CAC Performance with Self-Similar Traffic: Simulation Study and Performance Results

Yanping Wang, Carey Williamson
Department of Computer Science
University of Saskatchewan
Saskatoon, Saskatchewan
Canada S7N 5A9
E-mail: yaw073, carey@cs.usask.ca

Julie Doerksen
Department of Computer Science
University of Calgary
Calgary, Alberta
Canada T2N 1N4
E-mail: julie@cpsc.ucalgary.ca

Abstract

This paper investigates, through simulation, the performance of five Connection Admission Control (CAC) algorithms namely, PCR CAC, SCR CAC, AVG CAC, GCAC, and Norros CAC, when presented with synthetically generated homogeneous self-similar traffic sources. Various traffic and system parameters have been used. The simulation results show that statistical multiplexing gains both within a source and across sources should be exploited to improve network resource utilization. Source granularity and variability have been shown to have the most impact on the CAC performance. The limited impact of the Hurst parameter shows that the long-range correlation structure of the traffic sources can be neglected in evaluating CAC performance, at least when the buffer size is small. The CAC algorithms should be more conservative when link capacity is low and more aggressive when link capacity is high. While none of the CAC algorithms performs satisfactorily in all scenarios, the Norrors CAC and AVG CAC perform better than the others.

Keywords: *Connection-Admission-Control, ATM Networks, Simulation, Performance-Evaluation.*

1. Introduction

Connection Admission Control (CAC) is an important traffic management mechanism used in high-speed Asynchronous Transfer Mode (ATM) networks. A good CAC algorithm tries to achieve high network utilization, while meeting the QOS requirements of all accepted connections. To increase network utilization, multiplexing techniques are required. There are two types of multiplexing gains. Multiplexing gains *within a source* come from the bursty on-off nature of an individual traffic source and the ability of the switching system to buffer short-term bursts temporarily for later delivery. Multiplexing gains *across sources* come from the fact that most sources generate bursts independently (*i.e.*, they are not all "on" or "off" at the same time). As a result, the bandwidth required for *N* aggregate traffic sources is less than the sum of the bandwidths required for the individual traffic sources.

The difficulty in applying statistical multiplexing arises from the diverse characteristics of network traffic and our limited knowledge of these characteristics. Leland *et al.* [13] first demonstrated the self-similar fractal-like behavior of Ethernet traffic in 1993. Since then, self-similarity has been identified and studied in various types of traffic sources, such as VBR video traffic [2][8], TCP/IP WAN traffic [17], WWW traffic [3], and Frame Relay traffic [7]. It shows that real traffic is bursty and that the bursts exist over many time scales. Self-similarity influences the queuing behavior of aggregate traffic and has an impact on the design and analysis of many network engineering problems. This paper investigates its impact on the performance of Connection Admission Control algorithms.

Many connection admission control algorithms have been proposed in the past ten years [1][4][5][9][10][12]. They make different assumptions on traffic characteristics, QOS requirements and target network utilization. Most currently available CAC algorithms assume short-range dependent (SRD) traffic and ignore the existence of long-range correlations or heavy-tailed burst size distributions. Hence they tend to underestimate the impact of large sustained bursts on network performance. This typically results in overly optimistic performance predictions and inadequate network resource allocation [6].

Different schemes also assume different traffic and system parameters. Some algorithms depend exclusively on the Usage Parameter Control (UPC) parameters defined for ATM call signaling. This simplifies the admission process. However, the decisions made can be inaccurate due to poor, or even misleading, traffic parameters. Some schemes assume bufferless configuration [9] or additive bandwidth allocation [9][12].

These assumptions typically result in conservative CAC performance and inefficient use of network resources.

Therefore, it is difficult to compare the performance of different CAC algorithms based on the papers in which they were proposed. It is even more difficult to predict their performance when presented with self-similar traffic.

Discrete-event simulation is used in the experiments. The five CAC algorithms are PCR CAC, SCR CAC, AVG CAC, GCAC and Norros CAC. The simulations are conducted using the *ATM-TN* (ATM Traffic and Network) simulator, a cell-level simulator developed in the *TeleSim* project (http://www.wnet.ca/telesim).

The rest of the paper is organized as follows: Section 2 provides some background information on self-similar traffic and the CAC algorithms. Section 3 describes the experimental methodology. Section 4 presents the simulation results. Section 5 concludes the paper.

2. Background

2.1 Self-Similar Traffic

Intuitively, self-similarity refers to the presence of visually similar "burstiness" in network traffic across many time scales. This fractal-like behavior of aggregate traffic contradicts the general assumption of "Poisson-like" aggregate traffic, which becomes smoother as more sources are added.

Formally, a *covariance-stationary* process $X = \{x_k\}$ is *self-similar* if its autocorrelation function decays hyperbolically, *i.e.*,

$$\gamma_X(k) \sim |k|^{-\beta}, \text{ as } |k| \to \infty, \text{ where } 0 < \beta < 1. \quad (1)$$

Traditional traffic processes have autocorrelation functions that decay exponentially, if not faster, *i.e.*,

$$\gamma_X(k) \sim \alpha^{|k|}, \text{ as } |k| \to \infty, \ 0 < \alpha < 1. \quad (2)$$

Self-similarity manifests itself in different ways [6]. It has positive correlations across many time scales, resulting in a non-summable autocorrelation function. The variance of an aggregated self-similar process decreases more slowly than the reciprocal of the sample size used to compute the variance. The spectral density of a self-similar traffic source increases without limit as frequency goes to zero.

Hurst parameter, which is derived by $H = 1 - \beta/2$ where β is the hyperbolic decay parameter in Equation (1), has been used to characterize these phenomena in self-similar traffic [13]. It expresses the degree of self-similarity in a given traffic source. Short-range dependent (SRD) processes have $H = 0.5$. Long-range dependent (LRD) processes have $0.5 < H < 1$. Most traditional traffic models do not capture the LRD property of real traffic [13].

2.2 CAC Algorithms

Connection admission control algorithms can be classified as either deterministic or statistical schemes [18]. Deterministic schemes typically need simple traffic parameters, such as the Peak Cell Rate (PCR) or the Sustained Cell Rate (SCR). Statistical schemes, on the other hand, usually require more traffic parameters and an explicit traffic model. Multiplexing gains, either within or across sources or both, are often considered in statistical schemes. This makes them attractive for bursty traffic.

PCR CAC is a deterministic scheme [18]. It requires a bandwidth equivalent to the Peak Cell Rate (PCR) of the call to be reserved at the time of call arrival. No cell losses will occur, assuming that the stated PCR correctly reflects the maximum cell generation rate of the traffic source. However, the bandwidth is significantly underutilized, especially when the difference between the PCR and the SCR is large. The low bandwidth utilization makes PCR CAC impractical. Nevertheless, it provides an upper bound on the bandwidth required for the traffic in the network.

SCR CAC is another deterministic scheme. The call is accepted if the available bandwidth of the network is no less than the SCR of the incoming call, otherwise the call is rejected. In the long run, SCR allocation is adequate under the infinite buffering assumption, since (by definition) the network can eventually "catch up" with any bursts. However, delay can be significant when buffer size is too large, and the buffers cannot be infinitely large in practice. Therefore, SCR CAC is just an "ideal" scheme, which cannot be employed by any real system. It provides a lower bound on the bandwidth allocation required in the call admission process.

AVG CAC requires a bandwidth equivalent to the average of the PCR and the SCR of the call to be reserved at the time of call arrival. The call is accepted only when the available bandwidth is greater than the average value. The difference between the allocated bandwidth and the SCR value is the extra bandwidth allocated to handle the burstiness in the traffic.

GCAC (Generic CAC) is an algorithm described in the PNNI specification [1]. It is used in the path selection process by the edge nodes in the network to determine links that will likely have enough bandwidth for the incoming connection.

GCAC requires three generic parameters to be advertised by the switches. As shown in Figure 1, ASR (Aggregate Sustained Rate) is the sum of the SCRs of all accepted connections. The Cell Rate Margin (CRM) is the difference between the Actual Allocated Capacity (AAC)

and ASR. It represents the safety margin that the switching system has allocated to accommodate the existing traffic flows [1]. The Variance Factor (VF), which is CRM normalized by VAR, is defined as $VF = CRM^2 / VAR$, where $VAR = \sum_i SCR_i (PCR_i - SCR_i)$. VAR is an estimate of the variability of the traffic source, and is different from the variance or the Norros variance coefficient. The CRM is not pre-specified when the first connection enters the network. Thus the aggressiveness or conservatism of GCAC hinges on the VF chosen for the first call (*i.e.*, CRM). Once specified, it is always maintained as more connections are accepted into the network. For the experiments described below, the CRM of the first call is set to $(PCR_1\text{-}SCR_1)/2$.

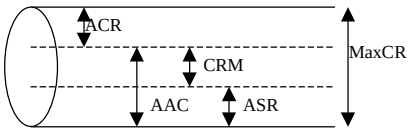


Figure 1. Parameters Used in GCAC

Norros CAC is based on a compact and parsimonous self-similar traffic model called *Fractional Brownian Motion* (FBM) [15][16]. Traffic characteristics are described in this model with three parameters: *m*, *a*, and *H*. The mean rate, *m*, measures the *volume* or *quantity* of the traffic. The other two parameters specify the *burstiness* or *quality* of the traffic. The Norros variance coefficient, *a*, measures the *peakedness* of the traffic (or the magnitude of fluctuations about the mean rate). It is defined as the ratio of the variance to the mean *at the unit time scale*. The Hurst parameter, *H*, describes the degree of self-similarity in the traffic (*i.e.*, how sustained the bursts and idle periods are when they occur). The bandwidth required is approximated in [16] as

$$C = m + \left(\kappa(H)\sqrt{-2\ln\varepsilon}\right)^{1/H} a^{1/(2H)} b^{-(1-H)/H} m^{1/(2H)},$$

where $\kappa(H) = H^H (1-H)^{1-H}$, b is the buffer size and ε is the target CLR.

Weibull distribution is used to approximate the heavy tail in the queue length distribution. The bandwidth allocated to the first source is composed of the SCR and a certain amount of extra capacity to tolerate bursts. The extra capacity allocated is sensitive to the buffer size and the target CLR, as well as the traffic parameters *m*, *a*, and *H*. This starting value can be adjusted to tune the aggressiveness of the CAC algorithm.

3. Experimental Methodology

This section describes the experimental methodology, including the chosen experimental factors and levels, the network topology, the simulation configurations and the simulation validation.

3.1 Experimental Design

Source granularity, source variability and long-range correlation structure, as characterized by the mean bit rate, the Norros variance coefficient, and the Hurst parameter, are used to examine the impact of traffic characteristics on CAC performance. The capacity of bottleneck link (*C*) is also changed to examine CAC performance under different link capacity. The buffer size is 1,000 cells. The target Cell Loss Ratio (ε) is 10^{-6}. The parameters used in the simulation are listed in Table 1.

Table 1. Factors and Levels Used in the Experiments

Factors	Levels
CAC Algorithms	PCR, SCR, AVG, GCAC, Norros
Source Granularity (*m*)	1 Mbps, **2 Mbps**, 3 Mbps
Source Variability (*a*)	150,000 bit-sec, **300,000 bit-sec**, 600,000 bit-sec
Long-Range Dependence (*H*)	0.5, **0.8**, 0.9
Buffer Size (*b*)	**1,000 cells**
Link Capacity (*C*)	10 Mbps, 20 Mbps, 30 Mbps, ..., 290 Mbps, 300 Mbps
Target CLR (ε)	**10^{-6}**

Note: the levels in bold are used in the baseline configuration.

A baseline configuration is chosen with traffic characteristics matching empirical measurements of Bellcore Ethernet LAN traffic [13]. The source granularity is 2 Mbps. The Norros variance coefficient is 300,000 bit-sec. The Hurst parameter is 0.8. Simulations of this configuration are conducted with the capacity of the bottleneck link ranging from 10 Mbps to 300 Mbps. The remaining simulations are conducted in such a way that one parameter is changed at a time to examine its effect on CAC performance. The performance metrics used are:

- Call Acceptance. This metric reflects the number of calls accepted by a CAC algorithm, at a given link capacity. Each CAC algorithm is presented with an initial list of candidate calls, in the same order. The differences in the call acceptance metric reflect different CAC behaviors.
- Link utilization. This metric reflects the carried load on the bottleneck link. It is measured as the number of cells delivered by the bottleneck link as a percentage of the capacity of the link. It reveals how efficiently the network resources are utilized, which is another major concern of service providers.

- Cell loss ratio. This metric expresses the number of cells lost due to buffer overflow compared to the number of cells sent to the switch. A measured CLR that is less than or equal to the target CLR is desirable.

3.2 Network Topology

A simple network topology with two switches, as shown in Figure 2, has been used for the experiments. One hundred sources are connected to switch A via 10 Mbps Ethernet links, and one hundred destinations are similarly connected to switch B. Each traffic source (Src1 ... Src100) generates bursts of data, which are segmented into cells. The link between switch A and B is the bottleneck ATM link with settable capacity. The five CAC algorithms under investigation are implemented in Switch A and B. In our experiments, CAC algorithms in switch A keep track of the allocated bandwidth and the available bandwidth of the bottleneck link. It determines whether another traffic source can be accepted. Once the admission decision is made, switch A will receive cells from all accepted sources and forward them on the bottleneck link, buffering as necessary. The capacity of the bottleneck link determines how quickly the buffer in switch A can be emptied. If the buffer is full when a cell arrives, the cell is lost.

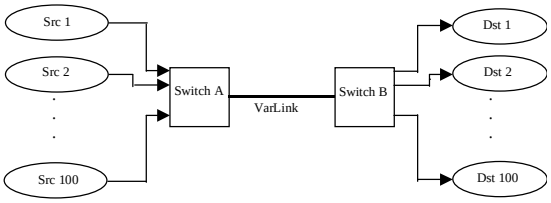


Figure 2. Network Topology Used in Simulations

3.3 Traffic Source Models

Nine sets of traffic sources are generated for the experiments. Their traffic parameters are calculated and listed in Table 2.

The first three sets (*i.e.*, m1, m2, and m3) are used to investigate CAC performance with respect to source granularity. The next three sets (*i.e.*, a1, a2 and a3) are used to investigate CAC performance with respect to source variability. The last three sets (*i.e.*, H1, H2, and H3) are used to examine CAC performance with respect to the long-range correlation structure of the traffic sources. The traffic sources in m2 are used in the baseline configuration. Each group contains 100 traces generated using the same Fractional Auto-Regressive Integrated Moving Average (F-ARIMA) process and the same set of mathematical transformations [20].

The traffic sources are considered *homogeneous* in each set, since they are generated using exactly the same model with the same parameters (through different random number seeds). This does not mean that all 100 sources are identical. Indeed, there can be a wide range in the individual source traffic characteristics generated, as shown by the minimum and maximum values in Table 2. However, all generated sources came from the same statistical distribution, and thus have very similar *m*, *a*, and *H* values.

The *F-ARIMA* based model used in our experiments has been shown in [20] to accurately capture the LRD feature of network traffic. Statistical tests have been used to verify the self-similar property of the generated traffic sources [19].

3.4 Simulation Issues

The warm-up phase of the experiments is 100 simulated seconds. This allows the switch buffer to be filled with some amount of cells before the performance data is collected. The simulations run for another 900 seconds after the startup phase and end when all the data in the sources are sent.

Due to the size of the experimental design, all simulations, except for the baseline configuration, have one replication. Thus no confidence intervals are provided on the graphs (e.g., CLR results). With 900-second simulation, the maximum observed number of cells transmitted by the bottleneck link when link capacity is 10 Mbps is 18,618,402 cells. The results of the CLR performance are reasonably accurate to the magnitude of 10^{-6} (*i.e.*, 10 to 20 cells are lost due to buffer overflow). The maximum number of cells transmitted by the bottleneck link when link capacity is 300 Mbps is 471,013,484 cells. The CLR values are accurate to the magnitude of 10^{-8} when 4 to 5 cells are discarded. Therefore, all the CLR values described in this paper are considered accurate if they are larger than 10^{-6}. The CLR value of 0 in our experiments means that no cell is lost during the 900-second simulation period. This should be interpreted to mean that the CLR value is likely less than 10^{-6} for these scenarios.

For the baseline configuration, simulation replications suggest that confidence intervals are reasonably tight. For example, the baseline configuration has five replications using different phasing parameters. The results in different replications are similar. It shows the limited impact of phasing effects within one data set. Furthermore, experiments for traffic sets m2, a2, and H2, which contain independently generated sets of traffic sources with similar traffic parameters (shown in Table 2), yield very similar CAC performance. Thus we do claim statistical significance for the relative performance differences seen among CAC algorithms, though not for the absolute CLR values reported.

Table 2. Characteristics of the Synthetically Generated Traffic Sources

Traffic Character-istics	Source Group	Number of Sources	*m* (Mbps)			*a* (bit-sec)			*H*
			Min.	Avg.	Max.	Min.	Avg.	Max.	
Source Granularity	m1	100	0.698	**1.060**	1.398	188,482	309,805	445,814	0.8
	m2	100	1.592	**2.026**	2.519	154,466	**309,615**	430,495	**0.8**
	m3	100	2.635	**3.081**	3.550	169,334	304,778	432,576	0.8
Source Variability	a1	100	2.519	2.088	1.817	76,498	**152,179**	232,588	0.8
	a2	100	1.673	1.997	2.368	182,211	**301,781**	486,152	0.8
	a3	100	1.404	2.075	2.722	364,576	**599,899**	817,761	0.8
Long-Range Dependence	H1	100	1.634	2.019	2.428	157,559	299,036	449,334	**0.5**
	H2	100	1.676	2.006	2.433	196,331	301,706	485,365	**0.8**
	H3	100	1.500	2.007	2.405	185,735	307,448	451,604	**0.9**

4. Simulation Results

The CAC performance in the baseline configuration is described in Section 4.1. The effects of different traffic parameters on the performance of CAC algorithms are studied in the rest of this section. These effects include that of the source granularity (Section 4.2), source variability (Section 4.3) and long-range correlation structure (Section 4.4).

4.1 Baseline Configuration

A typical traffic source in the baseline configuration generates 4,944,760 cells in total. Over 85% of the bursts are of size smaller than 100 cells. These bursts contain 46.92% of the total traffic. The maximum burst size is 1,795 cells. The minimum burst size is 1 cell. The peak-to-mean ratio is 3.48.

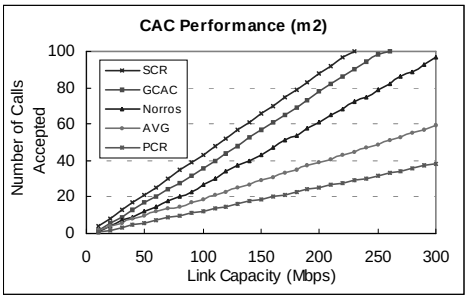
As shown in Figure 4(a), the numbers of calls accepted by all CAC algorithms increase with the increase in the link capacity. SCR CAC and PCR CAC provide the upper and lower bounds on the number of calls accepted. Within these bounds, GCAC is most aggressive and AVG CAC is most conservative. Norros CAC is in between.

Figure 4(b) shows the link utilization achieved by the different CAC algorithms. The CAC algorithm that accepts the most calls, *i.e.* SCR CAC, naturally has the highest link utilization (around 90%). The CAC algorithm that accepts the fewest calls, *i.e.* PCR CAC, has the lowest link utilization (around 30%). The link utilization of GCAC and Norros CAC increases as link capacity increases, or as more calls are accepted into the network. This is because both GCAC and Norros CAC consider multiplexing gains in their admission decisions. Multiplexing techniques play a key role in improving the resource utilization of the network. GCAC has relatively high link utilization (between 60% and 90%). The link utilization achieved by Norros CAC is between 40% and 70%. Although GCAC and Norros CAC have different link utilization, their utilization curves in Figure 4(b) are almost parallel to each other. This means that Norros CAC and GCAC exploit similar degrees of multiplexing gains as link capacity increases. The actual difference in the utilization of the two algorithms comes from their bandwidth allocation for the first call. The link utilization of AVG CAC does not change as link capacity increases since AVG CAC does not consider multiplexing gains across sources and all traffic sources are homogeneous.

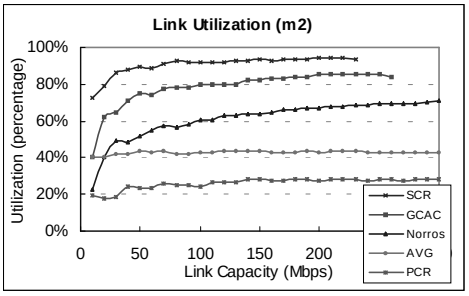
Figure 4(c) shows the CLR performance of the CAC algorithms. The actual CLR achieved by SCR CAC is in the order of 10^{-2} and that achieved by GCAC is in the order of 10^{-3}, which is far from the target of 10^{-6}. The CLR performance of AVG CAC is the best among all the CAC algorithms (besides PCR CAC). It fails to meet the target CLR when the number of calls accepted is small (less than 10). However, when the number of calls accepted increases, the target CLR is satisfied. The CLR performance of Norros CAC does not meet the target CLR, although it improves as more calls are accepted into the network. This shows that bursts in traffic sources have more significant impact on network performance when link capacity is small than when link capacity is large. For example, Norros assigns a bandwidth of nearly 20 Mbps for the first four connections whose individual PCR is 7.53 Mbps and whose aggregate SCR is 6.96 Mbps. This bandwidth allocation is not enough to meet the target CLR. The failures of both GCAC and Norros CAC in meeting the target CLR come from the fact that neither allocates sufficient safety margin for the first call.

An interesting observation for Norros CAC is that as link capacity increases, more calls are accepted, link utilization increases, and CLR performance improves. At first, it seems counterintuitive that higher link utilization is associated with better CLR performance. However, it highlights the importance of exploiting multiplexing gains across sources, especially when link capacity is large. When link capacity is low, the multiplexing gains that can be exploited are small since the number of calls accepted is small. The high bandwidth demand of individual self-similar traffic sources is obvious under such

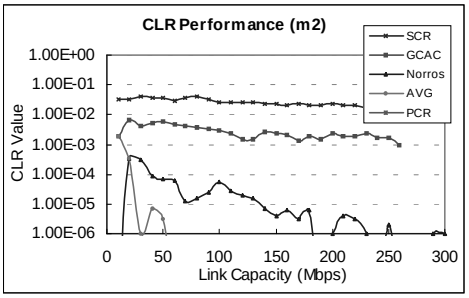
circumstances. The impact of large and persistent bursts on network performance can be observed from the poor CLR performance of all CAC algorithms (except PCR CAC) when link capacity is below 50 Mbps.



(a) Call Acceptance



(b) Link Utilization



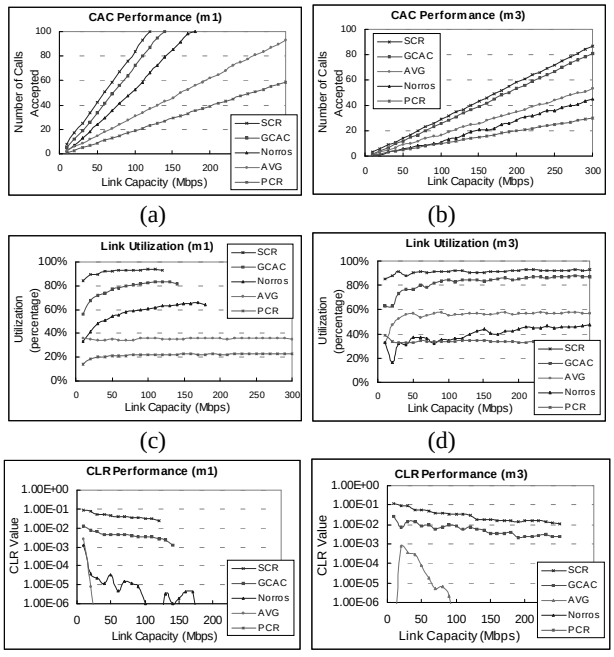
(c) CLR Performance

Figure 4. CAC Performance in the Baseline Configuration

An ideal CAC algorithm should have a bandwidth allocation strategy whose aggressiveness increases as the number of calls accepted increases. Therefore, multiplexing gains can be exploited at different levels to achieve maximum link utilization while meeting QOS requirements. For example, for the baseline configuration, a bandwidth allocation that is close to the PCR of the traffic can be used when link capacity is below 50 Mbps. When link capacity is between 50 Mbps and 100 Mbps, the bandwidth allocation can become close to that of AVG CAC. When link capacity is between 100 Mbps and 200 Mbps, the bandwidth allocation should be somewhere between that of AVG CAC and that of Norros CAC. When the link capacity is above 200 Mbps, bandwidth allocation close to that of Norros CAC is appropriate.

4.2 Effect of Source Granularity

The impact of source granularity on CAC performance is examined by changing the mean bit rate of the traffic sources from 2 Mbps in the baseline configuration to 1 Mbps and 3 Mbps, respectively. The left column of Figure 5 illustrates the CAC performance when the source granularity is 1 Mbps. The right column of Figure 5 illustrates the CAC performance when the source granularity is 3 Mbps. All other experimental factors remain the same.



(a) (b)

(c) (d)

(e) (f)

Figure 5. CAC Performance with Respect to Source Granularity

m = 1 Mbps: (a) Call Acceptance Performance (c) Link Utilization (e) CLR Performance

m = 3 Mbps: (b) Call Acceptance Performance (d) Link Utilization (f) CLR Performance

The numbers of calls accepted by all CAC algorithms increase when the source granularity decreases, and decrease when the source granularity increases. Compared with the baseline configuration, the change in source granularity influences Norros performance more than GCAC performance. AVG CAC is least sensitive to the changes in the source granularity. Although the number of calls accepted by GCAC decreases with increasing source granularity, the difference between the number of calls accepted by SCR CAC and GCAC decreases. This means that GCAC becomes more aggressive as the source granularity increases. Compared with that of PCR CAC, the call acceptance performance

of Norros CAC becomes more conservative as the source granularity increases.

The link utilization of AVG CAC increases from 35% to 43% and to 57% as the source granularity increases from 1 Mbps to 2 Mbps and to 3 Mbps. This increase largely reflects the different peak to mean ratios for these sets of traffic sources. The CLR performance of AVG CAC degrades slightly as the source granularity increases. For link capacity above 100 Mbps, AVG CAC meets the target CLR. It provides the best CLR performance with a link utilization at around 50%.

Although GCAC accepts fewer calls as the source granularity increases, the link utilization and CLR performance of GCAC are insensitive to the change in the source granularity. Its admission decision is too aggressive, regardless of the source granularity.

4.3 Effect of Source Variability

The performance of CAC algorithms with respect to the traffic source variability is discussed in this section. The Norros variance coefficient (a) is 300,000 bit-sec in the baseline configuration. Figure 6 shows the CAC performance when a is changed to 150,000 bit-sec and 600,000 bit-sec, respectively. Everything else is kept the same. The impact of the Norros variance coefficient on CAC algorithm performance is similar to that of source granularity. Interesting observations include:

- The number of calls accepted by all CAC algorithms decreases as a increases.
- Changing a has little effect on the performance of GCAC and AVG CAC.
- The number of calls accepted by Norros CAC is between that of GCAC and that of AVG CAC.

GCAC is the most aggressive algorithm besides SCR CAC. The CLR value achieved by GCAC decreases slightly as a increases. It falls between 10^{-2} and 10^{-4}, which fails to meet the CLR requirement of 10^{-6}. Link utilization achieved by GCAC is quite high and decreases only slightly as a increases. When link capacity is 10 Mbps, the link utilization achieved is 40%. It rises to 80% quickly as link capacity increases. As link utilization increases dramatically, the CLR performance of GCAC degrades slightly. The difference is so small that it is negligible. The performance of GCAC suggests that for self-similar traffic, link utilization over 70% will likely result in CLR performance worse than 10^{-4} assuming a buffer size of 1,000 cells.

Other than PCR CAC, AVG CAC yields the most conservative call acceptance performance, and therefore the lowest link utilization. As a increases from 150,000 bit-sec to 300,000 bit-sec and to 600,000 bit-sec, the link utilization of AVG CAC decreases from about 50% to 42% and to 35% and is insensitive to increasing link capacity. The CLR performance of AVG CAC satisfies the QOS requirement except when link capacity is below 50 Mbps.

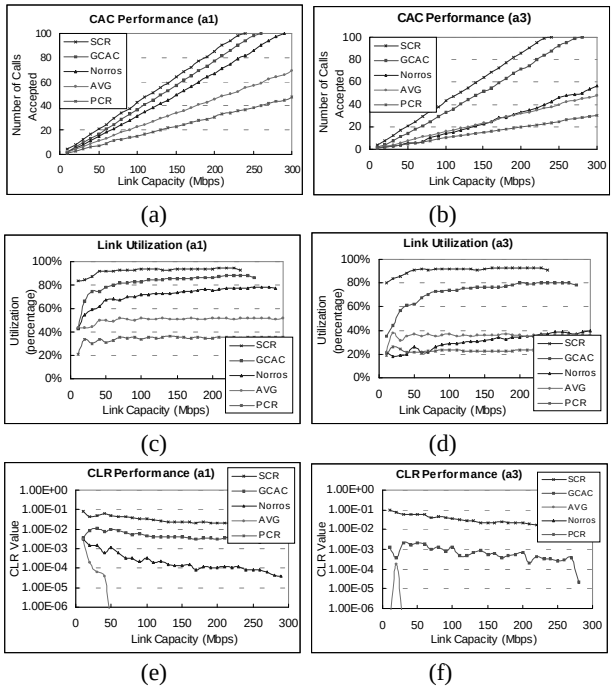


Figure 6. CAC Performance with Respect to Source Variability

a = 150,000 bit-sec: (a) Call Acceptance Performance (c) Link Utilization (e) CLR Performance

a = 600,000 bit-sec: (b) Call Acceptance Performance (d) Link Utilization (f) CLR Performance

Norros CAC is not very sensitive to the change of a from 150,000 bit-sec to 300,000 bit-sec, where the number of calls accepted decreases by only 1 or 2 when link capacity is low (below 150 Mbps) and by 4 to 5 when link capacity is high (above 150 Mbps). However, the number of calls accepted by Norros decreases by about 50% when a increases from 300,000 bit-sec to 600,000 bit-sec. Norros CAC becomes almost as conservative as AVG CAC when a becomes 600,000 bit-sec. The CLR performance of Norros is also more sensitive to the change of a. When a is 150,000 bit-sec, the CLR performance of Norros CAC decreases from 3×10^{-3} to 4×10^{-5} as link capacity increases from 10 Mbps to 290 Mbps. When a is 600,000 bit-sec, the CLR performance of Norros is below 10^{-6}.

The two statistical algorithms (*i.e.*, Norros CAC and GCAC) have link utilization curves for a =150,000 bit-sec and a = 300,000 bit-sec that are almost parallel to each other. In other words, the multiplexing gains exploited by Norros CAC and GCAC are at about the same level. However, this parallel behavior disappears when a increases to 600,000 bit-sec. The multiplexing gains exploited by Norros CAC decrease considerably as a

value increases. Figure 6(b) and 6(d) illustrate that the call acceptance performance and link utilization of Norros CAC when *a* is 600,000 bit-sec is worse than those of PCR CAC when link capacity is below 50 Mbps. They are still worse than those of AVG CAC when link capacity is between 50 Mbps and 200 Mbps. When link capacity rises to over 200 Mbps, the call acceptance and link utilization performance of Norros CAC becomes a little bit better than that of AVG CAC. This shows the problem of Norros CAC. While it realizes the necessity of changing the degree of multiplexing gains as traffic or system parameters change, its adjustment may not be adequate. For example, the multiplexing gains exploited by Norros CAC when *a* is 600,000 bit-sec is insufficient, and its bandwidth allocation is too conservative.

Although the CLR performance of GCAC does not meet the target CLR, it does not degrade as link capacity increases and more multiplexing gains are exploited to improve link utilization. This shows that the multiplexing gains exploited by GCAC is appropriate even when *a* is high (600,000 bit-sec).

4.4 Effect of Long-Range Dependence

The long-range correlation structure of the traffic sources is characterized by the Hurst parameter, *H*. CAC performance with $H = 0.5$ and $H = 0.9$ is given in the left and right columns of Figure 7.

Figure 7(a) shows an abnormal phenomenon in the performance of Norros CAC when *H* is 0.5. Its bandwidth allocation is extremely high, even higher than that of PCR CAC. With such a high bandwidth allocation, the CLR performance of Norros CAC satisfies the QOS requirement. The link utilization achieved is around 20% and it does not increase as link capacity increases. Hurst parameter has important influence on the performance of Norros CAC for several reasons:

First, Hurst parameter determines the degree of multiplexing gains *across sources* that can be exploited by Norros CAC. When Hurst parameter is 0.5, no multiplexing gains across sources are considered. As Hurst parameter increases, more gains are exploited. Therefore, when Hurst parameter becomes 0.8 or 0.9, Norros algorithm accepts too many connections into the network. Link utilization increases to around 75%, yet the CLR performance fails to meet the target CLR.

Second, Hurst parameter also controls the degree of multiplexing gains *within a source* exploited by Norros CAC. When Hurst parameter is 0.5, the extra capacity allocated by Norros CAC over the Sustained Cell Rate of the traffic source increases linearly with the increase in *m*, *a*, and $ln(\varepsilon)$, and decreases linearly with the increase in *b*. As *H* increases, this extra capacity allocation changes more slowly with the changes in *m*, *a*, $ln(\varepsilon)$, and *b*.

Third, for the experiments that we have conducted with Norros CAC, traffic sources with lower *H* value have been allocated higher bandwidth than traffic sources with higher *H* value. This may contradicts the traditional understanding of the self-similarity feature of the traffic sources: higher Hurst parameter should imply more persistent bursts in the traffic, which means that traffic sources with higher Hurst parameter should have more bandwidth allocated, if other parameters of the traffic sources are kept the same.

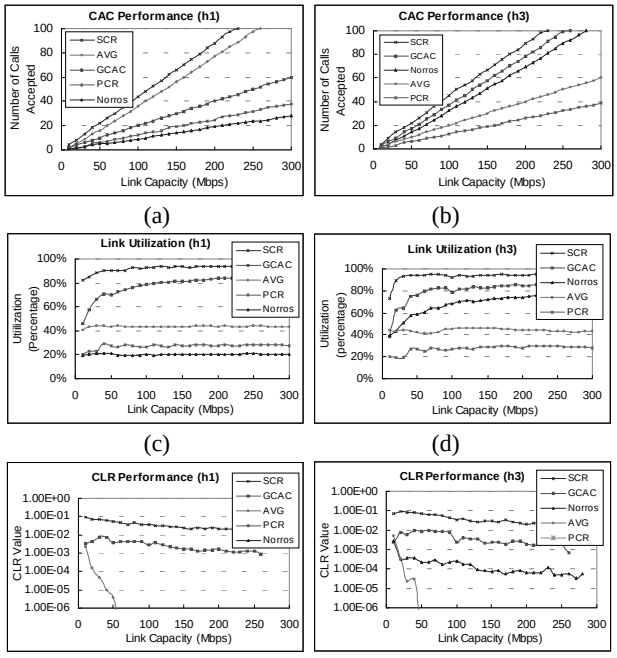


Figure 7. CAC Performance with Respect to Long-Range Dependence

$H = 0.5$: (a) Call Acceptance Performance (c) Link Utilization (e) CLR Performance

$H = 0.9$: (b) Call Acceptance Performance (d) Link Utilization (f) CLR Performance

This cross-over effect has been described by Neidhardt and Wang in [14]. It refers to the fact that the queuing performance of the traffic streams can be divided into three regions. The stream with high-*H* can have better queuing performance in one region, while the stream with low-*H* can have better queuing performance in another region. The cross-over occurs in the region in between where direct calculation is required to determine the queuing performance of the process. High-Hurst process can have better queuing performance since the queuing performance of self-similar traffic depends on a combination of various traffic and system parameters, such as buffer size, link capacity, peakedness, and Hurst parameter. It is true especially with high-speed networks, such as the ATM networks. It has been argued that high-

H process is smoother at small time scales with smaller variances and deviations, but at larger scales, the stronger correlations of the high *H* process produce larger fluctuations. If this is true, then Norros CAC is not necessarily making wrong bandwidth predictions when it allocates more bandwidth for $H = 0.5$ traffic. The only problem is that the traffic sources used in [14] have not only different Hurst parameter, but also different variances and deviations. In our experiments, Hurst parameter is the only parameter in which the traffic sources differ. And the fact that the bandwidth allocation of Norros CAC is higher than the peak cell rate of the traffic sources shows its inadequacy when Hurst parameter changes.

Buffer size is another factor that can cause this cross-over effect. Small buffer size of 1,000 cells has been used in the simulations. As buffer size increases, the extra bandwidth allocated by Norros CAC over the aggregate sustained cell rate of the traffic sources decreases proportionally. When buffer size is 8,000 cells, this reversed bandwidth allocation is still observed, but is much less pronounced.

None of the other CAC algorithms considers the correlation structure of the incoming traffic in their admission decision. Their performance is exactly the same as the Hurst parameter changes. GCAC is still aggressive, with CLR between 10^{-2} and 10^{-3} and link utilization around 80%. AVG CAC has low link utilization (around 45%). Its CLR performance improves from 10^{-2} to 10^{-6} as link capacity increases from 10 Mbps to 50 Mbps. The limited impact of the Hurst parameter illustrates that long-range correlation has no significant impact on network performance, at least when buffer size is small (1,000 cells here). This observation further supports the viewpoint expressed by Grossglauser and Bolot in [11] that while the correlations on all time scales have impact on network performance with infinite queuing assumption, only the correlations up to the implicit system time scale has an effect with finite buffers.

5. Conclusions

This paper evaluates CAC (Connection Admission Control) performance with respect to traffic and system parameters when presented with homogeneous self-similar traffic sources.

The simulation results show that for a target CLR of 10^{-6}, link utilization up to 70% may be achievable with link capacity over 150 Mbps. Source granularity and variability have significant impact on the network performance. Long-range correlation structure has very limited impact on network performance, at least when buffer size is small.

For aggregate self-similar traffic, both CLR performance and link utilization improve as link capacity increases. This illustrates the strong impact of bursts on CAC performance when link capacity is small and the importance of exploiting multiplexing gains to improve network utilization. Multiplexing gains both within a source and across sources should be exploited.

None of the CAC algorithms under investigation provides satisfying overall performance in all the scenarios. It is difficult for CAC algorithms to adjust appropriately to the changing parameters and to balance the trade-off between providing better QOS and reaching higher network utilization. The major problem with deterministic algorithms is that they are based only on one or two simple traffic parameters. While this makes the connection admission control decision simpler, it also makes the decision inaccurate. AVG CAC provides good CLR performance in the experiments conducted when link capacity is above 100 Mbps. However, its link utilization is relatively low, which means that bandwidth utilization is not maximized. Norros CAC is another promising schemes that considers the five key parameters in its admission decision and tries to exploit multiplexing gains. Simulation results in Section 4 show that while Norros CAC tries to adjust its CAC decision with respect to the different parameters, the magnitude of its adjustment is not adequate. The performance of an ideal CAC algorithm has been repeatedly shown to be somewhere between the performance of AVG CAC and Norros CAC. Both AVG CAC and Norros CAC should be more conservative when link capacity is small and more aggressive when link capacity is large.

Acknowledgements

Financial support for this research was provided by the Natural Sciences and Engineering Research Council (NSERC) of Canada, through a Postgraduate Scholarship, a Collaborative Research and Development grant (CRD183839) and Research Grant OGP0121969.
The authors are grateful to the TeleSim project team for building and maintaining the ATM-TN simulator. We also thank the anonymous reviewers for their constructive comments that helped improve the clarity and presentation of the final paper.

References

[1] ATM Forum Technical Committee, *Private Network-Network Interface Specification Version 1.0 (PNNI 1.0)*, af-pnni-0055.000, March 1996.
[2] J. Beran, R. Sherman, M. Taqqu, and W. Willinger, "Long-Range Dependence in Variable-Bit-Rate Video Traffic," *IEEE Journal on Selected Areas of Communications*, Vol. 43, No. 2/3/4, pp. 1566-1579, February/March/April 1995.

[3] M. Crovella and A. Bestavros, "Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes", *Proceedings of ACM SIGMETRICS'96*, Philadelphia, PA, pp. 160-169, May 1996.
[4] A. Elwalid, D. Heyman, T. Lakshman, D. Mitra, and A. Weiss, "Fundamental Bounds and Approximations for ATM Multiplexers with Applications to Video Teleconferencing," *IEEE Journal on Selected Areas in Communications*, Vol. 13, No. 6, pp. 1004-1016, August 1995.
[5] A. Elwalid and D. Mitra, "Effective Bandwidth of General Markovian Traffic Sources and Admission Control of High-Speed Networks," *IEEE/ACM Transactions on Networking*, Vol. 1, No. 3, pp. 329-343, June 1993.
[6] A. Erramilli, O. Narayan, and W. Willinger, "Experimental Queuing Analysis with Long-Range Dependent Packet Traffic," *IEEE/ACM Transactions on Networking*, Vol. 4, No. 2, pp. 209-223, April 1996.
[7] F. Foo and C. Williamson, "Network Traffic Measurements of IP/Frame Relay/ATM," *Proceedings of the Workshop on Workload Characterization in High Performance Computing Environments*, Montreal, PQ, pp. 1-14, July 1998.
[8] M. Garrett and W. Willinger, "Analysis, Modeling and Generation of Self-Similar VBR Video Traffic," *Proceedings of ACM SIGCOMM'94*, pp. 269-280, September 1994.
[9] R. Gibbens and P. Hunt, "Effective Bandwidths for the Multi-type UAS Channel," *Queueing Systems*, 9(1991), pp. 17-28.
[10] R. Gibbens, F. Kelly and P. Key, "A Decision, Theoretic Approach to Call Admission Control in ATM Networks", *IEEE Journal on Selected Areas of Communications*, Vol. 13, No. 6, pp. 1091-1100, August 1995.
[11] M. Grossglauser and J-C. Bolot, "On the Relevance of Long-Range Dependence in Network Traffic," *Proceedings of ACM SIGCOMM'96*, Stanford, CA, pp. 15-24, August 1996.
[12] R. Guerin, H. Ahmadi and M. Naghshineh, "Equivalent Capacity and Its Application to Bandwidth Allocation in High-Speed Network," *IEEE Journal on Selected Areas of Communications*, Vol. 9, No. 7, pp. 968-981, September 1991.
[13] W. Leland, M. Taqqu, W. Willinger, and D. V. Wilson, "On the Self-Similar Nature of Ethernet Traffic (Extended Version)," *IEEE/ACM Transactions on Networking*, Vol. 2, No. 1, pp. 1-15, February 1994.
[14] A. Neidhardt and J. Wang, "The Concept of Relevant Time Scales and Its Application to Queuing Analysis of Self-Similar Traffic," *Proceedings of ACM SIGMETRIC'98*, Madison, WI, USA, pp. 222-232, June 1998.
[15] I. Norros, "A Storage Model with Self-Similar Input," *Queueing Systems*, Vol. 16, pp. 387-396, 1994.
[16] I. Norros, "On the Use of Fractional Brownian Motion in the Theory of Connectionless Networks," *IEEE Journal on Selected Areas of Communications*, Vol. 13, No. 6, pp. 953-962, August 1995.
[17] V. Paxson and S. Floyd, "Wide-Area Traffic: The Failure of Poisson Modeling," *IEEE/ACM Transactions on Networking*, Vol. 3, No. 3, pp. 226-244, June 1995.
[18] H. Perros and K. Elsayed, "Call Admission Control Schemes: A Review," *IEEE Communications Magazine*, Vol. 34, No. 11, pp. 82-91, November 1996.
[19] Y. Wang, "CAC Performance with Self-Similar Traffic," M.Sc. thesis, Department of Computer Science, University of Saskatchewan, Canada, August 1999.
[20] C. Williamson, "Synthetic Traffic Generation Techniques for ATM Network Simulations," *Simulation Journal*, Vol. 92, No. 5, pp. 305-312, April 1999.