A Robust Packet Scheduling Algorithm for Proportional Delay Differentiation Services

Jianbin Wei and Cheng-Zhong Xu
Wayne state University, Detroit, Michigan 48202

Xiaobo Zhou
Univ. of Colorado, Colorado Springs, CO 80933

***Abstract*— Proportional delay differentiation (PDD) model is an important approach to relative differentiated services provisioning on the Internet. It aims to maintain pre-specified packet queueing-delay ratios between different classes of traffic at each hop. Existing PDD packet scheduling algorithms are able to achieve the goal in long time-scales when the system is highly utilized. This paper presents a new PDD scheduling algorithm, called *Little's average delay* (LAD), based on a *proof* of Little's Law. It monitors the arrival rate of the packets in each traffic class and the cumulative delays of the packets and schedules the packet according to their transient queueing properties in order to achieve the desired class delay ratios in both short and long time-scales. Simulation results show that LAD is able to provide predictable and controllable services in various system conditions and that such services, whenever feasible, can be guaranteed, independent of the distributions of packet arrivals and sizes. In comparison with other PDD scheduling algorithms, LAD can provide the same level of service quality in long time-scales and more accurate and robust control over the delay ratio in short time-scales. In particular, LAD outperforms its main competitors significantly when the desired delay ratio is large.**

I. Introduction

The past decade has seen an increasing demand for provisioning of different levels of quality of service (QoS) on the Internet to support different types of network applications and different user requirements. To meet this demand, two service architectures have been proposed: Integrated Services (IntServ) [3] and Differentiated Services (DiffServ) [2]. By reserving routing resources along the service delivery paths, IntServ is able to provide guaranteed service quality. In contrast, DiffServ aims to provide differentiated services among classes of aggregated traffic flows within a router. Two different schemes exist for DiffServ: Absolute DiffServ and relative DiffServ. Absolute DiffServ aims to guarantee a class's received resource, such as bandwidth. Relative DiffServ is to quantify the quality spacing between different classes.

Recently, Dovrolis, *et al.* defined a proportional delay differentiation (PDD) model in support of relative DiffServ [4], [5]. It ensures the quality spacing between classes of traffic to be proportional to certain pre-specified class differentiation parameters. Since then, many packet scheduling algorithms have been developed to implement the PDD model. Representatives of the PDD algorithms include backlog-proportional rate (BPR) [4], joint buffer management and scheduling (JoBS) [8], proportional average delay (PAD) [5], waiting-time priority (WTP) [5], adaptive WTP [7], hybrid proportional delay (HPD) [5], and mean-delay proportional (MDP) [9]. They demonstrated various characteristics in support of the PDD model in different class load conditions and different time-scales. Most of them are capable of achieving desired delay ratios, if the ratios are feasible, under heavy load conditions and in long time-scales. However, for light load conditions and in short time-scales, they exhibit various limitations. We shall compare them with our algorithm in Section IV.

In this paper, we present a new PDD algorithm, called Little's average delay (LAD), based on a proof of Little's Law. Little's Law regarding a queueing system states the *stationary* relationship between queue length, arrival rate, and queueing delay on average in the long run. Its proof reveals a *transient* property regarding the queueing length [10]. That is, the queueing length of a class at any time is equal to the product of the traffic arrival rate and the waiting time of backlogged packets, plus the experienced delay of departed packets. Accordingly, LAD monitors the average arrival rate of every traffic class and the queueing delay of arrived packets, including both the waiting packets in the queue and departed packets, for the purpose of controlling the delay ratio in both long and short time-scales.

Simulation results show that LAD overcomes the limitations of its main competitors: AWTP, HDP, and MDP. Specifically, whenever the PDD model of a desired class delay ratio is feasible, LAD is capable of providing more accurate and robust control over the delay ratio than its competitors in short time-scales. The improvement is significant when the desired delay ratio is large. In long time-scales, LAD performs no worse than its competitors under any load conditions. Moreover, the performance of LAD is independent of the distributions of packet arrivals and packet sizes because of the generality of Little's Law.

The remainder of the paper is organized as follows. Section II gives an overview of the PDD model and a brief review of the existing PDD algorithms. Section III presents the LAD algorithm and discusses its design and implementation issues. Section IV evaluates the algorithm via extensive simulation and compares it with other PDD algorithms. We conclude this paper in Section V.

II. Background and Related Work

We consider packet scheduling of a lossless, work-conserving, and non-preemptive link that services M ($M \geq 2$) first-come-first-served (FCFS) queues, one for each traffic class. The lossless property requires that the average arrival

rate of the aggregate traffic must be less than the link capacity and that there is enough queueing space to buffer backlogged packets. The work-conserving property is that the link is never left idle as long as there are backlogged packets waiting for service in the queues. The non-preemptive property requires the transmission of a packet cannot be interrupted. The aggregate traffic of the queueing system is determined by the superposition of the M traffic streams.

The objective of the PDD model is to control the quality spacing between different classes so that their average delay ratios be proportional to certain class differentiation parameters pre-defined by network operators. Let W_i denote the average delay of class i, and δ_i the pre-defined delay differentiation parameter. The PDD model requires to ensure that for any two classes i and j, $1 \leq i, j \leq M$,

$$\frac{W_i}{W_j} = \frac{\delta_i}{\delta_j}. \tag{1}$$

Notice that the PDD model is not always feasible. The upper bound of feasible delay ratio for a G/G/1 system can be estimated using a strict priority based scheduling algorithm [5].

The PDD model requires the differentiated services be predictable and controllable in the sense that network operators should be able to adjust the service quality spacing between any two classes by setting delay differentiation parameters and that the average delay ratios of different classes be consistent with their delay differentiation parameters in both long and short time-scales. Such consistency should also be maintained for individual packets departed successively from different classes. In addition, the service differentiation should be independent of class load traffic characteristics. Regardless of the distributions of packet arrivals and sizes, the consistency should be maintained whenever the PDD model is feasible.

Many packet scheduling algorithms have been proposed for PDD service model. Rate-based algorithms, as exemplified by BPR [4] and JoBS [8], adjust service rate allocations of classes dynamically to meet the proportional delay differentiation constraints. However, for accurate rate allocation, the system should operate under high load conditions, this limits the applicability of the rate-based PDD algorithms. In contrast, our algorithm has good performance in various load conditions.

Time-dependent priority based algorithms adjust the priority of a backlogged class according to the experienced delay of its head-of-line packet. In WTP, the priority of a backlogged class is adjusted to be proportional to its head-of-line packet's delay normalized with respect to its delay differentiation parameter. Albeit simple, WTP implements the PDD model only when the system utilization approaches unity[4]. To overcome such limitation, adaptive WTP adjusts the priority of a class not only according to its experienced delay, but also based on the current class load condition. We find out that such adjustment is valid for certain network traffic with small degree of self-similarity. In contrast, the performance of our algorithm is independent of network traffic characteristics and load conditions.

Some algorithms determine the next packet according to the average queueing delay of backlogged classes. It is known that at time t, arrived packets of a class in a time window $[t-\tau, t]$, can be in one of the two states: departed or waited in the queue. PAD considers the average delay of departed packets in the time window only. It is capable of achieving the PDD model constraints in various load conditions. However, PAD exhibits a pathological behavior in short time-scales; that is, occasionally higher classes to experience larger delays than lower classes, which is caused by its ignorance of those backlogged packets. To address this issue, HPD was proposed to take into account the average delay of departed packets, and the delay of the head-of-line packet simultaneously. HPD enhances the average control quality of PAD, and meanwhile avoids its pathological behavior problem. However, in Section IV, we will show that HPD achieves the class delay ratio with large statistical variations in short time-scales. MDP considers the delay of all arrived packets of each class in a time window $[t-\tau, t]$. In addition, it also takes into account the estimated delay of backlogged packets when they are departed. In Section IV, we shall show that MDP delivers performance comparable to HPD. However, its performance deteriorates as the target quality spacing between the classes is enlarged. Note that PAD, HPD and MDP schedule backlogged packets of different classes based on heuristic delay information of arrived packets. LAD presented in this paper is based on a proof of Little's Law [10]. It considers the delay of departed packets as well as the delay of the packets in the backlogged queue in the time window $[t-\tau, t]$.

III. Little's Average Delay Algorithm

A. Little's Law

For a G/G/1 queueing system, Little's Law states that the average number of packets in the system is equal to the product of average arrival rate of packets and the average waiting time of the packets in the system. Define $L(T)$ as the average number of the packets in the system during the time interval $[0, T]$, $W(T)$ as the waiting time per packet averaged over all packets, $\lambda(T)$ as the average arrival rate. Suppose $W(T)$ and $\lambda(T)$ have limits as $T \to \infty$, that is

$$W = \lim_{T\to\infty} W(T), \text{ and } \lambda = \lim_{T\to\infty} \lambda(T).$$

Then, the limit of $L(T)$, denoted by L, exists and

$$L = \lambda W. \tag{2}$$

The beauty of Little's Law (2) is that it does not depend upon any particular queueing discipline (packet scheduling algorithms); nor does it depend upon any specific assumptions regarding the packet arrival distribution or the packet size distribution. It is applicable to the queueing system of each traffic class in the PDD model.

LAD algorithm controls the delay ratios between different classes based on the Little's Law. Substituting L/λ for W, the objective of PDD model in (1) leads to a new constraint:

$$\frac{L_i}{\lambda_i \delta_i} = \frac{L_j}{\lambda_j \delta_j}, \tag{3}$$

for any two classes i and j. To ensure proportional delay differentiation between two classes, their normalized queue length with respect to their respective arrival rates and delay differentiation parameters should be kept equal. The LAD algorithm is to control the delay ratio by adjusting their average queueing lengths according to their arrival rates.

Notice that (2) reveals an asymptotic (or stationary) relationship between the queue length, packet arrival rate, and packet waiting time in the system. It is not enough to guide PDD scheduling because the objective of proportional delay needs to meet in small time windows. Because most of Web requests are small in size [1], provisioning of relative delay differentiation service in short time-scales is as important as in long time-scales. LAD algorithm is based on a transient property of the queueing system, as revealed by a proof of the Little's Law [10]. Following is a sketch of the proof.

Suppose that packets $p_1, p_2, \ldots$ arrive at time $t_1, t_2, \ldots (0 \leq t_i < t_{i+1})$, and depart at $t_1^d, t_2^d, \ldots$. The packets are not necessarily forwarded in FCFS discipline. Denote $N(T)$ the total number of arrived packets in the time interval $[0, T]$; $N^d(T)$ and $N^c(T)$ the number of departed packets and the number of waiting packets in queue, respectively. It follows that at time T,

$$N(T) = N^d(T) + N^c(T). \tag{4}$$

Define $I_i(t)$ as the presentation function of packet p_i at time t, that is

$$I_i(t) = \begin{cases} 1, & \text{if packet } p_i \text{ is present at time t;} \\ 0, & \text{otherwise.} \end{cases}$$

Then, we have

$$N^c(T) = \sum_{i=1}^{N(T)} I_i(t). \tag{5}$$

Since packet p_i stays in queue during the interval $[t_i, t_i^d]$ and its queueing delay $w_i = t_i^d - t_i$, we have

$$\int_0^T I_i(t)dt = \begin{cases} w_i, & t_i^d \leq T; \\ T - t_i, & t_i^d > T. \end{cases} \tag{6}$$

Therefore, the cumulative queue length in the interval $[0, T]$ is

$$\begin{aligned} \int_0^T N^c(t)dt &= \sum_{i=1}^{N^d(T)+N^c(T)} \int_0^T I_i(t)dt \\ &= \sum_{\{i:t_i^d \leq T\}} w_i + \sum_{\{i:t_i \leq T, t_i^d > T\}} \big(T - t_i\big), \end{aligned} \tag{7}$$

and the average queue length in interval $[0, T]$ is

$$L(T) = \frac{1}{T}\int_0^T N^c(t)dt = \lambda(T)W(T), \tag{8}$$

where

$$\lambda(T) = \frac{N(T)}{T}, \tag{9}$$

$$W(T) = \frac{\sum_{\{i:t_i^d \leq T\}} w_i}{N(T)} + \frac{\sum_{\{i:t_i \leq T, t_i^d > T\}} \big(T - t_i\big)}{N(T)}. \tag{10}$$

Assume that $\lambda(T)$ and $W(T)$ exist as $T \to \infty$, (8) leads to that

$$L = \lim_{T\to\infty} \lambda(T)W(T) = \lambda W. \tag{11}$$

This completes the proof.

B. The LAD Algorithm

The basic idea of LAD algorithm is to control the delay ratio of classes by monitoring their arrival rates and queueing delays of their arrived packets based on transient relationship between the queue length, arrival rate and waiting time, as revealed by (8). In particular, (10) defines the average waiting time per packet in a window of size T. The numerator of the first term actually represents the accumulated delays of all departed packets and the numerator of the second term represents the accumulated waiting time of the packets in the backlogged queue so far at time T. Accordingly, we define the LAD algorithm as follows.

For class i, the LAD scheduler maintains three control variables to monitor its traffic flow over finite time window T: the cumulative delays of departed packets W_i^d; the number of arrived packets N_i; and current queue length N_i^c. At the beginning of each time window, these variables are (re)initialized. Note that the size of T is in terms of number of successively departed packets from the system. These control variables are updated according to the following rules:

1) At the beginning of each time window, $N_i \leftarrow N_i^c$ and $W_i \leftarrow 0$.
2) Upon the receipt of a packet of class i, the packet is timestamped and $N_i \leftarrow N_i + 1$, and $N_i^c \leftarrow N_i^c + 1$.
3) After transmitting a packet of class i, $N_i^c \leftarrow N_i^c - 1$ and $W_i^d \leftarrow W_i^d + w$, where w is the measured delay of the packet.

Let W_i^c denote the current cumulative delay of backlogged packets in the queue i. According to (10), we set the priority of class i as

$$\mathcal{P}_i = \frac{W_i^d + W_i^c}{N_i \delta_i}. \tag{12}$$

Whenever the queueing system is available for packet transmission, a backlogged packet of class j^* with the highest priority is selected. That is,

$$j^* = \arg \max_{1 \leq i \leq M} \mathcal{P}_i. \tag{13}$$

Ties for the highest priority are broken by serving the packet that has entered the queueing system earliest. Note that the validity of Little's Law does not depend upon any particular queueing discipline. Therefore, the next packet can be any backlogged packet if a more complicated scheduling algorithm is needed.

There are some important issues in the implementation of the LAD algorithm. The foremost is the time window size T. It is known that Little's Law is valid when the time window is sufficiently large. However, provisioning PDD services in short time-scales is as important as in long time-scales. A good choice of T should strike a balance between system stability

and responsiveness. On one side, a large T would avoid abrupt changes of average queueing delay due to bursty traffic. Particularly, when T is sufficiently large, the average delay of the packets in the time window would hide the effect of the distributions of packet arrivals and packet sizes. On the other side, a small T would lead to an agile scheduler that responds to the change of traffic conditions quickly. Although we leave T to be an adjustable parameter by network operators, in our simulations, we show that LAD is able to provide PDD services in both long and short time-scales.

Another important implementation issue is the calculation of the cumulative delay of backlogged packets in each queue W_i^c. It is too costly to scan each queue to re-calculate W_i^c every time when a packet is to be transmitted and the priority of each class needs to be adjusted. Instead, we calculate W_i^c recursively in the following way. Suppose that at time u when the last transmitted packet was selected from class i , the class has m backlogged packets $p_1, p_2, \ldots, p_m$ and their arrival times are $t_1, t_2, \ldots, t_m$, respectively. Since the queueing system assumes no FCFS scheduling principle, the next packet to be selected for forwarding from class i can be any packet in the queue. Without loss of generality, we assume packet p_k is forwarded at time $u+\tau$, that is, the time interval between two successive packet departures is τ. Suppose there are n new packet arrivals during the interval and their arrival time are $t_{m+1}, t_{m+2}, \ldots, t_{m+n}$. It follows that

$$W_i^c(u+\tau) = W_i^c(u) - (u - t_k) + (m-1) \times \tau + \sum_{j=1}^{n} (u + \tau - t_{m+j}). \quad (14)$$

Recall that the traffic of class i has an arrival rate of λ_i. During the interval of τ, the average number of packet arrivals is $\lambda_i \tau$. Note that $E[\tau]$ is the average service time of a packet. For a stable system, it should be less than or equal to the average inter-arrival time. Thus, for $E[n]$, the average number of packets entering into the system during τ, we have $E[n] = \sum_{i=1}^{N} E[n_i] \leq 1$. Therefore, the main computation overhead of the updating is the multiplication, which is appropriate in real environment [5].

For each packet transmission, LAD needs to calculate and compare the priorities of all backlogged classes, which requires at most N calculations and $N-1$ comparisons. The calculation overhead is mainly due to the update of control variables and timestamping operations. The cost for update is small because it involves only a few number of add operations; the timestamping operation is assumed in the implementation of WTP and MDP as well.

IV. Simulation Results

In this section, we present simulation results of LAD to demonstrate its performance and properties. We also compare LAD with other PDD algorithms, including WTP, AWTP, PAD, HPD, and MDP. A primary performance metric is error between desired class delay ratio and achieved ratio. The results are an average of 1000 runs.

The experiments assumed the distributions of packet arrivals and sizes are similar to those in [5], [7]. That is, the inter-arrivals between packets of a class follow a Pareto or Poisson distribution. The packets size are variable with a small number of choices. The transmission time of a packet is proportional to its size.

A. Predictability of LAD

We investigated the predictability of LAD in experiments over three classes of Pareto distributed traffic ($\alpha = 1.5$). Their delay differentiation parameters ($\delta_1, \delta_2, \delta_3$) were set to $(4, 2, 1)$ and the class load distribution ($\lambda_1, \lambda_2, \lambda_3$) varies between $(1, 1, 1)$, $(1, 2, 4)$ and $(4, 2, 1)$.

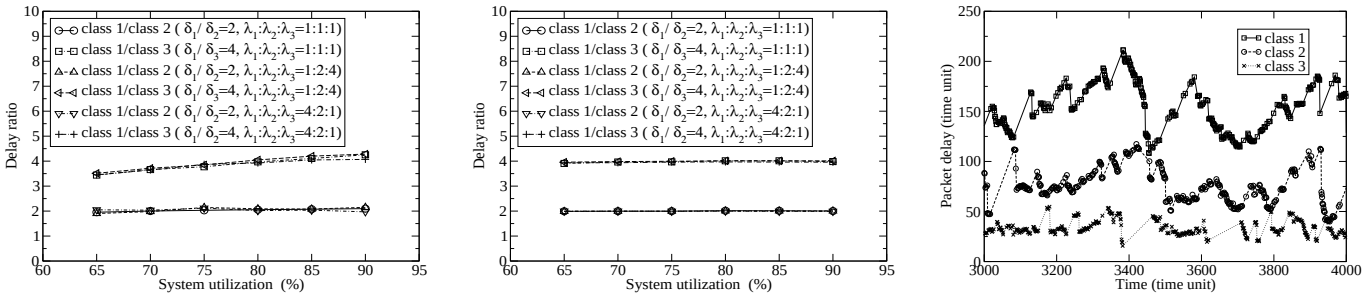
We obtained the simulations results in short ($T = 100$), moderate ($T = 1000$), and long ($T = 10000$) time-scales, as the system utilization rate ρ varies. Due to space limitation, Fig. 1 shows the results of short and long time-scales. In short time-scales, we can observe that the resulted error between achieved and desired delay ratios under moderate system utilization rates is larger than that under high system utilization. Such error is negligible in long time-scales. This is mainly because the system utilization difference in short and long time-scales. Although the experiment assumed stable system utilization rates in the long run, the system utilization rates were hardly maintained accurately in the short run. Due to the burstiness of the Internet traffic, the system transient utilization rates in short time windows were often lower than the stationary rates. That means even when the desired delay ratio can be achieved in long time-scales, it maybe infeasible in short time-scales. Such behavior can be observed when the system utilization is 65% and 80%.

Note that the feasibility of LAD is affected by the system utilization rate does not mean that LAD has requirements for an accurate estimation of the system utilization rate. It can be seen from the results from the setting of class delay ratio of 2 in Fig. 1. As long as that the utilization is high enough for the PDD ratio to be feasible, LAD can achieve ratio in both short and long time-scales, independent of class load distributions.

To investigate the behaviors of individual packets from different classes, we plot in Fig. 1(c) the individual delays of packets departed from 3000th to 4000th time unit and the system load is 90%. All classes had the same load (*i.e.,* $\lambda_1 = \lambda_2 = \lambda_3$). From this figure, we can see that individual packets of a higher class exhibit smaller changes of delay than those of a lower class in both dimensions of time (x axis) and delay (y axis). More importantly, the packets from higher class always have smaller delay than those from lower class. In other words, LAD can provide predictable differentiated service.

B. Controllability of LAD

We studied the controllability of LAD through an experiment over two classes of traffic with an equal load distribution (*i.e.,* $\lambda_1 = \lambda_2$). Fig. 2 plots the achieved class delay ratios in long time-scales, in comparison with the desired delay ratios as the system utilization rate changes. The results in short time-scales are ignored since there is no significant difference.



(a) $T = 100$ packets. (b) $T = 10\,000$ packets. (c) Individual packet behavior.

Fig. 1. Delay ratios of three classes in different system utilizations and time-scales.

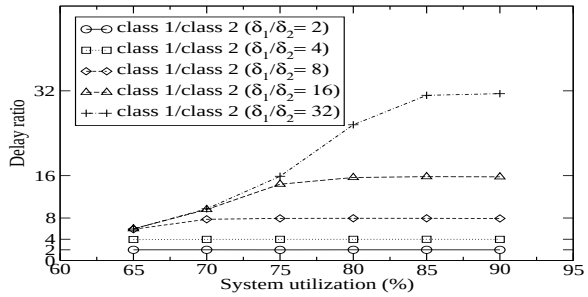


Fig. 2. Delay ratios of class 1 to class 2 in different system utilizations.

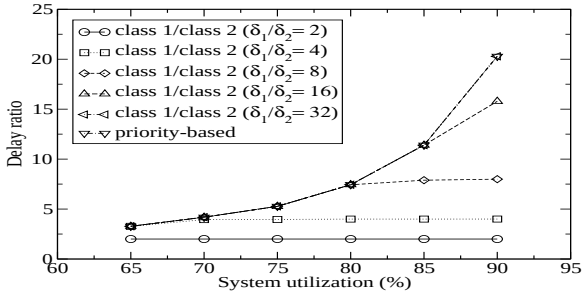


Fig. 3. Delay ratios of class 1 to class 2 using LAD and priority-based packet scheduling algorithms.

From this figure, it can be observed that LAD is capable of achieving the target delay ratio of 2 under medium or high system utilization rates. It shows that when the desired ratio goes up to 8, LAD cannot meet the PDD constraints in the long time-scale, unless the system utilization rate is higher than 70%. Aforementioned, this is because the ratio is infeasible when $\rho \leq 70\%$ in the long timescale. Whenever the ratio is feasible, the service difference between these two classes can be controlled accurately by network operators. With the increase of the desired class delay ratio δ_1/δ_2, the minimum system utilization rate that makes the given delay ratio feasible increases.

C. Generality of LAD

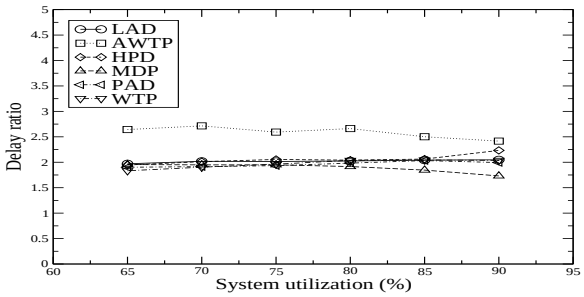
Recall that the generality of a PDD algorithm means that its performance should be independent of the distributions of packet arrivals and sizes. We studied the generality of LAD through experiments over two classes with equal class loads in the long timescale. In addition, we assumed that the packet arrivals of the same class followed a Pareto or a Poisson distribution and that the packet sizes of each class were variable. The variable packet sizes were set in the same pattern as in [4].

We carried out the experiments for traffic with different packet arrival distributions: Pareto, Poisson, and mixed distributions. Due to space limitation, we only illustrate the results for mixed distributions. In the mixed arrival distribution, we assumed packets of class 1 followed a Poisson distribution and packets of class 2 are Pareto distributed. To measure the actual feasible range for the same packet stream, the results due to the strict priority-based algorithm are included, as well.

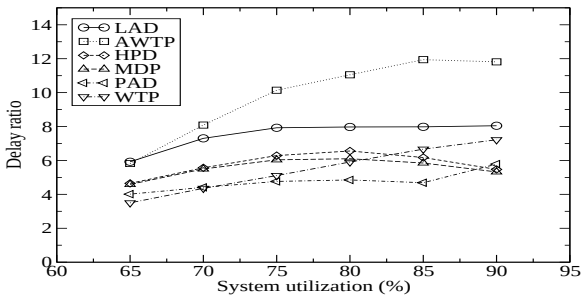
From Fig. 3, we can observe that the feasible delay ratio increases with the system utilization rate and that whenever the desired ratio is feasible, it can be achieved by LAD accurately; otherwise, the achieved ratio by LAD is very close to that measured by the strict priority-based algorithm. For example, Fig. 3 shows that the maximum achievable delay ratio is 22 when the system utilization rate $\rho = 85\%$. Although the desired ratio of 32 is infeasible, LAD achieves the maximum feasible ratio.

D. Comparison with Other PDD Algorithms

We compared LAD with other PDD algorithms, including WTP, AWTP, PAD, HPD, and MDP. In the experiments, we assumed two classes of traffic with the equal class loads. The packet arrivals of each class followed a Pareto distribution ($\alpha = 1.5$) and all the packets had equal size. We generated a stream of packets beforehand and assumed the same packet stream for all the experiments with different PDD algorithms. Recall that AWTP adjusts the feasible set of control parameters according to the delay of the head-of-line packet in each class and the system utilization. Our implementation of AWTP used the jumping window method, as suggested in [7], to estimate the arrival rate of traffic. HPD is a hybrid of WTP and PAD with a weighting parameter g. We set the parameter g to 0.875 as recommended in [5]. MDP takes into account the delay of departed packets and the estimated delay of all other waiting packets in the determination of class priorities. Although the MDP authors suggested a simplified method to approximate



(a) $\delta_1/\delta_2 = 2$.



(b) $\delta_1/\delta_2 = 8$.

Fig. 4. Delay ratios of class 1 to class 2 using different PDD algorithms in different system utilizations.

the average delay for all arrived packets to make a tradeoff between quality and run-time overhead [9], we implemented its original version in this experiment.

1) Comparison in short time-scales: We first compared the short timescale performance of the algorithms under different system utilization rates. The time window was set to $T = 100$ packets. The simulation results for the cases of $\delta_1/\delta_2 = 2$ and 8 are plotted in Fig. 4(a) and Fig. 4(b), respectively.

Fig. 4(a) shows that all the PDD algorithms, except AWTP, can meet the PDD constraints to an acceptable extent for a small delay ratio under moderate and high system load conditions. In particular, LAD achieves the desired delay ratio with minimum errors consistently. In contrast, HPD and MDP demonstrate good performance under moderate load conditions, but yield relatively large errors when the system utilization rate goes up to as high as 90%. Recall that HPD is a hybrid of WTP and PAD. Both WTP and PAD gain performance as the utilization rate increases, but their improvement rates are different. Hence, a linear combination of the WTP and PAD with a constant weighting parameter g in HPD is expected to generate a convex performance plot with respect to the utilization rate. This impact of linear combination can be seen more clearly in Fig. 4(b) for the case of a large desired delay ratio.

The reason for the inaccuracy of MDP in highly utilized systems is the estimation error of the delays of backlogged packets in a time window of $[t-\tau, \infty)$ at any time t. Although MDP can measure the delay of packets in the time window $[t-\tau, t]$, MDP uses a lower bound to estimate the delay of the packets in future $[t, \infty)$. With the increase of the system

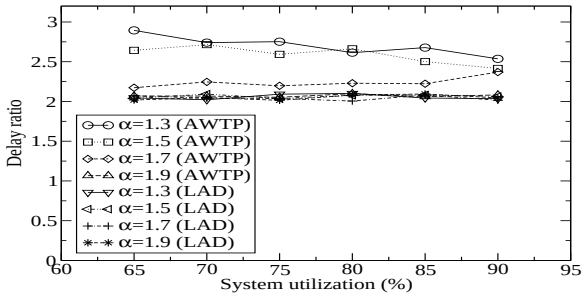


Fig. 5. Impact of scaling parameter α on AWTP when applied to Pareto distributed traffic. $\delta_1/\delta_2 = 2$.

utilization, there are more packets in a backlogged queue during the interval τ, and consequently the estimation error increases. When the system utilization rate goes beyond certain point, the impact of estimation accuracy becomes significant and the overall performance of MDP starts to deteriorate. Fig. 4(b) shows that the estimation error is exaggerated in the case of a large desired delay ratio and the gap between LAD and MDP is enlarged.

Fig. 4 shows that WTP yields relatively large errors when the system utilization rate is moderate. This is consistent with the findings of other researchers [5], [7]. AWTP was proposed as a remedy of this problem [7]. It relies on a policy iteration algorithm to adjust the feasible set of control parameters according to the delay of the head-of-line packet in each class and the class load distributions. The algorithm is based on an assumption that the arrival process of each traffic class is a Poisson distributions. The authors showed that AWTP be applicable to the traffic of a *Pareto* distribution with the shape parameter $\alpha = 1.9$.

We note that the shape parameter α of a Pareto distribution characterizes the degree of self-similarity and burstiness of network traffic [6]. The larger α, the less bursty and self-similar behaviors were observed in trace studies. Given the fact that $0 < \alpha < 2$, the Pareto distribution with $\alpha = 1.9$ bears much resemblance to a Poisson distribution. Fig. 4 shows the results from a Pareto distribution with $\alpha = 1.5$. We experimented with both AWTP and LAD for more Pareto distributions with various α and plotted the results in Fig. 5. From this figure, we observe that the control parameters of AWTP are unable to meet the PDD constraints over general Pareto distributed traffic. By contrast, LAD is insensitive to the Pareto distribution shape.

2) Comparison in long time-scales: We compared LAD with other PDD algorithms, focusing on their robustness in different timescales. The experiment settings remain the same as in the last one, except that the system utilization rate is fixed at 90%. Fig. 6(a) and Fig. 6(b) show three percentiles (the 5th, 50th, and 95th) of achieved delay ratios for the target ratio of 2 and 8, respectively. We give the numbers in the figures directly for some of the large percentiles.

Fig. 6 shows that LAD achieves the target ratios accurately in all of the timescales that we tested and outperforms its competitors consistently in terms of the errors in various

percentiles. This implies that LAD is more robust to keep the class delay ratio under control and deliver the desired ratio with small statistical variations. Although all the algorithms are able to meet the PDD constraints in terms of their medians with small deviations in long timescales, LAD is outstanding to provide tight and robust control in a statistical sense over the class delay ratio in short timescales.

Fig. 6(a) shows that all the PDD algorithms, except MDP, are able to achieve the delay ratio of 2 with a high probability in the short timescale of 100 packets. LAD demonstrates an excellent robustness because more than 90 percentage of the total runs would produce ratios between 1.6 and 2.4. MDP is robust, as well, but its achieved ratios center around 1.6. In contrast, AWTP, HPD, and PAD exhibit a "heavy tail" property in that majority of the runs, under the control of the algorithms, would lead to delay ratios that are close to the target ratio of 2, but the algorithms could lose the control in a few occasions.

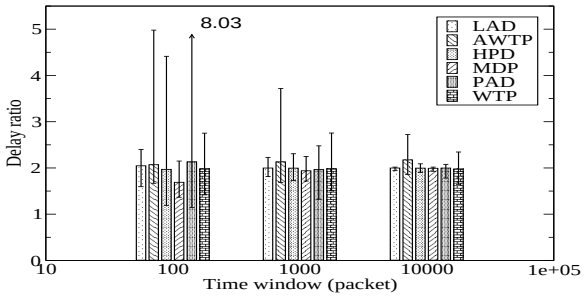
Fig. 6(a) also shows that the success probability of the algorithms increases with the time scale. In the long timescale of 10000 packets, all the algorithms are able to achieve the target delay ratio robustly.

In comparison with Fig. 6(b), we observe that all the PDD algorithms lose certain degrees of robustness when the desired delay ratio δ_1/δ_2 is large. In the short timescale of 100 packets, LAD performs slightly better than WTP and AWTP, but outperforms PAD, HPD and MDP significantly in terms of their medians. The goodness of WTP and AWTP are mainly due to the high utilization ratio (90%) that we assumed in this experiment. WTP and AWTP provide consistent levels of QoS, independent of the desired delay ratio. This is because they use extra control parameters to adjust the impact of the predefined delay ratio. But they are lack of robustness because of their medians with large statistical variations. PAD, HPD and MDP perform in a similar way to LAD. They differ in the way of delay estimation of arrived packets. Fig. 6(b) shows that their performance gap in short timescales gets larger as the delay ratio increases. As the timescale increases, all the PDD algorithms gain more control over the delay ratio. In the long timescale of 10000 packets, LAD provides similar levels of QoS to HPD and MDP.

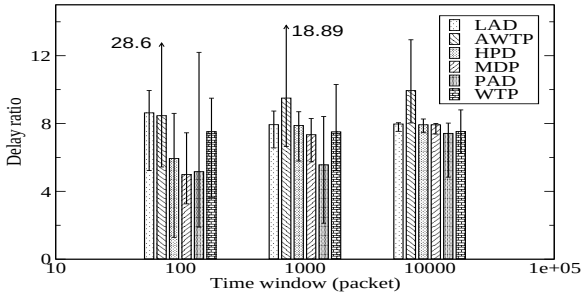
We conclude that in short timescales, LAD consistently outperforms its competitors for large target delay ratios. For small target ratios, most of the algorithms can provide an acceptable level of quality of service. Under heavy load conditions and in long timescales, LAD performs similarly to HPD and MDP. WTP, AWTP, and PAD are not as robust as the others due to their large statistical variations.

V. Conclusions

We have proposed a new proportional delay differentiation algorithm, called LAD, to implement the PDD model. The algorithm is derived from a proof of Little's Law. It monitors the arrival rate of the packets in each traffic class and their cumulative delays and achieves the desired class delay ratios in both short and long time-scales. Simulation results have shown that LAD is able to meet the PDD constraints, independent of the distributions of packet arrivals and packet sizes. In comparison with other PDD algorithms, LAD provides the same level of service quality in long time-scales and more accurate and robust control over the delay ratio in short timescales. Our future work will focus on the proportional loss differentiation model and combine it with the PDD model. The requirements for absolute QoS, such as the end-to-end delay will also be investigated.



(a) $\delta_1/\delta_2 = 2$.



(b) $\delta_1/\delta_2 = 8$.

Fig. 6. Percentiles of achieved delay ratios using different PDD algorithms in different timescales.

References

[1] M. Arlitt and C. Williamson, "Internet web severs: Workload characterization and performance implications", *IEEE/ACM Trans. on Networking* 5(5), Oct 1997.

[2] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, *An Architecture for Differentiated Services*, RFC 2475, Dec. 1998.

[3] R. Braden, D. Clark, and S. Shenker, *Integrated Services in the Internet architecture: An overview*, RFC 1633, Jun. 1994.

[4] C. Dovrolis, D. Stiliadis, and P. Ramanathan, "Proportional differentiated services: Delay differentiation and packet scheduling," in *Proceedings of SIGCOMM*, 1999, pp. 109–120.

[5] ——, "Proportional differentiated services: Delay differentiation and packet scheduling," *IEEE/ACM Transactions on Networking*, vol. 10, no. 1, pp. 12–26, 2002.

[6] W. Leland, M. Taqqu, W. Willinger, and D. Wilson, "On the self-similar nature of Ethernet Traffic", *IEEE/ACM Transactions on Networking*, vol. 2, no. 1, pp. 1–15, February 1994.

[7] M. K. Leung, J. C. Lui, and D. K. Yau, "Adaptive proportional delay differentiated services: Characterization and performance evaluation," *IEEE/ACM Transactions on Networking*, vol. 9, no. 6, pp. 801–817, December 2001.

[8] J. Liebeherr and N. Christin, "JoBS: Joint buffer management and scheduling for differentiated services," in *Proceedings of IWQoS 2001*, Karlsruhe, Germany, June 2001, pp. 404–418.

[9] T. Nandagopal, N. Venkitaraman, R. Sivakumar, and V. Bharghavan, "Delay differentiation and adaptation in core stateless networks," in *Proceedings of IEEE INFOCOM*, April 2000, pp. 421–430.

[10] S. J. Stidham, "A last word on $L = \lambda W$," *Operations Research*, vol. 22, no. 2, pp. 417–421, 1974.