Performance Evaluation of Resource Division Policies for PGPS Scheduling in ATM Networks

Amr S. Ayad, Khaled M. F. Elsayed, and Mahmoud T. El-Hadidi

Department of Electronics and Communications Engineering
Faculty of Engineering, Cairo University, Giza, Egypt 12613
asaad@ie-eg.com, khaled@ieee.org, hadidi@frcu.eun.eg

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 packets using the Packet-by-Packet Generalized Processor Sharing (PGPS) scheduling discipline. We propose an algorithm for call admission control (CAC), and present the simulation results used to evaluate the performance of the resource division policies used for mapping the end-to-end delay requirement into service rates to be reserved at each switch. The simulation results also show the performance gain when a simple resource-based routing algorithm is used.

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 packets from different applications according to their contracted QoS level. Many packet-scheduling disciplines have been proposed in the literature to implement such mechanisms (see [4], [8], [9]). Each scheduling discipline requires algorithms for performing call admission control (CAC) and resource reservation. In this paper, we propose such algorithms for the case of PGPS service discipline and calls requiring a hard (deterministic) bound on end-to-end delay. The paper addresses the following problems:

1. How to map the end-to-end delay requirement of a call into a local resource requirement to be reserved at each switch along the call's path?
2. How to divide the resource requirement among the schedulers on the call's path? That is, a simple even division policy would be to reserve the same amount of resources at all schedulers. However, it may be more efficient to use a policy that takes schedulers capacities and/or loading into account.
3. How much gain (if any) would be obtained from applying non-even resource division policies? What are the factors controlling the gain value?

The following terms will be used throughout the paper:

1. *Resource load on a switch*: It represents the amount of reserved resources at a switch to satisfy the guaranteed QoS of accepted calls. This value depends on the calls' traffic characteristics and the QoS level requested by each call. For PGPS, this is expressed in bit rate units (e.g. bps)
2. *Call load*: It represents the arrival rate and the holding time of incoming calls without regard to their resource load. The call load is measured in Erlangs.

The rest of this paper is organized as follows: section 2 gives an overview of the delay formulas associated with PGPS scheduling. Section 3 presents the proposed CAC algorithm, and the associated non-even resource division policies. Section 4 presents the simulation results of applying the proposed algorithms to several network models. Section 5 presents the simulation results showing the performance enhancement resulting from the use of resource-based routing. Section 6 concludes the paper.

2 PGPS Scheduling Discipline

2.1 Delay Formulas

Generalized Processor Sharing (GPS) (see [6]) is an ideal and non-realizable scheduling discipline that serves packets as if they are in separate logical queues, visiting each nonempty queue in turn and serving an *infinitesimally* small amount of data from each queue. Connections can be associated with service weights, and they receive service in proportion to this weight whenever they have data in the queue. PGPS scheduling (see [8]) closely approximates GPS by serving packets in ascending order of the service tags assigned to them. The service tags are computed as the finish times of those packets had a GPS scheduler having the same capacity and the same input traffic served them.

This paper builds on the work done in [1], [5], and [8] where it was shown that if a call (*f*) traverses a path of K_f PGPS schedulers and has traffic characteristics conforming to a leaky bucket with a maximum burst size of (σ_f) bits and a long term average rate of (ρ_f) bps, then an upper bound on the end-to-end delay is guaranteed for each ATM cell from call (*f*) by reserving a certain service rate at each switch along the call's path. The upper bound (D_f) is given by:

$$D_f \le \frac{\sigma_f - L}{g_f} + \left(\sum_{j=1}^{j=K_f} \alpha^j \right) + \left(\sum_{j=1}^{j=K_f} \frac{L}{g_f^j} \right). \qquad (1)$$

where:

g_f^j = The service rate reserved for call (*f*) at switch (*j*), $g_f = \min_{j \in [1, K_f]} g_f^j$,

$\alpha^j = \beta^j + \tau^{j,j+1}$, $\beta^j = L / C^j$, L= Length of the ATM cell (424 bits).

$\tau^{j,j+1}$ = The propagation delay on the link from switch (*j*) to switch (*j*+1)

C^j = The data rate of the link following switch (*j*) in bps. We will denote C^j as the switch capacity.

The above inequality is only valid when the following conditions are met at each switch (j), $j\in[1, K_f]$.

1-*The scheduler stability condition*, which requires that:

$$\sum_{k=1}^{N}\rho_k \le C^j . \tag{2}$$

where N is the number of accepted calls at switch (j).

The stability condition is necessary for all scheduling disciplines and is not specific to PGPS scheduling.

2-*The schedulability condition* for PGPS schedulers, which requires that:

$$\sum_{k=1}^{N} g_k^j \le C^j . \tag{3}$$

where N is the number of accepted calls at switch (j).

2.2 The Condition of Local Stability

A call (f) is said to be locally stable at a switch (j) if

$$g_f^j \ge \rho_f . \tag{4}$$

The local stability condition is not required for each call passing by a given switch if all the calls passing by this switch have a leaky-bucket constrained traffic. However, if some call is not, then (4) must hold true for all accepted calls. Therefore, the local stability condition is not necessary if the network operator uses leaky bucket traffic shapers for all the network's ingress traffic.

The implementation of the proposed resource division algorithms depends on whether local stability is imposed or not. For shortness, we will only consider the case in which all traffic is leaky-bucket shaped and, thus, the local stability condition need not be imposed when reserving rates for new calls.

The other case where this condition must be applied to all calls requires a modification of the algorithms presented in this paper and has been addressed in [2].

3 CAC Algorithm and Resource Division Policies

3.1 CAC Algorithm

The proposed CAC algorithm uses equations (1)-(3) to determine whether to accept or reject a new call. It operates as follows:

The first test compares the value of the end-to-end delay (D_f) requested by the incoming call with the value of the total transmission and propagation delay along the call's path. If the value of the required end-to-end delay is smaller, the call is rejected. The next test is to verify that the value of the required end-to-end delay of the incoming call is not less than the minimum end-to-end delay bound that the network can guarantee to the incoming call. This minimum value (D_f^*) is obtained from (1) with each switch along the call's path reserving a service rate equal to its remaining capacity. We define the remaining capacity of a PGPS scheduler (j) prior to the acceptance of call (f) as:

$$R_f^j = C^j - \sum_{k=1}^{N_f} g_k^j . \tag{5}$$

where N_f is the number of accepted calls prior to the acceptance of call (f). We denote the minimum remaining capacity along the path of call (f) prior to accepting it by R_f.

On passing the previous tests successfully, a division policy is used to map the required end-to-end delay into a local resource requirement $\{g_f^j\}$ to be reserved at each switch. Different division policies are discussed in the next section.

Finally a test is made to verify that the conditions in (2) and (3) hold true for all schedulers on the call path. If local stability is imposed, then the local stability condition in (4) must also hold true. If all conditions are met, the call is accepted

3.2 Resource Division Policies

Even policy (EVEN). The reserved rates are the same at all schedulers, i.e.

$$g_f^i = g_f \,\forall i\in[1,K_f] . \tag{6}$$

Substituting in (1), after converting it to an equality to reserve the least amount of resources required for meeting the delay bound of call (f), we get:

$$g_f^i = g_f = \frac{\sigma_f + (K_f - 1)L}{D_f - \sum_{j=1}^{K_f}\alpha^j}\,\forall i\in[1,K_f] . \tag{7}$$

Capacity proportional policy (CP). The reserved rate at a certain switch is proportional to the switch capacity, i.e.

$$g_f^i = \eta_f C^i \,\forall i\in[1,K_f] . \tag{8}$$

where η_f = Constant for the path and call (f) parameters.

Substituting in (1), after converting it to an equality to reserve the least amount of resources required to meet the delay bound and solving for η_f, we get:

$$g_f^i = \frac{\dfrac{\sigma_f - L}{C} + \sum_{j=1}^{K_f}(L/C^j)}{D_f - \sum_{j=1}^{K_f}\alpha^j} C^i \,\forall i\in[1,K_f] . \tag{9}$$

where $C = \min_j C^j$.

Remaining capacity proportional policy (RCP). The reserved rate at a certain switch is proportional to the remaining capacity of the switch i.e.

$$g_f^i = \eta_f R_f^i \,\forall i\in[1,K_f] . \tag{10}$$

where η_f = Constant for the path and call (f) parameters as long as no other calls are accepted at any of the schedulers along the call path's during call setup phase. Substituting in (1) after converting it to an equality to reserve the least amount of resources required for meeting the delay bound and solving for η_f, we get:

$$g_f^i = \frac{\frac{\sigma_f - L}{R_f} + \sum_{j=1}^{K_f}(L/R_f^j)}{D_f - \sum_{j=1}^{K_f}\alpha^j} R_f^i \forall i \in [1, K_f] \ . \qquad (11)$$

Note that computing the rate to be reserved at each switch using the above division policies may result in a case in which the rate computed from equations (7) or (9) is greater than the remaining capacity at one or more schedulers, i.e. $g_f^n \geq R_f^n$ for some $n \in [1, K_f]$. We denote such schedulers as *resource-limited schedulers.*

It can be shown [1] that using the RCP policy in conjunction with the proposed CAC algorithm guarantees the absence of resource-limited schedulers when accepting a new call. Thus, this case is only present with EVEN and CP policies. There are two approaches for handling the existence of resource-limited schedulers on accepting a new call:

a. Use an algorithm that reserves all of the remaining capacity (i.e. $g_f^i = R_f^i$) at such schedulers and then redistributes the rest of the delay requirement on other schedulers using (7), or (9).
b. Reject the incoming call.

The first approach seems to be more efficient. However, it requires more state information to be exchanged among the switches and also requires more computations to be made by the CAC algorithm. The presented simulations results are mainly based on the second approach (simply rejecting the new call). Reference [2] presents the simulation results when using the first approach.

4 Simulation Results

We have simulated the operation of the proposed CAC algorithm and the associated resource-division policies on several network models. The simulation consisted of generating a number of calls according to a Poission distribution with an average arrival rate of λ, and a holding time that is exponentially distributed with a mean of $1/\mu$. The value $\rho=\lambda/\mu$ characterizes the call load offered to the model.

The main objective is to compare the blocking probabilities of different division policies. An estimate of the blocking probability is computed as the number of blocked calls divided by the total number of generated calls. We simulated the following configurations of link capacities for each network model:

1-*Configuration (A)*: In this configuration, all links have the same capacity. Therefore, the results for the EVEN and CP policies are always the same.

2-*Configuration (B)*: In this configuration, link capacities are chosen in proportion to the expected call load.

3-*Configuration (C)*: In this configuration, link capacities are chosen in inverse proportion to the expected call load. It may be argued that this assignment of capacities is not typical for a properly planned network. However, we argue that there are two reasons leading to the importance of studying such configuration:

a. It may be difficult at the time of initial network planning to determine the actual load distribution pattern on the network. Furthermore, as the network evolves in terms of the number of nodes and the number of users, the actual load distribution pattern may deviate largely from the expected one. Hence, this configuration provides a "worst-case" condition of network planning.
b. The network may consist of several sub-networks, which are owned by multiple organizations, and consequently it becomes difficult to put a link capacity configuration for the whole network.

For brevity, we only consider the case in which resources are reserved in only one direction of the call (from calling party to called party). This is typical of real-time broadcast applications. The results of the more general case, in which resources are reserved in the two directions of the call has yielded similar results [2].

We start by introducing a relatively simple network model and simple traffic characteristics to allow us to explain the simulation results qualitatively. We then move to a more sophisticated network model in which we offer calls with more realistic traffic characteristics and delay requirements.

4.1 Model 1: Merge-Split Network

In this model (see Fig. 1), calling group 1 and calling group 2 initiate calls to called group 1 and called group 2 respectively. Generated calls are distributed equally between calling group 1 and calling group 2.

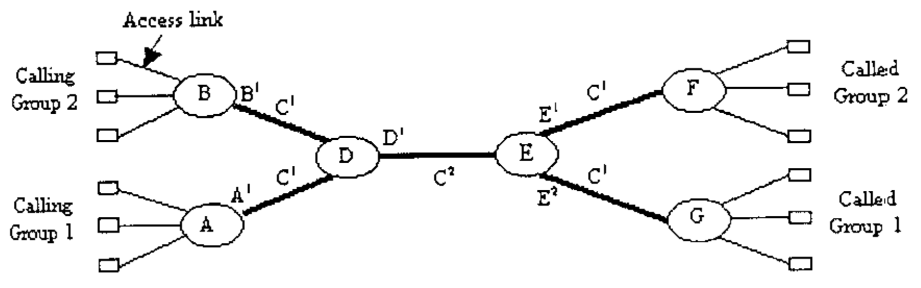

Fig. 1. Topology of network model 1

The network topology suggests that non-even division policies can introduce an improvement in the blocking probability by putting more resource load on the four branching links (A⇔D, B⇔D, E⇔F, E⇔G) and thus increasing the number of calls that can be served by the bottleneck link D⇔E.

In the absence of local stability, the number of calls that a switch can accept is limited by the minimum of the two bounds given by inequalities (3) and (4). However, only the scheduler stability bound limits the number of accepted calls at a switch when the reserved rates of accepted calls are lower than their average rates. We need to remove the switch stability bound when comparing the performance of different policies because they differ from each other in the reserved rate values. We do this by setting the average rate of incoming calls to zero whenever the reserved rate values for these calls are lower than the average rate used in simulations (32 Kbps). Those cases will be marked by (*) in the simulation results.

For a better interpretation of the simulation outcome, the results show the average allocated rate at each switch, which is computed as the sum of rates reserved to all the accepted calls at the switch in a simulation run divided by their number.

The call load distribution for a total offered call load of (ρ) is as follows:

Call load on link D⇔E= ρ

Call load on links A⇔D, E⇔G= call load from calling group 1= $\rho/2$

Call load on links B⇔D, E⇔F= call load from calling group 2= $\rho/2$

The results show different values of the blocking probability corresponding to the source traffic burst size in ATM cells. The average allocated rate at each switch is in Kbps units. Simulation parameters are as follows:

Generated calls per simulation run = 100000	Access speed = 128 Kbps
Source average rate = 32 Kbps	Call load (ρ)= 100 Erlangs
Required delay bound = 100msec	

Configuration (A). Here we take, $C^1 = C^2 = C$= 1.5 Mbps

Table 1. Simulation results for configuration (A) of model 1

Burst Size	EVEN/CP		RCP			
	Blocking	Av. allocated rate	Blocking	Av. allocated rate		
				D^1	A^1, E^1	B^1, E^2
1*	0.06872	14.8	$<10^{-5}$	11.16	17.91	17.83
5	0.57815	34.55	0.55986	33.16	64.63	64.27
10	0.75385	59.23	0.75534	59.23	114.54	112.99
20	0.87078	108.58	0.87003	109.71	210.35	208.46

This configuration shows that RCP can provide an improvement over the EVEN policy. RCP allocates the rates in inverse proportion to the call load, i.e. the rate reserved on link D⇔E is approximately half the rate allocated on the branching links. The results show that RCP gives a lower blocking probability than EVEN for smaller burst sizes, however, the RCP improvement over EVEN diminishes with the increase in burst size.

Configuration (B). Here we take, $C^1 = C$= 1 Mbps, C^2 = 2C

Table 2. Simulation results for configuration (B) of model 1

Burst Size	EVEN		CP			RCP		
	Blocking	Av. allocated rate	Blocking	Av. allocated rate		Blocking	Av. allocated rate	
				A^1,B^1, E^1,E^2	D^1		A^1,B^1, E^1,E^2	D^1
1*	0.00348	14.8	0.23019	12.3	24.7	0.00205	14.4	16.8
5	0.45772	34.6	0.69169	32.2	64.3	0.56713	43.1	44.8
10	0.68772	59.4	0.83234	56.9	113.8	0.74368	73.3	75.7
20	0.82455	108.9	0.91053	106.4	212.8	0.85997	131.2	136.3

The results show that the blocking probability of CP policy is higher than those of EVEN and RCP policies. This is because CP allocates a higher rate on the higher capacity and more loaded switch D1. This implies that CP defeats the main objective of proper network planning, which is to provide higher capacity to the more loaded links. We conclude that if a network is planned such that the capacity of each link is proportional to the expected call load offered to it, then CP policy should not be used.

Configuration (C). Here we take, C^1 =2C, $C^2 = C$=1 Mbps

Table 3. Simulation results for configuration (C) of model 1

Burst Size	EVEN		CP			RCP		
	Blocking	Av. allocated rate	Blocking	Av. allocated rate		Blocking	Av. allocated rate	
				A^1,B^1, E^1,E^2	D^1		A^1,B^1,E^1,E^2	D^1
1*	0.34602	14.8	0.06872	19.7	9.9	0.00032	26.5	8.0
5*	0.72378	34.6	0.67203	59.2	29.6	0.66369	110.6	28.9
10	0.83959	59.2	0.81855	108.6	54.3	0.82808	202.6	54.0
20	0.91053	108.6	0.91053	103.7	207.3	0.91265	362.2	104.7

The results show that CP/RCP performance is much better than EVEN policy. Yet, the performance gain decreases with the increase in burst size. We conclude that applying CP/RCP can prove useful in networks where actual call load pattern is drastically different from the anticipated one.

4.2 Model 2: Full Mesh 4-Nodes Network

In this section, a more sophisticated network model (see Fig. 2) is used. More realistic values of traffic characteristics are used in each simulation run. We also take the server stability limit into account, i.e. we don't set the source's average rate to zero when this rate is less than its reserved rate. In this model, calling group 1 and calling group 2 initiate calls to called group 1 and called group 2, respectively.

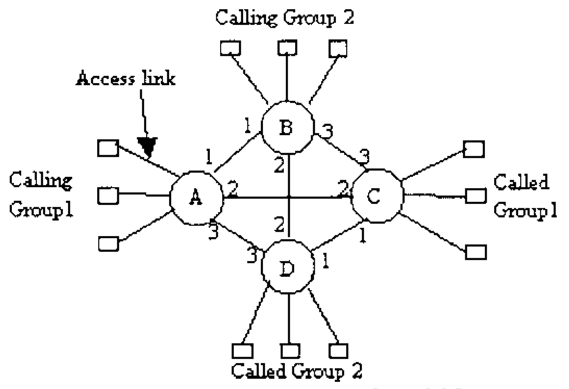

Fig. 2. Topology of network model 2

Generated calls are distributed equally between calling group 1 and calling group 2. Incoming calls from a calling group have five possible paths to the corresponding called group. The path of an incoming call is chosen at random from one of the five possible paths (i.e. random routing). Fig. 3 shows the call load pattern when using random routing.

The call load will be changed for each simulation run to keep the blocking probability in the order of 10^{-3}-10^{-5}. The offered traffic characteristic is selected as follows [3]:

Average rate (r)= 10^m Kbps, m is uniformly distributed on [0,3]
Burst size = y * r kbit, y is uniformly distributed on [0.5, 1.3]
Delay bound = $50*10^s$ msec, s is uniformly distributed on [0, 1.52]

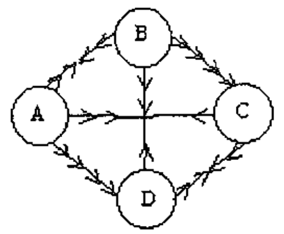

Fig. 3. Call load pattern with random routing

This range of generated traffic patterns include a typical MPEG video source with average rate = 518.4 kbps, and burst size = 576 kbit. It also includes a typical packetized voice source with average rate = 10 kbps, and burst size = 8 kbit. Link capacities are chosen from the standard Plesiochronous Digital Hierarchy (PDH) and Synchronous Digital Hierarchy (SDH) values. Access speed is chosen to be 6 Mbps, which is typical of ADSL (Asymmetric Digital Subscriber Line) units. The number of generated calls per simulation run is 100,000.

Configuration (A). Here we take the capacity of all links = C= 45 Mbps (T3)

Table 4. Simulation results for configuration (A) of model 2

Call load	Blocking	
	EVEN/CP	RCP
400	0.00017	0.00238
420	0.00086	0.00462
440	0.00098	0.00913
460	0.00277	0.01284
480	0.00381	0.01923
500	0.00465	0.02435

Configuration (B). Here we assign capacities in proportion to the call load shown in Fig. 3. We will not be able to strictly apply the rule at all links because some links, such as link A⇔C have different call loads in the reverse and forward directions and since we are assuming all the links to be symmetric (i.e. same capacity in the forward and backward paths), we will assign those links a capacity that is proportional to the higher call load. We thus have the following link capacity configuration:

A⇔B= A⇔C= C⇔D= D⇔B=C, and C⇔B= A⇔D=4C (instead of 2C to match the speed factor used in PDH). , C = 8.448 Mbps (E2)

Table 5. Simulation results for configuration (B) of model 2

Call load	Blocking		
	EVEN	CP	RCP
60	0.00054	0.00701	0.00115
80	0.00152	0.0148	0.00458
100	0.00478	0.03004	0.01141
120	0.00908	0.03835	0.02023
140	0.01356	0.05069	0.03313
160	0.01871	0.06644	0.04655

Configuration (C). Here we assign capacities in inverse proportion to the call load shown in Fig. 3. We thus have the following link capacity configuration:

A⇔B= A⇔C= C⇔D= D⇔B= 4C, and C⇔B= A⇔D= C, C=8.448 Mbps (E2)

Table 6. Simulation results for configuration (C) of model 2

Call load	Blocking		
	EVEN	CP	RCP
20	0.00006	0.00006	0.00006
40	0.00107	0.00107	0.00157
60	0.00686	0.00588	0.00974
80	0.01501	0.01452	0.02149
100	0.02473	0.02379	0.03505
120	0.03596	0.03533	0.05169

5 Enhancing Call Blocking Probability with Resource-Based Routing

This section presents the simulation results of model 2 when applying *Resource-based routing* in which the call path is selected as the least loaded path among the possible paths between the communicating parties. Except for configuration (A), where the number of generated calls per simulation run was taken to be 500,000 to allow accurate simulation of higher arrival rates associated with higher loads, all other parameters are the same as those of random routing simulations.

Configuration (A).

Table 7. Simulation results for configuration (A) of model 2, with resource-based routing

Call load	Blocking	
	EVEN/CP	RCP
780	0.000362	0.001382
800	0.0005	0.002508
820	0.000914	0.003128
840	0.00159	0.004544
860	0.00256	0.007326

Comparing tables 7,8, and 9 with tables 4,5, and 6, respectively shows the significant improvement introduced by the proposed resource-based routing. Note that comparable blocking probability values are achieved with much higher load values with resource-based routing.

Configuration (B).

Table 8. Simulation results for configuration (B) of model 2, with resource-based routing

Call load	Blocking		
	EVEN	CP	RCP
160	0.00003	0.03436	0.00006
180	0.00011	0.04173	0.00065
200	0.00029	0.05208	0.00162
220	0.00125	0.06039	0.00344
240	0.00251	0.06805	0.00688

Configuration (C).

Table 9. Simulation results for configuration (C) of model 2, with resource-based routing

Call load	Blocking		
	EVEN	CP	RCP
300	0.00002	0.00014	0.00001
320	0.00011	0.00057	0.00018
340	0.00067	0.00075	0.00075
360	0.00123	0.00271	0.00168
380	0.00295	0.00373	0.00331

6. Conclusions

We have studied resource allocation policies for ATM networks employing PGPS scheduling. We have addressed the problem of local mapping of end-to-end delay requirement into a local rate to be reserved at each switch along the path of an incoming call. Our findings can be summarized as follows:

1-For a network in which higher capacities are assigned to the links handling more call load, the CP policy is very inefficient because it allocates higher rates on those links. Furthermore, with larger burst sizes, CP cannot allocate smaller rates on lower capacity links and thus results in higher blocking.

2-In [7], it is argued that load imbalances can be viewed as resource imbalances, i.e., the node with a higher load may be thought of as a node with a load identical to other nodes but with smaller amounts of physical resources. Simulation results have revealed a flaw with the above statement. The imbalance in physical resources does not have the same effect as that of load imbalance because physical imbalance is a static effect, while load imbalance is a dynamic effect that depends on the network state. This explains why the amount of CP improvement (or deterioration) does not change with the offered call load while RCP does. This means that RCP can be better than EVEN for some value of call load and worse for another one. Thus, measuring RCP performance must be done at the expected call load "operating point".

3-Non-even division of end-to-end QoS (for PGPS or any other scheduling discipline) results in unfairness among network paths as it increases the number of acceptable calls on a certain path at the expense of the other intersecting paths. This means that when a non-even division policy achieves a lower overall blocking probability than that for EVEN policy, it comes as a result of reducing the blocking probability of some paths, while increasing it for other paths but with less amount. This is in contrast to routing, which works to select the path with minimum loading and may be configured to aim at equalizing the call load among the different paths. Therefore, we suggest the use of EVEN policy for topologies with many intersecting paths (e.g. model 2), and the use of RCP for simpler topologies (e.g. model 1) with small number of intersecting paths. For the particular case of using PGPS scheduling, the improvement over EVEN policy diminishes with the increase in burst size and increases with the increase in the delay bound.

4-The use of resource-based routing greatly enhances the performance of all policies. This is achieved at the expense of having to employ a link-state protocol for distributing schedulers remaining capacity and an algorithm for determining the least loaded path. This may result in longer call-setup times.

References

1. A. Ayad, K. Elsayed, M. El- Hadidi: Local Resource Allocation for Providing End-to-End Delay Guarantees in ATM Networks Using PGPS Scheduling. Proc. of Mascots'99 (1999)
2. A. Ayad: Resource Reservation for Deterministic end-to-end Delay Services in ATM Networks. M.Sc. Thesis. Faculty of Engineering. Cairo University (2000)
3. V. Firoiu, J. Kurose, D. Towley, Efficient Admission Control for EDF Schedulers. Proceedings of IEEE INFOCOM'97 (1997)
4. D. Ferrari, D. Verma: A Scheme for Real-Time Channel Establishment in Wide-Area Networks. IEEE JSAC, vol. 8, no. 4, pp. 368-379, April (1990)
5. P. Goyal, H.M. Vin: Generalized Guaranteed Rate Scheduling Algorithms: A Framework. TR-95-30, Department of Computer Sciences, University of Texas at Austin (1995)
6. S. Keshav: An Engineering Approach to Computer Networking: ATM Networks, the Internet, and the Telephone Network. (1995). Addison-Wesley
7. R. Nagarajan, J. Kurose, D. Towsley: Local Allocation of End-to-End Quality of Service in High-Speed Networks. Proceedings of IFIP Workshop on the Performance Analysis of ATM Systems, Martinique, Jan. (1993)
8. 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)
9. S. J. Golestani: A Framing Strategy for Congestion Management. IEEE JSAC, vol. 9, no. 7, September (1991)

Worst-Case Deterministic Delay Bounds for Arbitrary Weighted Generalized Processor Sharing Schedulers

Robert Szabó, Peter Barta, Felician Németh, and jozsef Bíró

High Speed Networks Laboratory
Dept. of Telecommunications and Telematics, Technical University of Budapest
Stoczek u. 2, H-1111, Budapest, Hungary
{szabor,barta,nemethf,biro}@ttt-atm.ttt.bme.hu

Abstract. Generalized Processor Sharing (GPS) is realized as the most important ideal fluid scheduling discipline in guaranteed QoS networks. The well-known problem of GPS-based rate-proportional servers is the bandwidth delay coupling which leads to inefficient resource utilization. In this paper we introduce the concept of non rate-proportional (or arbitrary) weighting of sessions in GPS systems. Such an approach in a GPS node of network constrained by leaky buckets can handle bandwidth and delay parameters independently, thus allowing better utilization of network resources. Moreover, we show that even under the traditional bandwidth delay coupled system (rate-proportional weighting) it is possible to determine tighter delay bounds of sessions than that of presented in earlier papers. A numerically inexpensive algorithm, which works in any arbitrary weighted GPS system is also presented for computing delay bounds. Besides the analytical work numerical examples are also shown.

1 Introduction

The primary QoS requirements of applications in multi-service telecommunication networks such as end-to-end packet delay, loss and bandwidth must be provided on a per-connection basis for guaranteed performance services. A network meets these requirements primarily by appropriately scheduling its resources, which in turn assumes the use of proper traffic scheduling algorithms within individual network elements (switches, routers) [1, 2, 7].

Traffic scheduling algorithms usually operate on packet/cell level and assume there are different *sessions*, which traffic is to be scheduled for. The task of service disciplines at switching nodes is to control the order in which incoming packets are served in order to provide (end-to-end) *performance bounds*. Based on delay and fairness properties, one of the most significant scheduling algorithms is Generalized Processor Sharing (GPS) [5, 6], which is a generalized version of Uniform Processor Sharing [4]. The packet-by-packet version of GPS (PGPS) is essentially the same as Weighted Fair Queuing (WFQ) [3], however they were independently developed.