BPR$^+$: A PACKET SCHEDULING ALGORITHM FOR PROPORTIONAL DELAY DIFFERENTIATION

Vladimir Vukadinović [1], Grozdan Petrović [2], Ljiljana Trajković [1]

[1] ***School of Engineering Science, Simon Fraser University, Vancouver, Canada***

[2] ***Faculty of Electrical Engineering, University of Belgrade, Belgrade, Serbia and Montenegro***

Abstract - *Proportional service differentiation models are simple and effective approaches to service differentiation in Internet. Schedulers for delay differentiation are important elements of these models. We describe a new scheduler for proportional delay differentiation called BPR$^+$: Optimized Backlog-Proportional Rate scheduler. BPR$^+$ considers waiting times of packets that are backlogged in queues in order to make precise scheduling decision. In this paper, we evaluate the performance of BPR$^+$, which is achieved by taking into account complete information regarding backlogged packets.*

1. INTRODUCTION

Internet applications have diverse service requirements. Real time applications, such as IP telephony and video on demand, require low delay and delay jitter while being relatively insensitive to packet losses. Applications dealing with bulk data transfer, such as File Transfer Protocol (FTP) and Simple Mail Transfer Protocol (SMTP), require sufficient throughput and reliability while being tolerant to network delay. Today's best-effort Internet service is not adequate for many applications and users. Considerable Internet research has been devoted recently to providing different levels of service to applications.

Initial proposals for service differentiation, such as Integrated Services (IntServ) model [1], provided strong service guarantees, with absolute per-flow bounds on delays, loss rates, and throughput. They required substantial modification to current Internet architecture. Furthermore, IntServ model raised scalability concerns because maintaining per-flow state imposes large computational and memory requirements on core routers in Internet. The interest has shifted recently to class-based service architectures that provide relative service guarantees to flow aggregates called classes: flows belonging to a higher priority class receive better performance compared to those of a lower priority class.

A service model known as Differentiated Services (DiffServ) [2] has also received considerable attention. Unlike in IntServ model, DiffServ routers employ per-class buffer management and packet scheduling, where the number of classes is significantly lower than the number of flows. There are two approaches for delivering QoS guarantees in DiffServ: *absolute* and *relative* service differentiation. Relative service differentiation is simpler and easier for deployment compared to absolute differentiation. Network administrator may vary quality spacing between traffic classes, such as delay and loss rate ratios.

Recently proposed Proportional Differentiated Services (PDS) model [3], [4] defines a service model where the ratios of loss rates and packet delays between successive priority classes remain approximately constant.

Scheduling and buffer management algorithms have to be designed in order to support relative service differentiation. Packet scheduling specifies the packet serving discipline in a network node. Incoming packets are organized by the buffer management algorithm into different logical queues. After a packet has been transmitted from the buffer, the packet scheduling algorithm decides which queue should be served next. Since QoS parameters of a traffic flow are significantly affected by the choice of scheduling mechanism, considerable research efforts have been focused on designing various scheduling schemes for relative delay differentiation [3]-[10]. However, these schemes proved to be efficient only under certain conditions (heavy traffic load, sufficiently large buffers) and only on certain timescales (short or long).

In this paper, we describe a new scheduling mechanism that aims to provide more precise and consistent relative delay differentiation between real-time and non-real-time (elastic) traffic. The new mechanism called Optimized Backlog-Proportional Rate scheduler (BPR$^+$) provides proportional delay differentiation. BPR$^+$ considers waiting times of all packets that are backlogged in queues in order to make a scheduling decision. We searched for an "optimal" scheduler for proportional delay differentiation, even if its implementation may be to complex.

The remainder of the paper is organized as follows: a brief description of existing scheduling mechanisms for proportional delay differentiation is given in Section 2. The Optimized Backlog-Proportional Rate scheduler (BPR$^+$) is described in Section 3. In Section 4, we evaluate the performance of the proposed scheme using various simulation scenarios. Conclusions are presented in Section 5.

2. RELATED WORK

Several schedulers for relative delay differentiation have been recently proposed. We review here two schedulers: Waiting Time Priority (WTP) and Backlog-Proportional Rate (BPR) schedulers [3], [4].

WTP scheduler is based on Kleinrock's Time-Dependent-Priorities (TDP) algorithm [11]. Priority of a packet from queue i at time t is proportional to the waiting time of the packet at time t:

$$p_i(t) = w_i(t)s_i, \tag{1}$$

where s_i is a service parameter of the class i. If head-of-line packets are served according to their priorities, it has been shown that, under heavy traffic load, relative average delay ratio of any two classes i and j is close to s_i/s_j. Hence, average delay ratio between classes i and j can be controlled by setting appropriate service parameters s_i and s_j. Note that scheduling decisions are based only on waiting times of head-of-line packets.

BPR scheduler was originally proposed [12] as Proportional Queue Control Mechanism (PQCM). BPR scheduler assigns class service rates r_i and r_j proportionally to queue lengths q_i and q_j, respectively, and a desired delay differentiation parameter $\delta \geq 1$:

$$\frac{r_i}{r_j} = \frac{1}{\delta}\frac{q_i}{q_j}. \tag{2}$$

If $C = r_i + r_j$ is the link capacity, service rates r_i and r_j can be calculated from (2). According to Little's law, average queuing delays of class i and class j packets are $d_i = q_i / r_i$ and $d_j = q_j/r_j$, respectively. The ratio of these delays approaches δ [3]:

$$\frac{d_i}{d_j} \to \delta. \tag{3}$$

In the case of BPR, scheduling decisions are based only on queue lengths q_i and q_j.

Both WTP and BPR schedulers perform well under heavy traffic load and utilization close to 100 %, when queues are sufficiently long. Their performance is significantly worse under light and medium load conditions. Furthermore, achieved delay ratio depends on traffic load distribution between classes.

3. BPR$^+$: AN OPTIMIZED BPR SCHEDULER

We propose Optimized Backlog-Proportional Rate (BPR$^+$), a new scheduler for proportional delay differentiation. BPR$^+$ assigns service rates to the queues based on waiting times of all backlogged packets. We consider a work-conserving scheduler serving two FCFS queues managed by an active queue management (AQM) algorithm. The two queues are marked as throughput-sensitive (TS) and delay-sensitive (DS). Our goal is to provide a relatively lower delay to delay-sensitive packets compared to throughput-sensitive packets, according to delay differentiation parameter δ specified by a network administrator.

BPR$^+$ is based on a fluid traffic model, with the following characterizations. A server is busy if there are packets in queues waiting to be transmitted. The input traffic curve $R^{in}(t)$ is defined as cumulative amount of traffic that has entered a queue since the current busy period. The output traffic curve $R^{out}(t)$ is the cumulative amount of traffic that has been transmitted from the queue since the current busy period [13], [14]. An example of input and output curves is shown in Fig. 1. At any time, service rate of a queue is equal to the slope of its output curve. Vertical and horizontal distance between input and output curve are current backlog and delay, respectively.

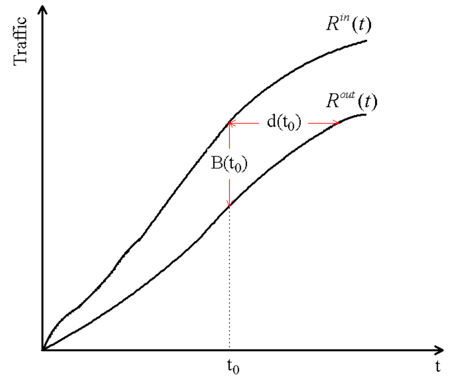

Fig. 1. Example of input and output traffic curves: vertical and horizontal distances between the curves are current backlog and delay, respectively.

Input and output curves for the TS class ($R_{TS}^{in}(t)$ and $R_{TS}^{out}(t)$) and the DS class ($R_{DS}^{in}(t)$ and $R_{DS}^{out}(t)$) are shown in Fig. 2. If the TS queue is served with rate $\dot{R}_{TS}^{out}(\tau)$ at the time τ, the average delay of TS backlog can be calculated as a sum of experienced delay $\bar{d}_{TS}^{E}$ and expected residual delay $\bar{d}_{TS}^{R}$:

$$\bar{d}_{TS}(\tau) = \bar{d}_{TS}^{E}(\tau) + \bar{d}_{TS}^{R}(\tau). \tag{4}$$

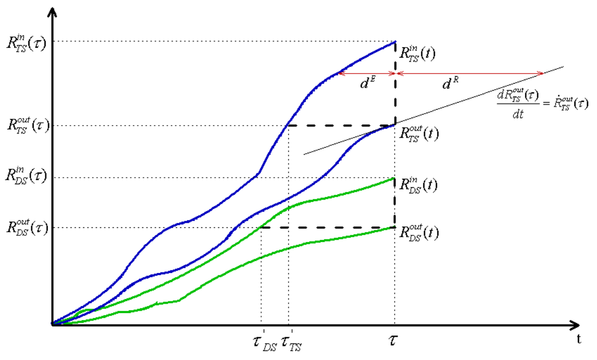

Fig. 2. The input and output curves for TS and DS classes. The average delay is the sum of experienced and residual delays.

The average experienced delay of TS backlog is

$$\bar{d}_{TS}^{E}(\tau) = \frac{1}{R_{TS}^{in}(\tau) - R_{TS}^{out}(\tau)} \int_{\tau'_{TS}}^{\tau} (R_{TS}^{in}(t) - R_{TS}^{out}(\tau))dt, \tag{5}$$

where τ'_{TS} is obtained from the condition $R_{TS}^{in}(\tau'_{TS}) = R_{TS}^{out}(\tau)$. The average residual delay of TS backlog is

$$\bar{d}_{TS}^{R}(\tau) = \frac{1}{\dot{R}_{TS}^{out}(\tau)(\tau - \tau'_{TS})} \int_{\tau'_{TS}}^{\tau} (R_{TS}^{in}(t) - R_{TS}^{out}(\tau))dt. \tag{6}$$

From (4), (5), and (6), expected average queuing delay for TS class is

$$\bar{d}_{TS}(\tau)=\left(\frac{1}{B_{TS}(\tau)}+\frac{1}{\dot{R}_{TS}^{out}(\tau)(\tau-\tau_{TS}^{'})}\right)I_{TS}(\tau)\,dt\,, \quad (7)$$

where $B_{TS}(\tau)=R_{TS}^{in}(\tau)-R_{TS}^{out}(\tau)$, $R_{TS}^{in}(\tau_{TS}^{'})=R_{TS}^{out}(\tau)$ and $I_{TS}(\tau)=\int_{\tau_{TS}^{'}}^{\tau}\left(R_{TS}^{in}(t)-R_{TS}^{out}(\tau)\right)dt\,.$

Using the similar procedure, expected average queuing delay for the DS class can be obtained as

$$\bar{d}_{DS}(\tau)=\left(\frac{1}{B_{DS}(\tau)}+\frac{1}{\dot{R}_{DS}^{out}(\tau)(\tau-\tau_{DS}^{'})}\right)I_{DS}(\tau)\,, \quad (8)$$

where $B_{DS}(\tau)=R_{DS}^{in}(\tau)-R_{DS}^{out}(\tau)$, $R_{DS}^{in}(\tau_{DS}^{'})=R_{DS}^{out}(\tau)$, and $I_{DS}(\tau)=\int_{\tau_{DS}^{'}}^{\tau}\left(R_{DS}^{in}(t)-R_{DS}^{out}(\tau)\right)dt\,.$

Service rates $\dot{R}_{TS}^{out}(\tau)$ and $\dot{R}_{DS}^{out}(\tau)$ that satisfy $\bar{d}_{TS}/\bar{d}_{DS}=\delta$ and $\dot{R}_{TS}^{out}(\tau)+\dot{R}_{DS}^{out}(\tau)=C$ can be calculated from

$$\begin{aligned}&\dot{R}_{TS}^{out\,2}+A_1\dot{R}_{TS}^{out}+A_0=0\,,\\&\dot{R}_{DS}^{out}=C-\dot{R}_{TS}^{out}\end{aligned} \quad (9)$$

where $Z(\tau)=I_{DS}(\tau)/I_{TS}(\tau)$ and

$$\begin{aligned}A_1&=\frac{1}{B_{TS}(\tau)-\delta Z(\tau)B_{DS}(\tau)}\left(\frac{1}{\tau_{TS}^{'}}+\frac{\delta Z(\tau)}{\tau_{DS}^{'}}\right)-C\,.\\A_0&=-\frac{1}{B_{TS}(\tau)-\delta Z(\tau)B_{DS}(\tau)}\frac{C}{\tau_{TS}^{'}}\end{aligned} \quad (10)$$

If (9) is satisfied for every τ, the ratio of average queuing delays for TS and DS class during busy periods will be δ.

The described BPR$^+$ server is based on fluid model of Internet traffic, and, hence, it cannot be implemented in practice. A heuristic approximation is used instead.

3.1. Heuristic approximation of BPR$^+$

We describe a heuristic approximation of the BPR$^+$ fluid server that reflects the packet nature of Internet traffic.

Each packet accepted to the queues is timestamped. These timestamps are used to calculate the average experienced delay of backlogged packets. For TS packets, the average experienced delay is

$$\bar{d}_{TS}^{E}=\frac{1}{N_{TS}}\sum_{k=1}^{N_{TS}}(t-t_k)\,, \quad (11)$$

where N_{TS} is number of the packets in TS queue and t_k is the arrival time of the kth packet. If served with rate r_{TS}, the average residual time of TS packets is

$$\bar{d}_{TS}=\frac{1}{N_{TS}}\sum_{k=1}^{N_{TS}}\frac{q_k}{r_{TS}}\,, \quad (12)$$

where q_k is the amount of traffic in TS queue that has to be served before kth packet (Fig. 3).

Therefore, expected average delay of TS packets is

$$\bar{d}_{TS}=D_{TS}+\frac{1}{r_{TS}}Q_{TS}\,, \quad (13)$$

where $D_{TS}=t-\frac{1}{N_{TS}}\sum_{k=1}^{N_{TS}}t_k$ and $Q_{TS}=\frac{1}{N_{TS}}\sum_{k=1}^{N_{TS}}q_k\,.$

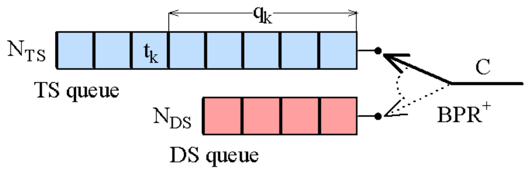

Fig. 3. In a heuristic approximation of BPR$^+$, timestamps are used to calculate the average experienced delay of backlogged packets.

Similarly, expected average delay of DS packets is

$$\bar{d}_{DS}=D_{DS}+\frac{1}{r_{DS}}Q_{DS}\,, \quad (14)$$

where $D_{DS}=t-\frac{1}{N_{DS}}\sum_{k=1}^{N_{DS}}t_k$ and $Q_{DS}=\frac{1}{N_{DS}}\sum_{k=1}^{N_{DS}}q_k\,.$

On each packet departure, service rates r_{TS} and r_{DS} that satisfy $\bar{d}_{TS}/\bar{d}_{DS}=\delta$ and $r_{TS}+r_{DS}=C$ are calculated from

$$\begin{aligned}&r_{TS}^2+\left(\frac{Q_{TS}+\delta Q_{DS}}{D_{TS}-\delta D_{DS}}-C\right)r_{TS}-\frac{Q_{TS}C}{D_{TS}-\delta D_{DS}}=0\,.\\&r_{DS}=C-r_{TS}\end{aligned} \quad (15)$$

Packets are scheduled based on virtual service functions $V_{TS}(t)$ and $V_{DS}(t)$. These functions approximate the difference between the amount of traffic that would have been transmitted from TS and DS queues during the current busy period, if these queues were serviced by rates r_{TS} and r_{DS}, respectively, and the amount of traffic that has been actually transmitted from these queues. The virtual service functions are updated on each packet departure as

$$V_{TS}(t)=\begin{cases}0 & if\ a_{TS}\ge t^d\\ V_{TS}(t^-)+r_{TS}(t)(t-t^d) & if\ a_{TS}<t^d\end{cases} \quad (16)$$

and

$$V_{DS}(t)=\begin{cases}0 & if\ a_{DS}\ge t^d\\ V_{DS}(t^-)+r_{DS}(t)(t-t^d) & if\ a_{DS}<t^d\end{cases}\,, \quad (17)$$

where $r_{TS}(t)$ and $r_{DS}(t)$ are service rates obtained from (15), t^d is the time of the previous departure, and a_{TS} and a_{DS} are arrival times of head-of-line packets in TS and DS queue, respectively. On each packet departure from TS queue, $V_{TS}(t)$ is updated as

$$V_{TS}(t)=V_{TS}(t^-)-l_{TS}\,, \quad (18)$$

where l_{TS} is the length of the departed packet. Similarly, on each departure from DS queue, $V_{DS}(t)$ is updated as

$$V_{DS}(t)=V_{DS}(t^-)-l_{DS}\,, \quad (19)$$

where l_{DS} is the length of the departed packet.

After the virtual service functions have been updated, the

BPR$^+$ scheduler chooses the next packet to be transmitted. If L_{TS} and L_{DS} are lengths of the head-of line packets in TS and DS queues, respectively, the TS queue is serviced if

$$L_{TS} - V_{TS}(t) < L_{DS} - V_{DS}(t). \quad (20)$$

Otherwise, the DS queue is serviced. Pseudocode for the heuristic approximation of the BPR$^+$ scheduler is given in the Appendix.

Alternative options for packet scheduling may not use the virtual service functions V_{TS} (t) and V_{DS} (t). One solution would be to randomly choose between the TS and DS queues with probabilities r_{TS}/C and r_{DS}/C, respectively.

4. PERFORMANCE EVALUATION

In the following simulation scenarios, we evaluate the effectiveness of the BPR$^+$ scheduler in providing proportional delay differentiation between the throughput-sensitive and delay-sensitive classes. The simulated topology in ns-2 network simulator [15] is shown in Fig. 4, with 5 TS sources ($N_{TS}=5$) and DS sources ($N_{DS}=5$). All sources are ON/OFF Pareto sources with an average packet size of 500 bytes, average durations of ON and OFF period 50 ms, and shape parameter 1.5. Capacities and propagation delays of links are shown in Fig. 4. BPR$^+$ is implemented in the access router R. The capacity of the buffer shared between the TS and DS class is 250 packets, except for the scenario where we vary the buffer size.

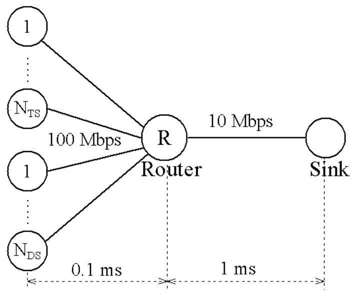

Fig. 4. The simulated topology employed for the evaluation of BPR$^+$ performance.

4.1. Influence of the traffic load

We varied the traffic load by changing the sending rate of the Pareto traffic sources during the ON periods. The traffic load has been varied from 70 % to 120 % of the output link's capacity. Simulation results for various delay differentiation parameters δ are shown in Fig. 5.

The goal of proportional delay differentiation is to make the average delay ratio as close as possible to the specified value δ. As shown in Fig. 5, BPR$^+$ performs better in the case of a heavily congested network. This result is expected because BPR$^+$ relies on the information about the backlogged packets to make a scheduling decision. When a network is underutilized, buffers in routers are empty most of the time and the schedulers are not able to provide desired delay differentiation. However, in that case delay differentiation is not needed because in an underutilized network all packets experience low delay.

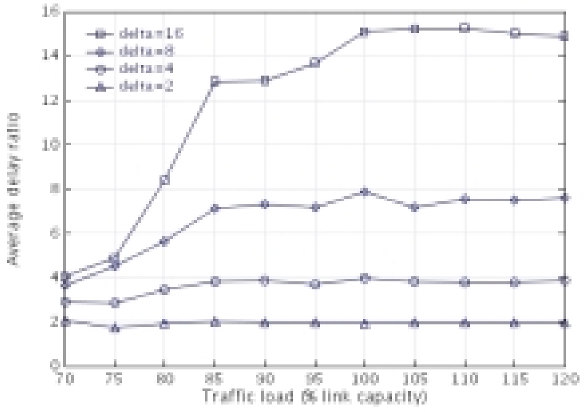

Fig. 5. Average delay ratios achieved by BPR$^+$ for various traffic loads and delay differentiation parameters δ.

4.2. Influence of the load distribution

In this scenario we varied the load distribution between the TS and DS classes while maintaining total load at 100 % of output link's capacity. Achieved average delay ratios are shown in Fig. 6.

In an ideal case, the achieved average delay ratio would not depend on the load distribution between classes. However, BPR$^+$, to a certain degree, depends on traffic load distribution. The traffic class that constitutes larger portion of the total traffic load is likely to experience higher delay than it would in a case of ideal delay differentiation. The most likely explanation for this effect is a certain bias toward the class with a smaller number of packets in the buffer, which is inherent to approximations made in order to adapt BPR$^+$ to the packetized traffic.

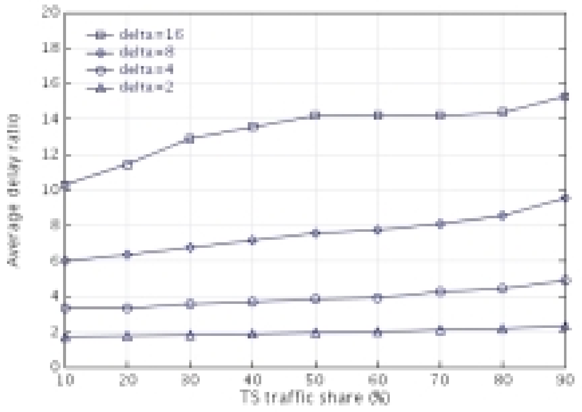

Fig. 6. Average delay ratios achieved by BPR$^+$ for various traffic load distributions and delay differentiation parameters δ.

4.3. Influence of the buffer size

We varied buffer sizes from 150 packets (60 ms of buffering on a 10 Mb/s link) to 350 packets (140 ms of buffering on a 10 Mb/s link). A rule of thumb for dimensioning the buffers in Internet routers is to provide approximately 100 ms of buffering. Simulation results shown in Fig. 7 indicate that the buffer size does not affect the performance of BPR$^+$. However, very small buffer size might affect the ability of the scheduler to provide desired delay differentiation because its decisions depend on information regarding backlog. We have not considered such a scenario in our simulations.

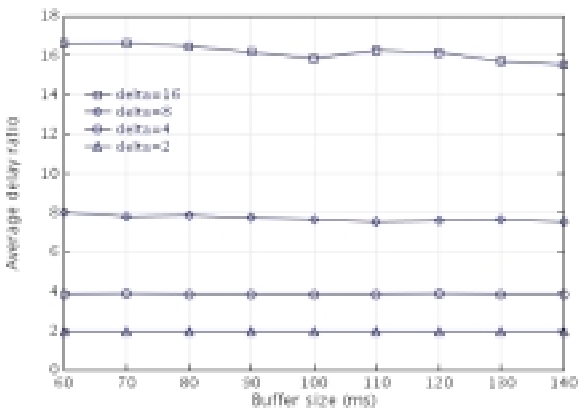

Fig. 7. Average delay ratios achieved by BPR$^+$ for various buffer sizes and delay differentiation parameters δ.

4.4. Influence of the timescale

The results in previous scenarios are obtained by averaging the achieved delay ratio over the entire simulation run. In order to evaluate the performance of BPR$^+$ on shorter timescales, we considered a scenario where the timescales are expressed in terms of *packet time constants*. In our case, one packet time constant is transmission time of a 500-byte packet on a 10 Mb/s link. For example, Fig. 8 shows the queuing delays for the TS and DS classes averaged over 1,000 packet time constants (0.4 s) for δ=4 and utilization 100 %. On this timescale, BPR$^+$ is able to provide consistent delay differentiation, i.e., the TS traffic always experiences higher delay than the DS traffic.

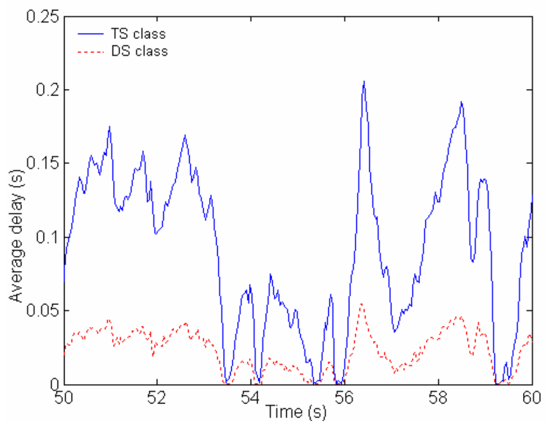

Fig. 8. Queuing delays of TS and DS packets in the case of δ=4, averaged over 1,000 packet time constants.

In order to investigate the performance of the scheduler on various timescales, the entire simulation run has been divided into a number of intervals whose duration is equal to the monitoring timescale. Average delay ratios are calculated for each interval. The procedure has been repeated for four different timescales, corresponding to 10, 100, 1,000, and 10,000 packet time constants. Fig. 9 shows 5 %, 25 %, 50 %, 75 %, and 95 % percentiles of the calculated ratios.

As the monitoring timescale increases, the percentiles of the average delay ratio converge to the specified value δ=4. Hence, BPR$^+$ provides more precise delay differentiation on coarser timescales because it aims to provide specified ratio of *average* delays. Certain schedulers, such as Waiting-Time Priority (WTP) [3], tend to provide specified delay ratio on packet-by-packet basis.

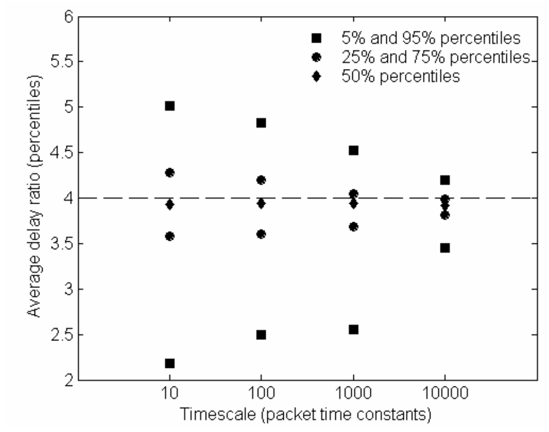

Fig. 9. Percentiles of the average delay ratios in the case of δ=4 over four distinct timescales.

5. CONCLUSIONS

Simulation results indicate that the BPR$^+$ is able to provide proportional delay differentiation between traffic classes. However, performance of BPR$^+$ is, to a certain degree, dependant on traffic load and averaging timescale. BPR$^+$ performs better for heavy traffic loads and coarser timescales. The BPR$^+$ scheduler could be improved by exploring new approaches to "packetize" the BPR$^+$ fluid server. Performance improvement achieved by considering waiting times of backlogged packets should be evaluated by comparing the BPR$^+$ with existing schedulers for proportional delay differentiation.

REFERENCES

[1] S. Shenker, C. Partridge, and R. Guerin, "Specification of guaranteed quality of service," RFC 2212, Internet Engineering Task Force, September 1997.

[2] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated services," RFC 2475, Internet Engineering Task Force, December 1998.

[3] C. Dovrolis, D. Stiliadis, and P. Ramanathan, "Proportional differentiated services: delay differentiation and packet scheduling," in *Proceedings of ACM SIGCOMM 1999*, Cambridge MA, September 1999, pp. 109-120.

[4] C. Dovrolis, D. Stiliadis, and P. Ramanathan, "Proportional differentiated services: delay differentiation and packet scheduling," *IEEE/ACM Transactions on Networking*, vol. 10, no. 1, pp. 12-26, February 2002.

[5] S. Bodamer, "A new scheduling mechanism to provide relative differentiation for real-time IP traffic," in

Proceedings of IEEE GLOBECOM 2000, San Francisco, CA, December 2000, pp. 646-650.

[6] C. C. Li, S.-L. Tsao, M. C. Chen, Y. Sun, and Y.-M. Huang, "Proportional delay differentiation service based on weighted fair queueing," in *Proceedings of IEEE International Conference on Computer Communications and Networks (ICCCN 2000)*, October 2000, pp. 418-423.

[7] J.-S. Li and H.-C. Lai, "Providing proportional differentiated services using PLQ," in *Proceedings of IEEE GLOBECOM 2001*, San Antonio, TX, November 2001, pp. 2280-2284.

[8] Y-C. Lai and W-H. Li, "A novel scheduler for proportional delay differentiation by considering packet transmission time," *IEEE Communications Letters*, vol. 7, no. 4, pp. 189-191, April 2003.

[9] H. Saito, C. Lukovzski, and I. Moldovan, "Local optimal proportional differentiated services scheduler for relative differentiated services," in *Proceedings of IEEE International Conference on Computer Communications and Networks (ICCCN 2000)*, Octobar 2000, pp. 544-550

[10] H-T. Ngin and C-K. Tham, "Achieving proportional delay differentiation efficiently," in *Proceedings of IEEE International Conference on Networks (ICON 2002)*, Singapore, August 2002, pp. 169-174.

[11] L. Kleinrock, *Queueing Systems*. New York: Wiley-Interscience, 1976, vol. 2.

[12] Y. Moret and S. Fdida, "A proportional queue control mechanism to provide differentiated services," in *Proceedings of International Symposium on Computer and Information Systems (ISCIS 1998)*, Belek, Turkey, Octobar 1998, pp. 17-24.

[13] N. Christin and J. Liebeherr, "A QoS architecture for quantitative service differentiation," *IEEE Communications Magazine*, vol. 41, no. 6, pp. 38-45, June 2003.

[14] J. Liebeherr and N. Christin, "Rate allocation and buffer management for differentiated services," *Computer Networks*, vol. 40, no. 1, pp. 89-110, September 2002.

[15] The Network Simulator - ns-2: http://www-mash.cs.berkeley.edu/ns.

APPENDIX

The pseudocode for heuristic approximation of the BPR$^+$ scheduler includes two functions. The first function, executed for each packet accepted to the buffer, appends the current timestamp to the packet and places it into corresponding queue. The second function, executed for each packet leaving the buffer, decides on the next packet to be served.

```
For each packet accepted to the buffer {
    TIME = current time;
    setTimestamp(pkt);
    if the packet belongs to the TS class {
        put packet in the TS queue;
        $D_{TS}$ += TIME; $Q_{TS}$ += length(pkt);
    if the packet belongs to the DS class {
        put packet in the DS queue;
        $D_{DS}$ += TIME; $Q_{DS}$ += length(pkt);
    }
}

For each packet leaving the buffer {
    TIME = current time;
    if the TS queue is empty and the DS queue is not empty {
        pkt = get a packet from the DS queue;
        $D_{DS}$ -= getTimestamp(pkt); $Q_{DS}$ -= $N_{DS}$ * length(pkt);
        $V_{DS}$ = 0; $V_{TS}$ = 0;
    }
    if the TS queue is not empty and the DS queue is empty {
        pkt = get a packet from the TS queue;
        $D_{TS}$ -= getTimestamp(pkt); $Q_{TS}$ -= $N_{TS}$ * length(pkt);
        $V_{TS}$ = 0; $V_{DS}$ = 0;
    }
    if none of the queues is empty {
        C = capacity of the output link;
        $N_{TS}$ = length of the TS queue;
        $N_{DS}$ = length of the DS queue;
        $Q_{TS} = Q_{TS} / N_{TS}$; $Q_{DS} = Q_{DS} / N_{DS}$;
        $D_{TS} = TIME – D_{TS} / N_{TS}$; $D_{DS} = TIME – D_{DS} / N_{DS}$;
        // a, b, and c are the coefficients in (15)
        $a = D_{TS} - \delta * D_{DS}$;
        $b = Q_{TS} + \delta * Q_{DS} – C*a$;
        $c = - Q_{TS}*C$;
        if (a != 0) {
            $r_{TS}$ = (- b + sqrt ($b^2$ - 4*a*c)) / (2*a);
        } else $r_{TS}$ = - c / b;
        $r_{DS} = C – r_{TS}$;
        if (timestamp of the HOL packet in the DS queue >= $t^d$) {
            $V_{DS}$ = 0;
        } else $V_{DS}$ += $r_{DS}$ * (TIME - $t^d$);
        if (timestamp of the HOL packet in the TS queue >= $t^d$) {
            $V_{TS}$ = 0;
        } else $V_{TS}$ + = $r_{TS}$*(TIME - $t^d$);
        if ($L_{DS} - V_{DS} > L_{TS} - V_{TS}$) {
            pkt = get a packet from the TS queue;
            $D_{TS}$ -= getTimestamp(pkt); $Q_{TS}$ -= $N_{TS}$ * length(pkt);
            $V_{TS} = V_{TS}$ - length(pkt);
        } else {
            pkt = get a packet from the DS queue;
            $D_{DS}$ -= getTimestamp(pkt); $Q_{DS}$ -= $N_{DS}$ * length(pkt);
            $V_{DS} = V_{DS}$ - length(pkt);
        }
    }
    $t^d$ = TIME;
}
```