Dynamic Priority Scheduling for Proportional Delay Differentiated Services

Lassaad Essafi[1], Gunter Bolch[1], and Hermann de Meer[2]

[1] Institute of Computer Science
University of Erlangen-Nuremberg
Martensstrasse 3, D-91058 Erlangen, Germany
ldessafi@aol.com, bolch@informatik.uni-erlangen.de

[2] Department of Computer Science
University College London
London WC1E 6BT, UK
H.DeMeer@cs.ucl.ac.uk

Abstract Recent results on the proportional differentiation model show that waiting time priority scheduling can be applied to implement proportional delay differentiatied services reasonably well under limited conditions. While earlier research focused on heavy load conditions only, more recent results provide insights into waiting time priority scheduling under moderate load conditions also, but the applicability of the algorithm has been limited to a two-service-class solution for numerical reasons. In this paper, in contrast, a dynamic adjustment of a waiting time priority scheduler is suggested to meet the differentiation requirements for any finite number of traffic classes. Our newly introduced approach is based on genetic algorithms. The dynamically optimized scheduling parameters can be determined with high accuracy. We apply an interpolation function to yield a continuum of parameters rather than discrete values and propose a simple look-up table for a dynamic adjustment of the scheduling parameters. We also focus on feasibility and implementation issues related to the suggested algorithm. Suitability of other time-dependent priority functions for proportional delay differentiation is also investigated.

1 Introduction

Two quality of service architectures in the Internet context have been proposed by the IETF: Integrated Services (IntServ) and Differentiated Services (DiffServ). IntServ was proposed as a model to introduce flow-based quality of service into the Internet, which has traditionally been limited to best-effort type services [4].

The differentiated services (DS) architecture is based on a model where traffic entering a DS domain is classified and conditioned at the edges, then marked and assigned to different behavior aggregates (BA) by setting the value of the DiffServ Code Point (DSCP). Within the core of the network, packets are forwarded according to the per-hop behavior (PHB) associated with the DS codepoint [7,8].

Two approaches to service differentiation have been defined: *absolute differentiated services* and *relative differentiated services*. The first relies on admission control and resource reservation mechanisms in order to provide guarantees and statistical assurances for performance measures, such as minimum service rate and end-to-end delay [3]. The approach of relative differentiated services groups network traffic in a small number of classes. These classes are ordered based on their packet forwarding quality, in terms of queuing delays and packet losses [2,3]. This ordering should guarantee that higher classes ($j > i, \forall i, j \in \{1, \ldots, R\}$, where R denotes the number of traffic classes) experience a better forwarding behavior.

The *proportional differentiation model* is a further refinement of the relative differentiated services approach. It is thought to control the quality spacing between classes at each hop, where the packet forwarding performance for each class are ratio-ed proportionally to certain differentiation parameters set by the network operator [3].

One of the main findings in [3] is that the waiting time priority (WTP) scheduler approximates the proportional differentiation model in heavy load conditions, even in short timescales. The WTP scheduler is a priority scheduler which assigns the priority of each packet depending on the packet waiting time in the queuing system. When the load tends to approach 100%, the delay ratios of two consecutive classes tend to converge to the reciprocals of the corresponding increasing rates of the priority functions.

We proposed in [1] an adjustment scheme of the WTP scheduler depending on the system load (characterized by the utilization of the different classes) and the delay differentiation parameters. For two traffic classes, we derived exact formulae for checking the feasibility of the delay differentiation parameters and setting the scheduler parameters. Similar results were also given in [6]. In [5], a similar approach for weighted fair queuing is proposed.

The iterative approach introduced in [1] for more than two traffic classes showed an unpredictable complexity and a limited accuracy. In contrast, the approach presented in this paper enables the calculation of the scheduler parameters with very high accuracy and is designed to meet the strict performance requirements in a router. It is based on an "offline" calculation of the optimal scheduler parameters for moderate load conditions. A table lookup or use of simple interpolation functions is proposed during runtime operation of the router. For the determination of the scheduler parameters, we show how genetic algorithms can be used to provide very accurate results. The interpolation functions are defined to deliver the optimal scheduler parameters in a continuous utilization range. Furthermore, we show how this optimization technique can be extended to other time-dependent priority functions.

The structure of this paper is as follows: in Section 2, we show why dynamic priority scheduling is required to meet the proportional delay differentiation requirements. We begin with presenting earlier results for two traffic

classes and then extend the approach to an optimization based on genetic algorithms for any finite number of traffic classes. In Section 3, we address some implementation issues related to the proportional delay differentiation. In Section 4, we investigate two other types of time-dependent priorities with quadratic functions and initial priorities. In Section 5, we conclude and summarize some of the open issues.

2 Dynamic Priority Scheduling

2.1 Proportional Delay Differentiation

The proportional differentiation model states that certain traffic class performance metrics should be proportional to the differentiation parameters the network operator chooses [3]. Suppose $E[q_i(t, t+\tau)]$ is the average performance measure for class i in the time interval $(t, t+\tau)$, where $\tau > 0$ is the monitoring timescale. The proportional differentiation model imposes that for all pairs of traffic classes and for all time intervals $(t, t+\tau)$ for which $E[q_i(t, t+\tau)]$ and $E[q_j(t, t+\tau)]$ are defined:

$$\frac{E[q_i(t, t+\tau)]}{E[q_j(t, t+\tau)]} = \frac{c_i}{c_j}. \tag{1}$$

where $c_1 < c_2 < \ldots < c_R$ are the generic quality differentiation parameters and R is the number of traffic classes.

The proportional differentiation model can be applied in the context of queuing delays as the performance measure, by setting $E[q_i(t, t+\tau)] = 1/E[W_i(t, t+\tau)]$, where $E[W_i(t, t+\tau)]$ is the average queuing delay of the class i packets departed in the interval $(t, t+\tau)$. If there are no such packets, $E[W_i(t, t+\tau)]$ is undefined. The proportional delay differentiation states that for all pairs (i, j) of traffic classes and for all time intervals $(t, t+\tau)$ for which $E[W_i(t, t+\tau)]$ and $E[W_j(t, t+\tau)]$ are defined:

$$\frac{E[W_i(t, t+\tau)]}{E[W_j(t, t+\tau)]} = \frac{\delta_i}{\delta_j}. \tag{2}$$

The parameters δ_i are known as the *Delay Differentiation Parameters (DDPs)*. As higher classes ($j > i, \forall i, j \in \{1, \ldots, R\}$)are better than lower classes with respect to delay, that is, they experience shorter delays, the DDPs satisfy the relation $\delta_1 > \delta_2 > \ldots > \delta_R$.

2.2 Dynamic Scheduling Model

The waiting time priority scheduler used to implement the proportional delay differentiation is a non-preemptive packet scheduling discipline, characterized by a set of parameters b_r, $1 \leq r \leq R$. The priority of a class-r packet at time t, which arrives to the system at time t_0, increases with the elapsed time in the queuing system linearly with the rate b_r. The priority of a class-r packet, $1 \leq r \leq R$, denoted by $q_r(t)$, is given by:

$$q_r(t) = (t - t_0)b_r, \qquad 1 \leq r \leq R. \tag{3}$$

The mean queuing delay of a class-r packet is given by [9]:

$$E[W_r] = \frac{E[W_{FIFO}] - \sum_{i=1}^{r-1} \rho_i E[W_i](1 - \frac{b_i}{b_r})}{1 - \sum_{i=r+1}^{R} \rho_i (1 - \frac{b_r}{b_i})}, \tag{4}$$

where ρ_i denotes the utilization in class i, $1 \leq i \leq R$. The queuing delay given in Eqn. (4) refers to the long-time average queuing delay and is therefore not time-dependent, as apposed to queuing delays denoted in Eqn. (2), which refer to the queuing delays in any time interval τ.

The proposed dynamic priority scheduling is depicted by Figure 1. The diagram shows the router classification, queuing and scheduling functions. The router is furthermore extended by a weight adjustment module, which -when triggered[1]- activates a new set of scheduler parameters based on collected measurement data. The measurements give evidence on the packet arrival

[1] see discussion in Section 3

rates in the different queues, the service rates of the different traffic classes and the backlog in the queues. As we assume infinite buffers in this paper, only arrival and service rates and the derived utilizations will be considered in this paper. The same architecture is also proposed in [5] for a dynamic weighted fair queuing model.

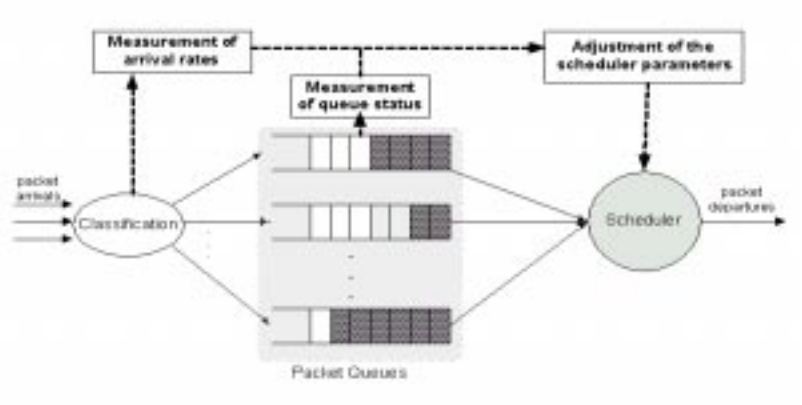

Figure 1. Dynamic priority scheduling

For the dynamic priority scheduling, we extend the priority function in Eqn. (3) as follows. Instead of defining a fixed increasing rate b_r for the priorities of class-r packets, the increasing rate b_r is to be dynamically adjusted depending on:

- the current system conditions, defined by the utilizations of the R traffic classes $\rho_1, \rho_2, \ldots, \rho_R$, and
- the differentiation requirements, defined by the delay differentiation parameters $\delta_1, \delta_2, \ldots, \delta_R$.

The increasing rates b_r for the priorities of class-r packets are regarded as functions of the parameters $\rho_1, \rho_2, \ldots, \rho_R$ and $\delta_1, \delta_2, \ldots, \delta_R$. We define the dynamic priority scheduler as a scheduler which applies the priority function:

$$q_r(t) = (t - t_0) b_r(\rho_1, \ldots, \rho_R, \delta_1, \ldots, \delta_R), \qquad 1 \leq r \leq R. \tag{5}$$

It has to be noted that the increasing rates for **all** classes have to adjusted simultaneously, in order to avoid the case that higher classes get lower increasing rates than lower classes ($b_j > b_i, \forall i, j \in \{1, \ldots, R\}$ and $j > i$).

2.3 Optimization Problem

Determining the increasing rates $b_1, b_2, \ldots, b_R$ of the priority functions q_1, $q_2, \ldots, q_R$ can in fact be considered as an optimization problem which can be stated as follows: given the delay differentiation parameters $\delta_1, \delta_2, \ldots, \delta_R$ and the utilizations of the traffic classes $\rho_1, \rho_2, \ldots, \rho_R$, determine the parameters $b_1, b_2, \ldots, b_R$ so that the scheduler achieves the required ratios of the average queuing delay (see Eqn.(2)). The ratios of the experienced queuing delays are ideally equal to the required delay differentiation ratios for all pairs of consecutive classes, which in terms of the sum of quadratic errors means:

$$\sum_{i=1}^{R-1} \left(\frac{E[W_i]}{E[W_{i+1}]} - \frac{\delta_i}{\delta_{i+1}} \right)^2 = 0, \tag{6}$$

where $E[W_i]$ is the mean queuing delay of a class-i packets which is given by Eqn.(4). In case the average delay differentiation is not feasible, we minimize the cost function

$$\sum_{i=1}^{R-1} g_i \cdot \left(\frac{E[W_i]}{E[W_{i+1}]} - \frac{\delta_i}{\delta_{i+1}} \right)^2 \rightarrow min, \tag{7}$$

or

$$\sum_{i=1}^{R-1} g_i \cdot \left| \frac{E[W_i]}{E[W_{i+1}]} - \frac{\delta_i}{\delta_{i+1}} \right| \rightarrow min, \tag{8}$$

which means that the ratios of the average queuing delay have to be as close as possible to the required DDP ratios. In Eqn. (7) the sum of quadratic errors is to be minimized, whereas in Eqn. (8) the sum of absolute errors is considered. The positive factor g_i in Eqn. (7) and Eqn. (8) is used to weight specific delay ratios, e.g. higher weights for the delay ratios of the higher priority classes. For the rest of the paper, all weights g_i, $1 \leq i < R$ are set to 1.

2.4 Exact Solution for two Traffic Classes

For two traffic classes, a set of results have been derived [1,6]:

1. A specific delay differentiation ratio $\frac{\delta_2}{\delta_1}$ is feasible, if and only if the system utilization ρ satisfies the condition:

$$\rho > 1 - \frac{\delta_2}{\delta_1}. \tag{9}$$

   For example a differentiation ratio of 4 ($\frac{\delta_1}{\delta_2} = 4$), cannot be achieved at a load less than or equal to 75%.

2. In order to meet predefined delay differentiation requirements specified by δ_1 and δ_2, the increasing rates of the priority functions have to be set to:

$$b_1 = 1 \text{ and} \tag{10a}$$

$$\frac{b_1}{b_2} = 1 - \frac{1}{\rho}(1 - \frac{\delta_2}{\delta_1}). \tag{10b}$$

3. As the load approaches 100%, the ratio of the scheduler parameters b_1/b_2 tends to the inverse of the corresponding DDPs (consider limit of Eqn. (10b) as the utilization ρ tends to 1).

4. The scheduler parameters don't depend on the class load distribution, the delay differentiation ratios neither. These parameters depend on the *total* utilization in the queuing system, which is in line with the simulation results presented in [3].

2.5 Optimization using Genetic Algorithms

The motivation behind the new approach is based on the following design requirements:

- fulfilling the strict performance requirements in a router and
- providing a high accuracy for the scheduler parameters, required for the priority function given in Eqn. (5).

In a router, where it thought to be inefficient to do complex[2] computations "online", the required sets of parameters $\{b_i\}$ can be computed "offline" for predefined load ranges and implemented in a lookup table. The router can switch between different sets of parameters depending on the current load, which is feasible even at high speeds. The same set of the "offline" calculated parameters can be used to estimate the control parameters of a simple fitting function, which then can be executed online. The adjustment of the weights can either be triggered on regular time intervals or due to important "events", e.g. overload of a specific traffic class (see discussion in Section 3).

In order to generalize the problem to any[3] number of traffic classes, the optimization problem in Eqn. (8) is to be solved by finding solutions for the set of $R-1$ non-linear equations given by the Eqn. (2) for all consecutive pairs of traffic classes. To our knowledge no analytical solution of this problem is available.

We concentrate in our work on using genetic algorithms for solving the optimization problem. Genetic algorithms simulate in a way some processes known in natural selection to get better and better solutions for a problem. A typical genetic algorithm has the following structure:

1. Randomly generate a population of individuals (bit strings).
2. Decode each individual and evaluate its fitness.
3. Generate a new population partly by cloning, partly by combining and partly by mutating the fittest individuals.
4. Repeat steps 2 and 3 until a stop condition holds.

For a comprehensive overview of genetic algorithms, refer to [11]. For the implementation of the genetic algorithms we consider the optimization problem in Eqn. (8) with the following constraints:

[2] such as time consuming optimizations

[3] given by the number of traffic classes, which is generally limited

- $b_i \geq 1, \qquad \forall i \in \{1, \ldots, R\},$
- $b_i \leq b_j, \qquad \forall i, j \in \{1, \ldots, R\} \text{ and } i \leq j,$
- $k_i b_i \leq b_{i+1}, \qquad \text{where } k_i = \frac{\delta_i}{\delta_{i+1}}, \forall i \in \{1, \ldots, R-1\}.$

The increasing rates $b_1, b_2, \ldots, b_R$ of the priority functions $q_1, q_2, \ldots, q_R$ are initialized with the values $1, \delta_2/\delta_1, \ldots, \delta_R/\delta_{R-1}$, respectively. The parameters $1, \delta_2/\delta_1, \ldots, \delta_R/\delta_{R-1}$ approximate the proportional delay differentiation for a very high utilization and is straight-forward to take as an initialization for an optimization algorithm.

Table 1 shows the results of the optimization using genetic algorithms for four traffic classes. The results show the required increasing rates b_1, b_2, b_3 and b_4, which are calculated for

- the utilizations 70%, 75%, 80%, 85%, 90%, 95% and 100% (these discrete values should be considered as the interval limits of the load range between 70% and 100%),
- the traffic load distribution [25%,25%,25%,25%] and
- the required delay ratio 2.

The optimization for the 100% utilization is meant to show that the algorithms converge to the expected set of scheduler parameters $[b_1, b_2, b_3, b_4] = [1, 2, 4, 8]$

In Table 2, we show the achieved mean waiting times for the four traffic classes and the delay ratios of consecutive classes, associated to the scheduler parameters shown in Table 1. For readability reasons we define one time unit as one millisecond. For an utilization equal to 100%, the mean waiting times are theoretically infinite. We show in Table 1 the values for an utilization equal to 99.99%.

The proposed approach differs significantly from the heuristic algorithm given in [1] and the iterative algorithm based on Gauss-Seidel iteration presented in [6]. Both algorithms have a too high complexity, that they are not

Table 1. Results of the optimization using genetic algorithms: Scheduler parameters $[b_1, b_2, b_3, b_4]$

ρ [in %]	b_1	b_2	b_3	b_4
70%	1.00	6.42	13.10	9999792
75%	1.00	6.53	20.61	999807
80%	1.00	3.61	16.06	218.29
85%	1.00	2.83	8.54	34.36
90%	1.00	2.42	6.01	16.72
95%	1.00	2.17	4.75	10.82
100%	1.00	2.00	4.00	8.00

Table 2. Results of the optimization using genetic algorithms: mean delays and delay ratios

ρ [%]	$E[W_1]$ [msec]	$E[W_2]$ [msec]	$E[W_3]$ [msec]	$E[W_4]$ [msec]	$\frac{E[W_1]}{E[W_2]}$	$\frac{E[W_2]}{E[W_3]}$	$\frac{E[W_3]}{E[W_4]}$
70%	6.465	3.232	2.424	1.212	2.00	1.33	2.00
75%	8.415	3.892	2.462	1.231	2.16	1.58	2.00
80%	10.667	5.333	2.667	1.333	2.00	2.00	2.00
85%	14.222	7.111	3.556	1.778	2.00	2.00	2.00
90%	21.333	10.667	5.333	2.667	2.00	2.00	2.00
95%	42.666	21.334	10.667	5.333	2.00	2.00	2.00
100%	21331	10667	5334	2668	2.00	2.00	2.00

suitable for the runtime operation of a DiffServ router. Especially the number of required iterations is not predictable. It even gets more unreliable, as no sufficient feasibility conditions are known and so cannot be verified beforehand. The "indeterministic" behavior of both algorithms together with the complexity of the operations to be executed is thought not to be suitable for runtime operation of a router. Though genetic algorithms are not less complex as the two considered algorithms, both the proposed offline calcu-

lation of the scheduler parameters and the achieved accuracy, as the results presented in Table 2 demonstrate, justify their use.

We notice that for an utilization equal to 70% and 75%, the required delay ratio 2 is not achieved. This can either be related to the optimization algorithm or due to the feasibility of the proportional differentiation requirements. We will address this issue in the next section.

3 Implementation of the Dynamic Priority Scheduling

3.1 Feasibility of the Differentiation Model

Necessary and sufficient conditions for the feasibility of the proportional differentiation model have been determined in the case of two traffic classes (see Section 2.4). The feasibility conditions for more than two traffic classes haven't been explicitly derived. In [6], several conditions were presented and discussed. They don't enable, however, to directly check the feasibility of certain DDPs under specific load conditions. From Table 2, it is not directly possible to identify, whether the values achieved for 70% utilization are not accurate due to the optimization algorithm or because the delay differentiation ratio 2 for four classes is not feasible because of non sufficient utilization (70%).

It is thought that delay differentiation is not required and is not applicable in all load conditions (e.g. experiments under 10% utilization in [6]). Based on Eqn. 4, we show in Table 3, the queuing delays and delay ratios for four traffic classes using the WTP and FCFS scheduler

- for the utilization range between between 10% and 95%,
- with the class load distribution [25%,25%,25%,25%] and
- the WTP scheduler parameters $[b_1, b_2, b_3, b_4] = [1, 2, 4, 8]$.

Table 3. Mean waiting times for FCFS and WTP scheduler

ρ [%]	$E[W_1]$ [msec]	$E[W_2]$ [msec]	$E[W_3]$ [msec]	$E[W_4]$ [msec]	$E[W_{FIFO}]$ [msec]
10%	1.1735	1.1318	1.0886	1.0506	1.1111
20%	1.3986	1.296	1.195	1.1103	1.25
30%	1.6994	1.506	1.3262	1.1826	1.4286
40%	2.1164	1.7838	1.4934	1.273	1.6667
50%	2.7234	2.1686	1.7164	1.3915	2
60%	3.6697	2.7382	2.0344	1.5577	2.5
70%	5.3068	3.6723	2.5375	1.8167	3.3333
80%	8.6957	5.5072	3.4944	2.3027	5
90%	19.1617	10.9138	6.2407	3.6838	10
95%	40.3785	21.625	11.6194	6.3771	20

These results are shown graphically in Figure 2.

In fact, the issue raised here is related to the need for quality of service in an over-provisioned network. The delay at 70% utilization is three times as much as the delay at 10%. In this utilization range the number of packets in the system is very small (Little's theorem). The delay at 95% utilization is 6 times higher than the delay at 70% utilization, though the utilization range is less than the half of the range from 10% to 70%. This surely allows more differentiation capability in this range (70% to 95% and 100%). A significant lower utilization limit, from which the proportional delay differentiation is to be applied, still needs to be addressed (e.g. 70%).

3.2 Adjustment of the Scheduler Parameters

In Figure 1 we presented the weight adjustment module, responsible for finding out the instantly optimal scheduler parameters and regulating the scheduling policy. We think here of two possible realization techniques:

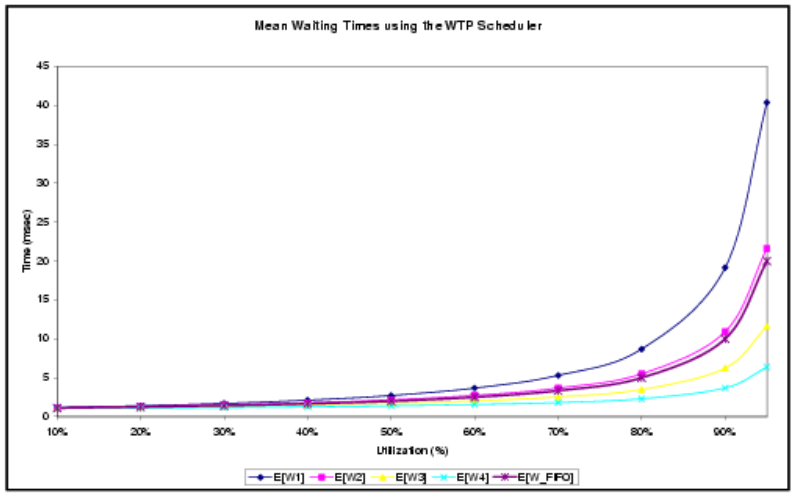

Figure 2. Mean waiting times of FCFS and WTP scheduling

Lookup tables: In a lookup table, the load range (e.g. [70%..99%]) is split in discrete intervals. Offline calculated parameter sets $[b_1, b_2, \ldots, b_R]$ are applied to the load ranges. For the offline calculation of the scheduler parameters, we recommend the use of genetic algorithms (see Table 2).

Use of a simple function: Similar to the function derived for two classes (see Eqn. (10b)), it is thought that a "simple" function can be defined to interpolate the ranges between the parameter sets (which can be calculated "offline" using genetic algorithms or other optimization methods). The use of this method can become unnecessary, if the intervals chosen for the lookup method are very small and so only minimal error is performed. Which class of functions optimally describes the changes of the scheduler parameters has not been addressed yet.

The use of the lookup table is always coupled to a discretization error, as one parameter set is used for a load range. Depending on the required accu-

racy, the utilization interval size has be set accordingly. An optimal solution should consider different sizes for the utilization intervals. Table 4 shows examples of the discretization errors based on the scheduler parameters in Table 1. The columns *error1, error2 and error3* refer to the differences between the calculated ratios and the desired differentiation ratios (in this case equal to 2). We notice that in most cases the calculated error is below 0.1, which corresponds to $0.1/2 = 5\%$.

For two classes, the scheduler parameters don't depend on the class load distribution, but on the overall load the system experiences (see Eqn. (10b)). Simulation results in [2,6] showed that the achieved delay ratios are independent from the class load distribution. It is assumed that only the overall utilization is relevant for the lookup tables (see Table 4).

Table 4. Discretization errors

measured utilization	considered utilization	error1	error2	error3
77.2%	75%	0.2769	-0.377	0.0824
79.2%	80%	-0.0302	-0.0303	-0.03
82%	80%	0.1832	0.359	-0.1458
86%	85%	0.0363	0.0375	0.0438
88.4%	90%	-0.051	-0.0529	-0.0647
92%	90%	0.069	0.073	0.0943
92.5%	90%	0.0872	0.0926	0.1207
92.5%	95%	-0.0735	-0.0769	-0.09801
97.2%	95%	0.0711	0.0764	0.1050
98%	100%	-0.0560	-0.0594	-0.0791

One issue related to the implementation of this approach, which we want to outline here, refers to the triggering of the weight adjustment, which dynamically adjusts the scheduler parameters (see Figure 1). Here we identify two possible triggers. Either a periodical adjustment of the scheduler param-

eters is performed. Or a trigger on particular events can be implemented. A significant utilization change in one or more traffic classes is an example of a significant event. The time window for the periodical adjustment has to be chosen in a way, that a reaction on the "recent" history is guaranteed. A long time window has the side effect, that past history becomes dominant and no short-time consideration is made. In this context, an approach how to deal with the backlog change in the queues needs to be investigated, especially if some queues remain empty for a long period of time.

4 Experiments with Other Priority Functions

In this section we present some preliminary results using two other priority functions[4]:

Quadratic priority function: which applies the priority function:

$$q_r(t) = (t - t_0)b_r^2. \tag{11}$$

Function with initial priorities: which applies the priority function:

$$q_r(t) = r_r + t - t_0, \tag{12}$$

where r_r is a initial priority which is assigned to all class-r packets.

Analogous to the WTP scheduler with linear increasing rates (defined by Eqn. (3)), we consider the optimization problem stated by Eqn. (8). For the mean queuing delay we use:

$$E[W_r] = \frac{E[W_{FIFO}] - \sum_{i=1}^{r-1} \rho_i E[W_i](1 - (\frac{b_i}{b_r})^2)}{1 - \sum_{i=r+1}^{R} \rho_i (1 - (\frac{b_r}{b_i})^2)}, \tag{13}$$

[4] the symbols used in the Equations (11) and (12) refer to the same entities as in Eqn. (3)

for the the quadratic priority functions. For the functions with initial priorities we use the approximation

$$E[W_r]\left(1-\sum_{i=r+1}^{R}\rho_i\Big(1-exp\big(\rho(r_r-r_i)E[W_r]\big)\Big)\right)=$$

$$E[W_{FIFO}]-\sum_{i=1}^{r-1}\rho_i E[W_i](\Big(1-exp\big(\rho(r_i-r_r)E[W_i]\big)\Big). \quad (14)$$

Eqn. (13) and Eqn. (14) are given in [9,10]. Table 5 and Table 6 show the solutions of the optimization problem Eqn. (8) using genetic algorithms for both priority functions. Table 5 shows the scheduler parameters b_1, b_2, b_3 and b_4 as defined in Eqn. (11). Table 6 shows the initial priorities r_1, r_2, r_3 and r_4 as defined in Eqn. (12). For both priority functions, we also show the difference between required and achieved delay ratios of consecutive traffic classes. The results are calculated for:

- the utilizations 70%, 75%, 80%, 85%, 90%, 95% and 100%,
- the traffic load distribution [25%,25%,25%,25%] and
- the required delay ratio 2.

Table 5. Results of the optimization using genetic algorithms for quadratic priority functions

ρ	b_1	b_2	b_3	b_4	$\frac{E[W_1]}{E[W_2]}-\frac{\delta_1}{\delta_2}$	$\frac{E[W_2]}{E[W_3]}-\frac{\delta_2}{\delta_3}$	$\frac{E[W_3]}{E[W_4]}-\frac{\delta_3}{\delta_4}$
70%	1.0	40.05	160.21	1000639	0.0	-0.67	-0.01
75%	1.0	22.61	243.94	1000508	0.0	-0.36	0.0
80%	1.0	13.04	258.06	47651	0.0	0.0	0.0
85%	1.0	7.99	72.88	1180.5	0.0	0.0	0.0
90%	1.0	5.87	36.16	279.43	0.0	0.0	0.0
95%	1.0	4.71	22.60	117.07	0.0	0.0	0.0
100%	1.0	2.0	4.0	8.0	0.0	0.0	0.0

The results using the quadratic priority functions show comparable results to those achieved using linear priority functions (i.e. WTP scheduler).

Table 6. Results of the optimization using genetic algorithms for functions with initial priorities

ρ	r_1	r_2	r_3	r_4	$\frac{E[W_1]}{E[W_2]} - \frac{\delta_1}{\delta_2}$	$\frac{E[W_2]}{E[W_3]} - \frac{\delta_2}{\delta_3}$	$\frac{E[W_3]}{E[W_4]} - \frac{\delta_3}{\delta_4}$
70%	1.0	27.11	3680.6	4187.9	0.0	0.08	-0.63
75%	1.0	20.0	35.81	15615	0.0	0.02	-0.48
80%	1.0	15.6	25.39	38.18	0.0	0.22	-0.26
85%	1.0	12.67	19.83	25.370	0.0	0.21	-0.24
90%	1.0	10.58	16.19	20.02	0.0	0.20	-0.22
95%	1.0	9.06	13.653	16.59	0.0	0.20	-0.21
100%	1.0	7.24	11.19	14.28	0.0	0.08	-0.09

The results using initial priorities are characterized by a higher optimization error. The used approximation could be one of the reasons. An improvement of the average delay ratios at an utilization equal to 75% and 70% cannot be noticed with both priority functions. This strengthens the assumption that the required delay differentiation parameters are not achievable at a utilization about 70% with work-conserving schedulers (see also [2]). Due to the simplicity of the WTP scheduler in comparison to the scheduler using quadratic priorities[5] and the lack of improvements, we don't see obvious reasons to further investigate this class of priorities for proportional delay differentiation.

5 Conclusion

In this paper we proposed a new approach of dynamic priority scheduling for implementing the proportional delay differentiation model. In contrast to earlier results, our approach applies equally well to high and moderate load conditions. The scheduling parameters are determined by "offline" computation and are operationally adopted by table look-up or interpolation func-

[5] at least one multiplication less for each priority calculation

tions. Based on formally stated optimization problems, genetic algorithms have been applied that yield accurate results under fairly general conditions which had not been achieved before. As extensions to earlier solutions, the proposed algorithm allows for more than two service classes to be considered as well as for dealing with extended load conditions. Some implementation issues have been addressed in this paper. Our results indicate that some further time-dependent priority functions, such as quadratic functions or functions with initial priorities, do hardly bear any advantage over linearly time-dependent priorities (i.e. WTP) for implementing the proportional delay differentiation model. In particular, no extension of the feasibility conditions could be achieved in these cases.

Future work will deal with performing simulation studies for validation purposes. Studying the feasibility conditions of the proportional delay differentiation may need more attention. More effort will also be devoted investigating the suitability of various interpolation functions and the applicability of measurement techniques. The impact of the infinite buffer approximation should be estimated.

References

1. *L.Essafi, G.Bolch, A.Andres:* "An Adaptive Waiting Time Priority Scheduler for the Proportional Differentiation Model", accepted paper in ASTC HPC'01, Seattle, Apr. 2001
2. *C.Dovrolis, D.Stiliadis, P.Ramanathan:* "Proportional Differentiated Services: Delay Differentiation and Packet Scheduling", ACM SIGCOMM '99, Cambridge, MA, Sep. 1999
3. *C.Dovrolis, P.Ramanathan:* "A Case for Relative Differentiated Services and the Proportional Differentiation Model", IEEE Network, September/October 1999
4. *P.P.White:* "RSVP and Integrated Services in the Internet: A Tutorial", IEEE Comm. Mag, May 1997

5. *C.C.Li, S.L.Tsao, M.C.Chen, Y.Sun, Y.M.Huang:* "Proportional Delay Differentiation Service Based on Weighted Fair Queuing", Proceedings of Ninth International Conference on Computer Communications and Networks, 2000
6. *M.K.H.Leung, J.C.S.Lui, d.K.Y.Yau:* "Characterization and Performance Evaluation for Proportional Delay Differentiated Services", Proceedings of IEEE International Conference on Network Protocols, Osaka Japan, Nov. 2000
7. *D.Black, S.Blake, M.Carlson, E.Davies, Z.Wang and W.Weiss:* "An Architecture for Differentiated Services", IETF RFC 2475, Dec. 1998
8. *K.Nichols, S.Blake, F.Baker, D.Black:* "Definition of the Differentiated Services Field (DS Field) in IPv4 and IPv6 Headers", IETF RFC 2474, Dec. 1998
9. *G.Bolch, S.Greiner, H.de Meer, K.S.Trivedi:* Queuing Networks and Markov Chains : Modeling and Performance Evaluation With Computer Science Applications, John Wiley & Sons, 1998
10. *G.Bolch, W.Bruchner:* "Performance evaluation of symmetrical multiprocessor systems with time-dependent priorities", Elektronische Rechenanlagen, No 26, Volume 1, 1984
11. *T.Bäcker:* "Evolutionary Algorithms in Theory and Practice", Oxford University Press, 1996