Proc. of the IEEE Global Telecommunications Conference (Globecom ‘00), San Francisco, Nov. 2000, pp. 646-650

A New Scheduling Mechanism to Provide Relative Differentiation for Real-Time IP Traffic

Stefan Bodamer

University of Stuttgart
Institute of Communication Networks and Computer Engineering
E-mail: bodamer@ind.uni-stuttgart.de

***Abstract*–In this paper, design alternatives for supporting real-time traffic in a differentiated services IP network node are studied. A scheduling algorithm called weighted earliest due date (WEDD) is proposed that provides tunable delay differentiation for applications like video or voice over IP. WEDD enhances the earliest due date (EDD) scheduling mechanism in such a way that not only different delay bounds but also different deadline violation probabilities are provided. The ratios of violation probabilities in different classes are easily specified by a set of parameters. Simulation studies are performed to show that the scheduler is able to maintain the specified ratios under various traffic conditions in short as well as in long time scales.**

I. Introduction

The best-effort service of the current Internet is not sufficient for many applications and users. Therefore a lot of research has been done to find an architecture that provides quality of service (QoS) in an appropriate manner. Two extreme approaches can be identified: reservation and prioritization.

Reservation of resources on a per-flow basis is used to give deterministic or statistical guarantees related to certain QoS measures. In the Internet community, this solution is represented by the integrated services (intserv) architecture using RSVP as a signalling protocol [3]. Similar traffic control functions as in ATM networks are necessary.

In a pure prioritization approach only relative guarantees are given, e.g., that flows belonging to a higher priority class receive better performance than those of a lower priority class (either in a deterministic or statistical sense). The differentiated services (diffserv) architecture defined in [2, 15] provides a framework that enables relative differentiation of traffic classes which are identified based on a set of bits in the IP packet header. An appealing feature of diffserv is that it handles flow aggregates thus avoiding per-flow state in the network. There is no end-to-end signalling of traffic parameters and QoS requirements at flow setup. If only relative differentiation is applied the network can do without admission control.

The realization of a diffserv network requires the definition of per hop behaviours (PHB) implemented in the network nodes. A proper design will consider requirements of different traffic types in the network in order to provide an appropriate kind of differentiation. Non-real-time (elastic) traffic (e.g., WWW, file transfer, e-mail) is usually based on TCP which enables applications to adapt their sending rate in case of congestion. This type of traffic needs a differentiation with regard to throughput. Real-time traffic produced by streaming applications like voice over IP or video conferencing additionally requires delay differentiation. As the requirements of elastic and real-time traffic are different the basic assumption in this paper is that both types are separated on each output link of a diffserv node. Such a separation is not unusual. The diffserv implementation SIMA (simple integrated media access), e.g., follows the same principle [13].

The separation of real-time and non-real-time traffic leads to a hierarchical resource sharing architecture (Fig. 1) which has some relationship to the assured forwarding (AF) PHB group definition [11]. Each type gets a share of the link bandwidth C (C_{rt} and C_{nrt}, respectively) that can be adapted dependent on the current load. Link sharing is realized by a fair queueing algorithm, e.g., self-clocked fair queueing (SCFQ) [10] or class-based queueing (CBQ) [9]. Within the non-real-time traffic category an active buffer management strategy with different congestion level thresholds based on random early detection (RED) [8] can be employed. For real-time traffic, however, buffer management is not sufficient. Delay differentiation requires a scheduling mechanism distinguishing several real-time classes.

The rest of the paper concentrates on finding a scheduling mechanism for the real-time classes that provides relative delay differentiation in an appropriate manner. Section II discusses several existing alternatives. In Section III, a new scheduler is proposed that assigns different deadlines to packets in different real-time classes and enforces weighted deadline violation probabilities. The performance of the scheduler is evaluated in Section IV.

II. Scheduling Algorithms for Delay Differentiation

A. Static Priority Queueing

A simple form of providing delay differentiation is to assign static delay priorities to different classes. In this scheme, packets in class i are not forwarded until all queues with priority index greater than i are empty. The static priority model, however, has several serious drawbacks. First, under heavy load conditions starvation may occur in lower priorities. Moreover, the grade of differentiation between classes extremely depends on the load and the load distribu-

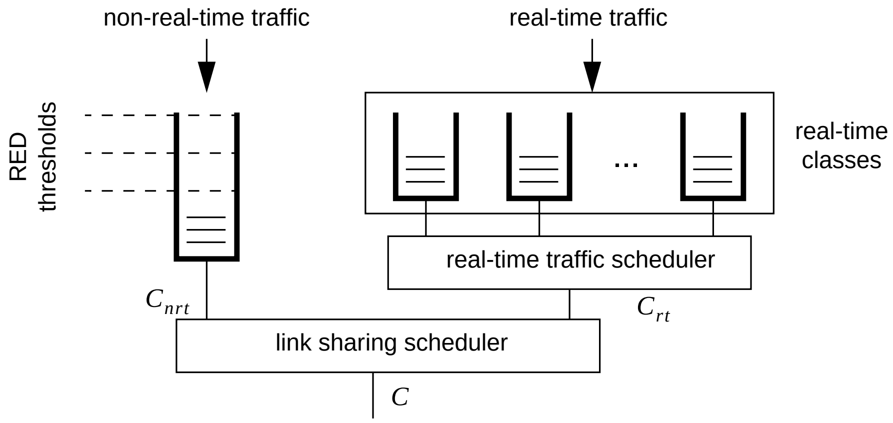

Fig. 1: Hierarchical resource sharing architecture

tion. Finally, a static priority scheme is not controllable as it misses any tuning knobs which could help the administrator to configure the system.

B. Fair Queueing

A different approach, which could be easily integrated into the architecture shown in Fig. 1, is to use a fair queueing mechanism also for real-time traffic. Increasing the weight of a class generally results in better performance with respect to delay. However, it is a complex task to find appropriate values for the weights even in an ideal generalized processor sharing (GPS) scheduler. The differentiation has to be significant on the one hand and the system should not degenerate to a static priority multiplexer on the other hand. This is a problem especially since the distribution of load on different classes, which is previously unknown in a diffserv network, has a huge impact. A solution could be an adaptation of the weights based on online traffic measurements. A long term adaptation, however, may result in lower performance of high priority classes in short time scales [7]. On the other hand, if the adaptation interval is very small, the algorithm has not much in common with fair queueing any longer.

C. Mean Delay Proportional Schedulers

Recently, several schedulers have been proposed that provide a relative differentiation with respect to mean delays [5, 7, 14]. In [5, 7] the well-known waiting time priority (WTP) scheduler is shown to give a rather exact proportional differentiation of mean delays in short and long time scales.

A disadvantage of this kind of schedulers is that they only provide delay differentiation based on average delays. The performance of multimedia tools, e.g. for IP telephony, will not so much depend on mean delays but on the probability that the transmission delay exceeds a certain threshold [1]. A WTP scheduler also differentiates with regard to the excess probabilities. But the ratio of the excess probabilities obtained for different classes depends on the load distribution and thus cannot be fixed by simply adjusting scheduling parameters.

D. Earliest Due Date (EDD) Scheduler

The EDD scheduler [12] also denoted as EDF (earliest deadline first) is a mechanism to provide absolute delay differentiation [6]. Each class i is associated with a delay bound δ_i. A packet of class i arriving at time t_A receives a tag $t_A + \delta_i$ representing its deadline. The packets to be forwarded are scheduled in increasing order of their deadlines.

An important property of an EDD scheduler is that in a homogeneous traffic scenario the probabilities of deadline violations due to congestion are equal in all classes. This property holds independent of the total load and the load proportion of the different classes.

An additional option in operating an EDD scheduler is to discard packets that exceed their deadlines before entering service. This leads to a scheduling mechanism called shortest time to extinction (STE) in [16]. The STE policy is, e.g., appropriate for voice packets which become useless if they do not reach their destination within a certain time interval.

III. Proposed Scheduler: WEDD

In this section, a new scheduler called weighted earliest due date (WEDD) is proposed. WEDD is an enhancement to EDD in the sense that it not only provides different deadlines but also different deadline violation probabilities. The violation probabilities p_i are weighted according to given parameters w_i such that

$$\frac{p_i}{p_j} = \frac{w_i}{w_j} \tag{1}$$

resulting in better performance for the class with lower weight parameter w_i.

First, this gives better controllability to the administrator as only the weight parameters have to be set. Moreover, such a scheme is useful if there is a direct relationship between the delay bounds specified in the scheduler and the playout delays in real-time applications like voice over IP. In this case packets exceeding their deadlines have to be discarded in the end device. So the proposed scheduler gives the ability to separately influence the delay and loss component which both decrease the quality perceived by the user.

The basic operation of WEDD, i.e. setting deadlines $t_A + \delta_i$ on arrival at t_A and scheduling packets in increasing order of their deadlines, is the same as for EDD. However, if there is more than one class being backlogged with the first packet having a deadline $t_A + \delta_i < t_S + \Delta_i$ (t_S denotes the current system time, Δ_i is a safety margin, e.g., $\Delta_i = \delta_i / 10$) the system is called to be in "congestion mode". In congestion mode, for each of the classes fulfilling the above condition a congestion tag c_i is calculated:

$$c_i = \frac{w_i}{E_i(t_S)}. \tag{2}$$

In this equation, $E_i(t_S)$ is the current measurement-based estimation of the real deadline violation ratio in class i. Now the packet with the lowest congestion tag c_i is served.

There are different ways to measure the real deadline violation ratio. The simplest approach is to have two counters m_i and n_i in each class that count the aggregate length of packets having exceeded their deadlines (m_i) and the total aggregate packet length in that class (n_i). The estimated deadline violation ratio is then simply given by the ratio of the current values of m_i and n_i. To avoid that the counter values become infinitely large the value of both m_i and n_i can be multiplied by some factor $\alpha \in [0, 1]$ each time n_i is updated. For $\alpha < 1$ a limitation of m_i and n_i can thus be achieved. Losses that occurred in the past are considered with an exponentially decreasing weight. One may argue that such a counter-based estimation of deadline violation ratios is critical with respect to implementation complexity. It has to be considered, however, that only a rather small number of real-time classes (e.g., 4, 8, 16, or 32) will be defined. Therefore the scalability problem known from networks with per-flow reservation does not occur in this context.

Like EDD, WEDD can be specified to discard late packets. This makes the WEDD scheduler similar to the one presented in [17]. The authors in [17], however, assume that packets are discarded on arrival. Furthermore, their mechanism aims at providing statistical guarantees as necessary in an ATM environment. In the context of this paper, absolute violation probabilities are not considered since there is no admission control and the focus is on relative differentiation.

IV. Performance Evaluation

This section concentrates on the evaluation of real-time scheduling taking only the right side of Fig. 1 into account. Non-real-time traffic is assumed to permanently consume its complete bandwidth share so that there is never more than C_{rt} available for real-time traffic.

First, a burst traffic model is considered where bursts arrive according to a Poisson process. The number of packets in a burst follows a geometric distribution with mean $b = 40$. During a burst packets of constant length $L = 200$ bytes arrive in fixed time intervals. The arrival rate h within a burst is set to 200 kbit/s, which is 1/50 of the link capacity $C_{rt} = 10$ Mbit/s reserved for real-time traffic. The burst arrival rates λ_i in the different classes as well as the total burst arrival rate $\lambda = \Sigma\lambda_i$ are varied during the following simulations. The WEDD estimator parameter α is set to 1.

The scenario comprises two classes ($i \in \{0, 1\}$) with delay bounds $\delta_0 = 100$ ms, $\delta_1 = 50$ ms, a ratio of deadline violation weights $w_0 / w_1 = 10$, and congestion margins $\Delta_0 = \Delta_1 = 10$ ms. The total burst arrival rate λ is fixed such that the offered load $\rho = \lambda \cdot L \cdot b / C_{rt}$ equals 95%. The ratio of burst arrival rates λ_1 / λ_0 is varied in a range between 0.1 and 10.

The simulation results (including 95% confidence intervals) of the deadline violation probabilities for EDD and WEDD in a system that discards late packets are depicted in Fig. 2. It is obvious that over the complete range of the load distribution WEDD is able to provide the desired ratio of violation probabilities very exactly. EDD on the other hand yields equal violation probabilities in both classes as expected. If packets that have exceeded their deadlines are not discarded the deadline violation probabilities increase by about one order of magnitude (Fig. 3). In this case the differentiation by the desired factor of 10 is not possible when the load share of class 1 traffic is very high. The reason for that is the fact that in a work-conserving system the difference in delay between classes can never be larger than that in the static priority model.

For the same scenario, the influence of the estimator parameter α has been investigated (Fig. 4). A value of $1 - 10^{-4}$ is enough if the same load is offered to both classes. If the offered load shares are very different, however, α has to be chosen closer to 1 (e.g., $\alpha = 1 - 10^{-8}$) to obtain the desired ratio of discard probabilities.

An assessment of the short term behaviour of WEDD has been performed by tracing the discard ratios occurring within consecutive time intervals (Fig. 5). If the intervals are small a great fluctuation of the ratio of discarded packets can be observed, while the traces are very much smoother for an interval size of 128 s (which is in the order of a typical IP phone call). The most important information that can be derived from the figure is, however, that the difference between the discard ratios is almost constant corresponding to the specified values of w_i. This widely holds for both values of the interval size so that one can conclude that WEDD behaves also well in reasonably small time scales.

In a second scenario, a system with three classes ($i \in \{0, 1, 2\}$) is evaluated. The parameters are: $\delta_0 = \delta_1 = 100$ ms, $\delta_2 = 50$ ms, $\Delta_0 = \Delta_1 = \Delta_2 = 10$ ms, $w_0 \div w_1 \div w_2 = 100 \div 10 \div 1$, $\alpha = 1$. The load shares of classes 0, 1, and 2 are fixed to values 50%, 25%, and 25%, respectively. Discarding of late packets is switched on. The

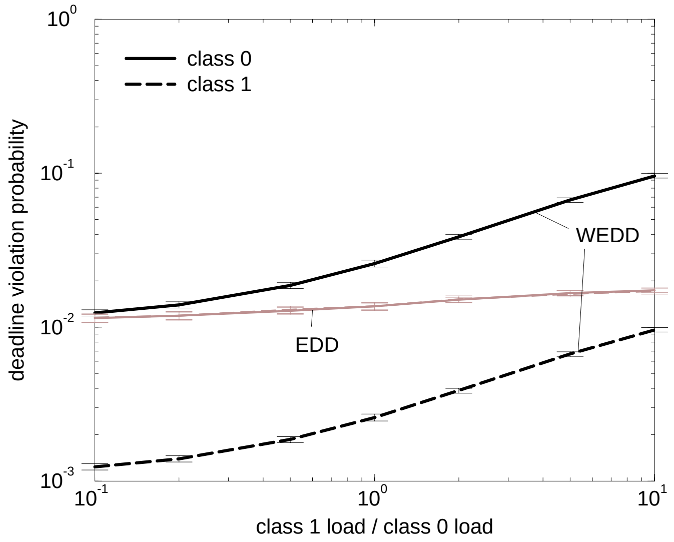

Fig. 2: EDD and WEDD with discarding,
2 classes with constant load 95%

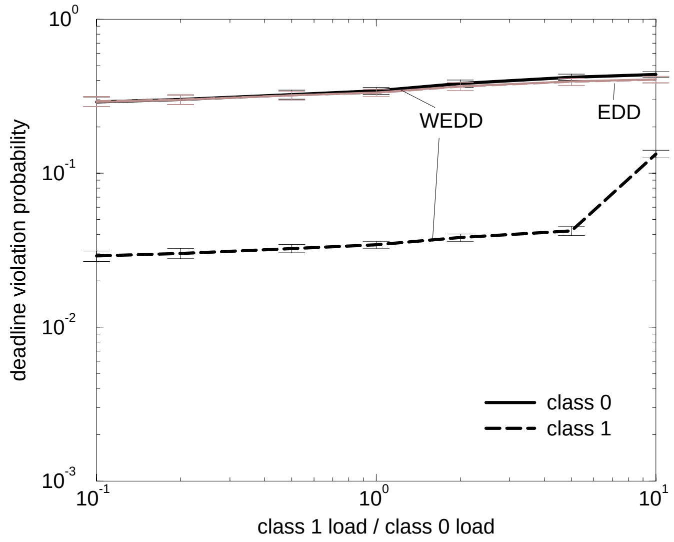

Fig. 3: EDD and WEDD without discarding,
2 classes with constant load 95%

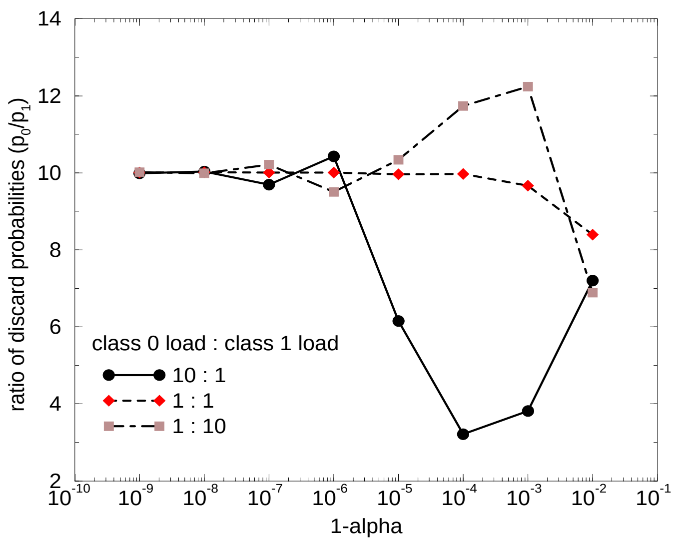

Fig. 4: Influence of α for WEDD with discarding,
2 classes with varying load shares, total load 95%

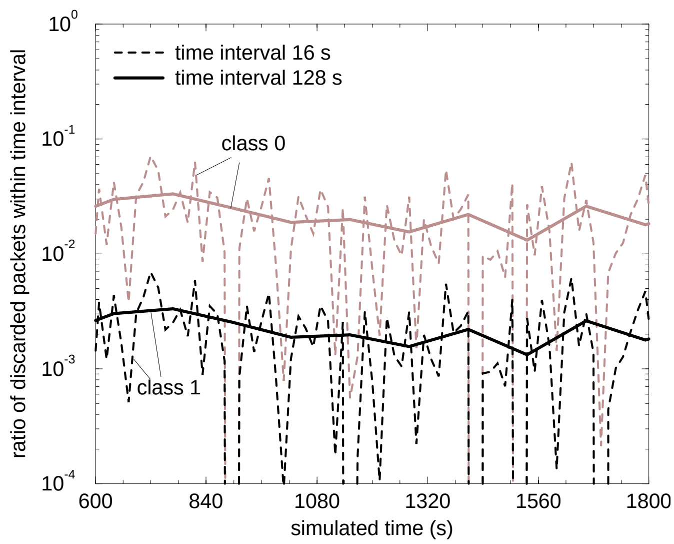

Fig. 5: Short term behaviour of WEDD with discarding,
2 classes with equal load shares, total load 95%

burst length distribution is now derived from measurements of the frame lengths produced by a video conference tool based on the H.261 ITU standard [4]. The measurements were performed with two different parameter settings yielding low and high video quality, respectively. Each burst is segmented into packets with a maximum size of 1000 bytes, which are sent at peak rates of 384 and 768 kbit/s for low and high quality, respectively. Low quality video is mapped to classes 0 and 1, while class 2 traffic corresponds to high quality video.

In Fig. 6, the deadline violation (i.e. discarding) probabilities under variable total offered load are shown. The ratios of violation probabilities follow the specified values of w_i in low as well as in high load regions. Note that even for a load greater than 100% no additional mechanism to limit the queue size is required as packets exceeding their deadlines are discarded.

Fig. 7 gives more insight into what happens to successfully transmitted packets. The complementary cumulative distribution functions (ccdf) of the transfer time have a sharp knee at time value $\delta_i - \Delta_i$ for the higher priority classes. This means that a significant amount of packets are "just in time", i.e. they have transfer times close to their delay bounds. This observation raises the question about the influence of the safety margin. Additional simulations have shown that the shape of the transfer time ccdf indeed depends on the choice of Δ_i. The deadline violation probabilities, however, are rather insensitive to the safety margins as long as very small values (e.g., less than 1 ms in the above configuration) are avoided.

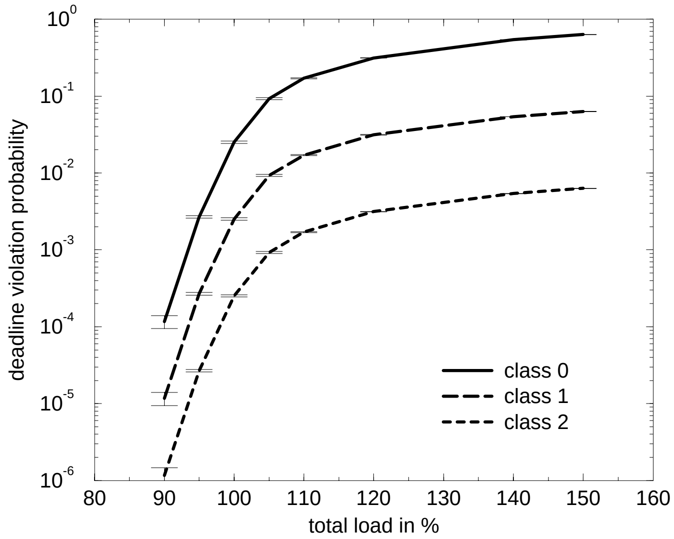

Fig. 6: WEDD with discarding, 3 classes with constant load shares (50%, 25%, 25%)

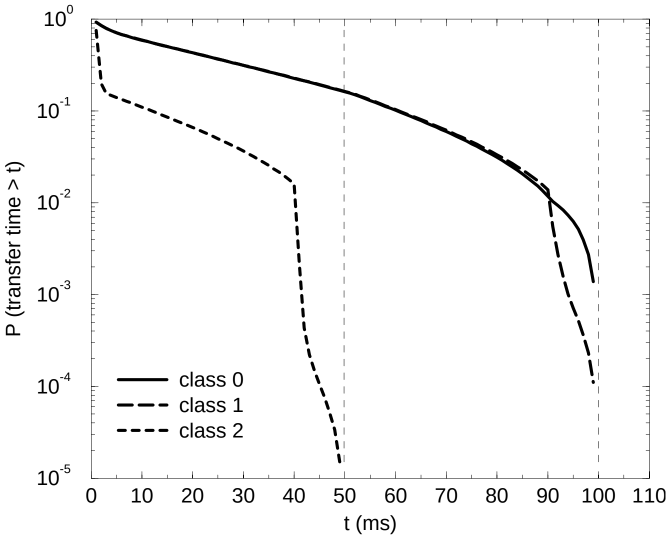

Fig. 7: WEDD with discarding, 3 classes with constant load shares (50%, 25%, 25%), total load 95%

V. Conclusions

The scheduling algorithm weighted earliest due date (WEDD) has been proposed which provides relative delay differentiation for real-time traffic in a diffserv network. WEDD enhances the well-known EDD service discipline such that the ratio of deadline violation probabilities can be fixed to certain values. This mechanism can be combined with the discarding of late packets. So real-time applications like voice over IP can be supported in a flexible way, as delay and loss, which are perceived by the user in different ways, can be influenced separately.

Simulation results have shown that WEDD is able to maintain the desired ratios of deadline violation probabilities under various load conditions independent of the traffic type in short as well as in long time scales. Thus, as only relative performance parameters are considered the scheduler easily provides an adaptation to the incoming traffic.

Acknowledgment

The author would like to thank Klaus Dolzer and Wolfgang Payer for providing the video traffic data.

References

[1] C. M. Aras, J. Kurose, D. Reeves, H. Schulzrinne: “Real-Time Communication in Packet-Switched Networks.” *Proceedings of the IEEE*, Vol. 82, No. 1, Jan. 1994, pp 122-139.

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

[3] R. Braden, D. Clark, S. Shenker: *Integrated Services in the Internet Architecture: an Overview*, IETF, RFC 1633, July 1994.

[4] K. Dolzer, W. Payer, M. Eberspächer: “A Simulation Study on Traffic Aggregation in Multi-Service Networks.” *Proceedings of the IEEE Conference on High Performance Switching and Routing (ATM 2000)*, Heidelberg, June 2000, pp. 157-165.

[5] C. Dovrolis, P. Ramanathan: “A Case for Relative Differentiated Services and the Proportional Differentiation Model.” *IEEE Network*, Vol. 13, No. 5, Sep.-Oct. 1999, pp. 26-34.

[6] C. Dovrolis, D. Stiliadis: “Relative Differentiated Services in the Internet: Issues and Mechanisms.” *Proceedings of ACM SIGMETRICS ’99*, Atlanta, May 1999.

[7] C. Dovrolis, D. Stiliadis, P. Ramanathan: “Proportional Differentiated Services: Delay Differentiation and Packet Scheduling.” *Proceedings of ACM SIGCOMM ’99*, Cambridge, USA, Sep. 1999.

[8] S. Floyd, V. Jacobsen: “Random Early Detection Gateways for Congestion Avoidance.” *IEEE/ACM Transactions on Networking*, Vol. 1, No. 4, Aug. 1993, pp. 397-413.

[9] S. Floyd, V. Jacobsen: “Link-sharing and Resource Management Models for Packet Networks.” *IEEE/ACM Transactions on Networking*, Vol. 3, No. 4, Aug. 1995, pp. 365-386.

[10] S. J. Golestani: “A Self-Clocked Fair Queueing Scheme for Broadband Applications.” *Proceedings of IEEE INFOCOM ‘94*, Toronto, June 1994, pp 636-646.

[11] J. Heinanen, F. Baker, W. Weiss, J. Wroclawski: *Assured Forwarding PHB Group*, IETF, RFC 2597, June 1999.

[12] C. Liu, J. Layland: “Scheduling algorithms for multiprogramming in a hard real-time environment.” *Journal of the ACM*, Vol. 20, No. 1, Jan. 1973, pp. 46-61.

[13] M. Loukola, J. Ruutu, K. Klikki: *Dynamic RT/NRT PHB Group*. Internet Draft <draft-loukola-dynamic-00.txt>, work in progress, Nov. 1998.

[14] T. Nandagopal, N. Venkitaraman, R. Sivakumar, V. Bharghavan: “Delay Differentiation and Adaptation in Core Stateless Networks.” *Proceedings of IEEE INFOCOM 2000* (to appear), Tel Aviv, March 2000.

[15] K. Nichols, S. Blake, F. Baker, D. Black: *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*, IETF, RFC 2474, Dec. 1998.

[16] S. S. Panwar, D. Towsley, J. K. Wolf: “Optimal Scheduling Policies of a Class of Queues with Customer Deadlines to the Beginning of Service.” *Journal of the ACM*, Vol. 35, No. 4, Oct. 1988, pp. 832-844.

[17] Y. Xie, T. Yang: “Cell discarding policies supporting multiple delay and loss requirements in ATM networks.” *Proceedings of the IEEE Global Telecommunications Conference (Globecom ’97)*, Phoenix, Arizona, Nov. 1997.