Performance Evaluation of Resource Reservation and Call Admission Policies for Deterministic Services in PGPS-based Packet Networks

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

Abstract. We address the issue of reserving resources at packet 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 a number of resource reservation policies that are used to map the end-to-end delay requirement into a local rate to be reserved at each switch. The proposed reservation policies are the uniform reservation (EVEN) policy, the capacity proportional (CP) policy, and the remaining capacity proportional (RCP) policy. We present extensive simulation results to evaluate the performance of these resource allocation policies for various topologies and traffic characteristics. We also propose a resource-based routing algorithm and show the performance gain when it is used.

Keywords: Resource reservation, call admission control, resource-based routing, packet-by-packet generalized processor sharing, deterministic services, ATM networks.

1. INTRODUCTION

One of the main promises of multi-service networks is to provide specific services 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 form different applications according to their contracted QoS level. Many packet-scheduling disciplines have been proposed in the literature to implement such mechanisms (see [3], [8], [12], and [15]). Each scheduling discipline requires algorithms for performing call admission control (CAC) and resource reservation. Most of the work in the literature assumes uniform resource reservation. In other words, a call entering into service is allocated the same amount of resources along its selected path. In this paper, we propose CAC algorithms and non-uniform resource allocation 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:

- 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?
- How to divide the resource requirement among the switches on the call's path? A simple even allocation policy would be to reserve the same amount of resources at all switches. However, it may be more efficient to use a policy that takes switches capacities and/or loading into account.
- How much gain (if any) would be obtained from applying non-even resource allocation policies? What are the factors controlling the gain value?

The following terms will be used throughout the paper:

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).

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.

There have been several trials to handle the problem of resource reservation and call admission control for packet networks providing deterministic service. In an early work [18], it is shown that if reservation is based on the peak rate of each connection, the network will be under-utilized by

guaranteed service traffic when the traffic is bursty. They first show that local deterministic delay bounds can be guaranteed over a link for bursty traffic even when the sum of the peak rates of all the connections is greater than the link speed. This allows a multi-fold increase in the number of admitted connections when the traffic is bursty. The results can be efficiently extended from a single switch to a network of arbitrary topology by using rate-controlled service disciplines at the switches.

In [11], development of a nodal metric defined as the "relative gain ratio" that predicts the relative performance of QoS allocation policies in a network setting is presented. Computation of the relative gain ratio and direct evaluation of allocation policy performance for two simple network models is done. It is found, however, that with the packet loss probability as the QOS metric, there is little difference in the performance of allocation policies in the regime of applications with low loss requirements. For applications which tolerate large packet loss or alternate QOS metrics, however, QOS allocation policies differ significantly in their performance. This work is related to the work presented here. We focus on the case of session requesting deterministic end-to-end delay bounds where the network implements PGPS scheduling.

Reference [9] discusses the problem of Resource partitioning which is useful for a number of applications, including the creation of virtual private subnetworks and of mechanisms for advance reservation of real-time network services. The paper gives the results of using admission control tests for resource partitioned servers for four representative scheduling disciplines, FIFO, WFQ, RCSP and EDF. The simulations confirm the intuition that resource fragmentation losses due to resource partitioning are small and that resource partitioning reduces the admission control computation overhead. An interesting result from the simulation experiments is that, under circumstances that arise naturally in multi-party communication scenarios, resource partitioning results in higher overall connection acceptance rate.

In a highly related work, [4] presents a general framework for admission control and resource reservation for multicast sessions. Within this framework, efficient and practical algorithms that aim to efficiently utilize network resources are developed. The problem of admission control is decomposed into several subproblems that include: the division of end-to-end QoS requirements into local QoS requirements, the mapping of local QoS requirements into resource requirements, and the reclaiming of the resources allocated in excess. However, we present here a more rigorous set of allocation policies and extensive simulation results on various topologies and traffic conditions.

In [17], a study of adaptive control policies and dynamic resource scheduling algorithms which would efficiently and optimally allocate network resources (e.g., bandwidth and buffers), while maintaining the best possible quality of service for the different classes of calls is presented. However, the results are not related to any particular scheduling discipline.

In [16], the problem of call admission and resource reservation control in ATM networks, where customers can book their connections in advance is addressed. Point-to-point as well as point-to-multipoint connections is considered. They present a detailed mathematical model and different solution techniques are proposed and their performance compared. They show by simulations that one can gain significant benefit from using bandwidth scheduling.

In [1], a study of resource allocation for PGPS based scheduling in packetized voice networks is presented. A similar policy to what is being proposed herein is presented in that work. However, the results were only carried out on a single test network and for the specific case of packetized voice.

In [14], the problem of resource allocation and CAC for EDF scheduling is discussed. Various non-uniform resource allocation policies were handled and their qualitative performance compared.

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 allocation policies. Section 4 presents the performance analysis 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

Packet-by-packet Generalized Processor Sharing (PGPS) [2, 12] is a non-preemptive scheduling discipline that closely approximates the behaviour of the non-realizable Generalized Processor Sharing (GPS) scheduling discipline. In PGPS and its variants, each packet is labelled with a service tag and packets are then served in the ascending order of service tags assigned to them. Connections can be associated with service weights, and they receive service in proportion to this weight whenever they have data in the queue.

In PGPS, service tags are equal to the *finish numbers* of the queued packets. The finish number of a packet is computed as the departure time of this packet had it been served by a GPS scheduler having the same capacity and the same input traffic as that of the PGPS scheduler. That is, PGPS simulates GPS "on the side" and uses the results of this simulation to determine the service order of queued packets. Thus, the higher the rate assigned to a certain call (*f*), the less finish time it gets under simulated GPS, and the earlier it starts to receive service under PGPS.

The computation of the finish number of a packet is a non-trivial operation because the service rate of a packet under GPS at time *t* depends on the number of backlogged calls at time *t*. Thus, the service rate is itself time varying and must be updated on the arrival and departure of each packet. This means that the finish number of one packet must be computed as if the packet has been served by variable service rates until it finishes its service. This is known as the problem of *iterated deletion.*

The complications of finish number computation led to the proposal of other GPS emulations that requires less computatinal power to compute the service tags of queued packets. Those variants include, for example, Self-Clocked Fair Queuing (SCFQ) proposed in [6], and Start-time Fair Queuing (STFQ) propsed in [8]. It should be noted, however, that variants of PGPS have different fairness characterstics when used to serve best-effort calls. PGPS has the following advantages:

- Because it emulates GPS, it protects calls from each other. This is an important feature to have in public data networks.
- Under certain assumptions, a call can obtain a worst-case end-to-end queuing delay that is independent of the behaviour of other calls. -This allows networks of fair queuing schedulers to provide real-time performance guarantees, which is important for calls with QoS guarantees.
- PGPS gives users an incentive to implement intelligent flow control mechanisms at the end-systems. With PGPS, a source is not required to send at a rate smaller than its currently allocated rate. However, a source that consistently sends more than its fair share is likely to lose packets from its own buffers, so it has an incentive to match its flow to the currently available service rate.

On the other hand, PGPS has a disadvantage that it requires per-call (or per-aggregate) scheduler state, which leads to implementation complexity and can be expensive for schedulers that serve large numbers of calls. PGPS also requires an expensive iterated deletion algorithm to compute the finish numbers that are used as service tags. Moreover, it requires explicit sorting of the output queue on the service tag, which requires time and complex hardware or software. Despite these disadvantages, many manufacturers are implementing weighted fair queuing in their router and switch products.

The PGPS discipline can be used to provide performance bounds to calls with QoS guarantees as well as fair resource sharing for best-effort calls. Using PGPS for providing QoS guarantees involves the following differences from using it for best-effort call:

- The calls must have a traffic characterization that conforms to a token-bucket traffic model.
- The operation must be subject to call admission control to avoid over-subscription of the scheduler capacity and the subsequent violation of the provided performance bounds.
- The PGPS weights are chosen to provide a call *f* with a service rate g_f that gives the required end-to-end delay bound.

Noting that PGPS allocates a minimum service rate to each call, Parekh and Gallager [12] proved an important bound on the worst-case end-to-end delay experienced by a call traversing a series of PGPS schedulers which is explored in the next section.

2.1 Delay Formulas

In [7] and [12], it was shown that if a call (*f*) traverses a path of K_f PGPS switches 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 packet from call (*f*) by reserving a certain service rate at each switch along the call's path. The upper bound on delay (D_f) is given by [7] as:

$$D_f^i \le \frac{\sigma_f}{g_f(i)} + \left(\sum_{j=1}^{K_f - 1} \max_{n \in [1, i]} \frac{l^n}{g_f^{n,j}} \right) - \left(\frac{l^i}{g_f(i)} - \frac{l^i}{g_f^{i,K_f}} \right) + \left(\sum_{j=1}^{j=K_f} \pi^j \right) \qquad (1)$$

where:

D_f^i is the upper bound on the end-to-end delay encountered by the i^{th} packet from call *f*, where the packets are numbered starting with the first packet to arrive in a busy period.

$g_f^{n,j}$ is the service rate of the n^{th} packet from call (*f*) at switch (*j*) along the call's path.

$g_f(i) = \min_{j \in [1, K_f]} g_f^{i,j}$, i.e. the minimum service rate received by the i^{th} packet along the call's path.

l^i is the length of the i^{th} packet in bits

$$\pi^j = \hat{\pi}^j + \hat{\omega}^{j,j+1}, \hat{\pi}^j = \frac{l(j)_{max}}{C^j}$$

$$\hat{\pi}^j = \frac{l(j)_{max}}{C^j}$$

$l(j)_{max}$ is the maximum length of the packets served at switch (*j*).

$\hat{\omega}^{j,j+1}$ is the propagation delay from switch (*j*) to switch (*j+1*).

Note that Equation (1) deals with the more general case in which each packet belonging to the same call may be assigned a different rate from other packets of the same call even at the same switch. We will not make use of this case, and will consider the more practical case in which all packets belonging to the same call are served at the same rate at a certain switch. Taking this into account, we get the following simpler form of (1):

$$D_f^i \le \frac{\sigma_f - l^i}{g_f} + \sum_{j=1}^{Kf-1} \frac{\max_{n \in [1,j]} l^n}{g_f^i} + \frac{l^i}{g_f^{K_f}} + \sum_{j=1}^{K_f} \pi^j \qquad (2)$$

where g_f^j = Service rate of all packets of call (*f*) at switch (*j*), $g_f = \min_j g_f^j$. Assuming packet size is of fixed length *L* (for example in ATM networks), (2) is modified to:

$$D_f \le \frac{\sigma_f - L}{g_f} + \sum_{j=1}^{K_f} \frac{L}{g_f^i} + \sum_{j=1}^{K_f} \delta^j \quad (3)$$

where L is the fixed packet length.

Equation (3) shows that larger values of burst size leads to a stronger dependency of the upper delay bound on the minimum allocated rate (g_f). Equations (2) and (3) are only valid when the following conditions are met at each switch (*j*), $j \in [1,2,..,K_f]$.

1-The server stability condition, which requires that:

$$\sum_{k=1}^{N_j} r_k^j \le C^j \quad (4)$$

where N_j is the number of accepted calls at switch (*j*) and r_k^j is the average rate of call (*k*) 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:

$$\sum_{k=1}^{N_j} g_k^j \le C^j \quad (5)$$

where g_k^j is the reserved rate of call (*k*) at switch (*j*).

2.2 The Condition of Local Stability

A call (*f*) is said to be locally stable at a switch (*j*) if [10]:

$$r_f \le g_f^j \quad (6)$$

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 (5) 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 allocation algorithms depends on whether local stability is imposed or not. For sake of brevity, 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 [13].

3. CAC ALGORITHMS AND RESOURCE RESERVATION POLICIES

3.1 CAC Algorithm

The proposed CAC algorithm uses equations (3) to (5) to determine whether to accept or reject a new call. The algorithm is illustrated in Figures 1 and 2. We define the remaining capacity of a PGPS switch (*j*) prior to the acceptance of call (*f*) as:

$$R_f^i = C^i - \sum_{k=1}^{N_f} g_k^i \qquad (7)$$

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.

First it checks for server stability condition and whether the connection mean rate is less than the remaining capacity. The second 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 (2) or (3) with each switch along the call's path reserving a service rate equal to its remaining capacity.

On passing the previous tests successfully, an allocation 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 allocation policies are discussed in the next section. If local stability is imposed, then the local stability condition in (6) must also hold true. If all conditions are met, the call is accepted

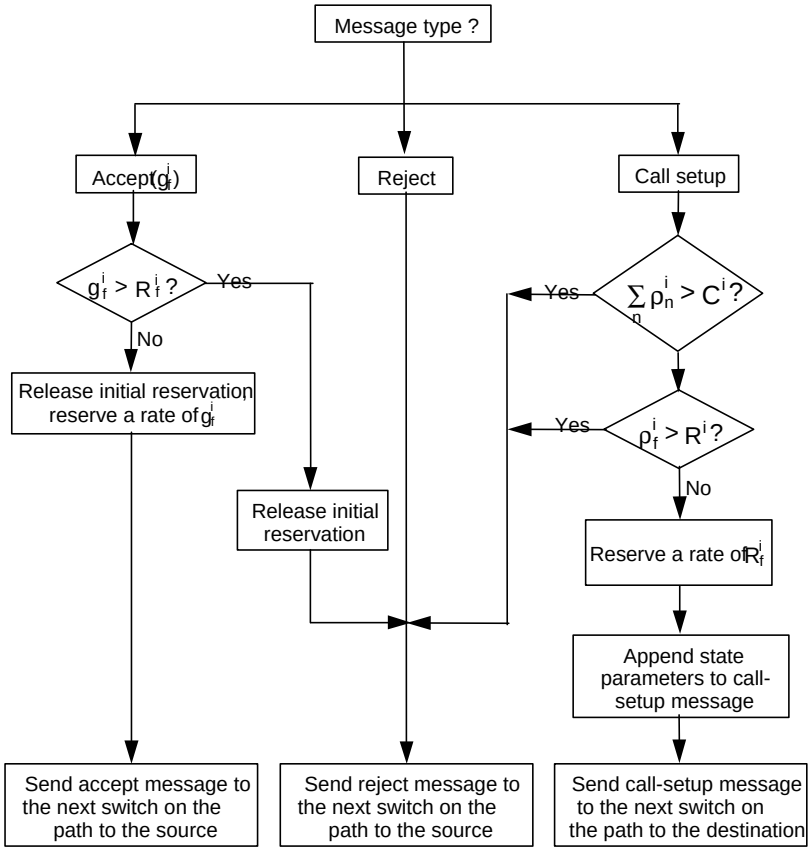

Figure 1. Part (A) of the CAC algorithm with local stability imposed

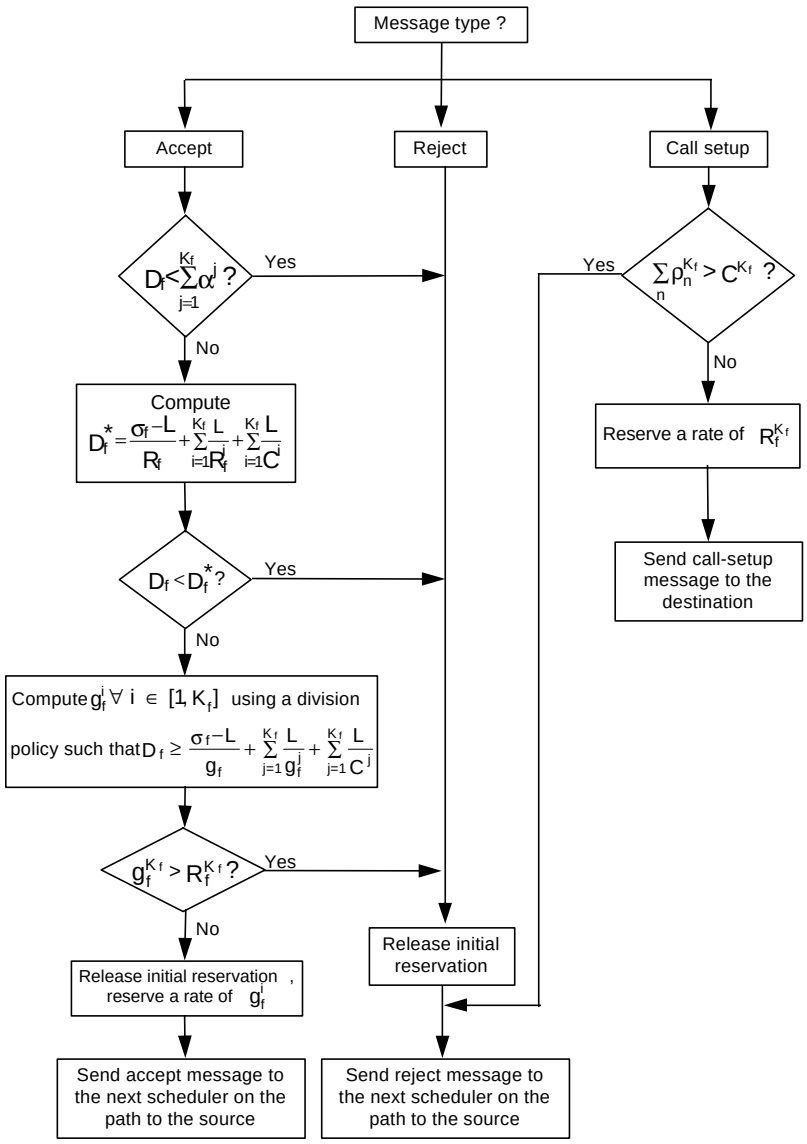

Figure 2. Part (B) of the CAC algorithm used with local stability imposed

3.2 Resource Reservation Policies

In this section, we derive the reserved rate for a call according to the various proposed policies.

Even Policy (EVEN): The reserved rates are the same at all switches, i.e.

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

Substituting in (3), after converting it to an equality to reserve the least amount of resources required for meeting the delay bound of call (*f*), we get:

$$g_f^i = g_f = \frac{s_f + (K_f - 1)L}{D_f - \sum_{j=1}^{K_f} a^j} \quad \forall i \in [1, K_f] \tag{9}$$

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

$$g_f^i = h_f C^i \quad \forall i \in [1, K_f] \qquad (10)$$

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

Substituting in (3), 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:

$$g_f^i = \frac{\frac{s_f - L}{C} + \sum_{i=1}^{K_f} \frac{L}{C^i}}{D_f - \sum_{i=1}^{K_f} a^i} \quad \forall i \in [1, K_f] \qquad (11)$$

where $C = \min_j C^j$.

Remaining Capacity Proportional Policy (RCP): Define $R^i = C^i - \sum_{k=1}^{N_i} g_k^i$ to be the remaining capacity at switch (*i*), where N_i is the number of calls currently allocated to switch (*i*), and g_k^i is the rate allocated to call (*k*) at switch (*i*). The reserved rate to call (*f*) at a certain switch is proportional to the remaining capacity of the switch:

$$g_f^i = h_f R_f^i \quad \forall i \in [1, K_f] \qquad (12)$$

where η_f = Constant for the path and call (*f*) parameters as long as no other calls are accepted at any of the switches along the call path's during call setup phase. Define R_f as min R_i, $i \hat{I}$ [1, K_f] and 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:

$$g_f^i = \frac{\frac{s_f - L}{R_f} + \sum_{j=1}^{K_f} \frac{L}{R_f^j}}{D_f - \sum_{j=1}^{K_f} a^j} R_f^i \quad \forall i \in [1, K_f] \qquad (13)$$

Note that computing the rate to be reserved at each switch using the above allocation policies may result in a case in which the rate computed from equations (9) or (11) is greater than the remaining capacity at one or more switches, i.e. $g_f^n \geq R_f^n$ for some $n \in [1, K_f]$. We denote such switches as **resource-limited switches**.

In Appendix A, we show that using the RCP policy in conjunction with the proposed CAC algorithm guarantees the absence of resource-limited switches 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 switches on accepting a new call:

Use an algorithm that reserves all of the remaining capacity (i.e. $g_f^i = R_f^i$) at such switches and then redistributes the rest of the delay requirement on other switches using (9), or (11).

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 [13] presents the simulation results when using the first approach.

4. PERFORMANCE ANALYSIS

Comparing the performance of the proposed policies requires the definition of a performance metric against which the different policies may be compared. A suitable performance metric would be the call blocking probability of the network. It is difficult to analytically obtain the call blocking probability for general network topologies with different types of traffic being offered to the network. Therefore, simulation is used in to compare the different policies for this case.

4.1 Single Path Network Model

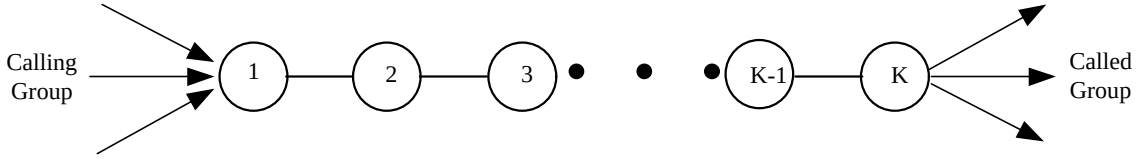

Figure 3. Single Path Network Model

Consider a network consisting of a series of *K* switches connected by links (see Figure 3). Originating calls are generated at node 1 and terminate at node *K*. For the case of a single path network, the maximum number of flows (N) supportable by the path while still meeting QoS guarantees of all accepted flows is a measure of the call blocking probability. The network may be modelled as a M/M/N/N system, in which case the blocking probability can be easily evaluated from Erlang B formula with N servers.

We define the relative gain value of a certain policy with respect to even allocation policy as:

$$G_{policy} = \frac{N_{policy} - N_{EVEN}}{N_{EVEN}} x100\% \qquad (14)$$

Where N_{policy} is equal to the maximum number of simultaneously supported calls on a single path network.

A problem that impedes the estimate of the gain value is that it depends on the following factors:

Link capacities configuration.

Source traffic characteristics.

Requested delay bound.

With the large number of possible combinations of the values of the above factors, it is not easy to derive a general result for the gain value. To evaluate the above policies, the following simplifying assumptions are made:

Source traffic parameters ($\acute{o}_f$,$\tilde{n}_f$) are the same for all flows. They are used as parameters against which the gain value is computed.

Only one class of QoS is assumed, i.e. all flows request the same value of end-to-end delay.

For even policy, (9) gives:

$$N_{EVEN} = \frac{C}{g_f} = \frac{D_f - \sum_{j=1}^{K_f} \alpha^j}{\sigma_f + (K_f - 1)L} C \qquad (15)$$

where $C = \min_j C^j$ and from (11), (13):

$$N_{CP} = N_{RCP} = \frac{C^j}{g_f^j} = \frac{D_f - \sum_{j=1}^{K_f} \alpha^j}{\frac{\sigma_f - L}{C} + \sum_{j=1}^{K_f} (L/C^j)} \qquad (16)$$

Note that the gain of CP & RCP is the same when all switches are initially unloaded, because the proportionality of the remaining capacities at all switches will always be the same as the proportionality of their total capacities (which are the initial remaining capacities for initially unloaded switches).

Using (14), (15), and (16), we get:

$$G_{CP} = G_{RCP} = \frac{(K_f/C) - \sum_{j=1}^{K_f} (1/C^j)}{\frac{(\sigma_f/L) - 1}{C} + \sum_{j=1}^{K_f} (1/C^j)} \qquad (17)$$

We note that the improvement factor is directly proportional to K_f and inversely proportional to σ_f/L.

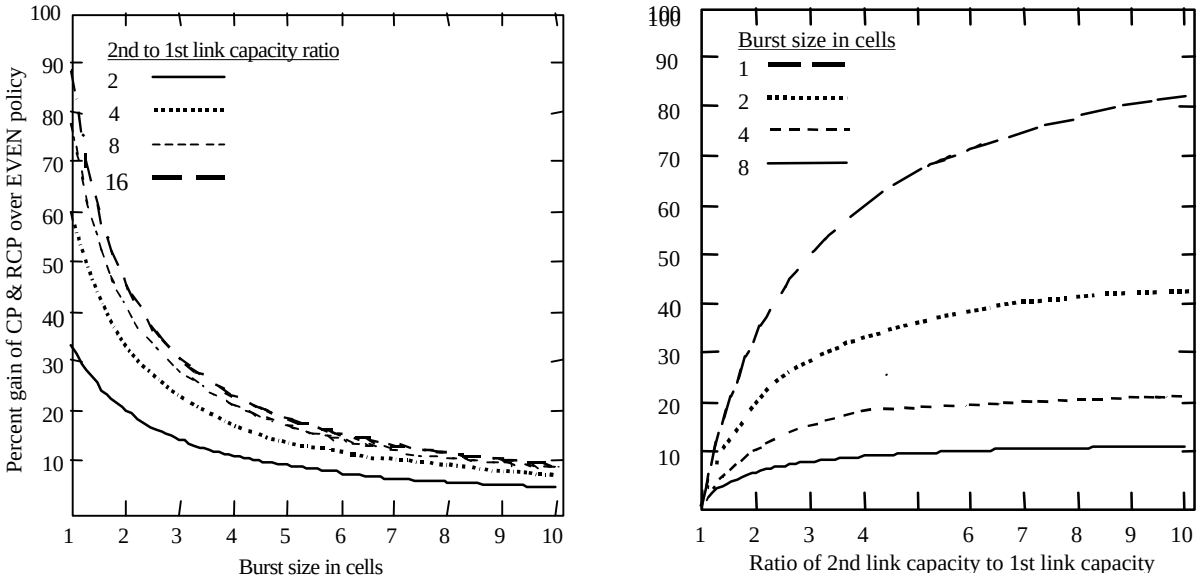

Figure 4. Results for three switches (2 links)

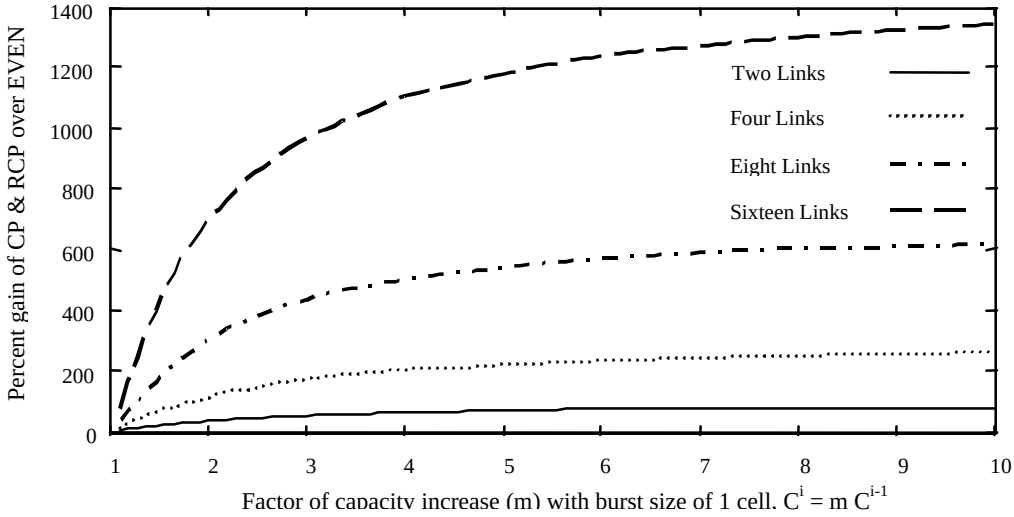

Figure 5. Gain of CP/RCP over EVEN policy with link capacity increasing exponentially along the path

4.2 Simulation Results for General Network Topologies

We have simulated of the operation of the proposed CAC algorithm and the associated resource-allocation policies on several network models. The simulation consisted of generating a number of calls according to a Poisson 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 a model. The simulation was done using C++ and the Sim++[*] simulation engine. All the results are obtained after running multiple simulations runs and obtaining the confidence intervals. We do not show the confidence intervals here since the intervals obtained were very narrow.

The main objective is to compare the blocking probabilities of different allocation 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:

Configuration A: We set all links to have the same capacity. Therefore, the results for the EVEN and CP policies are always the same.

Configuration B: We choose link capacities in proportion to the expected call load.

Configuration C: We choose link capacities 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. Firstly, 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. Secondly, 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 [13].

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

[*] Sim++ is available for download from: http://www.cise.ufl.edu/~fishwick/simpack/howtoget.html

network model in which we offer calls with more realistic traffic characteristics and delay requirements.

4.2.1 Model 1: Merge-Split Network

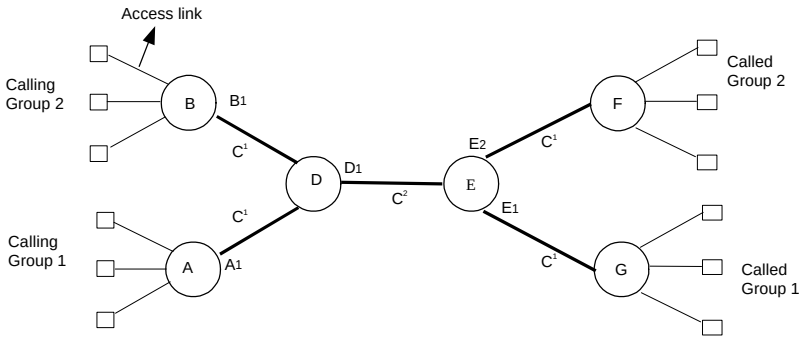

Figure 6. Merge-Split Network Model

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.

The network topology suggests that non-even allocation 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 server 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= r

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

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

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
Source average rate = 32 Kbps
Required delay bound = 100msec

Access speed = 128 Kbps
Call load (r)= 100 Erlangs

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 in all cases, however, the RCP improvement over EVEN decreases 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 never be used.

Configuration C: Here we take, $C^1 = 2C$, $C^2 = C = 1$ Mbps

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.

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

4.2.2 Model 2: Full Mesh 4-nodes Network

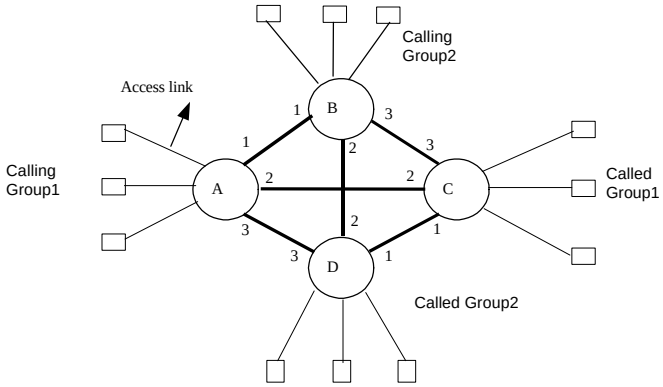

Figure 7. Four Node Full-Mesh Network

In this section, a more sophisticated network model (see Fig. 7) 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.

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. 8 shows the call load pattern when using random routing.

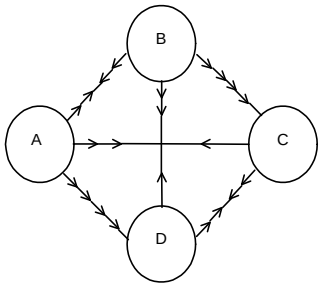

Figure 8. Call load pattern with random routing

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 [5]:

Average rate (r)= 10^{m} Kbps, m is uniformly distributed on [0,3]

Burst size = y * r kbits, y m is uniformly distributed on [0.5, 1.3]

Delay bound = $50*10^{s}$ msec, s m is uniformly distributed on [0, 1.52]

This range of generated traffic patterns include a typical MPEG video source with average rate = 518.4 kbps, and burst size = 576 kbits. It also includes a typical packetized voice source with average rate = 10 kbps, and burst size = 8 kbits. 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). The results are shown in Fig. 9.

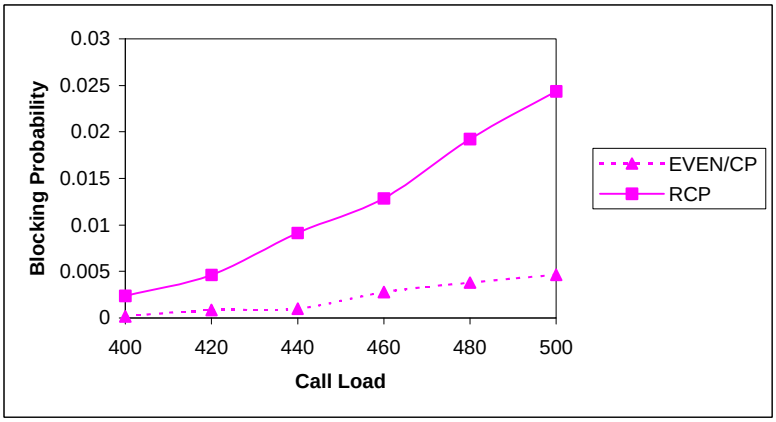

Figure 9. Simulation results for configuration A of model 2

Configuration B: Here we assign capacities in proportion to the call load shown in Fig. 8. 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). The results are shown in Fig. 10.

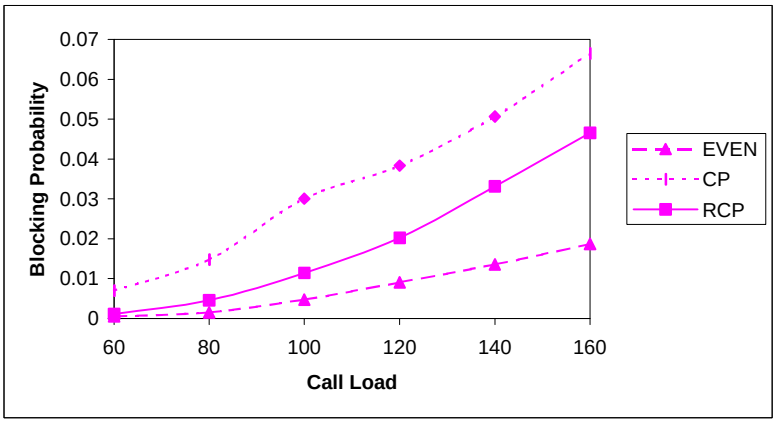

Figure 10. Simulation results for configuration B of model 2

Configuration C: Here we assign capacities in inverse proportion to the call load shown in Fig. 8. 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). The results are shown in Fig. 11.

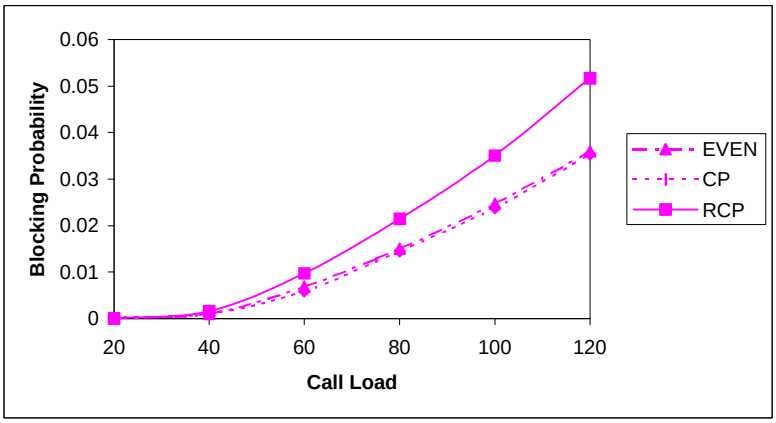

Figure 11. Simulation results for configuration C of model 2

5. ENHANCING CALL BLOCKING PROBABILITY WITH RESOURCE-BASED ROUTING

In this section, we present 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. For configuration A, 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: The results for Configuration A are shown in Fig. 12.

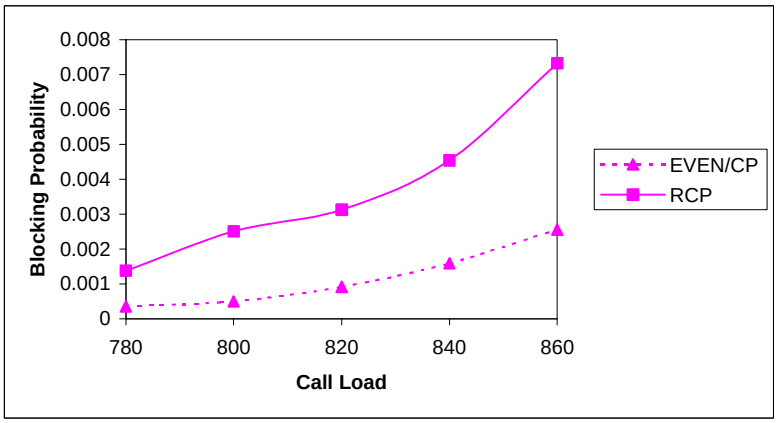

Figure 12. Simulation results for configuration A of model 2, with resource-based routing

Configuration B: The results for Configuration B are shown in Fig. 12.

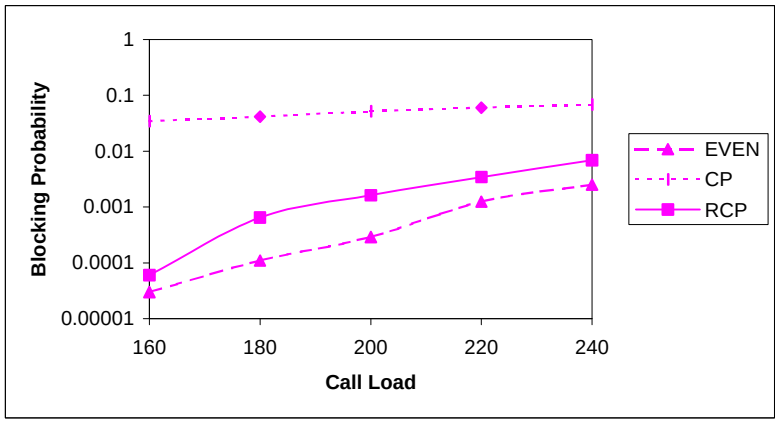

Figure 13. Simulation results for configuration B of model 2, with resource-based routing

Configuration C: The results for Configuration C are shown in Fig. 12

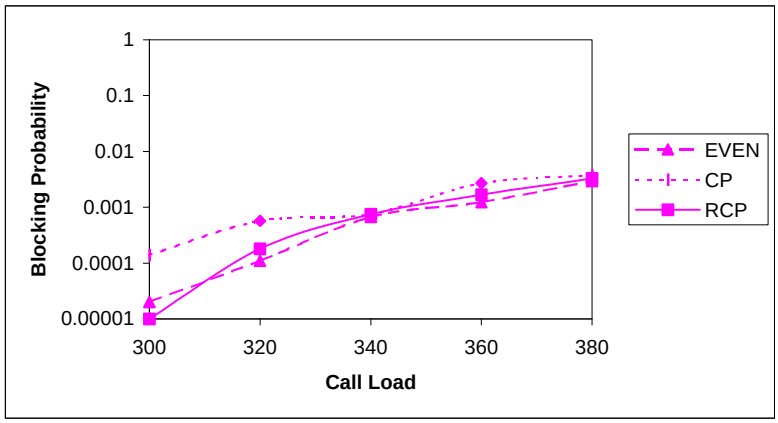

Figure 14. Simulation results for configuration C of model 2, with resource-based routing

Comparing figures 12, 13, and 14 with figures 9, 10, and 11, 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.

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:

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.

– In [11], 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 it does in the case of the RCP policy. This means that RCP can be better than EVEN for some values of the call load but worse for other values. Thus, measuring RCP performance **must** be done at the expected call load operating point.

– Non-even allocation 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 allocation 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.
– 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 the remaining capacity at each switch and an algorithm for determining the least loaded path. This may result in longer call-setup times.

A direct extension of the work presented here is the handling of the case of services requesting probabilistic (vs. deterministic case handled in this work) bounds on the end-to-end delay. The resource based routing presented here is not scalable. A scalable resource based routing algorithm would be a desirable extension.

APPENDIX A

Proof of the absence of the resource-limited switches problem when using the remaining capacity proportional allocation policy.

On the acceptance of a new call, the CAC algorithm imposes the following condition on all accepted calls

$$D_f \geq D_f^* \qquad \text{(A-1)}$$

where D_f^* is the minimum possible end-to-end delay bound that can be guaranteed to call (f) i.e. when reserving all of remaining network resources along the path, thus we have from (3):

$$D_f^* = \frac{\sigma_f - L}{R_f} + \sum_{j=1}^{K_f} \frac{L}{R_f^i} + \sum_{j=1}^{K_f} a^j \qquad \text{(A-2)}$$

Since the rates are assigned such that $g_f^i = \eta_f R_f^i \; \forall \; i \in [1, K_f]$, then we have from (3)

$$D_f = \frac{\sigma_f - L}{h_f R_f} + \frac{1}{h_f} \sum_{j=1}^{K_f} \frac{L}{R_f^i} + \sum_{j=1}^{K_f} a^j \qquad \text{(A-3)}$$

Substituting (A-2) and (A-3) into (A-1), we get

$$\frac{\sigma_f - L}{R_f} + \sum_{j=1}^{K_f} \frac{L}{R_f^i} \le \frac{\sigma_f - L}{\eta_f R_f} + \frac{1}{\eta_f} \sum_{j=1}^{K_f} \frac{L}{R_f^i} \qquad \text{(A-4)}$$

From which we get:

$$\eta_f \le 1 \qquad \text{(A-5)}$$

Since we have $g_f^i = \eta_f R_f^i \; \forall \; i \in [1, K_f]$, using (A-7) gives that:

$$g_f^i \le R_f^i \; \forall \; i \in [1, K_f] \qquad \text{(A-8)}$$

It follows from (A-8) that using the proposed CAC algorithm in conjunction with RCP policy guarantees that there are no resource-limited switches.

REFERENCES

[1] M. Baldi and F. Risso, Efficiency of Packet Voice with Deterministic Delay, IEEE Comm. Magazine, pp. 170-177, May 2000.

[2] A. Demers, S. Keshav, and S. Shenker, Design and Analysis of a Fair Queuing Algorithm, Proceedings of ACM SIGCOMM'89, Austin, 1989.

[3] D. Ferrari and D. Verma, A Scheme for Real-Time Channel Establishment in Wide-Area Networks. IEEE JSAC, vol. 8, no. 4, pp. 368-379, April 1990.

[4] V. Firoiu and D. Towsley, Call Admission and Resource Reservation for Multicast Sessions, Proceedings of IEEE INFOCOM'96, pp. 94 – 101, March 1996.

[5] V. Firoiu, J. Kurose, and D. Towsley, Efficient Admission Control of Piecewise Linear Traffic Envelopes at EDF schedulers, IEEE/ACM Transactions on Networking, Vol. 6, No. 5, pp. 558-570, Oct. 1998.

[6] S. J. Golestani, A Self-Clocked Fair Queuing Scheme for Broadband Applications, Proceedings of IEEE INFOCOM' 94, pp. 636-646, Jun. 12-16, 1994.

[7] P. Goyal and H.M. Vin, Generalized Guaranteed Rate Scheduling Algorithms: A Framework, IEEE/ACM Transactions on Networking, Vol. 5, No. 4, pp. 561-571, Aug. 1997.

[8] P. Goyal, H. Vin, and H. Cheng, Start-time Fair Queueing: A Scheduling Algorithm for Integrated Services Packet Switching Networks, IEEE/ACM Transactions On Networking, Vol. 5, pp. 690-704, Oct. 1997.

[9] A. Gupta and D. Ferrari, Resource Partitioning for Real-Time Communication, IEEE/ACM Transactions on Networking, Vol. 3, No. 5, pp. 501-508, Oct. 1995.

[10] S. Keshav, An Engineering Approach to Computer Networking: ATM Networks, the Internet, and the Telephone Network. Addison-Wesley, 1995.

[11] R. Nagarajan, J. Kurose, and 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.

[12] A. K. Parekh and R. G. Gallager, A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Multiple Node Case, IEEE Trans. on Networking, vol. 2, pp. 137-150, 1994.

[13] A. Saad, Resource Reservation for Deterministic end-to-end Delay Services in ATM Networks. M.Sc. Thesis. Faculty of Engineering. Cairo University, 2001.

[14] A. Saad, M. El-Hadidi and K. Elsayed, Resource Division Policies for EDF Scheduling in ATM Networks, Proceedings of IEEE International Symposium on Computers and Communications, pp. 249-254, July 2001.

[15] H. Sariowan, R. Cruz, and G. Polyzos, SCED: A Generalized Scheduling Policy for Guaranteeing Quality-of-Service, IEEE Trans. On Networking, Vol. 7, pp. 669-684, Oct. 1999.

[16] F. De Turck, P. Demeester, and H. Alaiwan, Bandwidth Scheduling Method for Efficient Resource Allocation in ATM Networks, Proceedings of IEEE ATM Workshop, pp. 385–393, May 1998.

[17] E. S. Valeroso and M. Alam, Performance Analysis of Resource Reservation Strategies in Broadband Networks, Proceedings of IEEE IPCCC'98, pp. 307 – 313, Feb. 1998.

[18] H. Zhang and D. Ferrari, Improving Utilization for Deterministic Service in Multimedia Communication, Proceedings of the International Conference on Multimedia Computing and Systems, pp. 295–304, 1994.