A Reservation-Based Scheduling Mechanism for Fair QoS Provisioning in Packet-Based Networks

S. Wittevrongel*, S. De Vuyst†, C. Sys*, H. Bruneel*
*SMACS Research Group,
Ghent University, Dept. of Telecommunications and Information Processing,
St.-Pietersnieuwstraat 41, B-9000 Gent, Belgium.
Phone: +32-9-2648901, Fax: +32-9-2644295
Email: {sw,cosys,hb}@telin.ugent.be
†Ghent University, Dept. of Industrial Management,
Technologiepark 903, B-9052 Zwijnaarde, Belgium.
Phone: +32-9-2645506, Fax: +32-9-2645847
Email: {stijn.devuyst}@ugent.be

***Abstract*—In this paper, we present a mechanism for scheduling multi-class traffic in a node of a packet-based communication network. We focus in particular on the output queues used for contention resolution at the output links of a network node. The mechanism, referred to as R-scheduling, is based on the use of in-queue reservations for future arriving data packets. Each traffic class has a given number of class-dedicated reservations at its disposal. If a data packet is enqueued, it seizes its reservation which is closest to the head of the queue by taking its place in the queue, after which a new reservation belonging to the same traffic class as the enqueued packet is created at the tail of the queue. The performance of the R-scheduling mechanism is investigated by means of extensive simulations for a wide parameter space. Based on these simulations, a meta-model is constructed that allows to translate given QoS (Quality of Service) targets into working parameters for the R-scheduling mechanism. We show that R-scheduling offers unique properties in terms of the QoS delivered to each of the traffic classes. In particular, it allows shaping the distributions of the queueing delay perceived by each traffic class in order to achieve a target spacing of their delay quantiles. Other important features that can be effectuated by R-scheduling are: isolation of traffic classes through temporal priority, mitigation of packet starvation for lower-priority traffic classes and early provision of tight delay bounds. The mechanism has low complexity and is suitable for dynamic applications.**

***Index Terms*—Quality of service; Reservation-based scheduling; Delay quantile spacing**

I. Introduction

In a packet-based communication network, all incoming packets in a network node with the same next-hop destination must be transmitted over a single output line towards the next node. Because of the limited bandwidth, an output queue is necessary to resolve contention at the output link. Current networks convey very diverse and heterogeneous traffic and should therefore be able to differentiate the QoS (Quality of Service) according to the specific needs of the individual traffic streams, i.e., deliver a better service for those applications that need it, in exchange for a worse service for other applications. This means the network nodes must be QoS-aware systems that treat packets differently depending on the application they belong to. In the DiffServ framework [1], streams are aggregated into a limited number, say M, of traffic classes, based on the DSCP (Differentiated Services Code Point) field in the IP header, and the handling of packets depends on the traffic class they belong to. Many QoS-aware algorithms exist, both at the application level (e.g. traffic shaping, congestion notification) and at the level of the network (e.g. congestion avoidance, active queue management, specific packet discarding and packet scheduling mechanisms) [2]. In this paper, the focus is on the latter, i.e., the scheduling algorithm for the packets in the queues at the output links of a network node.

A router's scheduling algorithm is responsible for determining which available packet is the next to be transmitted on the output link. The key technical challenge is to schedule packets in such a way that the joint QoS requirements of the different traffic classes are met as closely as possible, given a certain limited output bandwidth. Targeted QoS criteria usually are per-class output bandwidth, mean packet delay, delay jitter and packet loss probability. QoS provisioning further entails the amount of isolation, fairness, policing and the degree in which these service levels can be guaranteed and enforced.

There exists no single algorithm that can guarantee an *ideal* distribution of resources in all possible circumstances and under all output bandwidth constraints. Any measure taken to enhance a QoS criterion for one class necessarily degrades the performance of other classes to some extent. This allows for a fair amount of QoS tailoring: performance with regard to a high-utility criterion can be improved at the cost of a worse performance for low-utility criteria. Therefore, scheduling is always a matter of aiming for an agreeable trade-off in satisfying the targeted QoS requirements of the different traffic classes. Seen in this way, any particular scheduling mechanism implements only a subset of the possible ways in which multi-class scheduling can be done, thus covering only part of the possible trade-off choices that can be made. A scheduler is always based on a certain concept of 'fairness' [3], [4] between the different traffic classes. Usually, fairness entails that a predefined proportionality has to be maintained between the classes for a certain performance metric. However, it may also

978-0-9836283-9-2

be defined in a max-min sense, i.e. the goal is to maximise the smallest metric over the classes, or the proportional weights may tend to infinity as in the case of priority scheduling. The involved metric for the traffic classes may be the mean perceived delay, the obtained output bandwidth, the occupied queueing space, the loss probability, the lateness of packets or jobs with deadlines, etc. Advanced scheduling algorithms will often consider several of these metrics at the same time.

This paper presents a scheduling mechanism, referred to as R-scheduling, that makes use of in-queue reservations for future arriving data packets. Each traffic class has a given number of class-dedicated reservations at its disposal. If a data packet is enqueued, it seizes its reservation which is closest to the head of the queue by taking its place in the queue, after which a new reservation belonging to the same traffic class as the enqueued packet is created at the tail of the queue. As we will show, the mechanism offers unique properties in terms of the service delivered to each of the traffic classes. The mechanism is primarily designed for shaping the distributions of the queueing delay perceived by each traffic class, mainly with regard to their quantiles. If a certain spacing of these quantiles is targeted (e.g. in case of two traffic classes, the delay 5%-quantile of class 1 must be 25 ms smaller than the delay 5%-quantile of class-2 packets), setting the parameters of R-scheduling accordingly will achieve this. We refer to this main functionality of R-scheduling as *delay quantile fairness*. However, other possible features that can be effectuated by R-scheduling are: isolation through temporal priority, mitigation of starvation and early provision of tight delay bounds, all of which will be discussed in the paper.

The outline of the paper is as follows. In Section II, we give an overview of prior work on class-based scheduling mechanisms and explain how R-scheduling is different from any existing technique. The operation of the R-scheduling mechanism is described in detail in Section III. Section IV presents a meta-model for the parameter dependence of the delay α-quantiles of the different traffic classes; this meta-model has been tested and validated via extensive simulations. Section V concentrates on the main functionality of R-scheduling, which is delay quantile spacing, and indicates how given QoS targets can be translated into working parameters for the R-scheduler. Other nice features of R-scheduling are discussed in Section VI. Finally, concluding remarks are given in Section VII.

II. Related work

Various multi-class scheduling mechanisms for packet-based networks have been proposed in the past and some of them are extensively used in present-day routers. Their goal is to provide QoS differentiation, as opposed to the simple FIFO (First-In First-Out) scheduling where all packets are regarded as equally important. In the realm of class-based scheduling the main classical players are absolute priority, proportional-rate scheduling and deadline-based scheduling.

Absolute priority (AP), which is also known as Head-of-Line (HoL) scheduling, see e.g. [5]–[7], provides an extreme QoS trade-off in the sense that minimizing the delay of class-i packets ($i = 1, \ldots, M$) is infinitely more important than the delay of packets of classes $i + 1, \ldots, M$. A separate queue for each class is used and the scheduler always visits the queue that has the lowest class number and is non-empty. The drawback of AP is that no further control over the QoS differentiation is possible and that it is only practical for few classes with clearly distinct service requirements. Additionally, classes with lower priority may experience a dramatically high delay, especially when the partial load of the higher-priority classes is high, an effect known as packet starvation. R-scheduling inherently avoids this phenomenon.

Some modifications to soften the rigidity of AP have been proposed. In [8], a probabilistic priority scheme is presented that assigns a small probability p_i to class i by which the server may service a class-$(i+1)$ packet, even though class-i packets are available. Another idea is to allow packets to promote to a higher-priority class under certain well-specified conditions, or conversely, to degrade to a lower class; this is referred to with the term priority jumps, see e.g. [9]–[11]. Nevertheless, the resulting per-class delay distributions of these solutions are poorly understood and rules to translate delay targets to working parameters are lacking.

Proportional-rate schedulers, also called *fair-queueing* schedulers (FQ), consider it fair to guarantee each traffic class a predefined fraction of the output bandwidth. In other words, FQ-schedulers assign the available output rate proportionally, in line with the Proportional-DiffServ framework [12] for QoS provisioning in IP networks. Implementations always use a separate queue for each traffic class (as does AP) and dequeueing is done by visiting these queues in some periodical manner (or otherwise) such as to guarantee rate proportionality as much as possible. Weighted Fair Queuing (WFQ) [13], [14] optimally achieves proportionality but is computationally very complex, as it needs to recalculate virtual finish times for *all* queues each time a packet is either enqueued or dequeued anywhere. Many modifications exist: start-time Fair Queueing [15], self-clocked Fair Queueing [16], worst-case Fair Weighted Fair Queueing [17], frame-based Fair Queueing, Weighted Round-Robin, Deficit Round-Robin [18], all of which have their specific advantages, see [19], [20]. Unlike FQ however, R-scheduling focusses on proportionally spacing the delay quantiles instead of the output rate. The objective is therefore entirely different.

Deadline-based schedulers require each packet to have a strict deadline by which that packet should have been transmitted at the very latest. The QoS provisioning is considered fair as long as all deadlines can be met. If a packet cannot be transmitted before its deadline, the time it is overdue is defined as its lateness. The Earliest Deadline First (EDF) scheduler, see e.g. [21], [22], always transmits the packet with the nearest deadline and is known to achieve the smallest possible overall lateness. The drawback of such deadline-based schedulers, however, is their need to maintain a priority list of all queued packets, resulting in a high complexity. As with previous solutions, the actual per-class delay distribution, or even the

978-0-9836283-9-2

per-class lateness, of deadline-based schedulers is hard to control or predict. Hence, the delay quantile proportionality we obtain with R-scheduling is not achieved with deadline-based schedulers. Additionally, R-scheduling does not require each packet to have a deadline.

Our R-scheduling mechanism relies on the use of in-queue reservations, which has some precedents in the prior art. In [23], a Priority Forcing Scheme (PFS) is suggested. This scheduler requires the involved applications to make the reservations by sending R-packets (reservation packets) through the network in advance of D-packets which contain the actual data. So contrary to our solution, only network users (applications) that are aware of the Priority Forcing Scheme used in the network nodes can benefit from it. It is also clear that malicious users can easily flood the network with R-packets, thereby gaining access to all of the resources at the expense of the other users. Our R-scheduling mechanism on the contrary provides QoS differentiation as determined by the network administrator and not the applications. Another important difference is that with PFS, once an R-packet is seized by a D-packet, no new reservation is made. In [24], a queueing system with two traffic classes and a single reservation for the high-priority class is described (the reservation is reused, like the reservations in our R-scheduling mechanism). The resulting delay differentiation is therefore very small, static and uncontrollable. The performance of this queueing system is studied further in our own work [25]–[28]. In [29], the same system with two traffic classes is considered, but with multiple reservations for the high-priority class. None of these papers however provide clues as to how delay quantile spacing should be achieved or how early delay bounds should be estimated.

III. Operation of the R-scheduling mechanism

We consider a queue for the storage of data packets at an output link of a network node. Time is assumed to be divided into fixed-length intervals called slots. Arriving packets may have different QoS requirements; there are M different traffic classes, each with a particular required QoS. Contrary to many existing schedulers, where dedicated queues are used for the traffic classes, a *single* queue is considered in which all data packets of all traffic classes are ranked. Besides the data packets themselves, the queue can also contain class-dedicated reserved places or 'reservations'. More specifically, we allocate for each traffic class i, $i = 1, \ldots, M$, a number of reservations N_i in the queue. The assigned numbers of reserved places are thus given by the vector $\mathbf{N} = [N_1 \ \ldots \ N_M]^T$. So, if $N = N_1 + \ldots + N_M$ then an empty queue (i.e., without data packets) would contain reservations in the first N positions.

A data packet of traffic class i arriving at the queue will be enqueued in either of two manners: (1) if traffic class i has non-zero reservations at its disposal ($N_i > 0$), the packet 'seizes' the first reservation (i.e., the one closest to the head of the queue) of its traffic class, or (2) if traffic class i has no reservations ($N_i = 0$), the packet is placed at the end of the queue in the usual FIFO way. Seizing a reservation of traffic class i at a certain position means that the data packet of class i is stored in this position in the queue *and* that a new class-i reservation is made at the tail of the queue. This way, it is assured that the number of reservations for each traffic class is always kept constant.

When the output link becomes available at the beginning of a slot, a packet from the queue will be chosen to be transmitted. This will be the data packet with the lowest position in the queue (i.e., the packet closest to the *head* position), say at position n. The chosen data packet is dequeued and all entries (both data packets and reservations) with positions higher than n shift one position to close the gap. If there are no reservations before the data packet in the lowest position, that data packet will be at the head position; reservations with a position smaller than n will not be affected by the transmission of the data packet.

The evolution of the queue according to the above rules is illustrated in Figure 1 for $M = 3$ traffic classes. This figure shows the time axis divided into slots. Above the axis we see how data packets and reservations are ranked in the queue at the beginning of the subsequent slots (with positions closer to the axis being closer to the head of the queue). The arrows above the axis indicate the data packets arriving at the queue during the different slots. Below the axis, the data packets under transmission are shown, with arrows indicating the transmission times.

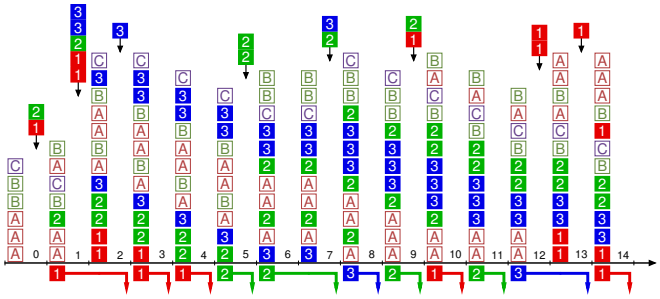

Fig. 1. Example of a queue with R-scheduling in case of $M = 3$ traffic classes depicted as 1, 2, 3 respectively. There are $N_1 = 3$ reservations A, $N_2 = 2$ reservations B and $N_3 = 1$ reservation C.

IV. Meta-model and simulation results

We have developed a simulation tool to evaluate the performance characteristics of the R-scheduling mechanism. The tool performs discrete-time simulations of a router output queue with M classes of arriving traffic, while keeping record of various performance statistics such as per-class distributions of delay and queue content. The numbers of packet arrivals per slot for each traffic class are assumed to be independent from slot to slot and Poisson distributed. The service times of all the packets are constant and equal to one time slot.

Several thousands of simulations have been performed for a wide parameter space (various numbers of traffic classes, various numbers of reservations for each traffic class, various values for the total system load and for the partial loads of each traffic class). Each simulation run took 10 billion time slots

and we have focused in particular on the tail behaviour of the delay distributions of the different traffic classes. An algorithm was developed that can automatically detect the decay rate and the intercept (in logarithmic scale) of the geometric tail in the obtained delay histograms. For every simulation, the parameters and the tail characteristics for each traffic class were stored in a database. By querying this database of several thousands of simulations, we have constructed a meta-model that shows the parameter dependence of the delay α-quantiles of the different traffic classes. The α-quantile d_i^α of the delay d_i perceived by packets of traffic class i is the value d_i^α such that a fraction α of the packets of class i has a delay of at least this value, i.e.,

$$\mathrm{Prob}[d_i \geq d_i^\alpha] = \alpha\,.$$

According to our meta-model, the delay α-quantile for traffic class i is, for small enough α, given by

$$d_i^\alpha = \frac{\log\alpha}{r_{\text{FIFO}}} + C_1 + C_2 \sum_{\substack{j=1\\ j\neq i}}^{M} N_j + C_3(L, L_i)\, N_i\,,$$

or equivalently,

$$d_i^\alpha = \frac{\log\alpha}{r_{\text{FIFO}}} + C_1 + C_2\ (N - N_i) + C_3(L, L_i)\, N_i\,. \qquad (1)$$

The constant r_{FIFO} in (1) is the logarithmic tail decay rate of the corresponding FIFO system and depends only on the total system load. Under our assumption of Poisson traffic with a mean of λ_i arriving packets per slot for class i and single-slot transmission times, it is known that the delay distribution of an arbitrary packet in case of FIFO has generating function [30],

$$D(z) = \frac{1-\lambda}{\lambda}\, z\, \frac{e^{\lambda(z-1)} - 1}{z - e^{\lambda(z-1)}}\,,$$

with $\lambda = \sum_{i=1}^{M} \lambda_i$. This is in fact a discrete-time version of the well-known Pollaczek-Khinchine formula. The expression has a singularity for $z = z_d$ which is the unique real positive root larger than 1 of the denominator, i.e. $z_d = e^{\lambda(z_d-1)}$. An expansion of $D(z)$ around this singularity gives as asymptotic delay distribution $\mathrm{Prob}[d = n] \approx -\theta z_d^{-n-1}$ and consequently,

$$\mathrm{Prob}[d \geq n] \approx -\theta \frac{1}{z_d - 1} z_d^{-n}\,.$$

Here, θ is the residue of $D(z)$ in this expansion, i.e.

$$\theta = \mathrm{Res}_{z_d} D(z) = \lim_{z\to z_d} (z - z_d) D(z) = \frac{1-\lambda}{\lambda} z_d \frac{z_d - 1}{1 - \lambda z_d}\,.$$

Therefore, on a logarithmic scale, we have

$$\begin{aligned}\log(\mathrm{Prob}[d \geq n]) &= \log(-\frac{\theta}{z_d - 1}) - n \log z_d \\ &= \log(\frac{1-\lambda}{\lambda}\ \frac{z_d}{\lambda z_d - 1}) - n \log z_d\,, \qquad (2)\end{aligned}$$

and the geometric tails will look like straight lines with slope $r_{\text{FIFO}} = -\log z_d$.

The parameters C_1 and C_2 in (1) are constants. The value of the parameter C_3 depends on the total system load (L) and on the partial load (L_i) of traffic class i. These model parameters are extracted from the database of simulation results using the least squares method. For instance, for two traffic classes and a total load of 95%, a set of around 1550 simulation results was used, corresponding to different numbers of reservations and different relative loads of class i. This resulted in respective values 1.1 and 1.0 for the constants C_1 and C_2. Some values for the parameter C_3 are given in Table I. Note that the value C_3 is negative: increasing the number of reservations N_i of class i will decrease the delay α-quantiles d_i^α of that class.

TABLE I
DEPENDENCE OF PARAMETER C_3 ON TOTAL LOAD (L) AND RELATIVE LOAD ($RL_i = \frac{L_i}{L}$).

relative load	10%	20%	30%	40%	50%	60%
total load = 95%	-6.1	-3.2	-2.0	-1.3	-0.9	-0.7
total load = 90%	-4.7	-2.7	-1.7	-1.2	-0.8	-0.6
total load = 85%	-3.8	-2.3	-1.5	-1.1	-0.8	-0.6

In order to illustrate the performance of the R-scheduling mechanism and the validity of the constructed meta-model, we now present a selection of simulation results for a system with four traffic classes ($M = 4$). The system load, i.e., the average number of arriving packets per slot, is 95%. The relative loads (RL) of the different packet flows are respectively 10%, 20%, 30% and 40%. Thus, the respective partial loads are: 9.5%, 19%, 28.5%, and 38%.

In Figures 2–6, $\log(\mathrm{Prob}[d_i \geq n])$ is plotted versus n, for the different traffic classes, i.e., for $i = 1, 2, 3, 4$. The numbers of reservations for each traffic class (N_1, N_2, N_3 and N_4) vary from figure to figure. In all the figures, dots correspond to results from simulation and solid (dashed) lines correspond to the aforementioned meta-model (i.e., equation (1) for the delay α-quantile dependence on α). Note that all shown geometric delay tails have slope $r_{\text{FIFO}} = -0.1017$ in logarithmic scale as predicted by (2), a property which we will exploit in the next section.

Figure 2 corresponds to the case where there are no reservations ($N_1 = N_2 = N_3 = N_4 = 0$). We observe that in this case, the system behaves as a FIFO system, with no differentiation between the different traffic classes.

From the moment reservations are introduced, differentiation between the traffic classes follows. We discuss two examples where all traffic classes have the same number of reservations. In Figure 3, all traffic classes have two reservations ($N_1 = N_2 = N_3 = N_4 = 2$). Comparing two traffic classes with different partial loads, the reservations of the traffic class with the lower load will, on average, be in lower positions (i.e., be further in the queue, closer to the head of the queue) and thus an arriving packet of that traffic class will seize, on average, lower positions in the queue. Thus, for traffic classes with the same number of reservations, the lower the relative load RL_i of a particular traffic class i, the lower its delay α-quantiles d_i^α.

In Figure 4, all traffic classes have five reservations ($N_1 = N_2 = N_3 = N_4 = 5$). The delay quantile spacings are now a factor 2.5 larger than in Figure 3. In general, if all the numbers of reservations are multiplied with the same factor, the delay quantile spacings increase with that factor.

Increasing the number of reservations of a particular traffic class while keeping the numbers of reservations of the other classes, will decrease that class's delay quantiles, and vice versa. In Figure 5, the numbers of reservations of the different traffic classes have been tailored in such a way ($N_1 = 2$, $N_2 = 5$, $N_3 = 10$ and $N_4 = 20$) to inverse the 'natural' order. In this example, the higher the relative load of a traffic class i, the lower its delay α-quantiles d_i^α (for small enough α).

From equation (1), it follows that the tail behaviour of a traffic class i depends only on the number of reservations of that class N_i and the total number of reservations N. Figure 6 illustrates this feature: compared to the previous Figure 5, only the numbers of reservations of traffic classes 2 and 4 are different, while the total number of reservations remains the same ($N_1 = 2$, $N_2 = 15$, $N_3 = 10$ and $N_4 = 10$). As a consequence, the tail behaviour of traffic classes 1 and 3 is the same in Figure 5 and Figure 6, i.e., for small enough α their delay α-quantiles d_i^α remain the same.

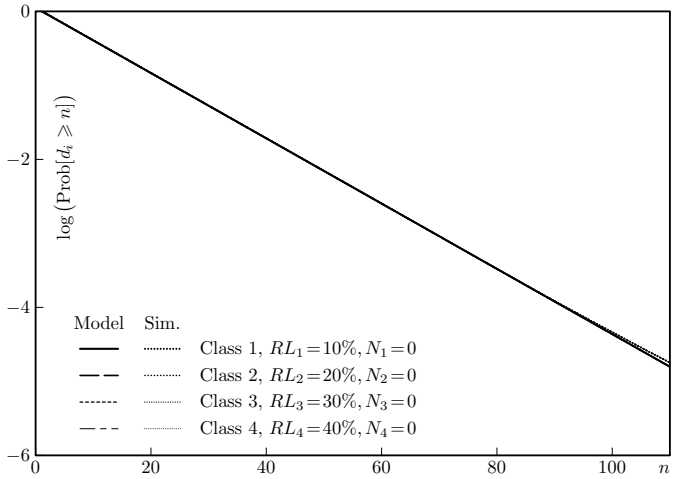

Fig. 2. $\log(\text{Prob}[d_i \geq n])$ versus n: simulation results (dots) and meta-model results (solid lines) for $N_1 = 0$, $N_2 = 0$, $N_3 = 0$ and $N_4 = 0$.

V. Delay quantile spacing

The meta-model established in the previous section indicates that R-scheduling exhibits an attractive statistical property regarding the tail behaviour of the per-class delay distributions. As already observed from the discussed examples in the previous section, it turns out that the decay rate of the tails is the *same for all classes* and equal to the decay rate in the equivalent FIFO system (i.e., if no QoS-aware scheduling were used):

$$\log(\text{Prob}[d_i \geq n]) \propto r_{\text{FIFO}} \cdot n\,. \tag{3}$$

Note that for the case of two traffic classes ($M = 2$) this property can also be shown analytically, as explained in [29],

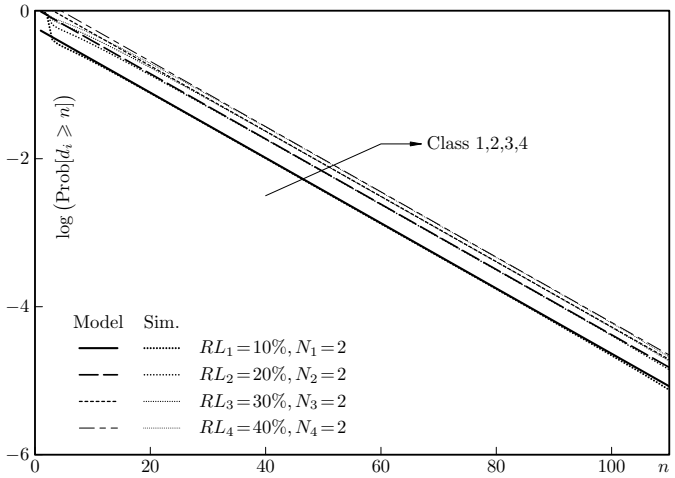

Fig. 3. $\log(\text{Prob}[d_i \geq n])$ versus n: simulation results (dots) and meta-model results (solid lines) for $N_1 = 2$, $N_2 = 2$, $N_3 = 2$ and $N_4 = 2$.

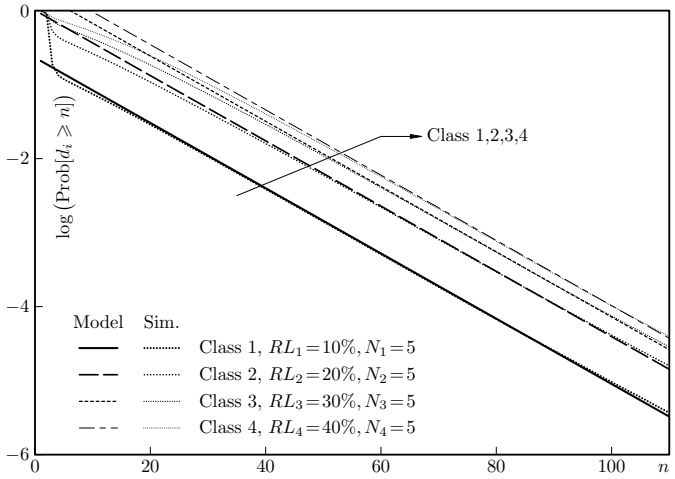

Fig. 4. $\log(\text{Prob}[d_i \geq n])$ versus n: simulation results (dots) and meta-model results (solid lines) for $N_1 = 5$, $N_2 = 5$, $N_3 = 5$ and $N_4 = 5$.

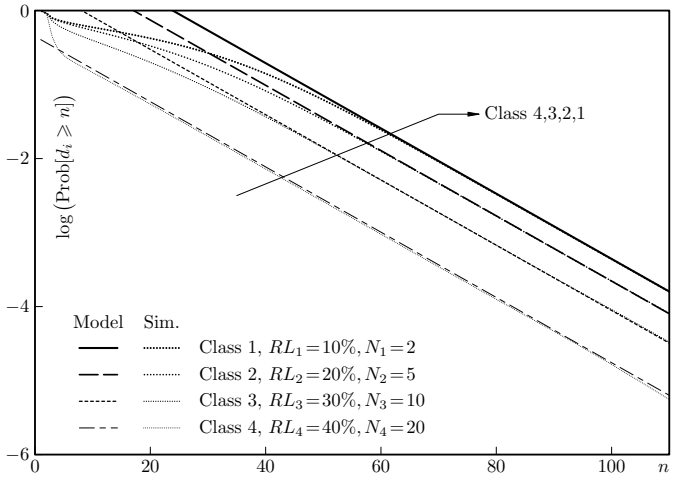

Fig. 5. $\log(\text{Prob}[d_i \geq n])$ versus n: simulation results (dots) and meta-model results (solid lines) for $N_1 = 2$, $N_2 = 5$, $N_3 = 10$ and $N_4 = 20$.

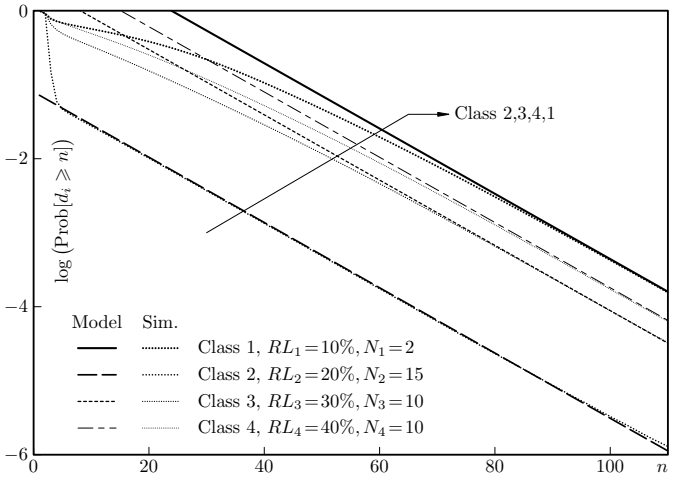

Fig. 6. $\log(\text{Prob}[d_i \geq n])$ versus n: simulation results (dots) and meta-model results (solid lines) for $N_1 = 2$, $N_2 = 15$, $N_3 = 10$ and $N_4 = 10$.

[30]. Based on (3), for small enough α (5%, 1%, ...), the difference between the delay α-quantiles of traffic classes i and $i+1$ is independent of α and a (linear) function of the numbers of reservations N_i and N_{i+1} of the respective classes. The spacings

$$\Delta_i^* = d_{i+1}^\alpha - d_i^\alpha \,, \quad i = 1, \ldots, M-1 \,, \tag{4}$$

between the delay α-quantiles of different traffic classes are therefore independent of α, as illustrated in Figure 7, and can be controlled by adjusting the numbers of reservations of the traffic classes i and $i+1$, as we will see further in equation (5).

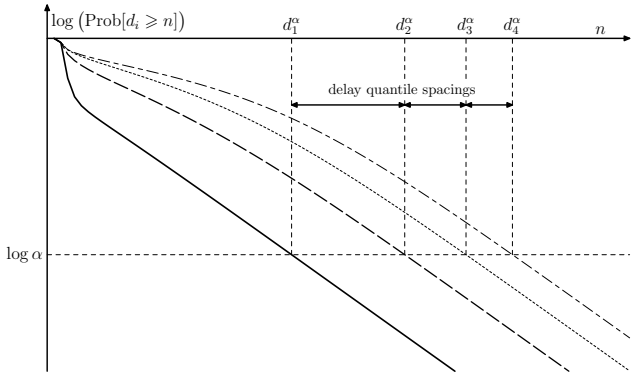

Fig. 7. For small enough α, the delay quantile spacings are independent of the value of α.

This property assures an 'asymptotic delay fairness' between the traffic classes that can be smoothly controlled. Additionally, the fact that all traffic classes have their delay distributions decaying at the same rate prohibits the situation where some unfortunate classes have extremely slow decaying delay tails, and therefore excessively high delay quantiles, manifested by the phenomenon of packet starvation for these traffic classes.

Based on our meta-model, we can obtain formulas for determining the working parameters (N_i, $i = 1, \ldots, M$) for R-scheduling if targeted delay quantile spacings between the traffic classes are given. In particular, the (constant) delay quantile spacing for small enough α follows from equation (1) as

$$\begin{aligned} \Delta_i^* = C_2 \, (N_i - N_{i+1}) + C_3(L, L_{i+1}) \, N_{i+1} \\ - C_3(L, L_i) \, N_i \,. \end{aligned} \tag{5}$$

Notice that the i-th delay quantile spacing Δ_i^* depends linearly on the numbers of reservations of the traffic classes i and $i+1$. Setting the working parameters in such a way as to obtain predetermined delay quantile spacings, comprises solving a (limited) set of linear equations. Such a set of linear equations has one degree of freedom since there are M unknowns (numbers of reservations) to be determined and there are $M-1$ equations (delay quantile spacings).

The solutions of such a set of linear equations can be easily obtained with standard mathematical techniques. Choosing the number of reservations N_M of traffic class M as a parameter, the numbers of reservations of the other classes are:

$$N_i = \frac{1}{r_i} \left[\sum_{j=i}^{M-1} \Delta_j^* + r_M \, N_M \right] \,, \quad i = 1, \ldots, M-1 \,, \tag{6}$$

where $r_i = C_2 - C_3(L, L_i)$.

The calculation of the numbers of reservations (N_i, $i = 1, \ldots, M-1$) only requires a few basic mathematical operations (addition and multiplication). As a consequence, the numbers of reservations can be acquired almost instantaneously and thus our technique has low complexity and is suitable for dynamic applications.

We anticipate that being able to guarantee delay quantile fairness in multi-class queueing systems may prove useful in many applications, not only in digital telecommunications but also for example in production scheduling or software engineering. Any application where multiple classes of items need to be scheduled for service by a single server and where the main concern is with the *excessive queueing delays* that may occur, could benefit by our proposed R-scheduling algorithm. Indeed, it is clear the delay quantile spacings in (5) are directly related to the proportionality between the delay tail probabilities of the traffic classes themselves. Suppose for example, in case of two traffic classes 1 and 2, the network administrator wants to target a fixed ratio between the probabilities that packets of either class reach a delay n or above. For high enough delay level n, this ratio follows from (3) as

$$\frac{\text{Prob}[d_2 \geq n]}{\text{Prob}[d_1 \geq n]} = (z_d)^{\Delta_1^*} \,,$$

and is therefore determined by our meta-model for the delay quantile spacings.

VI. Other advantages of R-scheduling

We have seen that the R-scheduling mechanism realises delay quantile fairness in the sense that the delay quantile spacings between the traffic classes are proportional. Moreover, it is straightforward to translate given administrator's QoS targets

into a set of working parameters for the mechanism. Other nice features of R-scheduling are the generality of the framework, the isolation of traffic classes through temporal priority and the possibility of calculating accurate delay bounds early. All of these are explained next.

A. Generality of the framework

The R-scheduling mechanism is very versatile in the number of configurations that can be chosen (number of classes M and the number of reservations for each traffic class N_i, $i = 1, \ldots, M$) and consequently, in the kinds of QoS differentiation that can be obtained. For instance, FIFO ($N_i = 0$) and absolute priority ($N_i = (M - i)K$ with large K) are limiting cases of R-scheduling. For the latter however, it must be kept in mind that the tails of the delay distributions will remain geometrical with FIFO decay, as is characteristic for R-scheduling. Nevertheless, as K increases, the importance of these tails decreases with it. That is, the probabilities α for which the model (1) is valid become arbitrarily low as $K \to \infty$, up to the point where they become an extremely rare event and likely no longer impact the performance of the system. In contrast however, choosing K large but not *too* large, results in delay distributions of which the body resembles that for strict priority scheduling but of which the FIFO tails mitigate packet starvation. This is an intended application of R-scheduling that does not necessarily rely on model (1).

Besides FIFO and absolute priority, R-scheduling also allows for many other useful QoS trade-offs in between. R-scheduling in particular enables a fine-tuned differentiation between several traffic classes of comparable importance (e.g. multimedia classes with different profiles or tariffs), with specific QoS spacing targets.

B. Isolation: temporal priority

R-scheduling can be used to effect a temporal priority for isolating traffic classes from one another. The idea of temporal priority is to be understood more or less as follows. Consider a traffic class i with N_i reservations at its disposal. If for some time, there is no incoming traffic of class i, these reservations will slowly migrate towards the head of queue and remain there until a new burst of class-i traffic comes in. The first N_i packets of this burst will seize the reservations in the front positions and therefore be scheduled prior to the other packets in the queue. However, for the remainder of the burst, the reservations will usually have less favourable positions. Hence, the priority is 'temporal' in the sense that only the first N_i packets of a burst receive quick scheduling. Additionally, the longer the absence of class-i traffic lasts, the further in the queue its reservations will be allowed to advance. The mechanism can thus be interpreted as granting a certain maximum 'priority credit' to each of the traffic classes. If the arrival rate is high, the credit is gradually used up and when depleted, no significant prioritization is granted to the packets. Absence of traffic however, gradually restores the priority credit up to its maximum. R-scheduling can therefore be used to grant a traffic class very fast access to the transmitter for limited bursts of packets. The longer a traffic class abstains from sending packets before the burst, the faster the burst will be processed. Clearly, the maximum priority credit of traffic class i is directly related to the number of reservations N_i that are assigned to it. Since each class has its own dedicated reservations, it is protected from packet flooding by other classes. The higher the number of reservations of a particular traffic class, the more severe its isolation.

C. Early delay bound calculation

With R-scheduling, as a consequence of storing all data packets and reservations in the same queue, we are able to estimate the total queueing delay of packets *before they are enqueued*. Having such an early estimate can be advantageous in several ways. It can be used to gather congestion information of the network that can be used locally or globally in the network. We may e.g. use it for packet discarding (similar to RED) by deciding not to enqueue an arriving packet with a large foreseen delay at all. The delay information could also be sent back to the source in order to warn for imminent upstream congestion (similar to TCP).

We preferably use a three-point estimation: a lower bound (best-case), upper bound (worst-case) and best estimate. A packet P that is enqueued sees a number of data packets and reservations in front of itself in the queue. The lower delay bound corresponds to the sum of the sizes of these packets and thus assumes the possibility that none of the reservations will be seized prior to the transmission of P. The upper delay bound on the contrary accounts for the worst-case where all of those reservations are seized by packets. So, in case the number of reservations ahead of arriving packet P is relatively small (usually so for packets of higher-priority traffic classes), both bounds may be very close together. Using a probabilistic algorithm we can calculate a best delay estimate for P by predicting the number of reservations ahead of P that will be seized. Optionally, per-class traffic measurements can be used to enhance this prediction.

VII. Conclusions

In this paper, we have presented the R-scheduling mechanism as a feasible and practical alternative to existing schedulers. We have shown that the R-scheduler aims at tailoring the per-class delay distributions in a specific way: these distributions have geometrically shaped tails, shifted relatively to each other by a specific amount. None of the existing class-based schedulers have this goal: the deadline-based schedulers are complex and aim at some hard upper bound for the delay that should never be crossed. Absolute priority targets the most extreme delay differentiation possible between the traffic classes and consequently is very rigid, cannot be controlled, and is vulnerable to starvation. The fair queueing schedulers aim for output rate proportionality instead of proportionality in the delay α-quantiles d_i^α, i.e., the delay levels such that packets of traffic class i have a delay of at least d_i^α with probability α.

978-0-9836283-9-2

The independence of the delay quantile spacing of α (for small enough α) has been verified for a wide parameter space (number of traffic classes, number of reservations for each traffic class, total system load, relative load for each traffic class). A precise and simple algorithm for translating QoS targets into system parameters is available.

The presented meta-model of the R-scheduling mechanism was constructed under the assumption of fixed-sized packets with constant transmission times of one slot. Future work will consider the extension of the meta-model to the case of variable-length transmission times.

ACKNOWLEDGEMENT

This research has been funded by the Interuniversity Attraction Poles Programme initiated by the Belgian Science Policy Office.

REFERENCES

[1] S. Blake et al., An architecture for Differentiated Services, Internet RFC 2475, December 1998.

[2] P. Gevros, J. Crowcroft, P. Kirstein, S. Bhatti, Congestion control mechanisms and the Best Effort service model, IEEE Network, May-June 2001, pp. 16-26.

[3] A. Wierman, Fairness and scheduling in single server queues, Surveys in Operations Research and Management Science, vol. 16, 2011, pp. 39-48.

[4] T. Bonald, L. Massoulié, A. Proutière, J. Virtamo, A queueing analysis of max-min fairness, proportional fairness and balanced fairness, Queueing Systems, vol. 53, 2006, pp. 65-84.

[5] T. Takine, B. Sengupta, T. Hasegawa, An analysis of a discrete-time queue for broadband ISDN with priorities among traffic classes, IEEE Transactions on Communications, vol. 42, 1994, no. 2-4, pp. 1837-1845.

[6] B.D. Choi, D.I. Choi, Y. Lee, D.K. Sung, Priority queueing system with fixed-length packet-train arrivals, IEE Proceedings-Communications, vol. 145, 1998, no. 5, pp. 331-336.

[7] J. Walraevens, B. Steyaert, M. Moeneclaey and H. Bruneel, Delay analysis of a HOL priority queue, Telecommunication Systems, vol. 30, 2005, no. 1-3, pp. 81-98.

[8] C.-K. Tham, Q. Yao and Y. Jiang, A multi-class probabilistic priority scheduling discipline for differentiated services networks, Computer Communications, vol. 25, 2002, pp. 1487-1496.

[9] Y. Lim and J.E. Kobza, Analysis of a delay-dependent priority discipline in an integrated multiclass traffc fast packet switch, IEEE Transactions on Communications, vol. 38, no. 5, 1990, pp. 659-685.

[10] T. Maertens, J. Walraevens, M. Moeneclaey and H. Bruneel, Performance analysis of a discrete-time queueing system with priority jumps, International Journal of Electronics and Communications (AEU), vol. 63, no. 10, 2009, pp. 853-858.

[11] A.Z. Melikov, L.A. Ponomarenko and C.S. Kim, Approximate method for analysis of queuing models with jump priorities, Automation and Remote Control, vol. 74, no. 1, 2013, pp. 62-75.

[12] C. Dovrolis and P. Ramanathan, A case for relative differentiated services and the proportional differentiation model, IEEE Network, Sept./Oct., 1999, pp. 2-10.

[13] A. Demers, S. Keshav, S. Shenker, Analysis and simulation of a fair queueing algorithm, Proceedings of the ACM Symposium on Communications Architectures & Protocols, SIGCOMM '89 (19-22 September 1989, Austin, TX, USA), pp. 1-12.

[14] H. Alnuweiri, H. Tayyar, Analysis of virtual-time complexity in weighted fair queueing, Computer Communications, vol. 28, 2005, pp. 802-810.

[15] P. Goyal, H.M. Vin, H. Cheng, Start-time fair queueing: A scheduling algorithm for integrated services packet switching networks, IEEE-ACM Transactions on Networking, vol. 5, 1997, no. 5, pp. 690-704.

[16] S.J. Golestani, Network delay analysis of a class of fair queuing algorithms, IEEE Journal on Selected Areas in Communications, vol. 13, 1995, no. 6, pp. 1057-1070.

[17] J.C.R. Bennett, H. Zhang, WF^{2}Q: Worst-case fair weighted fair queueing, Proceedings of the 15th Annual Joint Conference of the IEEE Computer and Communications Societies: Networking the Next Generation (24-28 March 1996, San Fancisco, CA, USA), pp. 120-128.

[18] M. Shreedar, G. Varghese, Efficient fair queueing using deficit round-robin, IEEE/ACM Transactions on Networking, vol. 4, 1996, no. 3, pp. 375-385.

[19] C. Semeria, Supporting differentiated service classes: queue scheduling discipline, Juniper Networks white paper, 2001.

[20] R. Guérin and V. Peris, Quality-of-Service in packet networks: basic mechanisms and directions, Computer Networks, vol. 31, 1999, pp. 169-189.

[21] B. Doytchinov, J. Lehoczky and S. Shreve, Real-time queues in heavy traffic with earliest-deadline-first queue discipline, Annals of Applied Probability, vol. 11, no. 2, 2001, pp. 332-378.

[22] P. Moyal, On queues with impatience: stability, and the optimality of Earliest Deadline First, Queueing Systems, vol. 75, no. 2-4, 2013, pp. 211-242.

[23] W. Burakowski and M. Fudala, Priority Forcing Scheme: a new strategy for getting better than best effort service in IP-based network, Proceedings of the IFIP Workshop on Internet Technologies, WITASI 2002 (10-11 October 2002, Wroclaw, Poland), pp. 135-150.

[24] W. Burakowski and H. Tarasiuk, On new strategy for prioritising the selected flow in queuing system, Proceedings of the COST 257 11th Management Committee Meeting (20-21 January 2000, Barcelona, Spain), COST-257 TD(00)03.

[25] S. De Vuyst, S. Wittevrongel and H. Bruneel, Delay differentiation by reserving space in queue, Electronics Letters, vol. 41, no. 9, 2005, pp. 564-565.

[26] S. De Vuyst, S. Wittevrongel and H. Bruneel, Place reservation: delay analysis of a novel scheduling mechanism, Computers and Operations Research, Special Issue on 'Queues in Practice', vol. 35, no. 8, 2008, pp. 2447-2462.

[27] B. Feyaerts, S. De Vuyst, S. Wittevrongel and H. Bruneel, Analysis of a discrete-time priority queue with place reservations and geometric service times, Proceedings of the 6th Symposium on Design, Analysis and Simulation of Distributed Systems, DASD 2008 (16-19 June 2008, Edinburgh, UK), pp. 140-147.

[28] B. Feyaerts and S. Wittevrongel, Performance analysis of a priority queue with place reservation and general transmission times, Proceedings of the Fifth European Performance Engineering Workshop, EPEW 2008 (24–25 September 2008, Palma de Mallorca, Spain), Lecture Notes in Computer Science, vol. 5261, 2008, pp. 197–211.

[29] S. De Vuyst, S. Wittevrongel, D. Fiems and H. Bruneel, Controlling the delay trade-off between packet flows using multiple reserved places, Performance Evaluation, vol. 65, no. 6-7, 2008, pp. 484-511.

[30] S. De Vuyst, S. Wittevrongel, H. Bruneel, Transform-domain analysis of packet delay in network nodes with QoS-aware scheduling, Lecture Notes in Computer Science, vol. 5233, 2011, pp. 921-950.