Resource Division Policies for EDF Scheduling in ATM Networks

Amr S. Ayad asaad@ie-eg.com
Mahmoud T. El-Hadidi hadidi@frcu.eun.eg
Khaled M. Fouad Elsayed khaled@ieee.org
Department of Electronics and Communications Engineering
Faculty of Engineering, Cairo University
Giza, Egypt 12613

Abstract

The paper addresses the issue of reserving resources at ATM switches along the path of calls requiring a deterministic bound on end-to-end delay. The switches are assumed to schedule outgoing cells using the Earliest- Deadline -First (EDF) scheduling discipline. EDF is known to be an optimal scheduling discipline when providing delay bounds to a number of calls being served by a single scheduler. We present the algorithm for call admission control (CAC), and propose a number of resource division policies used for mapping the end-to-end delay requirement of a call into local delay deadlines to be reserved at each scheduler.

1. Introduction

One of the main promises of ATM networks is to provide users with Quality-of-Service (QoS) guarantees, such as Cell Transfer Delay (CTD) and Cell Loss Ratio (CLR). Handling the variety in QoS requirements of different applications requires the network to use a mechanism for serving cells from different applications according to their granted QoS level. Many scheduling disciplines have been proposed in the literature to implement such mechanism (see [1], [2], and [3]). Each scheduling discipline requires algorithms for performing Call Admission Control (CAC) and resource reservation. The paper proposes such algorithms for the case of EDF service discipline and for calls requiring a hard (deterministic) bound on the end-to-end delay experienced by their cells. The paper addresses the problem of how to map the end-to-end delay requirement of a call into a local resource requirement to be reserved at each scheduler along the call's path.

The paper is based on the use of the general EDF schedulability condition given in [4], and the EDF schedulability condition for token-bucket-shaped traffic given in [5]. The reader is encouraged to review those references (all available for download from the Internet) in order to get in the context of the paper.

The rest of this paper is organized as follows: Section 2 presents a brief discussion on the EDF scheduling discipline. Section 3 presents the CAC algorithm for bounded end-to-end delay services. Section 4 proposes a number of resource division policies that are applicable when using EDF schedulers. Section 5 concludes the paper.

2. Defining EDF operation

The operation of an EDF scheduler is described as follows: a deadline is assigned to each newly arriving cell from call (f). The deadline is computed as the sum of the arrival time of the cell and the local delay bound reserved for call (f) at this scheduler. The scheduler serves cells in the ascending order of their deadlines.

Maintaining the delay guarantee made to call (f) is equivalent to having all the cells belonging to it transmitted completely before their assigned deadlines. Consequently, all cells from call (f) do not get delayed beyond the delay bound reserved for call (f) at this scheduler. We denote the case in which a cell misses its deadline, i.e. not transmitted completely before its deadline, as a case of *violation*.

The conditions under which a single EDF scheduler operates without violations are:

1- The stability condition:

This is a condition that must hold true for all work-conserving disciplines in general. If the condition does not hold true, a scheduler with finite buffers will always overflow and drop cells. For a scheduler (k) with capacity C^k (which is the data rate of the link following the scheduler), the stability condition states:

$$\sum_{j=1}^{N} \rho_j \le C^k \qquad (1)$$

where ρ_j is the average rate (in bits per second) of the traffic source of call (*j*) and *N* is the number of calls that are being served by scheduler (*k*).

2- The schedulability condition:

The schedulability condition guarantees that a scheduler will not make violations and will, therefore, honor the QoS commitment made during the call set-up phase. The form of the schedulability condition differs, in general, for each scheduling discipline. If the schedulability condition is sufficient but not necessary, then there is a possibility of under-utilizing the scheduler since a violation of the sufficient conditions does not necessarily mean that a given set of calls is not schedulable. On the other hand, a schedulability condition that is both necessary and sufficient guarantees that there is no under-utilization of the scheduler's resources.

2.1 Schedulability conditions for token-bucket traffic

Here, we use the results presented in [5] which applies Theorem 1 in [4] to deduce the schedulability conditions for token-bucket traffic models. The analysis in [5] assumed the use of a preemptive EDF scheduler which is equivalent to the use of negligible cell transmission time which is typical in the case of ATM networks with small cell size and high speed links. For negligible cell transmission time, [5] defines the function $F(t)$ as:

$$F(t) = C\,t - \sum_{j \in N} A_j^*(t - d_j) \qquad (2)$$

Where *C* is the data rate of the link serving the EDF scheduler in bits per second, d_j is the delay bound reserved for call (*j*) in seconds, and $A_j^*(t)$, in bits, is the traffic-constraint function on the traffic arrivals from call (*j*) up to time (*t*). In [5], it is shown that the schedulability condition of an EDF scheduler is equivalent to verifying that:

$$F(t) \geq 0 \qquad \forall\, t \geq 0 \qquad (3)$$

As in [5], we will be dealing with the case in which $A_j^*(t)$ represents token-bucket traffic models (σ,ρ,c), and (σ,ρ).

3. CAC for bounded delay service in ATM networks.

For bounded delay service, the application of the CAC algorithm on a call path, which typically consists of more than one scheduler, requires the computation of the minimum delay that each scheduler along the path of the call can guarantee to this new call. This allows the CAC algorithm to determine the minimum achievable end-to-end delay bound for this call and, thus, to determine if the network can guarantee the requested delay bound and to, consequently, decide if the call is accepted or not.

This section starts by describing the assumed CAC operation when accepting a new call. We then discuss the computation of the minimum delay bound that an EDF scheduler can guarantee to calls with token-bucket traffic models.

3.1 CAC operation

The CAC operation proposed in [5] assumes that the call is established using a setup protocol such as ATM signaling. The operation of this protocol proceeds as follows: the calling party wishing to establish a call (*f*), sends a SETUP message to the called party, including information such as the flow's traffic characteristics (Peak Cell Rate (PCR), Sustained (average) Cell Rate (SCR), Maximum Burst Size (MBS)), and the required end-to-end delay bound (D_f). This message travels over *K* schedulers belonging to the call path *P* selected by the routing algorithm in use. At each scheduler *i* on *P*, the minimum delay that a scheduler *i* can guarantee to call *f*, d_f^{i*}, is computed and added to d_f^*, which is the cumulative delay sum included in the setup message. If at some scheduler, the cumulative delay exceeds the required delay bound, then the call cannot be accepted and a RELEASE message is returned to the calling party. Otherwise, the setup message reaches the last scheduler which checks if $D_f \geq D_f^*$, where $D_f^* = \sum_{i=1}^{K} d_f^{i*}$ is the minimum achievable end-to-end delay for call *f*. If the condition is true, the call is accepted, and a CONNECT message is then returned on the same path to the calling party, reserving a delay bound $d_f^k \geq d_f^{k*}$ to call *f* at each scheduler (*k*) such that $\sum_{k \in P} d_f^k \leq D_f$. The values of d_f^k are chosen according to some delay division policy. The type of the division policy in use is the main subject of this paper.

We assume that only the final scheduler checks the validity of condition ($D_f \geq D_f^*$) and makes the irreversible decision of accepting or rejecting the call. However, some resource division policies exhibit the problem of having one or more resource-limited schedulers. We define a scheduler to be resource-limited for a call (*f*) if the resource division policy in use results in $d_f^i < d_f^{i*}$. We present algorithms for handling the case of having resource-limited schedulers in Section 4.

3.2 Minimum delay bound of a new call

The CAC algorithm depends on computing the minimum delay bound that can be guaranteed by an EDF scheduler to a new call given its traffic characteristics, delay requirement and the current state of the scheduler. The computation of this value allows a scheduler to determine if it has enough capacity to serve the incoming call. As described in Section 3-1, this value is essential to the CAC operation for a call spanning multiple schedulers along its path. This value will also be used when applying the proposed resource division policies as an indicator of the loading state of a scheduler. The solution of this problem for the (σ,ρ,c), and (σ,ρ) traffic models is given in [5].

4. Resource division policies

In this section, we propose policies for dividing the end-to-end delay requirement among the schedulers along the call path. It is impossible to devise a single division policy that maximizes the acceptable number of calls for the general case in which a network consists of multiple paths and with calls having different traffic characteristics and delay requirements. Therefore, we will derive resource division policies for a single path, initially unloaded network with all calls having the same traffic characteristics and delay requirement. The derived policies can then be applied to more general scenarios and their relative performance can be evaluated using simulation.

We start by broadly categorizing resource division policies into the following categories:

i- *Static policies:* These are policies in which the delay requirement assigned to a scheduler is independent of the loading state of this scheduler or other schedulers on the call path.

ii- *Dynamic policies:* These are state-dependent policies in which the delay requirement assigned to a scheduler depends, in general, on its state and the state of other schedulers on the call path.

We will study each category in a separate section and present the results for both (σ,ρ,c) and (σ,ρ) models. We will denote the number of schedulers along the single path network by K and we assume that all calls have an end-to-end delay requirement of D_f. The formulas used for implementing the following policies depend on the assumption of having a rate-controller before each scheduler. The delay contribution of a scheduler to the end-to-end delay is, therefore, independent of other schedulers' contribution, and we have:

$$D_f = \sum_{i=1}^{K} d_f^i \qquad (4)$$

4.1 Static division policies

In static division policies, the delay bound assigned to a scheduler is independent of its loading state. Therefore, the same delay will be assigned to a given scheduler for all calls of the same traffic characteristics and delay requirements offered to a single path network. Let us denote the maximum number of acceptable calls by a scheduler (i) as N^i. Hence the maximum number of acceptable calls (N_{max}) on the path is given by:

$$N_{\max} = \min_{1 \le i \le K} N^i \qquad (5)$$

4.1.1 Optimal static policy. An optimal policy is one that maximizes the number of calls acceptable on the call path.

Lemma: The optimal policy is the one which satisfies the condition that a scheduler (i) assigns a delay d_f^i to call (f) such that:

$$N^i = \text{const} = N_{\max} \qquad \forall\ 1 \le i \le K \qquad (6)$$

The proof is of the above lemma is as follows: assume that another policy can accept a larger number of calls than $N_{\max}$. This means that some scheduler (k) can accept a number of calls $N_k \ge N_{\max}$ and has a delay $d_f^k > d_f^k(N_{\max})$. Since the end-to-end delay is fixed, another scheduler (l) must have delay $d_f^l < d_f^l(N_{\max})$. Consequently, $N_l \le N_{\max}$. From (5), the number of acceptable calls would then be $N \le min\ (N_k, N_l) \le N_{\max}$ Therefore, no policy can accept more calls than $N_{\max}$ which is given by the policy satisfying (6).

In the following, we derive the formulas for the optimal static policy satisfying (6):

i- (σ,ρ, c) *model:* We start by deriving the formula for N^i. Since all calls will be assigned the same delay bound at a given scheduler, applying the schedulability condition in (3), we have:

$$C^i t \ge n^i A_f^*(t - d_f^i) \qquad \forall\ t \ge 0 \qquad (7)$$

We solve (6) for $N^i = \max(n^i)$ by considering Figure 1, which shows the two sides of (7) for (σ,ρ, c) traffic model.

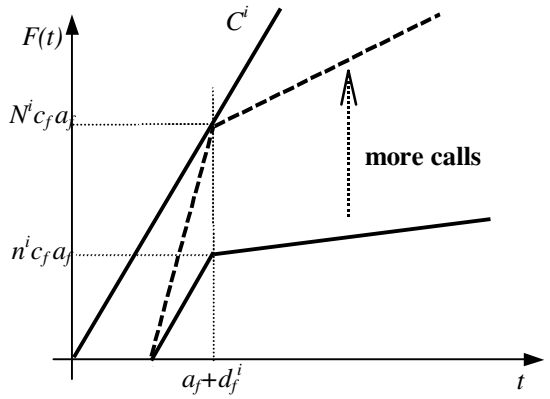

Figure 1: Deriving the value of N^i for a scheduler *i* along the path from source to destination

From graph, we find that:

$$N^i = \frac{C^i(a_f + d^i_f)}{c_f a_f} \tag{8}$$

To satisfy (6), we must have:

$$C^i(a_f + d^i_f) = C^j(a_f + d^j_f) \quad \forall\, 1 \le i,j \le K \tag{9}$$

The solution of (9) under the condition in (4) is as follows: from (9), we have:

$$d^i_f = \frac{C^j(a_f + d^j_f)}{C^i} - a_f \tag{10}$$

$$\Rightarrow \sum_{\substack{i=1 \\ i\neq j}}^{K} d^i_f = \sum_{\substack{i=1 \\ i\neq j}}^{K} \left(\frac{C^j(a_f + d^j_f)}{C^i} - a_f\right) \tag{11}$$

$$\Rightarrow D_f - d^j_f = C^j d^j_f \sum_{\substack{i=1 \\ i\neq j}}^{K} \frac{1}{C^i} + a_f C^j \sum_{\substack{i=1 \\ i\neq j}}^{K} \frac{1}{C^i} - a_f(K-1) \tag{12}$$

$$\Rightarrow d^j_f = \frac{1}{\sum_{i=1}^{K} \frac{1}{C^i}} \left[\frac{D_f}{C^j} + a_f\left(\frac{K}{C^j} - \sum_{i=1}^{K} \frac{1}{C^i}\right)\right] \quad \forall\, 1 \le j \le K \tag{13}$$

Substituting in (8), we get

$$N_{\max} = \frac{D_f + K a_f}{c_f a_f \sum_{i=1}^{K} \frac{1}{C^i}} \tag{14}$$

ii-(σ,ρ) *model:* We note that the (σ,ρ) model is a (σ,ρ,c) model with $c \to \infty$. Using $c_f \to \infty$ in (13), and (14) (note that as $c \to \infty$, a should tend to 0 so that $ac \to$ finite value = σ), we get:

$$d^j_f = \frac{\frac{1}{C^j}}{\sum_{i=1}^{K} \frac{1}{C^i}} D_f \qquad \forall\, 1 \le j \le K \tag{15}$$

$$N_{\max} = \frac{D_f}{\sigma_f \sum_{i=1}^{K} \frac{1}{C^i}} \tag{16}$$

Note that (15) depicts an inverse capacity proportional policy, as the delay bound assigned to a scheduler is inversely proportional with its capacity, and thus a scheduler with smaller capacity is required to reserve larger delay bound (i.e. less resource load).

4.1.2 ***Handling resource-limited schedulers.*** There is a possibility of having resource-limited schedulers when applying static policies. A resource limited scheduler (*i*) is one for which the assigned delay bound value d^i_f as computed from (13) (or (15)) is less than the minimum delay bound d^{i*}_f achievable by the scheduler given its loading state.

There are two approaches for handling this case: one approach is to reject the call; the other approach is to accept it and redistribute the remaining of the local delay bound on other schedulers. Note that assigning the minimum delay bound d^{i*}_f to some call (*f*) does not necessarily imply that scheduler (*i*) cannot accept any more calls as it can still accept other calls with less intense traffic characteristics or more relaxed delay requirement. In the following, we give the algorithms for re-distributing the delay among other schedulers when there exists one or more resource-limited schedulers.

i. (σ,ρ,c) *model:* Rewriting equation (13), we have:

$$C^j(d^j_f + a_f) = \frac{D_f + a_f K}{\sum_{i=1}^{K} \frac{1}{C^i}} \qquad \forall 1 \le j \le K \tag{17}$$

We use this equation to reduce the execution time of the algorithm by keeping a sorted list of the quantity $C^j(d^{j*}_f + a_f)$ and comparing it with the quantity $\frac{D_f + a_f K}{\sum_{i=1}^{K} \frac{1}{C^i}}$ to identify resource-limited schedulers. The algorithm is given in the following steps.

- Initialize $S=\{i: 1\le i \le K\}$
- Sort S in ascending order of $C^j(d_f^{j*}+a_f)$
- Set $B=\phi$
- Set $d_f^i=\frac{1}{\sum_{j=1}^{K}\frac{1}{C^j}}\left[\frac{D_f}{C^i}+a_f\left(\frac{K}{C^i}-\sum_{j=1}^{K}\frac{1}{C^j}\right)\right]$ *//Initial assignment of local delay bounds*

$\forall i\in S$

- Find the least index (l) for which $C^j(d_l^{j*}+a_l)<\frac{D_l+a_lK}{\sum_{i=1}^{K}\frac{1}{C^i}}$ *// Identify resource-limited schedulers*
- $B=B\cup\{i: i\in S, i\ge 1\}$ *// Add resource-limited schedulers to B*
- If $B=\phi$ then terminate *// No resource-limited schedulers⇒ terminate*
- Set $d_f^i=d_f^{i*}\quad \forall i\in B$ *// Reserve the remaining delay at resource-limited schedulers*
- $S=S-B$ *// Update S to remove resource-limited schedulers*
- $D_f=D_f-\sum_{i\in B}d_f^{i*}$ *// Update the delay requirement*
- Return to 3 *// Loop until no resource-limited schedulers*

ii. (σ,ρ) model: Rewriting equation (15), we have:

$$C^j d_f^j=\frac{D_f}{\sum_{i=1}^{K}1/C^i} \tag{18}$$

We use this equation to reduce the execution time of the algorithm by keeping a sorted list of the quantity $C^j d_f^{j*}$ and comparing it with the quantity $\frac{D_f}{\sum_{i=1}^{K}1/C^i}$ to identify resource-limited schedulers. The algorithm is shown in the following steps.

- Initialize $S=\{i: 1\le i\le K\}$
- Sort S in ascending order of $C^j d_f^{j*}$
- Set $B=\phi$
- Set $d_f^i=\frac{\frac{1}{C^i}}{\sum_{i=1}^{K}\frac{1}{C^j}}D_f\ \forall i\in S$ *// Initial assignment of local delay bounds*
- Find the least index (l) for which $C^l d_f^{l*}>\frac{D_f}{\sum_{i=1}^{K}\frac{1}{C^i}}$ *// Identify resource-limited schedulers*
- $B=B\cup\{i: i\in S, i\ge 1\}$ *// Add resource-limited schedulers to B*
- If $B=\phi$ then terminate *// No resource-limited schedulers⇒ terminate*
- Set $d_f^i=d_f^{i*}\quad \forall i\in B$ *// Reserve the remaining delay at resource-limited schedulers*
- $S=S-B$ *// Update S to remove resource-limited schedulers*
- $D_f=D_f-\sum_{i\in B}d_f^{i*}$ *// Update the delay requirement*
- Return to 3 *// Loop until no resource-limited schedulers*

4.1.3 Even policy (EVEN). We use the even policy as a reference policy against which other policies may be compared. In EVEN policy, all schedulers are required to reserve the same amount of delay, hence:

$$d_f^i=\frac{D_f}{K} \tag{19}$$

From (8), we have for (σ,ρ,c) model:

$$N^i=\frac{C^i(a_f+(D_f/K))}{c_f a_f} \tag{20}$$

and using (5),

$$N_{EVEN}=\frac{C^{\min}(a_f+(D_f/K))}{c_f a_f} \tag{21}$$

Comparing with the expression of $N_{\max}$ in (14), we get the relative gain value of the optimal static policy with respect to even policy:

$$\frac{N_{\max}}{N_{EVEN}}=\frac{K}{\sum_{i=1}^{K}\frac{C^{\min}}{C^i}} \tag{22}$$

From (8), by taking $c\to\infty$, we have for (σ,ρ) model:

$$N^i=\frac{C^i D_f}{K\sigma_f} \tag{23}$$

and using (5), (23)

$$N_{EVEN}=\frac{C^{\min}D_f}{K\sigma_f} \tag{24}$$

As in (22), from (16), the gain is:

$$\frac{N_{\max}}{N_{EVEN}} = \frac{K}{\sum_{i=1}^{K} \frac{C^{\min}}{C^i}} \quad (25)$$

Handling resource-limited schedulers, if the admission policy allows their acceptance, can be done using similar algorithms to those used for the case of the optimal static policy.

4.2 Dynamic division policies

In dynamic division policies, the delay bound assigned to a scheduler depends, in general, on its loading state as well as the loading states of other schedulers on the call path. This complicates the analysis of such policies because even when all calls have the same traffic characteristics and delay requirement, the assigned delay bound at a scheduler for a certain call is not, in general, equal to the one for previous or subsequent calls. Therefore, one cannot compute values such as $N_{\max}$ as for static policies and, hence, it's not possible to devise an optimum dynamic division policy. Instead, we propose three adhoc policies. The proposed policies are extensions of a policy suggested in [1] for dividing the end-to-end delay bound. This policy is based on the assumption that each scheduler initially reserves the tightest possible delay value for the incoming call. It then suggests subsequent relaxation of this reservation by equally redistributing the excess end-to-end delay on the schedulers. The excess end-to-end delay is defined as:

$$\overline{D}_f = D_f - D_f^* \quad (26)$$

4.2.1 Even distribution of excess delay. This policy is the one suggested in [1]. The delay bound formula is given by:

$$d_f^j = d_f^{j*} + \frac{\overline{D}_f}{K} \qquad \forall\ 1 \le j \le K \quad (27)$$

4.2.2 **Capacity proportional distribution of excess delay.** In this policy, the excess delay is distributed in inverse proportion to the scheduler capacity. The delay bound formula is given by:

$$d_f^j = d_f^{j*} + \frac{\frac{\overline{D}_f}{C^j}}{\sum_{i=1}^{K} 1/C^i} \qquad \forall\ 1 \le j \le K \quad (28)$$

4.2.3 Remaining-delay proportional distribution of excess delay. In this policy, the excess delay is proportional to the minimum delay bound that the scheduler can guarantee to the incoming call. The delay bound formula is given by:

$$d_f^j = d_f^{j*} + \frac{\overline{D}_f}{D_f^*} d_f^{j*} \qquad \forall\ 1 \le j \le K \quad (29)$$

$$\Rightarrow \qquad d_f^j = d_f^{j*} \frac{D_f}{D_f^*} \qquad \forall\ 1 \le j \le K \quad (30)$$

5. Conclusion

This paper discussed the use of non-even resource division policies when performing resource reservation in order to provide bounded delay service in an ATM WAN. We have derived the required local delay allocations to obtain an optimal static policy for a single path network.

The use of non-even resource division policies when performing resource reservation has the potential of obtaining more efficient utilization of network resources. We have derived an expression for the gain in the number of accepted calls along a single path of schedulers, due to the use of non-even static resource division policies. This gain value increases with the number of schedulers on the call path and with the imbalance in their link capacities.

The use of dynamic policies can provide even higher gain in situations where the call path is initially loaded, which is common in a network with many intersecting call paths.

Current research activity is focusing on evaluating the performance of the proposed policies for networks with general topologies, using simulation. Initial results have shown considerable gain improvement using such policies.

6. References

[1] D. Ferrari and D. C. Verma, "A Scheme for Real-Time Channel Establishment in Wide-Area Networks", IEEE Journal on Selected Areas in Communications, Vol. 8, No. 3, pp. 368-379, April 1990.

[2] A. K. Parekh, "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks", Ph.D. dissertation, Dep. Elec. Eng. Comput. Sci., M.I.T, Feb 1992.

[3] S. J. Golestani, A Framing Strategy for Congestion Management. IEEE JSAC, vol. 9, no. 7, September (1991)

[4] J. Liebeherr, D. E. Werge, and D. Ferrari, "Exact Admission Control for Networks with a Bounded Delay Service", IEEE/ACM Transactions on Networking, Vol. 4 No. 6, December 1996. Downloadable from "ftp://ftp.cs.virginia.edu/pub/jorg/papers/ton-95-1328.pdf"

[5] V. Firoiu, J. Kurose, and D. Towley, "Efficient Admission Control for EDF Schedulers", Dept. of Com. Sci., University of Massachusetts, downloadable from "ftp://ftp.cs.umass.edu/pub/techrept/techreport/1996/UM-CS-1996-046.ps"