\mu_j - \hat{\lambda}_j)^2},
\]

for any classes \( i \) and \( j \), \( 1 \leq i, j \leq N \). Combining with the constraint \( \omega \), the set of equations \( \hat{q} \) leads to the optimal allocation

\[
\mu_i^* = \lambda_i + (B - \lambda) \frac{\lambda_i^2}{\sum_{i=1}^{N} \lambda_i^2},
\]

where \( \hat{\lambda}_i \) is the normalized request arrival rates and \( \hat{\lambda}_i = \delta_i \lambda_i \). As a result, the optimal quality factor of class \( i \) is

\[
\hat{q}_i^* = 1 + (B - \lambda) \frac{\lambda_i^2}{\lambda_i \sum_{i=1}^{N} \lambda_i^2},
\]

To show the implications of the derived bandwidth allocation scheme on system behavior, we give the following basic properties regarding the controllability and dynamics due to the allocation scheme:

1. If the differentiation parameter of a class increases, the quality factor of all other classes decreases, while the quality factor of that class increases.
2. The quality factor of a class \( i \) decreases with the increase of arrival rate of each class \( j \).
3. Increasing the load of a higher class causes a larger decrease in the quality factor of a class than increasing the load of a lower class.
4. Suppose that a fraction of the class \( i \) load shifts to class \( j \), while the aggregate load remains the same. If \( i > j \), the quality factor of class \( j \) increases, while that of other classes decreases. If \( i < j \), the quality factor of class \( j \) decreases, while that of other classes increases.

In addition, we can show that the optimal allocation scheme has the property of fairness. That is,

**Theorem 3.1.** The optimal allocation of \( \hat{q} \) guarantees a proportional share distribution of excess bandwidth over the minimum requirements between streams with different differentiation weights.

**Proof.** Define \( \hat{\mu}_i \) as the allocation of excess bandwidth to class \( i \) over the minimum requirement \( \lambda_i \). According to \( \hat{q} \), we have

\[
\hat{\mu}_i = \mu_i^* - \lambda_i = (B - \lambda) \frac{\lambda_i^2}{\sum_{i=1}^{N} \lambda_i^2}.
\]

The allocation of \( \hat{\mu}_i \) yields an increment of quality factor, \( \hat{q}_i = \hat{\mu}_i / \lambda_i \), to the minimum quality factor of one. That is, \( \hat{q}_i = q_i^* - 1 \).

By comparing the quality factor increments of two classes \( i \) and \( j \), we obtain the quality spacing between the classes that

\[
\frac{\hat{q}_i}{\hat{q}_j} = \frac{\lambda_j \lambda_i^2}{\lambda_i \lambda_j^2} = \sqrt{\frac{\lambda_j}{\lambda_i}} \sqrt{\frac{\delta_i}{\delta_j}}.
\]

(12) indicates that the ratio of excess bandwidth allocation between the two classes with given arrival rates is proportional to their differentiation parameters. This completes the proof. \( \Box \)

Note that when system load is heavy, that is, \( \lambda \) is close to the bound \( B \), all requests are going to be allocated the lower bound of the bit rate, which would in turn minimize the rejection rate when the system is heavily loaded.

Recall that the quality factor \( q_i \) must be less than or equal to the upper bound of streaming bit rate \( K \). According to \( \omega \), the computed \( q_i \) could be greater than \( K \) when the system is lightly loaded. As a result, certain request classes may not be able to use all their allocated channels. To improve the channel utilization, we can re-distribute the excess channels to other request classes or simply leave them for calculating the channel release rate in the next allocation interval.

According to \( \omega \), given fixed \( \lambda_i \), the classes with higher \( \delta_i \) get more portion of available network-I/O bandwidth. However, by \( \omega \), \( \hat{q}_i \geq \hat{q}_j \) if only if \( \frac{\delta_i}{\lambda_i} \geq \frac{\delta_j}{\lambda_j} \) holds. Otherwise, the property of predictability of service differentiation becomes violated. For differentiation predictability, one solution is temporary weight promotion, as suggested in the context of server node partitioning \( 33 \). That is, the request scheduler temporarily increases quality differentiation weight \( \delta_i \) based on the current request arrival rates, so as to ensure \( \frac{\delta_i}{\lambda_i} \geq \frac{\delta_j}{\lambda_j} \).

### 4. Implementation Issues

We built a simulation model for the evaluation of the network-I/O bandwidth allocation and scheduling schemes on streaming servers. The model divides the simulation process into a sequence of short intervals and performs bandwidth allocation and scheduling functions based on predicted arrival rates of the request classes and measured server bandwidth release rate in each interval.

**Estimation of Request Arrival Rate.** The request arrival rate of each class \( \lambda_i \) is estimated by counting the number of requests from each class occurring in a moving window of certain immediate past periods. The moving window estimation approach based on history information was used in many similar experiments \( 2 \). A smoothing technique based on a decaying function is applied to take weighted averages over past estimates.

**Measure of Bandwidth Allocation Rate Bound (\( B \)).** Since the focus of this paper is on service differentiation provisioning, we assume that the streaming servers provide support for a video playback function only. Because of the continuous feature of streaming services, it is feasible to measure the bandwidth to be released in the current allocation interval. We employed a similar smoothing window to calculate the bound of bandwidth allocation rate in each allocation interval. It takes into account the bandwidth to be released in the current allocation interval and the excess bandwidth in the past allocation intervals.

**Admission Control.** The derived bandwidth allocation schemes in \( 5 \) and \( 9 \) ensure that the requests in higher classes always get better services in terms of streaming bit rate. However, streaming services usually have a maximum acceptable waiting time \( 2 \). If the
aggregate arrival rate of all classes (λ) exceeds the bound of bandwidth allocation rate (B) in the current allocation interval due to some bursty traffic, the dispatcher in the streaming cluster imposes admission control and drops those requests which have waited in the queue for more than the maximum acceptable waiting time. The requests to be rejected are first from the lowest differentiated class, and then the second lowest class, and so on [11, 33].

**Feedback Queue.** We note that estimation accuracy of the arrival rate in an allocation interval is affected by the variance of arrival distributions. It fluctuates due to the existence of bursty traffic. When the actual arrival rate exceeds an estimated one, streaming bit rates would be over-allocated to current requests, leading to queueing delay of subsequent requests. On the other hand, some network-I/O bandwidth would be wasted due to under-measured streaming bit rates if the actual arrival rates are over-estimated. To reduce the impact of the variance on performance, we introduced a feedback queue as a smoothing technique. It takes into account the backlogged requests into the estimation of the arrival rates. It calculates \( \lambda_i = \lambda_i + \alpha \times l_i \), where \( l_i \) is the number of backlogged requests of class \( i \) at the end of the past allocation interval and \( \alpha, 0 \leq \alpha \leq 1 \), is a scaling parameter, indicating the percentage of the waiting queue to be included.

5. PERFORMANCE EVALUATION

In this section, we examine the impact of service differentiation on system performance and the impact of various bandwidth allocation schemes on service differentiation provisioning in terms of streaming bit rate and queueing delay. We assume the streams are delivered according to a First-Come-First-Served with First-Fit backfill (FCFS/FF) scheduling policy, unless otherwise specified. In the FCFS/FF scheduling, requests are responded in the order they arrive, as long as sufficient system resources are available to meet their requirements. In the case that the head-of-the-line request is blocked due to the lack of sufficient resources, first-fit backfilling searches further down the listen queue for the first request that is able to be scheduled immediately.

The experiments assumed that the videos were encoded with a bit rate of 8 Mbps, a typical value for high-quality MPEG DVD. The minimum acceptable bit rate was set to 1 Mbps, an acceptable one for low-quality MPEG movies. Thus, 8 Mbps and 1 Mbps was the upper and lower bound of streaming bit rate for transcoding, respectively. The streaming cluster consisted of 8 servers and each server had 1.8 Gbps network-I/O bandwidth. Totally, the streaming capacity of the cluster ranged from 20 requests at 8 Mbps per minute to 160 requests at 1 Mbps per minute. Thus, the aggregate arrival rate ranged from 20 to 160 requests per minute. Like related experiments in [2, 17], the access patterns were generated by a Poisson process and each video lasted 2 hours. The maximum acceptable queueing delay was set to 4 minutes [2]. If the queueing delay of a request exceeded 4 minutes, either the admission control rejected the request or the client dropped the request.

We adopted a service differentiation workload from an eBay online auction that was used for analysis of queueing-delay differentiation in [33]. It consisted of three classes of client requests, 10% requests from registered clients for bidding and posting (Class A, premium), 40% requests from registered clients for browsing and searching (Class B, ordinary), and 50% requests from unregistered clients (Class C, basic). That is, their request arrival ratios \( (\lambda_a, \lambda_b, \lambda_c) \) were \( (1, 4, 5) \). Their quality differentiation weights \( (\delta_a, \delta_b, \delta_c) \) were assumed to be \( (4, 2, 1) \). Each representative result reported in this section is an average of 500 runs.

5.1 Impact of DiffServ on System Performance

We first examined the system performance due to various bandwidth allocation schemes. In addition to the proportional-share and harmonic proportional allocations, two static bandwidth allocation schemes are included. A static non-uniform bandwidth allocation scheme provides absolute service differentiation. It allocates the fixed streaming bit rate 4Mbps, 2Mbps and 1Mbps to requests from class A, B and C, respectively. A static uniform bandwidth allocation scheme supports no service differentiation; we considered three uniform encoding bit rates: 2Mbps, 3Mbps and 4Mbps for all the requests.

Figure 2 shows the impact of service differentiation on the average rejection rate. It shows that the harmonic allocation scheme guarantees service availability. This is achieved by degrading the streaming bit rates adaptively with transcoding according to system load. The proportional-share allocation scheme achieves similar results to the harmonic allocation. The absolute allocation scheme cannot adapt to system load dynamically and so that it cannot guarantee service availability. Like the system with the absolute differentiation allocation, the average rejection rate in the system without service differentiation increases abruptly after arrival rate exceeds corresponding knee points. The figure reveals that both the harmonic and proportional bandwidth allocation schemes can achieve high throughput and high service availability when servers are heavily loaded.
Figure 3 shows the impact of service differentiation on average queuing delay. It can be seen that without service differentiation or with the absolute service differentiation, the average queuing delay increases abruptly after arrival rate exceeds certain knee points and rejection occurs at the corresponding levels, as shown in Figure 2. The average queuing delay is approaching and bounded by the maximum acceptable waiting time (4 minutes). In contrast, the harmonic and proportional allocation schemes maintain the average queuing delay in acceptable degrees at various arrival rates. The queuing delay is due to the variance of interarrival time distributions. It also shows that the harmonic allocation scheme yields slightly lower queuing delay than the the proportional allocation when the system load is light.

Figure 4 shows that the harmonic allocation scheme enables the streaming system to efficiently and adaptively manage its available network-I/O bandwidth. The proportional allocation scheme obtains similar results, which were omitted for brevity. On the other hand, the absolute allocation does not provide such an adaptivity. Like the system without any service differentiation, the system with the absolute differentiation allocation wastes considerable streaming bandwidth when system load is light. They can fully utilize their bandwidth when arrival rate exceeds some knee points. However, as shown in Figure 2, this utilization comes at the cost of driving rejection rate to unacceptable levels.

In summary, in comparison with the absolute DiffServ strategy, both the harmonic and proportional-share bandwidth allocation schemes make it possible for a streaming server to achieve high throughput, high service availability, and low queuing delay when the server is heavily loaded.

5.2 Impact of Bandwidth Allocation Schemes

The second experiment was on the impact of the network-I/O bandwidth allocation schemes on service differentiation provisioning in details.

Figure 5 shows a microscopic view of the streaming bit rate of individual requests in the three classes due to the proportional and harmonic allocation schemes, when arrival rate is low (50 requests/minute), medium (80 requests/minute), and high (110 requests/minute), respectively. The simulation at each arrival rate was run for 60 minutes. Each point represents the streaming bit rate of individual requests in the classes in consecutive recording time units. It can be seen that both bandwidth allocation schemes consistently enforce pre-specified quality ratios between the request classes. We find that both the harmonic and proportional-share allocation schemes achieve short-term objective of service differentiation in terms of streaming bit rate. Although the harmonic allocation scheme was proposed to maximize the overall quality factor of streams, it provides the same level of differentiation control over the inter-class quality spacings as the proportional-share scheme.

Figure 6 shows the average streaming bit rate of requests from each class due to the proportional and harmonic allocation schemes at various arrival rates. The transcoding-enabled bandwidth allocators degrade the streaming bit rate of each request class adaptively with varying system load. When system load is light, requests from class A tend to receive the upper bound of streaming bit rate, i.e., 8 Mbps. When system load is moderate, all request classes get their fair shares. When system load is heavy, all request classes tend to receive the lower bound of streaming bit rate, i.e., 1 Mbps. Furthermore, requests from class A receive higher streaming bit rate by the use of the harmonic allocation scheme than by the use of the proportional one. In general, the harmonic allocation scheme favors requests from higher classes more than the proportional one. The proportional allocation scheme adjusts the quality levels of request classes in proportion to their differentiation weights. The harmonic allocation scheme is also proportional, as shown by (12) and proved in Theorem 3.1. In all cases, requests from higher classes
so that the impact of the backlogged requests on the bandwidth queueing delay of subsequent requests. As the arrival rate further increases, requests in lower classes due to another popular strict priority request scheduler. In this experiment, we investigate the relationship between the harmonic proportional bandwidth allocation with request scheduling approaches.

Recall that the QoS-aware bandwidth allocation schemes are implemented in combination with the FCFS/FF request scheduling by default. The preceding experiments show that the FCFS/FF scheduler does not differentiate queueing delay of requests from different classes. In this experiment, we investigate the relationship between the harmonic proportional bandwidth allocation with request scheduling approaches.

Figure 7 shows the queueing delay of requests from the three classes due to the harmonic proportional and proportional-share allocation schemes at various arrival rates. The queueing delay is due to the variance of interarrival time distributions. When system load is light ($\lambda < 30$), the queueing delay of all request classes is trivial. It is because some network-I/O bandwidth was unused during the past allocation intervals due to the existence of upper bound of streaming bit rate. When arrival rate exceeds 40 requests/minute, unexpectedly, we find a queueing-delay dip scenario. That is, the queueing delay initially increases and then marginally decreases as arrival rate increases and then increases significantly as arrival rate is close to the system’s streaming capacity. Note that the queueing delay is not only affected by the variance of interarrival time distributions, but also affected by differentiated bandwidth allocations. Because the backlogged requests in queues are not considered in the calculation of arrival rate of classes, the streaming bit rates are over-allocated to current requests, leading to higher queueing delay of subsequent requests. As the arrival rate further increases, the requests are allocated with lower streaming bit rates so that the impact of the backlogged requests on the bandwidth over-allocation decreases and thus the impact on queueing delay of subsequent requests decreases. The impact of the variance of interarrival time distributions on queueing delay dominates and it is significant when system load is close to the system’s streaming capacity ($150 < \lambda < 160$). In Section 5.4, we will show that the feedback queue technique can significantly mitigate this kind of queueing-delay dip scenarios.

Figure 7 also shows that by the use of the two bandwidth allocation schemes, requests from class A have higher queueing delay compared to requests from class B and C. Recall that the simulation assumed FCFS/FF request scheduling in combination with bandwidth allocation. Although FCFS/FF scheduling does not provide any queueing-delay differentiation between different request classes, the QoS-aware bandwidth allocation schemes affect the performance metric of queueing delay, as well. Requests from higher classes tend to be allocated with higher streaming bit rates. This results in higher queueing delay. Due to the first-fit feature of the FCFS/FF scheduler, on the other hand, requests from lower classes have higher probabilities to be processed with the differentiated streaming bit rates. Figure 7 also shows that compared with the proportional allocation scheme, the harmonic allocation postpones the emergence of queueing delay. We also obtained 95% confidence intervals for the QoS metrics. All bounds were uniformly tight. Due to the space considerations, they are omitted here.

In summary, we observed that both the harmonic and proportional allocation schemes can achieve the objective of service differentiation provisioning (in terms of streaming bit rate) in long and short timescales. They have positive side effects on the request queueing delay and the harmonic allocation scheme leads to lower queueing delay when the system is lightly loaded.

5.3 Impact of Request Scheduling Schemes

Recall that the QoS-aware bandwidth allocation schemes are implemented in combination with the FCFS/FF request scheduling by default. The preceding experiments show that the FCFS/FF scheduler does not differentiate queueing delay of requests from different classes. In this experiment, we investigate the relationship between the harmonic proportional bandwidth allocation with request scheduling approaches.

Figure 8 shows the queueing delay of requests from the three classes due to another popular strict priority request scheduler. In the strict priority request scheduling, requests in queue $i$ cannot be serviced until the higher priority queues are all empty in the current allocation interval. The figure shows that the priority scheduler imposes certain degrees of control over queue-delay differentiation between the request classes. The queueing delay of requests from class A is rather limited since the priority scheduler favors the requests of higher classes. This is achieved at the cost of higher queueing delay of requests in lower classes. Note that the strict priority scheduling itself can not guarantee quality spacings between various classes. Time-dependent priority scheduling schemes, widely addressed in the packet scheduling in the network side [12, 20], deserve further studies in providing queueing-delay differentiation in streaming services.

5.4 Impact of the Feedback Queue

Figures 7 and 8 illustrate the queueing-delay dip scenarios. That is, the queueing delay increases and then marginally decreases and significantly increases again as the arrival rate varies within a certain range. As we discussed above, it is due to the impact of the backlogged requests on the bandwidth allocation, together with the variance of interarrival time distributions. In this experiment, we investigate the impact of the feedback queue technique on mitigating...
Figure 8: Impact of strict priority request scheduling.

Figure 9: Impact of the feedback queue technique.

The feedback queue technique reduces queueing delay at the cost of reducing streaming bit rates of request classes because their arrival rates are less underestimated. We experimented with bandwidth allocation schemes with and without the presence of the feedback queue and found only marginal differences between the streaming bit rates of request class A when the system is lightly loaded. The feedback queueing technique found no impact on the streaming bit rates of classes B and C at all.

5.5 Impact of Request Arrival Ratio

In the previous experiments, the request arrival ratios of the three classes \((\lambda_a, \lambda_b, \lambda_c)\) were fixed to (1, 4, 5). In this section, we examine the impact of these parameters on differentiation performance. In the following experiments, the harmonic proportional allocation scheme was adopted.

Figure 10 shows a microscopic view of the differentiated streaming bit rate of requests due to various arrival ratios among the three classes. We varied the request arrival ratios \((\lambda_a, \lambda_b, \lambda_c)\) from (1, 4, 5) to (2, 3, 5), (5, 5, 10), and (3, 2, 5), respectively. The aggregate arrival rate of the three classes was kept to be same at 50 requests/minute. That is, the class C load did not change. The quality differentiation weight of the three classes \((\delta_a, \delta_b, \delta_c)\) were fixed equal to (4, 2, 1). The simulation at each request arrival ratio was run and recorded for 40 minutes. It can be seen that as a fraction of the class B load shifts to class A, the quality factor of class B increases, while that of class A and C decreases. This verifies the property 4 of the controllability and dynamics achieved by the allocation scheme (10).

We also performed a wide range of sensitivity analyses. We varied the video duration, the server outgoing bandwidth, the maximum acceptable waiting time, and the differentiation weight ratio. While we do not have space to present all of the results, we note that we did not reach any significantly different conclusions regarding to the differentiation predictability, controllability, and fairness achieved by the proposed bandwidth allocation schemes.

6. RELATED WORK

Streaming technology is critical to many popular multimedia applications, such as VoD, distance learning, digital library etc. Early studies focused on data layout and data retrieval, memory buffering, admission control and disk scheduling on individual servers; see [27] for an example. Recent focus is on scalability and reliability in distributed servers. Streaming services are I/O-intensive. Multicasting and periodic broadcasting strategies have been stud-
ied for reducing disk-I/O and network-I/O requirements at server side; see [2, 13, 15, 17] for representatives. In those bandwidth allocation schemes, the primary QoS metric (streaming bit rate) is fixed and constant and it is the same to all streams. In this article, we investigate QoS-adaptive bandwidth allocation schemes for providing differentiated streaming services.

Service differentiation aims to provide differentiated levels of QoS to different classes by dynamically allocating available resources. Its idea stemmed from the network side. Existing efforts focus on QoS guarantees and QoS adaptation in packet scheduling and packet dropping. For example, Dovrolis et al. defined a proportional differentiation model to enable the network operator to adjust the quality spacing between packet classes independent of the class loads [12]. They applied the model on queueing-delay differentiation in packet forwarding, which guarantees the quality spacing between packet classes be proportional to certain pre-specified class differentiation parameters. They further applied the model on loss differentiation in packet dropping.

Servers are a major force in end-to-end service differentiation provisioning. There were many early efforts on providing differentiated Web services. Strategies are mostly based on priority-based resource allocation and scheduling with admission control. Almeida et al. modified Apache server by adding a new Scheduler process [3]. It sets different upper bounds on the number of concurrent processes allocated to the process pool for requests of each class. Response-time differentiation is hence achieved by limiting the per-class process pool size and process concurrency. Bhatti et al. modified Apache server at application level by creating a new process, called connection manager, to intercept all incoming HTTP requests [8]. Connection manager categorizes the requests and puts them into the appropriate queues for the corresponding service classes. Child processes are scheduled to handle the requests in the multiple queues in a strict priority order and hence response-time differentiation is achieved. Zhu et al. adopted a multi-server queueing model to guide node-based resource allocation optimization in server clusters [33]. Based on the measured workload of different request classes and their priority levels, a dynamic node partitioning strategy adaptively partitions the server nodes of a cluster and allocate them to handle requests of different classes. OS support for service differentiation provisioning has also been addressed in prior efforts, as exemplified by resource containers [7], cluster reserves [6], and QLinux [28]. The priority-based resource allocation and scheduling strategies are often used in combination with various admission control strategies to provide performance guarantees and differentiation in Web services. For example, Abdeulzerah et al. proposed control-theoretical approaches for response time guarantees [1]. Lee et al. proposed queuing-theoretical approaches for proportional queueing delay differentiation [19]. In those efforts, CPU power is often treated as the resource bottleneck and response time is the primary performance metric. The approaches cannot be used to provide differentiated streaming services with respect to streaming bit rate.

A few studies focused on network-I/O bandwidth scheduling for service differentiation provisioning in multimedia servers. Fox et al. adopted a proxy-based approach to content adaptation, in which proxy agents placed between clients and servers perform aggressive computation and storage on behalf of clients [16]. Adaptation is achieved via application-level datatype specific transcoding techniques, which preserve information of image and text objects that has the highest semantic value. The proxy-based approach enables itself to transparent incremental deployment, and provides a smooth path for rapid prototyping of new client devices, content formats and network protocols. In their work, content adaptation techniques are used to adapt content to network and client variations. However, the approach does not provide guarantees on predictable quality spacings between various classes.

Chandra et al. designed a quality-ware image transcoding technique to provide differentiated multimedia Web services at application level [10]. Image transcoding is used by the server to customize the size of image objects (JPEG) and hence manage the available network-I/O bandwidth on the server. The primary performance metric is the image quality factor. The transcoding technique can provide graceful degradation of image quality factors so that the preferred clients are served at quality factors that closely follow the original images and non-preferred clients are served at a lower image quality factor. The transcoding-enabled network-I/O bandwidth management schemes allow the server to provide acceptable access latency to clients by trading off image quality for image size. Like the work in [16], this approach also does not guarantee predictable quality spacings between difference classes.

Schojer al. proposed an architecture of a video proxy cache, which uses video adaptation to improve the cache replacement strategies in the proxy [26]. Our work differs from those prior approaches in that it offers predictable quality spacings to different classes based on theoretical foundation of resource allocation. We prove that the optimal bandwidth allocation strategy provides a guarantee of proportional share between the request classes with respect to their predefined differentiation weights.

7. CONCLUSIONS AND FUTURE WORK

Recent advance of real-time transcoding technology makes it possible for streaming servers to manage the limited network-I/O bandwidth available to the Internet and control the quality spacing between request classes at a fine-grained level. In this article, we have presented a bandwidth allocation mechanism to facilitate the delivery of high bit rate streams to high priority requests without over-compromising low priority requests on streaming servers. We have formulated the bandwidth allocation problem as optimization of a harmonic system utility function and derived the optimal streaming bit rates under various server load conditions. We have proven that the optimal bandwidth allocation guarantees proportional sharing between requests with different priorities.

We have evaluated the bandwidth allocation schemes via extensive simulations. Simulation results have shown that

1. The harmonic bandwidth allocation scheme enables the streaming servers to efficiently and adaptively manage their network-I/O bandwidth and hence achieve high service availability and maintain low queueing delay when the servers are heavily loaded.

2. The harmonic bandwidth allocation with respect to the overall system utility function can achieve the objective of proportional sharing (in terms of the streaming bit rate) in both short and long timescales.

3. The harmonic bandwidth allocation scheme is based on estimates of request arrival rates of different classes in moving windows. The feedback queue technique increases the estimation accuracy and helps reduce the allocation variations caused by bursty traffic.

4. The harmonic allocation scheme, in combination with the strict priority request scheduling provides certain degrees of control over the queueing-delay differentiation of the request classes, although no quality spacing between the classes is guaranteed.
We note that this article considered no CPU overhead for on-line transcoding. As a matter of fact, the transcoding technology enables efficient utilization of network-I/O bandwidth at the cost of CPU cycles. The cost modelling of continuous media transcoding is still an open research issue. Given the cost model of transcoding, it is important to make a tradeoff between the network-I/O bandwidth and CPU power. Therefore, transcoding-enabled service differentiation deserves a further study.

8. REFERENCES


