Proportional Differentiation: A Scalable QoS Approach

Yang Chen and Chunming Qiao, SUNY Buffalo
Mounir Hamdi and Danny H. K. Tsang, HKUST

ABSTRACT

The *proportional service model* has gained attention recently as an effective solution for quantitative service differentiation in IP networks. In particular, this differentiation scheme is controllable, consistent, and scalable. Thus, it gives network operators convenient management of services and resources even in a large-scale network. In this article we give an overview of recent research efforts on this QoS model. Details about implementation strategies for various QoS metrics are provided. We also discuss how to achieve absolute service bounds in this relative differentiation model with different approaches. Several problems such as feasibility in differentiation are mentioned as open research issues at the end.

INTRODUCTION

Internet traffic has been growing at an exponential rate. Driving this growth is the fact that the Internet is gradually and successfully becoming a mission-critical platform for conducting business. The prevailing *best effort* mechanism does not provide any service differentiation or guarantee for this new commercial infrastructure. Hence, the current Internet cannot satisfy diverse quality requirements and different user expectations. As a result, there is urgent demand for the provisioning of quality of service (QoS) guarantees with the evolution of the Internet. Two broad approaches have emerged as solutions to this demand: integrated services (*IntServ*) and differentiated services (*DiffServ*).

The IntServ approach is based on resource reservation. Every router should be capable of reserving resource for each traffic flow. Keeping flow-specific states will result in a scalability problem at the core routers, which manage millions of traversing flows. Furthermore, incremental deployment is impossible because path setup and resource reservation cannot be achieved unless each router is able to participate (i.e. IntServ-capable). These factors present practical obstacles to the deployment of IntServ infrastructure.

The DiffServ model was proposed as a solution to the scalability problem of the IntServ model. In this case, packets are classified into a small number of service classes at the edge router according to their service requirements. The core router will differentiate between packets on a class-by-class rather than flow-by-flow basis. DiffServ has several desirable characteristics. It avoids the scalability problem associated with IntServ. In particular, involved operations such as packet classification and per-hop behavior (PHB) encoding are done at the edge router. As a result, the core router can be built with an architecture that has fewer functional units but forwards packets faster. In addition, DiffServ can be deployed incrementally. A heterogeneous network with non-DiffServ-capable routers is also able to provide service differentiation with DiffServ-capable routers on the congested links.

However, the DiffServ model, or more specifically the relative DiffServ model, can only guarantee that traffic of a high-priority class will receive no worse service than that of a low-priority class. Recently, it has been further refined into a quantitative scheme: the proportional QoS model [1]. This model provides network operators with adjustable and consistent differentiation between service classes, which cannot be achieved with other relative differentiation models such as strict prioritization. With this QoS model, the service differentiation level between classes can be adjusted according to prespecified factors, and these quantified differentiations are stable even in a short time period. More important, as a variant of relative DiffServ, the proportional QoS model keeps the benefit of scalability.

Various QoS metrics for packet loss and delay have been investigated in this proportional paradigm [1–4]. Although the initial work studied differentiation on a particular QoS metric in a per-hop manner, recent efforts are reported on proportional differentiation provisioning over multiple QoS metrics simultaneously [5, 6] and end to end [7, 8]. Because of the simplicity of the relative QoS model, absolute QoS guarantees cannot be directly provided in a proportional QoS model. Different approaches were proposed to achieve absolute QoS bounds within this proportional paradigm [6, 9–11].

The rest of the article is organized as follows. We introduce the proportional differentiation model and its generalized representation form. We discuss details of implementations of proportional differentiation on various QoS metrics. How to achieve absolute QoS bounds on packet delay and loss is explained later. We conclude with some open research issues.

THE PROPORTIONAL QOS MODEL

The advantage of a proportional QoS model over a conventional relative QoS model is that while scalability is retained, the differentiation level can be quantitatively adjusted to be proportional to differentiation factors set in advance. If q_i is the QoS metric of interest and s_i the differentiation factor for class i, in the proportional QoS model, we should have $q_i/q_j = s_i/s_j$ ($i, j = 1 \ldots N$). For example, in an IP network, assume that l_1, l_2 are the packet loss rates for class 1 and 2, respectively. If s_1 is 1 and s_2 is 2, we should have $l_1/l_2 = 1/2$, which means the packet loss rate of class 2 is twice that of class 1.

It is desirable that the proportional differentiation model hold over not only long timescales but also short ones [1]. Then within a short time period τ, the following should hold: $\bar{q}_i(t, t + \tau)/\bar{q}_j(t, t + \tau) = s_i/s_j$ ($i, j = 1 \ldots N$), where $\bar{q}_i(t, t + \tau)$ is the QoS metric measurement in time period τ.

In a real system, we cannot expect the differentiation ratio to be exactly the same as that specified by proportional factors. It is reasonable to allow some deviations. In [11], the following equation is used to describe a proportional service differentiation system (e.g., an IP router):

$$\frac{\bar{q}_i(t,t+\tau)}{\bar{q}_j(t,t+\tau)} = \frac{s_i}{s_j} \pm \Delta. \qquad (1)$$

The value of Δ represents the deviation from the relation specified by proportional factors between service classes i and j. Suppose we have N service classes and class 1 is the highest priority one (this convention will be followed in the following sections). We measure the QoS metric of each class and compute the absolute value of $\{\bar{q}_i(t, t + \tau)/\bar{q}_j(t, t + \tau) - s_i/s_1\}$ for i from 2 to N. The maximum of these $(N - 1)$ values is considered the system's deviation.

This QoS model is controllable, consistent, and scalable: by changing the service differentiation factor s_i, the differentiation between certain service classes can be adjusted within a bounded deviation; if τ is small enough, the traffic in a higher-priority class will consistently receive better service than a lower-priority class independent of the load fluctuation; since proportional differentiation is a variant of relative DiffServ, no flow-specific state needs to be recorded or managed, which guarantees the scalability of this QoS model.

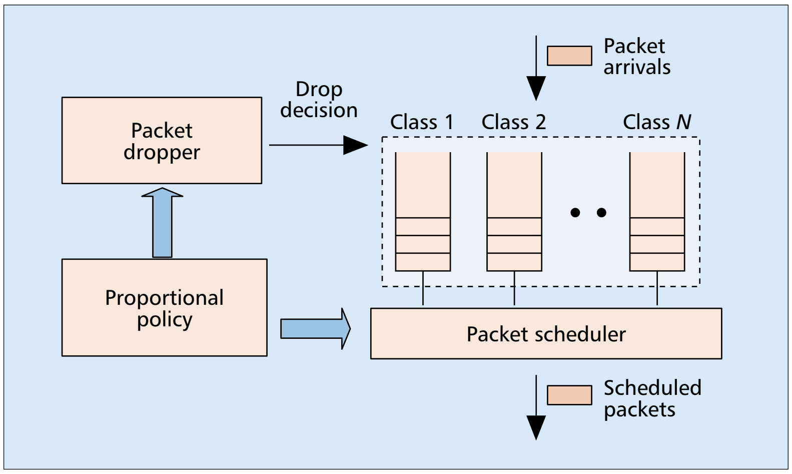


Figure 1. *Proportional differentiation provisioning architecture in a router.*

GENERAL FRAMEWORK AND IMPLEMENTATIONS

In order to provide differentiated services in an IP network, we need two modules at routers: the packet dropper and packet scheduler. Packets from different service classes enter the corresponding logical queues. When a packet needs to be forwarded, the packet scheduler decides from which service class a packet will be served. Packet dropper makes the drop decision when a packet needs to be dropped according to the buffer management schemes such as Random Early Detection (RED). However, in order to achieve quantitative differentiation between classes, we need a proportional policy unit which takes the real-time measurement of the QoS metrics and manages the packet dropper and scheduler as illustrated in Fig. 1.

The first QoS metric discussed in proportional QoS provisioning is average packet delay [1] with the Wait Time Priority (WTP) scheduler. The work in [2] extended proportional differentiation to packet loss rate using a proportional loss rate (PLR) dropper. The performance of real-time applications (e.g. IP telephony) does not depend on average packet delay. As a result, the proportional QoS model has been applied to deadline violation probability in [3] and is a more suitable performance parameter for those applications. Recently, jitter [4] has also been included in this paradigm. In this section we give an overview of the details in achieving proportional differentiation over various QoS metrics.

PROPORTIONAL PACKET DELAY

Weighted Fair Queuing (WFQ) has been used to reserve a fixed portion of bandwidth for a service class by assigning a certain weight to its logical queue, which represents the priority level of the queue in packet scheduling. However, average packet delays are not proportional to these fixed weights. The weights need to be adjusted dynamically based on the measurement or estimation of packet delay to achieve a proportional differentiation.

The WTP scheduler was first proposed to provide proportional differentiation in average packet delay in [1]. The priority of a particular

A traffic fluctuation might make the differentiation ratio deviate from the specified value within a short time period. The reason is that the dropping decision is made on the entire history of packet arrival and loss, which might not accurately represent the current traffic load condition.

class i at time t is $p_i(t) = w_i(t)/s_i$ where $w_i(t)$ is the waiting time of the packet at the head of logical queue i, and $s_N > s_{N-1} > \ldots > s_1 = 1$ are the proportional factors for each class. When a packet needs to be forwarded at time t, a packet from class i with the maximum $p_i(t)$ will be scheduled. The WTP scheduler is able to achieve average packet differentiation proportional to the factors assigned when the traffic load is heavy [1]. Further study in [12] shows the dependence of delay differentiation on the traffic load. Considering average packet delay as the QoS metric, Eq. 1 turns into

$$\frac{\bar{d}_i(t,t+\tau)}{\bar{d}_j(t,t+\tau)} = \frac{f(s_i,\bar{\lambda}_i)}{f(s_j,\bar{\lambda}_j)} \pm \Delta, \qquad (2)$$

where $\bar{\lambda}_i$ is the arrival rate for traffic class i within period τ and $f(.)$ represents the joint effect of proportional factor and traffic load on the service differentiation. A dynamic proportional factor adjustment scheme with periodic arrival rate measurement of each service class is proposed in [12] to provide accurate proportional differentiation on packet delay even when traffic load is not heavy.

In the WTP scheduler, a time monitoring and related computation process is initiated whenever a packet needs to be forwarded. A scheme with less computational and operational complexity, called Dynamic WFQ, was proposed late in [5]. Instead of using a packet's waiting time directly, average packet delay is predicted based on arrival and service rates for each class. Queues' weights (i.e., service rates) are adjusted periodically with those predicted values. Simulation results show that with a short weight adjustment period, Dynamic WFQ can provide proportional differentiation on average packet delay comparable to that of the WTP scheduler [5].

For digital continuous media applications such as audio and video streaming, delay jitter is required to be within certain limits. In order to manage this service parameter as the volume of real-time traffic keeps increasing, the proportional QoS model was applied to jitter differentiation [4]. Average jitter is predicted using recorded packet arrival information as $\bar{j}_i(t, t + \tau)$, and a Relative Jitter Packet Scheduler (RJPS) selects a packet from the class with maximum normalized jitter $\bar{j}_i(t, t + \tau)/s_i$.

PROPORTIONAL PACKET LOSS

In order to achieve proportional differentiated packet loss rate, two droppers, PLR(M) and PLR(∞), were proposed in [2] that make the dropping decision based on loss rate measurements. There are two counters, A_i and D_i, for each service class i. A_i records the packet arrivals while D_i records the number of dropped packets; $s_N > s_N - 1 > \ldots > s_1 = 1$ are the proportional factors. A packet from the logical queue with minimum normalized loss rate D_i/A_is_i will be dropped. The only difference between PLR(∞) and PLR(M) dropper is the timescale on which they make the packet loss rate measurement.

For the PLR(∞) dropper, the counters start recording the information once the system begins operation. It is simple to implement but not adaptive to load fluctuation [2]. Similar to delay differentiation, the accuracy of the proportional loss differentiation is related to traffic load and composition when the PLR(∞) dropper is used. Traffic fluctuation might make the differentiation ratio deviate from the specified value within a short time period [2, 11]. The reason is that the dropping decision is made on the entire history of packet arrival and loss, which might not accurately represent current traffic load conditions.

On the other hand, the PLR(M) dropper is more adaptive to load fluctuation because the counters only record the information within the last M packet arrivals. A cyclic queue of size M will be maintained for the packet loss and arrival of each class. The PLR(M) gives better performance than PLR(∞) when the load fluctuates but requires an extra interior tag for each packet and corresponding tag operations. In addition, after every packet arrival or loss, the queue needs to be updated by replacing the oldest record with the latest one. This increases the processing complexity and might cause a scalability problem at the core router. More important, the value of M should be chosen with care. For large M, PLR(M) performs like PLR(∞). When M is too small, targeted loss rate ratio is well approximated in short time periods but not over long timescales.

Instead of calculating average loss rate with PLR(M), the authors of [13] estimated average drop distance (ADD) for each service class. The estimated ADD is the number of successfully transferred packets between two packet losses, denoted $\bar{d}_i$ for service class i. ADD is adaptive to the traffic load, so the problem associated with fixed M in PLR(M) is avoided, while average loss rate $\bar{l}_i$ is simply $1/\bar{d}_i$.

In [11], an active counter resetting process was proposed for a PLR(∞) dropper following the idea behind Eq. 1. When the system's deviation is less than a predetermined value *ERROR*, all the counters will be reset. Using this approach, counter overflow [3] is avoided. In addition, we can achieve proportional differentiation in a short timescale without a complicated operation in a PLR(M) dropper since the decision is always made based on recent nformation after the previous resetting. Furthermore, the parameter *ERROR* acts as a trade-off between the accuracy of the proportional relationship and adaptivity to the traffic fluctuation. A larger *ERROR* will result in larger deviation, but resetting occurs more frequently (i.e. the dropping decision will be made on more recent history).

JOINT AND END-TO-END QOS PROVISIONING

Besides the efforts discussed in previous sections on proportional QoS provisioning on single QoS metrics and in single-hop manner, research work has been done on differentiation over multiple QoS metrics and end to end.

Proportional Packet Loss and Delay — The Probabilistic Longest Queue (PLQ) mechanism was used to provide proportional differentiation over packet loss and delay instead of combined use of WTP scheduler and PLR dropper [14]. The queue length of each service class will be used to make the scheduling and dropping decisions. A probability p_i will be assigned to service class i as

$$p_i = (L_i / s_i) / \sum_{j=1}^{N} (L_j / s_j),$$

where L_i is the queue length for class i. A packet will be forwarded from the logical queue i with a probability p_i. Since the scheduling decision is made probabilistically, starvation for a low-priority class that might occur with a WTP scheduler is avoided [14]. Similarly, a dropping decision is made probabilistically using queue length information.

In another approach, joint buffer management and rate allocation formulates the service differentiation as an optimization problem [6]. With a fluid flow assumed, packet loss and delay can be geometrically illustrated after packet arrival and departure processes are drawn on the service curve figure for each service class. With system constraints such as buffer size and QoS constraints such as proportional differentiation, the objective function tries to minimize the amount of traffic to be dropped and the variance on the rate allocation (i.e., the frequency of adjustment on packet scheduling). However, due to its high computational complexity, a heuristic approximation is provided in [6]. In this approximation, the original optimization is decomposed into several smaller subproblems. When a buffer overflows due to physical buffer size limits, proportional differentiation over packet loss is taken care of. Differentiation in packet delay is checked periodically, and the rate allocation is adjusted if the proportional relation is violated.

For real-time applications, average packet delay is not a meaningful QoS parameter since packets whose delay is larger than a threshold will be dropped regardless of the value of the average delay. An Earliest Due Date (EDD) scheduler is usually used for those applications. There is a delay bound d_i for each class. When a packet arrives at time t, it will receive a tag $t + d_i$ that indicates its deadline. The scheduler will always serve a packet with a minimum tag first. Any packets exceeding their deadlines will be dropped directly. Hence, deadline violation probability is used as the QoS metric here. An enhancement of the EDD scheduler, Weighted EDD (WEDD), was proposed in [3] following the same idea applied to proportional loss rate differentiation. Two counters, D_i and L_i, will record the deadline violating packets and the total packets leaving the queue of class i, respectively. A safety margin S_i (usually $S_i = d_i/10$) is set for each class. When a packet needs to be forwarded at time t and there are more than one backlogged class with the first packet having a deadline less than $t + S_i$, the system is said to be in congestion mode. For each class fulfilling the above condition, the priority is calculated as $p_i(t) = D_i/L_i s_i$, where $s_N > s_N - 1 > \ldots > s_1 = 1$ are the proportional factors used. A packet from a class with maximum $p_i(t)$ will be scheduled. If the system is not in congestion mode, a WEDD scheduler is the same as an EDD scheduler.

End-to-End Solution — Per-hop proportional differentiation on packet delay can easily be translated into proportional differentiation in end-to-end delay. In other words, if $d_{im}/d_{jm} = s_i/s_j$ holds at any intermediate node m between the source and the destination, we have

$$\sum_{M=1}^{N} d_{jm} = s_i / s_j,$$

where N is the total number hops. Simulation results in [6, 12] confirmed that end-to-end proportional delay differentiation can be achieved by applying a proportional QoS policy at each hop. Since jitter always occurs at the ingress point rather than in the backbone, using the Relative Jitter Packet Scheduler (RJPS) proposed in [4] at the ingress router and a WTP scheduler at the core to achieve proportionally differentiated end-to-end delay and jitter was studied via simulation in [15].

However, it is not straightforward to achieve end-to-end proportionally differentiated packet loss rates via a single-hop approach. The mismatch is illustrated by the following example, used in [7]. In a two-hop network, suppose the loss rate of class i is $s_i{}^*l_a$ at node a and $s_i{}^*l_b$ at node b, respectively; then the end-to-end loss rate relations is

$$\begin{aligned}\frac{l_i}{l_j} &= \frac{s_i * l_a + (1 - s_i * l_a) * s_i * l_b}{s_j * l_a + (1 - s_j * l_a) * s_j * l_b} \\ &= \frac{s_i * (l_a + l_b) - l_a * l_b * s_i^2}{s_j * (l_a + l_b) - l_a * l_b * sj} \neq \frac{s_i}{s_j}.\end{aligned} \quad (3)$$

While the summation of the loss rates over all the intermediate routers follow Eq. 1, the end-to-end loss rate ratio contains products of the loss rates on individual routers and will deviate from Eq. 1. This is also shown by the simulation results in [6]. One possible solution proposed in [7] is that the dropping decision at each intermediate node should calculate loss rate with local packet loss and the total number of packets sent out from the source instead of packet arrival at this particular node. The information about packet loss occurring in the upstream nodes are encoded in the packet header and is passed along the downstream direction. Each router can recover the total number of packets sent out from the source with number of packet arrivals and packet loss at previous nodes. The product form will then be removed from Eq. 3, and end-to-end proportional differentiation on packet loss can be achieved.

Achieving Absolute QoS Bounds

There are many types of traffic that require strict QoS guarantees. For example, a real-time application puts a stringent requirement on packet delay. A data transfer operation cannot bear packet loss exceeding a certain threshold. Since the proportional differentiation model is a relative QoS model, absolute QoS guarantees cannot be provided directly. There are two primary approaches to absolute QoS bound provisioning in the proportional paradigm: end-to-end and single-hop.

The End-to-End Approach

Integration with Admission Control — Admission control was introduced into the proportional paradigm in [10]. A probing packet will be sent to the destination before the request for transmission is accepted. At each intermediate node, the packet loss rate is recorded. When

Since the proportional differentiation model is a relative QoS model, absolute QoS guarantees cannot be provided directly. There are two primary approaches to absolute QoS bound provisioning in the proportional paradigm: end-to-end and single-hop.

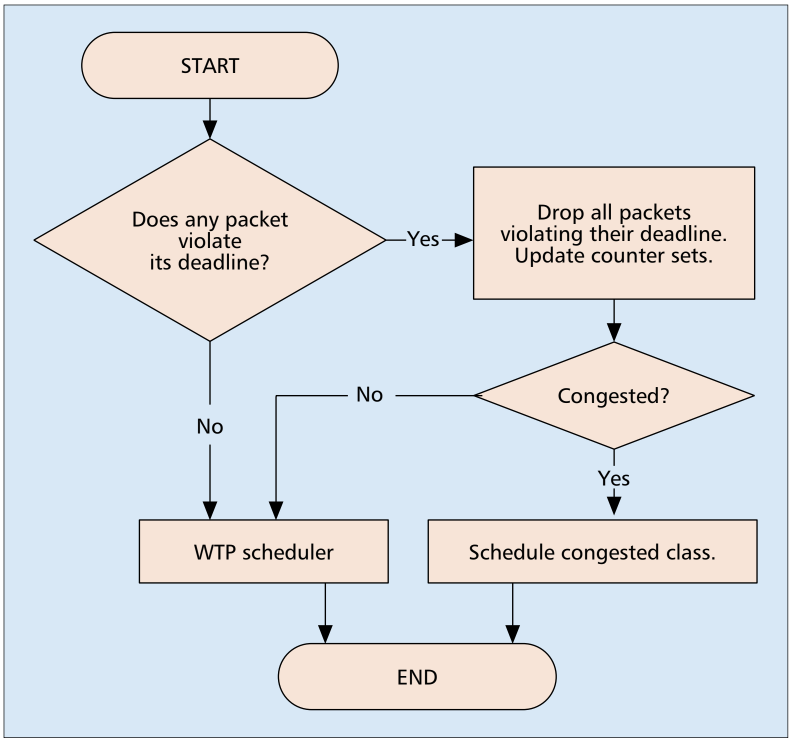

START
Does any packet violate its deadline?
Yes
Drop all packets violating their deadline. Update counter sets.
Congested?
No
No
Yes
WTP scheduler
Schedule congested class.
END

Figure 2. *Joint packet scheduling and dropping algorithm.*

the packet loss rate exceeds a prespecified threshold, a congestion indicator is set. When the probing packet detects congestion at any intermediate node from the source to the destination, or the end-to-end packet loss rate estimation based on per-hop packet loss information is higher than the requested level, admission will not be granted.

Adaptive Class Selection — Within a network that only offers per-hop relative delay and loss differentiation on a class-by-class basis, end-to-end QoS provisioning can be obtained via dynamically selecting the suitable service class. A sender can start transmission in an arbitrary service class i. Then the ingress router periodically sends out special packets to measure the end-to-end QoS for this flow [8], or the receiver monitors the end-to-end QoS and notifies the sender [9]. This flow will be moved to a higher-priority class if the QoS measurement exceeds the required absolute bound; otherwise, the flow's priority level is decreased. This scheme is used to achieve end-to-end delay guarantee in [8] and is also applicable to packet loss bounds provisioning [9].

In the above two approaches, admission control or end-to-end signaling for single traffic flows is required, which might cause too much overhead and a scalability problem. In the following section, we focus on achieving absolute bounds in a per-hop manner and on a class-by-class basis.

Single-Hop Approach

As mentioned previously, the joint buffer management and rate allocation approach [6] considers QoS provisioning an optimization problem. Absolute QoS bounds can be treated as the constraints in this problem, and these constraints have higher priority than proportional differentiation constraints. When all the constraints cannot be fulfilled at the same time, those with lower priorities will be relaxed (ignored). Below we introduce a simple yet effective heuristic algorithm proposed in [11] with constraint relaxation.

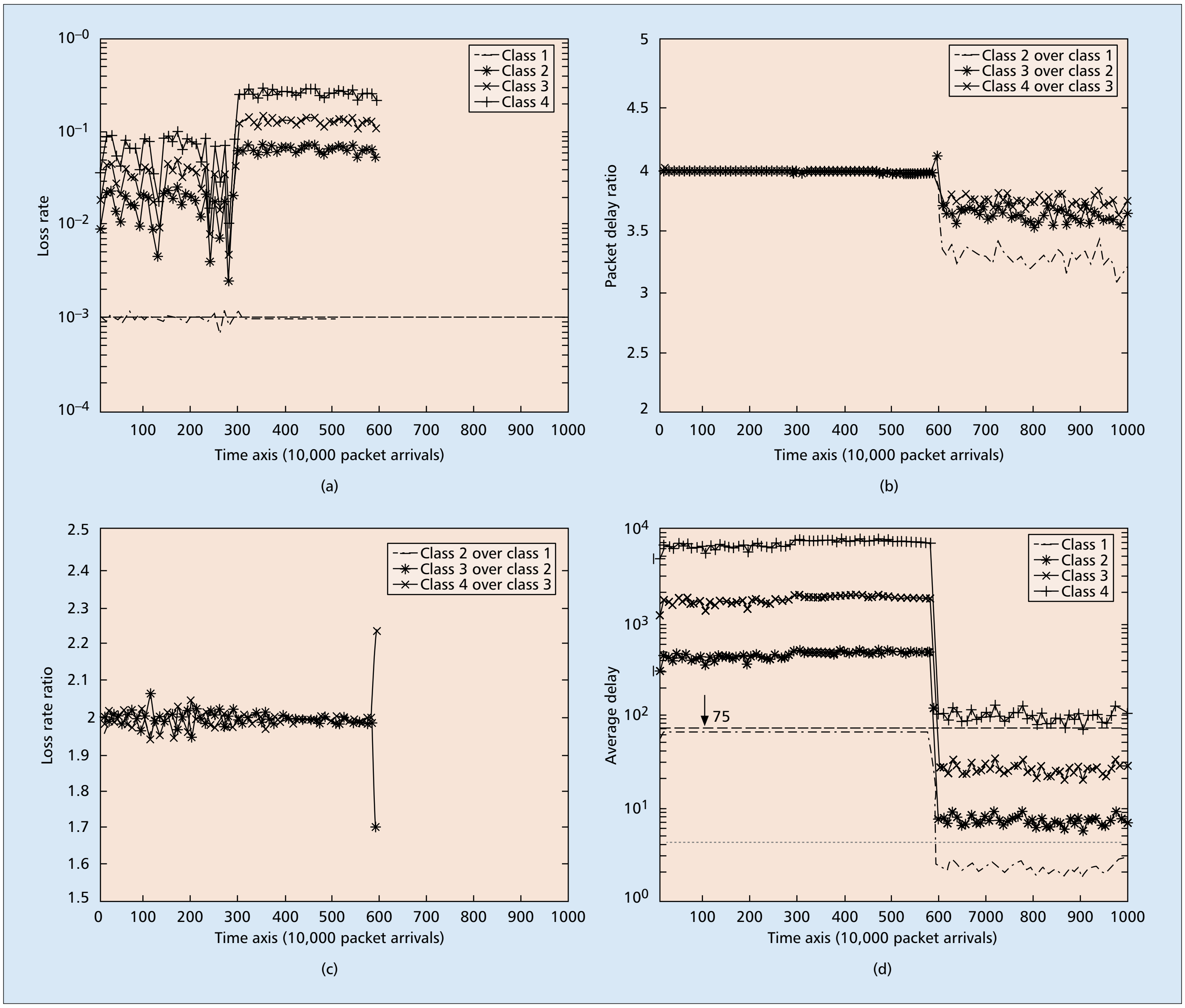
Joint Packet Scheduling and Dropping Algorithm — When a buffer overflows, a real-time measurement will be made as in previously discussed schemes such as PLR. If a particular class's loss rate is equal to or higher than the bound, its normalized loss rate l_i/s_i is set to be 1. This guarantees that no packet from the classes whose loss rates are around the bounds will be dropped. However, this might cause deviation in proportional differentiation and force it to be relaxed since an absolute QoS constraint has higher priority. An ordinary PLR dropper then makes the final dropping decision and maintains the proportional loss differentiation among the classes that do not violate their loss bounds.

Most of the computation complexity with the work in [6] is caused by the need to predict average packet delay. In [11], a delay bound d is used to replace the absolute constraint on average delay. This not only simplifies operation but is reasonable because a fixed delay bound for each packet is more meaningful than an average delay for a time-stringent application. Providing absolute delay guarantee in the proportional delay differentiation model should operate in the following manner: When a packet needs to be forwarded, all the packets violating their deadline are dropped first; then a WTP scheduler finishes the packet scheduling. Dropping deadline violating packets guarantees the absolute delay constraint while the WTP scheduler provides proportionally differentiated delay to those classes that do not violate their delay bounds. However, packets dropped due to deadline violation will be counted in the total packet loss, which might lead to a high loss rate for the service class that has an absolute delay constraint [11].

As mentioned above, dropping deadline violating packets will affect not only packet delay but also packet loss. Thus, packet dropping and scheduling should operate jointly. When the link is congested, there might be a lot of packets with absolute delay bounds waiting in the logical queue; it is very possible that many of them will be dropped due to deadline violation. However, if we can proactively increase the service rate for classes with absolute QoS requirements when the link is congested, queue length and delay of packet for those service classes will be reduced.

A modification is proposed in [11] following the line of thought in [3]. More specifically, if we set a delay bound d_i for a particular class i, a safety margin S_i (usually, $S_i = d_i/10$) is also set for this class. When a packet of class i is dropped due to deadline violation, if this class is still backlogged with the first packet having a deadline less than $t + S_i$, this class is said to be in congested mode. A packet from this class is scheduled directly instead of the decision made by a WTP scheduler. Hence, we can increase the service rate of class i to avoid dropping too many packets.

The modified algorithm is illustrated in Fig. 2, and detailed simulation results are given in



■ **Figure 3.** *Delay and loss differentiation with absolute QoS bounds: a) loss rate: ALC; b) packet delay ratio: ALC; c) loss rate ratio: ADC; d) packet delay: ADC.*

the next section to show the performance of this joint packet scheduling and dropping algorithm.

Simulation Results — In this set of simulations, we use a relatively heavy traffic load (90–110 percent) to emulate the situation on a congested link where our scheme is supposed to take effect. An effective proportional differentiation provisioning scheme should adapt to load fluctuation quickly. Thus, we increase the traffic load from 100 to 110 percent after 30,000,000 packet arrivals and decrease it to 90 percent after 60,000,000 packet arrivals.

Suppose packets have the same size (various size packets will lead to similar results) normalized to be 1 over the transmission speed. The buffer size is 200. Packet interarrival time follows Pareto distribution. There are four service classes that share equal portions in the total traffic load. We choose the proportional factors for packet loss rate as 1:2:4:8; for delay the factors are 1:4:16:64. The packet loss rate and delay are measured every 100,000 packet arrivals.

At first, we put an absolute loss (rate) constraint (ALC) on class 1 of 0.001. The simulation results are shown in Fig. 3a and b. The loss rate for class 1 is kept at 0.001 most of the time, while the proportional differentiations are kept among classes 2–4. At the same time, the WTP scheduler guarantees proportional delay differentiation. However, we notice that the proportional ratios between class 1 and other classes do not approximate proportional factor ratios well. This indicates that in order to fulfill the absolute loss rate bound for class 1, proportional differentiation constraints over class 1 are relaxed. When traffic load is 90 percent, there is no packet loss due to enough buffer size being provided.

Then we show the joint algorithm's performance in Fig. 3c and d when there is an absolute delay constraint (ADC) on class 1 of 75. The average delay for class 1 is kept strictly under 75 while the proportional differentiation constraint is kept when they do not contradict the absolute constraint. Instead of dropping too many class 1 packets, the modified scheme obtains propor-

Instead of dropping too many class 1 packets, the modified scheme obtains proportional differentiation on packet dropping. Note that besides being able to guarantee absolute QoS bounds, this scheme is also adaptive to load fluctuations due to active counter resetting.

tional differentiation in packet dropping. Note that besides being able to guarantee absolute QoS bounds, this scheme is also adaptive to load fluctuations due to active counter resetting.

Open Issues and Conclusion

Feasibility of Differentiation

Given a certain traffic load and a set of differentiation factors, average packet delay proportional to the factors might not be achieved [12]. When it is impossible to achieve service differentiation according to proportional factors, we call it infeasibility. Algorithms [9, 12] were proposed to test feasibility or compute the feasible proportional factor set with statical traffic load on a single hop. Fast algorithms for computing feasible solutions with real-time traffic load input and in a multihop scenario are still under investigation.

Resource Provisioning

On the other hand, with traffic load and composition, how to provide enough link capacity and allocate resources to each class appropriately in order to fit the QoS requirement is studied in [9] in a single-hop scenario. How to efficiently provide resources to achieve network-wide proportional differentiation remains an open problem.

Summary

Recently, the proportional differentiated service model has received attention because it is controllable, consistent, and scalable. This article covers recent research work in proportional QoS provisioning on various QoS metrics. Different approaches to providing absolute QoS guarantee are presented. Several open research issues are pointed out as well.

References

[1] C. Dovrolis and D. Stiliadis and P. Ramanathan, "Proportional Differentiated Services: Delay Differentiation and Packet Scheduling," *Proc. ACM SIGCOMM*, 1999, pp. 109–20.

[2] C. Dovrolis and P. Ramanathan, "Proportional Differentiated Services, Part II: Loss Rate Differentiation and Packet Dropping," *Proc. IWQoS*, 2000, pp. 52–61.

[3] S. Bodamer, "A New Scheduling Mechanism to Provide Relative Differentiation for Real-Time IP Traffic," *Proc. GLOBECOM*, 2000, vol. 1,pp. 646–50.

[4] T. Quynh *et al.*, "Relative Jitter Packet Scheduling for Differentiated Services," *Proc. 9th IFIP Conf. Perf. Modeling and Eval. of ATM & IP Networks*, 2001.

[5] Chin-Chang Li *et al.*, "Proportional Delay Differentiation Service Based on Weighted Fair Queuing," *Proc. IEEE Int'l. Conf. Comp. Commun. and Nets.*, 2000, pp. 418–23.

[6] J. Liebeherr and N. Christin, "Rate Allocation and Buffer Management for Differentiated Services," *Comp. Nets.*, vol. 40, no. 1, Sept. 2002, pp. 89–110.

[7] A. Kumar, J. Kaur, and H. M. Vin, "End-to-End Proportional Loss Differentiation," Tech. rep. TR-01-33, Univ. of TX Austin, Sept. 2001.

[8] T. Nandagopal *et al.*, "Delay Differentiation and Adaptation in Core Stateless Networks," *Proc. INFOCOM*, 200, vol. 2, pp. 421–30.

[10] W. Wu *et al.*, "Forwarding a Balance between Absolute and Relative: A New Differentiated Services Model," *Proc. IEEE Wksp. High Perf. Switching and Routing*, 2001, pp. 250–54.

[11] Y. Chen *et al.*, "Proportional QoS Provision: A Uniform and Practical Solution," *Proc. ICC*, 2002, vol. 4, pp. 2363–67.

[12] M. K. H. Leung, J. C. S. Lui, D. K. Y. Yau, "Adaptive Proportional Delay Differentiated Services: Characterization and Performance Evaluation," *IEEE/ACM Trans. Net.*, vol. 9, no. 6, Dec. 2001, pp. 801–17.

[13] U. Bodin, A. Jonsson and O. Schelen, "On Creating Proportional Loss-rate Differentiation: Predictability and Performance," *Proc. IWQoS*, June 2001.

[14] J.-S. Li and H.-C. Lai, "Providing Proportional Differentiated Services Using PLQ," *Proc. GLOBECOM*, 2001, vol. 4, pp. 2280–84.

[15] T. Quynh *et al.*, "The Influence of Proportional Jitter and Delay on End to End Delay in Differentiated Service Network," *Proc. IEEE Int'l. Symp. Net. Comp. and Apps.*, 2001.

Biographies

YANG CHEN (yangchen@acsu.buffalo.edu) received his B.S. degree in electrical engineering from Xi'an Jiaotong University, China, in 1999 and his M.Phil. degree in electrical and electronic engineering from Hong Kong University of Science and Technology. He is currently a Ph.D. candidate in the Department of Computer Science and Engineering, State University of New York at Buffalo (SUNY Buffalo). His research interests include performance analysis, resource management, and service provisioning in optical networks.

MOUNIR HAMDI [M] (hamdi@cs.ust.hk) received his B.S. degree (with distinction) in computer engineering from the University of Southwestern Louisiana in 1985, and M.S. and Ph.D. degrees in electrical engineering from the University of Pittsburgh in 1987 and 1991, respectively. He has been a faculty member in the Department of Computer Science at Hong Kong University of Science and Technology since 1991 where he is now associate professor of computer science and director of the Computer Engineering Program. In 1999 and 2000 he held visiting professor positions at Stanford University and the Swiss Federal Institute of Technology. His general areas of research are networking and parallel computing. He has guided more than 10 M.S./Ph.D. students in this area of study. Currently, he is working on high-speed networks including the design, analysis, scheduling, and management of high-speed switches and routers, and WDM networks and switches. He is a member of ACM.

CHUNMING QIAO [M] (qiao@computer.org) is currently an associate professor at SUNY Buffalo, where he directs the Laboratory for Advanced Network Design, Evaluation and Research (LANDER). He has published more than 100 papers in leading technical journals and conference proceedings, and is recognized for his pioneering research on optical Internet and, in particular, the optical burst switching (OBS) paradigm. His work on integrated cellular and ad hoc networking systems (iCAR) is also internationally acclaimed. He is an editor of several journals and magazines including *IEEE/ACM Transactions on Networking* and the IEEE Optical Communications Supplement of *IEEE Communications Magazine*, as well as a guest editor for several issues of *IEEE JSAC* and other publications. He has chaired and co-chaired many conferences and workshops including the Symposium on Optical Networks at ICC '03 and Opticomm '02. He is also the founder and chair of the Technical Group on Optical Networks sponsored by SPIE, and a vice chair of the IEEE Technical Committee on Gigabit Networking.

DANNY TSANG [SM] (eetsang@ee.ust.hk) received a B.Sc. degree in mathematics and physics from the University of Winnipeg, Canada, in 1979, and B.Eng. and M.A.Sc. degrees in electrical engineering from the Technical University of Nova Scotia, Halifax, Canada, in 1982 and 1984, respectively. He also received a Ph.D. degree in electrical engineering from the University of Pennsylvania, Philadelphia, in 1989.He joined the Department of Mathematics, Statistics and Computing Science, Dalhousie University, Halifax, Canada, in 1989, where he was an assistant professor in the computing science division. He joined the Department of Electrical and Electronic Engineering of Hong Kong University of Science and Technology in summer 1992, where he is currently an associate professor. His current research interests include congestion controls in B-ISDN/ATM networks, wireless ATM, Internet QoS and applications, application-level multicast and QoS, and optical networks. He served as a member of the Technical Program Committee for IEEE INFOCOM from 1994 to 1996. He was General Chair of IFIP Broadband Communications '99 in Hong Kong. He also received the Outstanding Paper from Academe Award at the IEEE ATM Workshop '99. He is currently an Associate Editor for *OSA Journal of Optical Networking*. During his leave from HKUST in 2000–2001, he assumed the role of principal architect at Sycamore Networks in the United States. He was responsible for the network architecture design of Ethernet MAN/WAN over SONET/DWDM networks. He invented 64B/65B encoding and also contributed to the 64B/65B encoding proposal for transparent GFP in the ITU T1X1.5 standard.