A Multi-constrained QoS Aware Scheduler for Class-based IP Networks

Pedro Sousa, Paulo Carvalho and Vasco Freitas
Universidade do Minho, Departamento de Informática, 4710-057 Braga, Portugal
email: {pns,pmc,vf}@di.uminho.pt

Abstract: This article presents a novel modular scheduler with powerful semantics able to differentiate simultaneously multiple QoS metrics in class-based IP networks. In opposition to traditional scheduling mechanisms, this scheduler encompasses rate, loss and delay differentiation capabilities in a flexible way. This behaviour stems from new relative and mixed differentiation models able to bound QoS parameters on high sensitive traffic classes.

1 Introduction

The scheduler proposed in this work is able to achieve independent control of delay, loss and rate differentiation, through the use of two priority disciplines acting at distinct points of the proposed scheduler architecture. The delay differentiation modules are based on theoretical schemes [1] and, in particular, proportional differentiation [2, 3] is considered as one of the possible options for delay differentiation. Other differentiation schemes are also supported [4, 5, 6] by the scheduler, including an hybrid model specially devised for real-time differentiation. These delay models aggregate a packet drop mechanism in order to provide *(i)* loss differentiation or *(ii)* rate allocation with distinct work conserving behaviour. If required, for specific scenarios, the packet drop mechanism is able to provide simultaneously loss and rate differentiation semantics. The present scheduling proposal can be viewed as a modular traffic control mechanism able to be configured with distinct semantics depending on each class QoS requirements, enhancing the scheduling QoS capabilities of a network node. The proposed model has been implemented and tested in the network simulator (*NS-2*).

2 Reactive Rate Differentiation

This section focuses on one of the roles of the packet drop mechanism associated with the scheduler. The mechanism is able to induce output rate differentiation among multiple traffic classes by controlling the corresponding loads. Consider that the traffic arriving at a network node, to be forwarded to a specific output link, is classified in N distinct traffic classes contributing with individual loads $R\_in_i(t)$ with $0 \leq i \leq N-1$. From queuing theory, the server associated with the corresponding output link enters in an unbalanced state ($\rho > 1$)[1] when the total traffic class load at the input exceeds the output capacity of the link, C. This situation, illustrated in Eq. (1), leads to packet loss and to different levels of throughput share depending on the service discipline, class load and buffering resources.

$$C < \sum_{i=0}^{N-1} R\_in_i(t) \tag{1}$$

$$C \geq \sum_{i=0}^{N-1} min(R\_in_i(t), R\_max_i) \tag{2}$$

$$C \geq \sum_{i=0}^{N-1} R\_max_i \tag{3}$$

The first step in the mechanism design assures that Eq. (1) is not verified, i.e. the total arriving load does not exceed the output capacity of the server. Thus, to each $Class_i$ is assigned a value, R_max_i, which is the maximum input rate to be submitted to the server. If $R\_in_i(t)$ measures $Class_i$ input load at time t then Eq. (2) is valid and assures that the server is always under a balanced state ($\rho \leq 1$)[2]. Assuming N distinct classes, it is clear that the sum of R_max_i values should not exceed the output capacity of the server, as denoted by Eq. (3). $R\_in_i(t)$ is estimated resorting to an adaptive exponential weighted moving average, Eq. (4), where l_i^k is the length of the k^{th} packet of $Class_i$ and $\Delta t_i^k = t_{0_i}^k - t_{0_i}^{k-1}$ is the inter packet arrival time. The parameter T acts as a reference value which should have a similar order of magnitude of the time period for which the estimation module is expected to provide average rate information. In addition, the dropping mechanism was conceived so that the unused share of bandwidth of $Class_i$ is assigned to a variable $credit_i(t)$ (see Eq. (5)) representing the amount of bandwidth provided by $Class_i$ to the differentiation node for subsequent distribution. The sum of all $credit_i(t)$ values is represented by $Credits(t)$[3].

$$R\_est_i = (1 - 2^{-\frac{\Delta t_i^k}{T}}) \cdot \frac{l_i^k}{\Delta t_i^k} + 2^{-\frac{\Delta t_i^k}{T}} \cdot R\_est_i^{old} \tag{4}$$

$$credit_i(t) = \begin{cases} R\_max_i - R\_in_i(t) & if\ !(cong_i) \\ 0 & if\ (cong_i) \end{cases}$$

$$Credits(t) = \sum_{i=0}^{N-1} credit_i(t) \tag{5}$$

---

[1] $\rho = \lambda \cdot \bar{S}$, λ is the arrival rate and $\bar{S}$ the average service time.

[2] This means that, assuming enough buffering resources, the server is able to forward all traffic, i.e. on average, the R_max_i will also represent the output rate share obtained by the $Class_i$.

[3] The boolean variable, $cong_i$, is true if $R\_in_i(t) \geq R\_max_i$.

Within this work conserving behaviour, Eq. (6) determines the server operating under a balanced state. The function $limit_i(t)$ defines the maximum throughput share for each class. If the traffic class exceeds its R_max_i then $limit_i$ will increase R_max_i of a value given by a given credit distribution function, $dist(t)$[4]. The dropping mechanism associated with Eq. (6) is now ruled by Eq. (7) assuring a reactive response to load oscillations and redirecting the unused bandwidth to the congested classes[5].

$$limit_i(t) = \begin{cases} R\_max_i & if\ !(cong_i) \\ R\_max_i + dist(Credits(t)) & if\ (cong_i) \end{cases}$$

$$C \geq \sum_{i=0}^{N-1} min(R\_in_i(t), limit_i(t)) \tag{6}$$

$$drop\_prob_i(t) = 1 - \frac{limit_i(t)}{R\_in_i(t)}\ \ if\ \ (R\_in_i(t) > limit_i(t)) \tag{7}$$

3 Enhanced Delay Differentiation

This section overviews four delay differentiation models included in the proposed scheduler [5, 6]. Lets consider N classes $Class_{i(0 \leq i \leq N-1)}$ having C_0 the highest priority.

3.1 Proportional Model

Assume that $p_i(t)$ is the priority function associated with the queue i and U_i the corresponding differentiation parameter. In the proportional model this function is given by Eq. (8) , with t_{0_i} denoting the arrival time of packet to queue i and $U_0 > U_1 > ... > U_{N-1}$. Under heavy load conditions, it is expected that Eq. (9) is valid for all classes ($0 \leq i, j < N$) where $\bar{d}_i, \bar{d}_j$ are the mean queuing delays of the classes i and j, i.e. the proportional delay relations are ruled by the U_i parameters.

$$p_i(t) = (t - t_{0_i}) * U_i \tag{8}$$

$$\frac{U_i}{U_j} \approx \frac{\bar{d}_j}{\bar{d}_i} \tag{9}$$

3.2 Additive Model

The additive model differentiates queues by an additive constant as expressed by Eq. (10), with $U_0 > U_1 > ... > U_{N-1}$. The focus of this model is on the possibility of achieving additive differentiation in class delays, as expressed by Eq. (11), denoting that high priority classes may have a delay gain over low priority classes similar to the difference between the differentiation parameters.

$$p_i(t) = (t - t_{0_i}) + U_i \tag{10}$$

$$[\bar{d}_i - \bar{d}_j] \approx [U_j - U_i] \qquad (i > j) \tag{11}$$

3.3 Upper Time Limit Model

This model tries to impose a finite queuing delay, reflected by U_i (see Eq. 12) and, the lower the boundary time is, the higher the priority function slope will be. At the limit ($(t - t_{0_i}) \geq U_i$), the server is *forced*[6], to dispatch the packet waiting service. This model protects high priority classes, giving that packets remain queued for a maximum value U_i, with $U_0 < U_1 < ... < U_{N-1}$. This allows to establish delay bounds on the highest priority class and, simultaneously, achieve proportional differentiation between the other classes. For instance, $Class_1$ can be *protected* by a realistic upper time limit, and $Class_2$ and $Class_3$ by *virtual* limits (e.g. $U_2, U_3 \gg U_1$). Proportionality between $Class_2$ and $Class_3$ is obtained as explained by Eq. (8).

$$p_i(t) = \begin{cases} \frac{(t - t_{0_i})}{U_i - t + t_{0_i}} & if\ \ t < t_{0_i} + U_i \\ \infty\ \ if\ \ t \geq t_{0_i} + U_i \end{cases} \tag{12}$$

3.4 An Hybrid Delay Model

This model is useful to distinguish real-time traffic with distinct sensibilities to queuing delays and excess delays. In this model, the priority function assumes values starting from an infinity negative reaching zero when the queuing time of the packet matches the upper time parameter. After that, if the packet is still enqueued, the scheduler switches to a new working region of positive values, where the priority behaviour is ruled by a *congestion parameter*[7] which determines the slope of the priority function. The final priority function is given by (13).

$$p_i(t) = \begin{cases} \frac{\delta_t - U_i}{\delta_t} & \text{if } \delta_t < \mathrm{U}_i \\ (\delta_t - U_i) * C_i & \text{if } \delta_t \geq \mathrm{U}_i \end{cases} \tag{13}$$

with $\delta_t = t - t_{0_i}$ and $0 \leq i \leq N - 1$.

$$d_i = d_i^{\circ} + d_i^{\bullet} \tag{14}$$

The total delay (d_i) affecting $Class_i$ can be divided in two components: one induced by the priority function when it assumes negative values ($t < t_{0_i} + U_i$), which we call *upper time delay*, d_i°, and the other one when the function assumes positive values, which we call *congestion delay*, $d_i^{\bullet}$ (see Eq. (14)). The magnitude of

---

[4] An example of $dist(t)$ can be a *strict priority* function where credits are allocated to traffic classes according to their priority, i.e server credits are firstly allocated to high priority classes (see details in Sec. 5.1).

[5] Relaxed versions of the rate differentiation module are possible. For instance, this mechanism may operate only during specific probing periods or the $limit_i$ and $drop\_prob_i$ values are only computed for specific time intervals, despite the class rate estimation being continuously updated.

[6] When congestion occurs or the load of high priority classes becomes very high, the time limit may be exceeded.

[7] In this context, the term *congestion* is used in a relaxed way as it may reflect heavy load conditions in the server; heavy load conditions in $Class_i$ impairing the expected upper time limit or feasibility problems in the differentiation parameters.

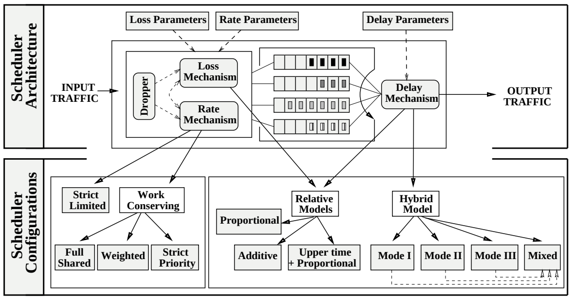

Figure 1: The scheduler architecture implemented in *NS-2*.

d_i° is controlled by U_i whereas C_i controls the magnitude of $d_i^{\bullet}$. This means that fundamental differentiation relations among classes, i.e. $d_0 \leq d_1 \leq ... \leq d_{N-1}$, can be achieved through different combinations of d_i° and $d_i^{\bullet}$, and consequently by different combinations of U_i and C_i. In summary, a distinct delay behaviour can be induced depending on the relations between the *upper time* and *congestion* delays of the traffic classes[8].

4 Enhanced Loss Differentiation

As regards packet loss, the model resorts to Eqs. (8), (10) and (12) to achieve loss differentiation. In this case, instead of using the packets queuing time, i.e. $(t - t_{0_i})$, the models use the ratio l_i/A_i, with l_i being the number of packet drops and A_i the number of packet arrivals for $Class_i$[9]. Whenever the buffer overflows, the class selected to drop a packet is the one with the lowest $p_i(t)$ value. The traffic classes are configured with loss differentiation parameters $L_0 > L_1 > ... > L_{N-1}$.

5 A Multi-constrained QoS Engine

The scheduling architecture presented in Fig. 1 aggregates all the previously explained differentiation mechanisms and was implemented in *NS-2*. This section illustrates that the proposed scheduling architecture is able to decouple the rate, loss and delays differentiation behaviour, i.e. the differentiation mechanisms can act jointly but, simultaneously, can provide independent QoS metric differentiation. Due to the high number of possible differentiation schemes this section only covers examples of specific configuration modes. The selected examples were taken from a scenario where three classes contend for a $4.5 Mbps$ capacity link, with packet lengths of 500 bytes uniformly distributed over the interval $[250, 750]$. The scheduler was tested successfully for distinct traffic sources as CBR, exponential, pareto and combinations thereof[10].

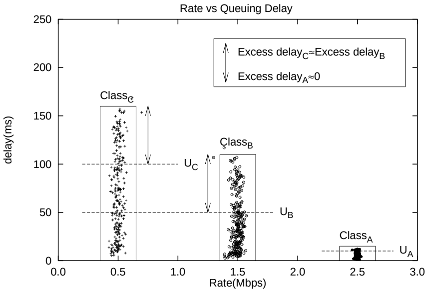

Figure 2: Rate differentiation with hybrid delay model (conf. II+III) for $(R\_max_A, R\_max_B,\ R\_max_C) = (2.5Mbps, 1.5Mbps, 0.5Mbps)$, $(U_A, U_B, U_C) = (10ms, 50ms, 100ms)$ and $(C_A, C_B, C_C) = (20, 1, 1)$.

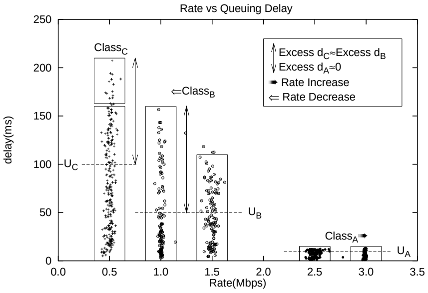

Figure 3: Strict priority rate with hybrid delay model (conf. II+III) for $(R\_max_A,\ R\_max_B, R\_max_C) = (2.5Mbps, 1.5Mbps, 0.5Mbps)$, $(U_A,\ U_B, U_C) = (10ms, 50ms, 100ms)$ and $(C_A, C_B, C_C) = (20, 1, 1)$.

5.1 Rate vs. Delay

Strict Priority Rate Model with Hybrid Delay - This example illustrates the use of the hybrid delay differentiation module and the rate differentiation module for the configuration parameters shown in Fig. 2. In the delay configuration mode, $Class_A$ is the highest protected class as regards both rate and delay violations and $Class_B$ and $Class_C$ have distinct upper time parameters but similar congestion parameters, meaning that they have similar sensibility to absorb excess delays despite having different upper time delays. Fig. 2 shows the average output rate (x-axis) and queuing delays (y-axis) obtained by the classes, clearly corroborating the expected differentiation behaviour. Fig. 3 illustrates this delay differentiation mode and the strict priority rate differentiation. The rate credits of the server are now first distributed to the high priority classes and the remaining credits, if any, are allocated to low priority classes. With this purpose, Fig. 3 plots the differentiation behaviour when $Class_B$ decreases its rate to $1Mbps$. As plotted in Fig. 3, only $Class_A$, which has the highest priority, has assigned extra bandwidth, being shifted to the right side of the graph with an offset of $0.5Mbps$, exactly the share provided by $Class_B$. As a consequence, a new delay distribution is achieved by the server and both $Class_B$ and $Class_C$ delays increase. For $Class_C$, all plots are

---

[8] For instance, in Sec. 5.1, the classes are configured with distinct upper time parameters, having two of them similar congestion parameters. In the example included in Sec. 5.2, the classes have distinct congestion parameters, having two of them similar upper time parameters.

[9] l_i and A_i are measured for a specific time interval which can be configured in the differentiation node.

[10] For a simulation period of $120s$ with a QoS metric evaluation interval of $1s$ and the overall class load above the link capacity to force packet loss.

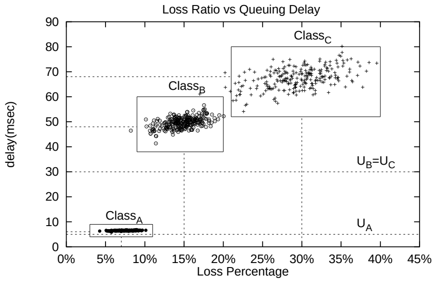

Figure 4: Proportional loss and hybrid delay model for Conf. I+II, with $(U_A, U_B, U_C) = (5ms, 30ms, 30ms)$, $(C_A, C_B, C_C) = (40, 2, 1)$, $(L_A, L_B, L_C) = (4, 2, 1)$.

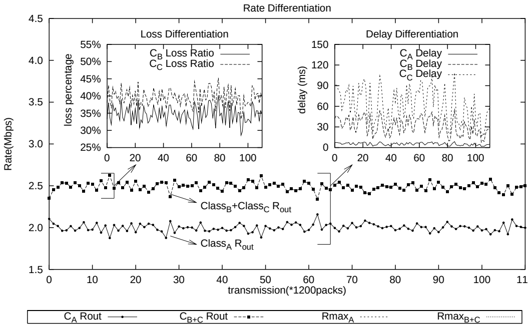

Figure 5: Rate differentiation with additive loss and upper time delay model with $(R\_max_A, R\_max_{B+C}) = (2Mbps, 2.5Mbps)$, $(L_B, L_C) = (0.05, 0)$, $(U_A, U_B, U_C) = (10ms, 100ms, 200ms)$.

still centered on $0.5Mbps$ as this class has not received any extra bandwidth share. The increase in $Class_C$ excess delays is represented by a second box above the previous obtained delays. The magnitude of $Class_B$ and $Class_C$ excess delays is still similar even after the rate sharing, while $Class_A$ delay violations keep a low value due to its high C_A parameter.

5.2 Loss vs. Delay

Proportional Loss and Hybrid Delay - In this example the classes were configured to have proportional loss differentiation with $(L_A, L_B, L_C) = (4, 2, 1)$. They are also configured with the hybrid delay differentiation mechanism in the mixed configuration I+II with $(U_A, U_B, U_C) = (5ms, 30ms, 30ms)$ and $(C_A, C_B, C_C) = (40, 2, 1)$. This means that a proportional packet loss is expected and, due to a very high congestion parameter, $Class_A$ should have queuing delays close to $5ms$. In addition, the congestion delays of $Class_C$, i.e. the difference between the obtained delays and the target delay of $30ms$, should be twice the congestion delay of $Class_B$, which have a similar delay target of $30ms$, but a congestion parameter two times higher than $Class_C$. This behaviour is illustrated in Fig. 5.2 showing the delay vs. loss experienced by the classes during the simulation.

5.3 Rate vs. Loss vs. Delay

Rate Differentiation with Additive Loss and Upper Time Delay - This example illustrates the three differentiation modules acting together. It was assumed that $Class_A$ is used for high loss and time sensitive traffic and the traffic load is controlled at network edges imposing to differentiation nodes a bandwidth allocation of $2Mbps$ for the class. $Class_B$ and $Class_C$ are used for low priority traffic and, depending on the network conditions, packet loss is likely to occur. In this context, the rate parameters were configured as $(R\_max_A, R\_max_{B+C}) = (2Mbps, 2.5Mbps)$. The additive model was used to guide loss differentiation between $Class_B$ and $Class_C$ with $(L_B, L_C) = (0.05, 0)$, meaning that $Class_B$ should experience a loss percentage which is 5% lower than the obtained by $Class_C$. Finally, the upper time model was used to limit the queuing delay of $Class_A$ to a maximum value of $10ms$, with proportional relations between $Class_B$ and $Class_C$. As depicted in Fig. 5, the results show the correctness of this configuration.

6 Conclusions

This article presents a modular scheduler architecture providing enhanced rate, loss and delay differentiation behaviour. The diversity of the configuration modes for the three QoS metrics turns the proposed scheduler in an useful component to be used in network scenarios aiming at QoS differentiation. The proposed scheduler allows to achieve independent QoS metrics differentiation behaviour, avoiding coupling effects which may affect other differentiation mechanisms. Due to the enhanced differentiation semantics, many combinations of rate, loss and delay differentiation behaviour are possible using a small set of simple and intuitive configuration paramaters.

References

[1] G. Bolch et al. *Queueing Networks and Markov Chains - Modeling and Performance Evaluation with Computer Science Applications*. John Wiley and Sons INC., 1998.

[2] C. Dovrolis et al. A case for relative differentiated services and the proportional differentiation model. *IEEE Network Magazine*, 1999.

[3] C. Dovrolis et al. Proportional differentiated services: delay differentiation and packet scheduling. *IEEE/ACM Transactions on Networking*, 10(1), Feb. 2002.

[4] P. Sousa et al. End-to-end delay differentiation of IP traffic aggregates using priority queueing models. In *Proc. of the IEEE Workshop on High Performance Switching and Routing (HPSR2002)*, pages 178–182, Kobe, Japan, May 26-28 2002.

[5] P. Sousa et al. Tunning delay differentiation in IP networks using priority queueing models. In E. Gregori et al, editor, *Proc. 2^{nd} International IFIP-TC6 Networking Conference*, pages 709–720, Pisa, Italy, 2002. LNCS 2345, Springer-Verlag.

[6] Pedro Sousa et al. Scheduling Time-Sensitive IP Traffic. In G. Goos et al, editor, *Proc. 6th IFIP/IEEE MMNS International Conference*, pages 368–380, Northern Ireland, Belfast, 2003. LNCS 2839, Springer-Verlag.