A DYNAMIC TIME SCALE APPROACH FOR ON-LINE MEASUREMENT-BASED CAPACITY ALLOCATION

Fatih Hacıömeroğlu and Michael Devetsikiotis

Abstract - Current high-speed networks handle a variety of services, requiring different QoS constraints. The choice of appropriately accurate but also *practically implementable* measurement algorithms in this context becomes crucial. In this paper, we perform a comparative study of alternative on-line bandwidth allocation algorithms, we analyze their complexity, and perform comparisons via simulation experiments. Moreover, we argue the relevance of a dynamic measurement time scale approach and demonstrate the effectiveness of this new approach over the static one. Our motivation is to use these algorithms in the data plane of "self-sizing" frameworks, and make use of their output in taking control plane decisions either locally or globally, in an *on-line* fashion. Previously, no such comprehensive comparison of relevant methods has been carried out, especially from a combined accuracy versus implementation complexity point of view and from the perspective of changing the measurement time scale "dynamically".

Keywords: On-line measurements, bandwidth allocation, effective bandwidth.

Resumo - As atuais redes de alta velocidade transportam uma variedade de serviços com requisitos heterogêneos de Qualidade de Serviço. A escolha de algoritmos de medição precisos e implementáveis é de suma importância para a provisão destes requisitos. Neste artigo, algoritmos para a avaliação em tempo real da capacidade efetiva de fluxos de tráfego são comparados. Tanto a complexidade quanto o desempenho dos mesmos são avaliados. Enfatiza-se a importância da natureza dinâmica dos intervalos de amostragem quando comparada à abordagem tradicional de intervalos definidos estaticamente. As conclusões da comparação são úteis na adoção de algoritmos baseados em medidas no plano de controle de redes auto-ajustáveis.

Palavras-chave: Medições em tempo real, alocação de banda passante, capacidade efetiva.

1. MOTIVATION AND INTRODUCTION

The demand on high-speed networks is getting higher and tougher to satisfy everyday, with the invention and commercialization of new bandwidth-hungry applications. Network resources are to be increased accordingly, to sustain an acceptable level of service. However, it is not always feasible to increase resources at the same pace of the increase in the traffic demand. In this regard, the bandwidth allocation in high-speed QoS-oriented networks is critical and needs to be made dynamic, adaptive and measurement-based, rather than static, to attain a more efficient use of resources. Especially for network links shared through statistical multiplexing, adaptive bandwidth allocation algorithms based on traffic measurements can achieve important gains.

For these reasons, we performed a comparison of on-line, dynamic measurement-based bandwidth allocation algorithms, which have low time complexity, and are based only on measurements instead of unreasonable assumptions about the incoming traffic. To the best of our knowledge, no such comprehensive comparison of relevant methods has been previously carried out, especially from a combined accuracy versus implementation complexity point of view.

We are also motivated by the fact that such algorithms can be used in the data plane of "self-sizing" network frameworks such as [1, 2] and in which every node in the network runs a measurement-based bandwidth allocation algorithm for every traffic class or "band". Periodically, the output of the algorithms, which give the required capacity demands of the traffic types, are collected, and either locally or globally, a control plane action is taken (i.e., the virtual links or scheduling allocations are re-calculated) so as to minimize a predefined objective such as bandwidth cost or maximize revenue.

In this context, it is very important to choose measurement methods that satisfy stringent constraints in terms of both accuracy and complexity. Most of the algorithms we used in this paper naturally originated from the effective bandwidth concept, since effective bandwidth is the amount of the required bandwidth to be allocated for the satisfaction of a QoS constraint. Furthermore, in the literature, we identified two research areas which are related to our aim. These are network traffic prediction [3, 4, 5, 6] and measurement-based admission control (MBAC) [7, 8, 9].

MBAC algorithms are composed of separate measurement procedure and admission criterion. We investigated the applicability of these separable measurement procedures for our purposes. However, other than the trivial Gaussian Approximation bandwidth estimator, such methods are not suitable for on-line re-sizing due to the fact that either they rely on unreasonable information (i.e., they presume that a priori traffic descriptors, such as present number of connections are known) or they have unacceptable computational complexity.

These incompatibilities stem from the different design considerations between measurement-based estimators in MBAC and *self-sizing* frameworks. First, MBACs are designed to operate only in the ingress nodes, where admission decisions are taken. Second, the period of execution of MBAC algorithms is at the *connection level* time scales. Our aim is to

F. Hacıömeroğlu and M. Devetsikiotis are with the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, USA. E-mails: fatih61@ieee.org, mdevets@ncsu.edu.

obtain *on-line* algorithms, working on every node in the network. Therefore their timescale and computational complexity are smaller than connection level timescales. Similar to MBAC algorithms, traffic predictors do not suit our consideration either. The reason this time is not because they are centralized and computationally complex as in MBAC algorithms, but that they do not target a QoS constraint.

The algorithms in this paper take a window of traffic measurements as input, estimate the parameters they need in a bandwidth allocation calculation formula and output the required amount of capacity to be reallocated. The measurements correspond to the amount of incoming traffic during a *slot duration*, t_{slot}. Consequently, the algorithm is called periodically every $N * t_{slot}$ seconds (the reallocation period), where N is the window size. The choices of t_{slot} and N are critical [10] and affect significantly the performance of algorithms.

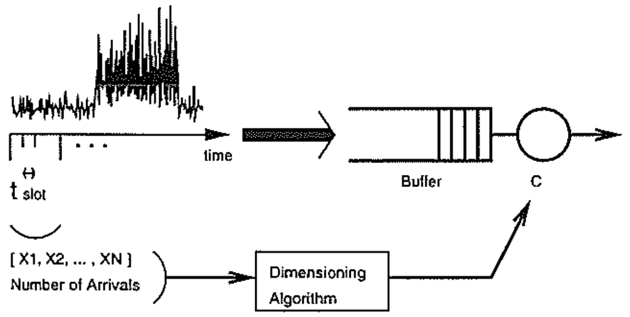

Figure 1. Simulation scenario.

The on-line measurement-based resource allocation algorithms are implemented and tested in a simulation scenario as shown in Figure 1. In our simulations, the performance metric is related to a QoS constraint, in our case packet loss probability. To quantify the amount of expanded resources, we use two cost metrics, namely,

- **Allocation Ratio** (Average Capacity Allocation divided by Average Traffic Rate)
- **Average Queue Occupancy**

Our purpose is to obtain a feasible algorithm which is able to use bandwidth *optimally* (i.e., dynamically as in Figure 2) while still obtaining a performance close to the QoS target. The ideal algorithm should not require any knowledge or unreasonable assumptions on the traffic, and be based completely on measurements.

The remainder of the paper is structured as follows. After describing the algorithms in Section 2, we compare their response to different scenarios (i.e., variable buffer size, level of aggregation of sources, long-range dependence of traffic, computational and memory complexity requirements) and choose promising ones for further simulation study in Section 3. Then we explain our simulation methodology in Section 4. Later, simulation results highlighting the importance of measurement time scales in algorithm performance are presented in section 5. Section 6 introduces the use of a *measurement-based* time scale and shows its effectiveness over the static time scale approach. Finally, we conclude with a discussion of the results and outline prospects for further enhancements.

2. BANDWIDTH ESTIMATION ALGORITHMS

2.1 DIRECT EB ALLOCATION (DEB)

DEB algorithm relies on a direct analytical evaluation, using the definition in [11].

$$eb(s,t) = \frac{\ln(E(e^{sX(0,t)}))}{st} \qquad (1)$$

$X(0,t)$ is the amount of incoming work during a duration of t. The (s,t) parameters are usually called the space and time parameters. Parameter s is calculated by using Large Deviations Theory (LDT) and by making a large buffer assumption. The overflow probability is calculated from an asymptotically exponential decrease assumption.

$$P(B < Q) = e^{-s(C)B} \qquad (2)$$

The time parameter t is related to time scales which are responsible for buffer overflow. It should be chosen small enough so that traffic is observed for buffer overflow analysis. Finally, the expectation in (1) is approximated by a time average, as suggested in [12].

The empirical evaluation of (1) is simulated and compared with analytical effective bandwidth of known Poisson and ON-OFF source types in [13].

2.2 COURCOUBETIS EB ALLOCATION (CEB)

This method [14] is based on LDT and a large buffer assumption, similar to the DEB algorithm:

$$eb = m + \frac{ID\,s}{2B} \qquad (3)$$

The parameters m, B, s and ID are the mean rate, buffer size, space parameter and index of dispersion of $X[0,t]$. The space parameter s is calculated using (2).

ID estimation process has a computational complexity proportional to N^2.

$$ID = Var(X(0,t)) \left(1 + 2\sum_{k=1}^{1/4N}\left(1 - 4\frac{k-1}{N}\right)AC(k)\right) \left(\sum_{k=1}^{N}\frac{X(0,t)}{N}\right)^{-1} \qquad (4)$$

2.3 MANY SOURCES ASYMPTOTIC EB ALLOCATION (MSAEB)

This approach [15] is also based on the effective bandwidth approach similar to the first two algorithms, but unlike them, this algorithm uses a different way of estimating the time and space parameters in (1). An assumption of many sources is made, instead of a Large Buffer assumption, while using LDT to solve the problem of estimation of the space and time parameters (s,t). As described in [15], if M sources are multiplexed in the buffer B, r_j is the percentage of streams of type j, and maximum allowed buffer overflow probability to be guaranteed is e^{-a}, then minimum required bandwidth can be calculated by solving

$$C = \sup_s(\inf_s(R(s,t))) \tag{5}$$

where

$$R(s,t) = \frac{stM\sum_j r_j eb(s,t) + a}{st} - \frac{B}{t} \tag{6}$$

In (6), the $eb(s,t)$ term is found from (1).

For a given t, the $R(s,t)$ is a unimodal function of s, having a unique minimizer. Then, $R(s,t) = R_t(s)$ is solved by using a golden section search method as described in [15]. This process is repeated for a range of t values smaller than the measurement window time, and the maximum among them is taken as the required capacity to be allocated.

2.4 ON-OFF EB ALLOCATION (OOEB)

The idea is to obtain estimation values of an equivalent ON-OFF traffic model from measurements, and substitute them in the specific analytical effective bandwidth formula (7) for ON-OFF sources [16].

$$eb(s,t) = \frac{-sr + a + b - 1/2\,(-sr + a - b)^2 - 2\,ba}{2s} \tag{7}$$

Parameters a, b and r denote a ON-OFF traffic model where ON and OFF periods are exponentially distributed with parameters a and b respectively, and r is the constant traffic generation rate in the ON state. These parameters can be estimated by matching the first three moments of N data measurements falling into the window. These estimations have difficulties and require search algorithms, since direct solution of high order equations is not trivial. Moreover, this method is weak because of the limited fitting spectrum of ON-OFF model [17].

2.5 NORROS EB ALLOCATION (NEB)

In [18], besides introducing modeling of real traffic by fractional Brownian motion (FBM), an effective bandwidth formula (8) for FBM is also given.

$$\begin{aligned} eb = m\ &+ \\ &K(H)\sqrt{-2\,\ln(P_{loss})}^{1/H} \\ &*\ a^{1/(2H)} * B^{-(1-H)/H} * m^{1/(2H)} \end{aligned} \tag{8}$$

where $K(H) = H^H\,(1-H)^{1-H}$ and m, H, $Ploss$, x and a are the mean, Hurst parameter, buffer overflow probability, buffer size and coefficient of variation respectively. Parameter a, is approximated by the index of dispersion. In fact, this is a valid assumption only when the traffic is short range dependent.

The Hurst parameter can be set from a priori measurements. However, to react to unexpected traffic changes, a measurement-based on-line algorithm is favored. Difficulties of H estimation methods are analyzed in [19], where the comparison of several H estimation algorithms indicated that the Abry-Veitch estimator (AV estimator) based on Wavelet theory is a very effective approach [20].

2.6 DRDMW (IMPROVED EMPIRICAL EB ALLOCATION)

This method in [21] is an improved version of empirical effective bandwidth methods. A unified phenomenological framework to estimate overflow probability of both long range dependence (LRD) and short-range dependence (SRD) is put forward by including the Hurst parameter in traditional analytical effective bandwidth methods.

$$P(B < Q) = e^{-s(C)B^{2-2H}} \tag{9}$$

Second, the difficulty of measuring the effective bandwidth of real-time traffic online by using direct estimator [21] is alleviated by using an approach based on dual recursive algorithm with double moving windows (DRDMW), which is introduced in an empirical calculation of analytical effective bandwidth formula instead of using the direct estimator.

2.7 GAUSSIAN APPROXIMATION ALLOCATION (GA)

The simplest resource allocation method existing in literature is the GA method [22], where link buffer is ignored and server capacity is set according to Gaussian arrival rate distribution:

$$C = m + \sigma * \sqrt{-2 * \ln(P_{loss}) - \ln(2 * \pi)} \tag{10}$$

where m and σ are the mean and standard deviation of the arrival rate distribution.

3. COMPARISON OF THE ALGORITHMS

The first algorithm, namely Direct Effective Bandwidth Allocation algorithm, relies on the effective bandwidth formula, and possesses the problem of finding appropriate values for s and t, which depend on QoS requirements and the system parameters. The space parameter is estimated using the Large Buffer Assumption. The time parameter estimation is left somewhat arbitrary, for the time being.

The second algorithm uses (3), which is an alternative generic effective bandwidth definition in terms of the mean rate, index of dispersion, QoS parameter and buffer size. It is simpler, but it still doesn't address long range dependent traffic.

The Many Sources Asymptotic Effective Bandwidth algorithm relies on the effective bandwidth formula (1) and encounters the problem of estimation of (s,t). This method accomplishes it by solving a functional optimization problem. Although it is a very innovative approach, this may be too slow for our motivational *self-sizing* scenario where every node takes on-line measurements of every traffic type.

The ON-OFF Effective Bandwidth formula (7) for the fourth method is obtained by substituting an ON-OFF arrival process instead of $X(0,t)$ in the analytical effective formula. With regard to a practical usage of such expressions, we encountered other problems than estimation of (s,t) parameters, such as model parameter estimation, and goodness of fit of the model.

The Norros Effective Bandwidth Allocation and Gaussian Approximation methods are alternatives not including (s,t) estimation. They are approximate expressions, which are derived independently of the effective bandwidth formula. The Gaussian Approximation algorithm assumes a bufferless link. This will overestimate required capacity. Moreover, the gaussian assumption is not valid for traffic formed by small number of sources. This places a constraint on the source type, however our aim is to have an algorithm capable of functioning without unreasonable assumptions.

The self-similarity is addressed only in the NEB and in the DRDMW method. Others do not discriminate between short range dependence and long range dependence. Although the index of dispersion in the Courcoubetis formula of the second algorithm stands for burstiness of the source, the formula is not for long range dependent traffic. Thus the effective bandwidth approximation on which the Courcoubetis formula is based, (i.e., exponential decay of buffer overflow probability with increasing huffer size) is not valid for self-similar traffic (the decay is hyperbolic and slower than exponential). We provide a summary of our performance comparisons with respect to various network scenarios in Table 1.

The Gaussian Approximation algorithm is the easiest to implement, and suitable to be used as an algorithm setting an *upper bound*, since it does not consider buffer size. The Courcoubetis Effective Bandwidth Allocation is also another easy, and promising one, since this one takes into account buffer also. But neither of the previous two algorithms is designed with long range dependent traffic in mind. Norros effective bandwidth and DRDWM algorithms are the only ones incorporating the Hurst parameter, therefore addressing to long range dependent traffic. Although DRDMW is designed to alleviate the numerical overflows in the direct effective bandwidth allocation, that problem can not be completely alleviated due to the structure of (1). Therefore, we picked the following three algorithms for further simulation analysis:

- Gaussian Approximation (GA)
- Courcoubetis Effective Bandwidth Allocation (CEB)
- Norros Effective Bandwidth Allocation (NEB)

4. SIMULATION METHODOLOGY

The simulation scenario given in Figure 1 is a single server queue simulation where the service rate is changed, in an on-line fashion, periodically based on recent traffic measurements. We used the Sup-FRP traffic model [23]. The simulation flow slides packet by packet, emulating a real case scenario as in an Ethernet card passing packets to upper network layers.

Figure 2 gives a visual representation of how algorithms adjust service rates, tracking fluctuations in the incoming traffic rates, so as not to waste resources.

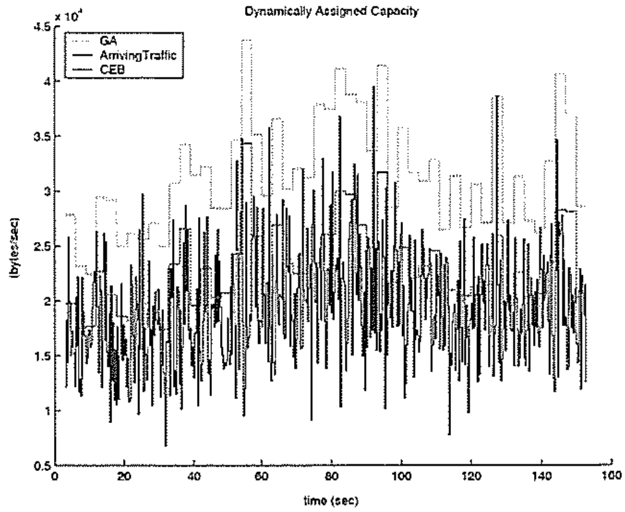

Figure 2. A visual example view of dynamic capacity allocation.

We performed simulations with 5 different t_{slot} values (0.01, 0.05, 0.1, 0.5, 1 s) and 5 window sizes (N) values (3, 6, 30, 60, 300 slots) in every method. Therefore, we had 25 simulations per method. As a total, we present here results of 75 simulations. Also note that the measured statistics in this paper resulted after 30 simulation replications and confidence intervals are insignificant.

In all of the simulations in this section, we generated traffic with the same mean value of 20 Kbytes/s, the same Hurst parameter of 0.7 and the same buffer size of 5 Kbytes. We set the QoS target packet loss probability to 10^{-3}, so as to have a common ground for the performance comparisons of algorithms in the simulation scenario (Figure 1).

Note that the average traffic rate of 20 Kbytes/s is the product of an average packet size of 200 bytes and average packet arrival rate of 100 packets/s. Therefore, the average time between two consecutive packets is 0.01 seconds and the t_{slot} values chosen in the simulations, which are $(0.01, 0.05, 0.1,$ $0.5, 1$ s), correspond to cases where 1, 5, 10, 50 and 100 packet arrivals take place on average in a slot time duration, respectively. Also note that the choices for t_{slot} and N are made deliberately to have simulations where reallocation takes place in every 3 seconds, but with different measurement resolution in the recent history of the measurement data. For example, a simulation with (0.01s, 300 slots) includes 300 measurements, whereas the one with (1s, 3 slots) includes three measurements in the same recent 3 seconds history.

We did not implement an on-line H estimation [24]. We provided the value of H (i.e., 0.7) to the algorithms beforehand, so that we can examine the performance of the bandwidth allocator, independent from the performance of the H estimator. The combined, on-line Hurst parameter and EB estimation is beyond the scope of this paper and is left as future work.

	MSAEB	DEB	OOEB	CEB	NEB	DRDMW	GA
Small Buffer	Very Good	Poor	Poor	Poor	Good	Poor	Perfect
Large Buffer	Very Good	Very Good	Very Good	Very Good	Very Good	Very Good	Poor
SRD Traffic	Good	Very Good	Good	Very Good	Good	Very Good	Good
LRD Traffic	Good	Poor	Poor	Poor	Perfect	Poor	Poor
Many Sources	Perfect	Very Good	Poor	Very Good	Very Good	Very Good	Perfect
Single Source	Poor	Very Good	Good	Very Good	Good	Very Good	Poor
Computational Complexity	Poor	Poor	Poor	Very Good	Good	Good	Perfect
Memory Requirement	Good	Good	Perfect	Good	Good	Perfect	Perfect

Table 1. Performance Comparisons

5. SIMULATION RESULTS

In this section, we first present performance and cost plots. We demonstrate and observe the importance of time scale choice in measurement-based algorithms. At the end of this section, we provide the processing time plots with respect to the window sizes.

5.1 PERFORMANCE AND COST RESULTS

Tables 2, 3 and 4 show respectively the performance and cost results of the GA, CEB and NEB algorithms. We observe how the algorithms response against different t_{slot} and N values in the simulation scenario given in Figure 1.

In Table 2, we observe that the QoS target, which is 10^{-3}, is satisfied in every (t_{slot}, N) combination with the exception of ($t_{slot} = 1$ s, $N = 3$ slots). Note that GA is used as an upper band of resource allocation for comparison purposes. It does not consider buffer size. In fact, it assumes there is no buffer. This is why, it is expected to be more generous than other algorithms. The fact that a violation of QoS took place in this method implies trouble for other methods. The average allocation ratio and the average queue occupancy values agree with P_{loss} values and show that as t_{slot} is increased for a constant N, the allocation ratio decreases towards 1. We also observe that for constant t_{slot}, increasing N results in a larger capacity allocation. But this rate of increase in capacity allocation depends on the t_{slot} value. So, once t_{slot} is *properly* chosen, choosing N looses its importance, since the change in the ratio values are much smaller.

Table 3 tells us that the CEB's performance changes similar to GA against t_{slot} and N variations, but P_{loss} values are relatively about an order of magnitude *higher*. The QoS target is violated for the following (t_{slot}, N) pairs: (0.5 s, 3 slots), (1 s, 3 slots), (0.5 s, 6 slots) and (1 s, 6 slots). This algorithm, considering the presence of buffer, theoretically permits lesser resource usage than GA. Compared to GA, we observe that bigger t_{slot} values lead to better allocation ratios (i.e., ratios closer to 1) and the choice of N has a greater effect on CEB. With proper choice of t_{slot} and N, the same performance can be achieved with lesser resource usage. Here, similar to GA, we see the importance of choosing t_{slot} properly. But unlike GA, here choosing N is also important. This is because the rate of increase of ratio values when N is increased is much more significant. Choosing a large N leads to serious over-allocation.

From Table 4, we observe that the loss probability results of NEB are between the ones of GA and CEB for N values of 3 and 6. However, when N is either 30, 60 or 300, P_{loss} is smaller than other algorithms, which implies an over-allocation of bandwidth, which is justified by the values of the allocation ratio metric. When it comes to the choice of t_{slot}, as in the previous methods, a bigger t_{slot} resulted in smaller resource allocation. This method is the only one which allocates more capacity to the traffic possessing higher long-range dependence. Overall, it can be said that the NEB includes similar performance and cost changes as the ones of CEB, but performance values are around one order of magnitude better, and consequently, cost values are higher. This shows that the NEB has a tendency of allocating more resources than the CEB, and results in better QoS constraint satisfaction.

5.2 COMPLEXITY

Figure 3 shows the processing times of the algorithms as a function of window size values. The processing time is the time required for the algorithm to re-calculate capacity. The processing time is seen to be related to N for CEB and NEB, with a complexity of $O(N^2)$. This result is in parallel with our expectations. CEB and NEB calculate autocorrelations of measurements falling into the measurement window of size N, and this requires a processing time proportional to N^2. Whereas, GA uses only the mean and variance of the measurements, whose calculations are fully on-line. As a result, GA has a complexity of $O(1)$ and regardless of N, its execution time remains much smaller compared to other algorithms.

Window Size (N)		t_{slot} 0.01 s	t_{slot} 0.05 s	t_{slot} 0.1 s	t_{slot} 0.5 s	t_{slot} 1 s
3 slots	Ratio	4.2160	2.5588	2.1108	1.4898	1.3416
	Queue	89	183	254	568	764
	P_{loss}	0	0	0.0001	0.0008	0.0015
6 slots	Ratio	5.0263	2.8748	2.3148	1.5783	1.4044
	Queue	68	126	180	405	566
	P_{loss}	0	0	0	0.0001	0.0004
30 slots	Ratio	5.8932	3.1018	2.4610	1.6431	1.4557
	Queue	45	99	142	322	459
	P_{loss}	0	0	0	0	0.0001
60 slots	Ratio	6.2305	3.1541	2.4843	1.6537	1.4673
	Queue	43	97	139	316	447
	P_{loss}	0	0	0	0	0.0001
300 slots	Ratio	9.2851	3.5462	2.6659	1.7013	1.5080
	Queue	43	96	135	302	416
	P_{loss}	0	0	0	0	0.0001

Table 2. Results for GA

Window Size (N)		t_{slot} 0.01 s	t_{slot} 0.05 s	t_{slot} 0.1 s	t_{slot} 0.5 s	t_{slot} 1 s
3 slots	Ratio	4.8650	1.9259	1.4730	1.0930	1.0453
	Queue	79	354	635	1685	1988
	P_{loss}	0	0	0.0005	0.0098	0.0134
6 slots	Ratio	8.4596	2.7438	1.8578	1.1658	1.0814
	Queue	38	167	349	1278	1697
	P_{loss}	0	0	0.0002	0.0049	0.0087
30 slots	Ratio	30.67	7.13	3.9855	1.5826	1.2927
	Queue	7	35	72	380	741
	P_{loss}	0	0	0	0.0001	0.0008
60 slots	Ratio	58.25	12.17	6.3863	2.0612	1.5456
	Queue	3	18	38	199	394
	P_{loss}	0	0	0	0	0.0001
300 slots	Ratio	540.14	66.78	29.78	6.4247	3.8839
	Queue	1	3	7	37	70
	P_{loss}	0	0	0	0	0.0001

Table 3. Results for CEB

Window Size (N)		t_{slot} 0.01 s	t_{slot} 0.05 s	t_{slot} 0.1 s	t_{slot} 0.5 s	t_{slot} 1 s
3 slots	Ratio	2.3612	1.8967	1.7369	1.4664	1.3797
	Queue	175	345	435	705	820
	P_{loss}	0	0	0.0002	0.0018	0.0022
6 slots	Ratio	3.1974	2.4621	2.1945	1.7531	1.6156
	Queue	140	189	231	356	427
	P_{loss}	0	0	0	0.0002	0.0003
30 slots	Ratio	6.6640	4.7717	4.0782	2.9392	2.6033
	Queue	41	57	68	108	131
	P_{loss}	0	0	0	0	0
60 slots	Ratio	9.9601	6.8413	5.7430	3.9993	3.5108
	Queue	26	36	43	69	82
	P_{loss}	0	0	0	0	0
300 slots	Ratio	39.21	21.04	16.50	10.61	9.2576
	Queue	9	12	14	21	25
	P_{loss}	0	0	0	0	0

Table 4. Results for NEB

6. DYNAMIC VS. STATIC TIME SCALE

Section 5.1 shows how drastically the performances of the measurement-based capacity allocation algorithms change depending on the time scale choice.

Mainly, we observed that increasing the measurement slot, t_{slot}, results in a decrease in the capacity allocation and consequently an increase in P_{loss} in all of the algorithms. This

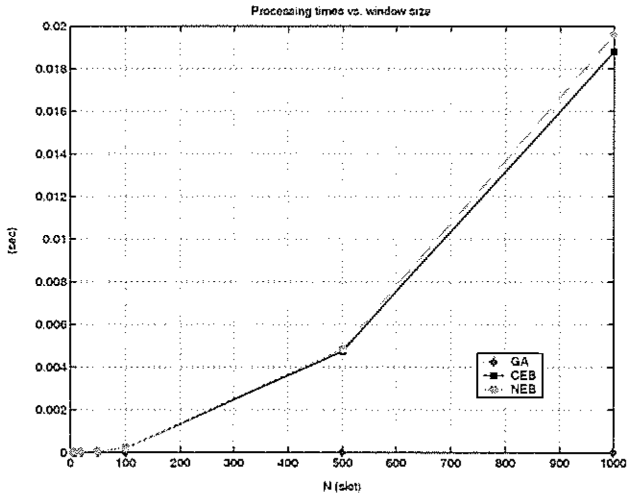

Figure 3. Processing times of algorithms vs. window size.

can be explained intuitively from the structure of the formulas used in the capacity allocation algorithms. To illustrate, consider the Gaussian Approximation Algorithm's formula (10), where m and σ are the mean and standard deviation of the most recent N measurements, $[X_1, X_2, ..., X_N]$. Each X_i represents incoming traffic load in consecutive t_{slot} durations. By the law of large numbers, as t_{slot} increases, say $t_{slot} \rightarrow A$, where A is a time parameter, which is *large* (dependent on the traffic characteristic), then X_i approaches $m * A$ for all i. This causes σ in (10) to go to zero, and the capacity value, *eb* to approach to the mean rate, m. A similar reasoning can be given for other methods, in which not only standard deviation, but also autocorrelations of measurements, $[X_1, X_2, ..., X_N]$ are used.

On the other hand, we also observe that taking t_{slot} arbitrarily *small* leads to over-allocation of resources. As t_{slot} decreases, the measurement history ($t_{slot} * N$) decreases too. This decreases the confidence and increases the randomness in the formula parameter estimations, yielding over-allocations.

We could obtain empirical t_{slot} values from the performance and cost metrics plots, so that the QoS is satisfied with minimum resource allocation. However this particular t_{slot} value would be useful only for the traffic that we used in our simulations. Consider using a traffic whose mean is $m * K$ (that is K times bigger). This time, on average K times more traffic load will fall on average into the slots. Relatively, this is the same experiment as using K times bigger t_{slot} measurement slots, with mean traffic rate m. In other words, the measurement time scale is *relative* to the traffic characteristics. A static t_{slot} may correspond to cases where we described previously as *small* or *large*, depending on the incoming traffic.

As a result, we believe that the measurement time scale t_{slot} should also change dynamically based on measurements $[X_1, X_2, ..., X_N]$, in order to keep the algorithms always working close to their best.

In [25], the Maximum Time-Scale (MaxTS = t^*) is used as the time scale of interest for queueing systems feed by a fractal Brownian motion (fBm) process:

$$t^* = \frac{k\sigma H}{(C-m)}^{\frac{1}{1-H}} \qquad (11)$$

where $k = \sqrt{-2 * \ln(P_{loss})}$, m is the mean traffic rate, σ is the standard deviation of the traffic rate and C is the capacity of the server.

The value of t^* is derived from (12), where $\hat{A}_H(t)$ is the probabilistic envelope process of the fBm cumulative arrival process $A_H(t)$ ($A_H(0) = 0$), such that $P(A_H(t) > \hat{A}_H(t)) \approx P_{loss}$:

$$\frac{d\hat{A}_H(t^*)}{dt} = C \qquad (12)$$

On the basis of the law of large numbers, as $t \rightarrow \infty$, $\frac{d\hat{A}_H(t)}{dt}$ converges to the mean arrival rate. $\hat{A}_H(t)$ increases with a decreasing rate after t^*. This means that the probability that the average arrival rate exceeds the link capacity decreases for $t > t^*$.

In the remaining of the paper, we test using a *dynamic time scale* by estimating t^* using the recent N measurements and taking $t_{slot} = t^*$ as the measurement slot duration for the next N measurements. In other words, besides effective capacity, t_{slot} is also recalculated after every N measurements.

Instead of the $(C - m)$ term in (11), we used $L * m$, where L is taken as a constant $L = (AllocationRatio) - 1$. The reason is that we allocate capacity dynamically and do not have a constant C [1].Table 5 shows the improvement of using a dynamic $t_{slot} = t^*$ against static t_{slot} choices (GA is used as the capacity allocation algorithm, and the P_{loss} target is set to 10^{-3} as in the previous simulations). As the mean rate increases, the performance of the static t_{slot} cases changes (the ratio decreases and P_{loss} increases), whereas the performance of the dynamic $t_{slot} = t^*$ case remains the same. This shows that on-line measurement-based algorithms with constant measurement intervals are heavily dependent on the incoming traffic's mean rate, whereas the ones with dynamic measurement intervals are more robust [2].

Table 5. Performance Metrics vs. Time Scale

Mean (Kbytes/s)		t_{slot} 0.08s	t_{slot} 0.4s	t_{slot} 2s	t_{slot} t^*
4	Ratio	4.751	2.657	1.745	1.915
	P_{loss}	0	0	0.000003	0.000391
20	Ratio	2.952	1.739	1.353	1.914
	P_{loss}	0	0.000009	0.000382	0.000696
100	Ratio	1.742	1.334	1.183	1.913
	P_{loss}	0.000010	0.000332	0.002649	0.000910

To illustrate the benefits visually, we generated a traffic trace of 1000 s, where the mean rate of traffic between 200

[1] We tried using the average capacity allocation instead of C, but this caused a multiplicative effect, such that, when capacity allocation increases, $C - m$ term in (11) decreases. But decreasing t_{slot} results in increased capacity allocation measurement in the next window, and this loop ends up having $t^* \approx 0$.

[2] The particular performance figures for dynamic t_{slot} case in Table 5 are dependent on the value of L (we used $L = 1.5$). However, note that the choice of L does not affect the robustness of the algorithm.

and 800 s [illegible] mean rate at the remaining intervals. Figure 4 shows the dynamic capacity allocations. Note that the allocation ratios in the static t_{slot} cases change in the region of traffic with high mean rate. But the allocation ratio remains roughly the same in the dynamic time scale case. This is achieved by adjusting t_{slot} as shown in Figure 6.

The number of packets dropped increases when the mean is increased gradually in Figure 5 (for $t_{slot} = 0.01$ s, no loss occurred, due to over-allocation). The t^* case performs again in between the static t_{slot} cases. But note that when $t_{slot} = t^*$, the algorithm can self-adjust and perform similarly against traffic mean changes, whereas the performance of an algorithm with static t_{slot} is dependent on the traffic. To illustrate, a method with static $t_{slot} = 0.01$ s case will over-allocate significantly when the mean rate decreases much below of 4 Kbytes/s, and a method with static $t_{slot} = 2$ s case will suffer significant degradation of the QoS target when the mean rate increases much above 100 Kbytes/s.

7. SUMMARY AND CONCLUSIONS

In this paper, we presented and compared measurement-based on-line capacity allocation algorithms and proposed a way to improve their robustness.

We distinguished such algorithms from MBAC and traffic predictors due to their smaller time scales and QoS-oriented use. We observed that their performance is directly dependent on the involved measurement time scales. Mainly, we saw that when the time slot length is increased while the window size is kept constant, due to increasing aggregation of packets in the slot interval, the variations between the measurements in the measurement window decrease and the allocated capacity approaches the mean traffic rate. This causes the loss probability to increase.

Since the measurement time scale is directly related to the measured traffic, the result of a measurement-based algorithm using constant time scale is open to the performance degradations due to the changes in traffic trends. However, our goal was to obtain an algorithm which does *not* require any *a priori* traffic knowledge, and which is based fully on the measurements. Therefore we incorporated the Maximum Time-Scale (MaxTS) parameter and tested successfully adapting the measurement time scales based on measurements themselves.

To sum up, in this paper, we

- identified on-line measurement-based capacity allocation algorithms,
- compared their performances analytically,
- simulated promising ones,
- observed significant affects of the choice of measurement time scale,
- proposed to vary measurement time scale adaptively,
- through an example, showed the performance robustness of measurement-based algorithms, in which measurement time scale is adaptive (*measurement-based*).

The outcomes of this study can be used for choosing algorithms to be implemented in real switches, taking into account trade-offs of complexity, accuracy and robustness.

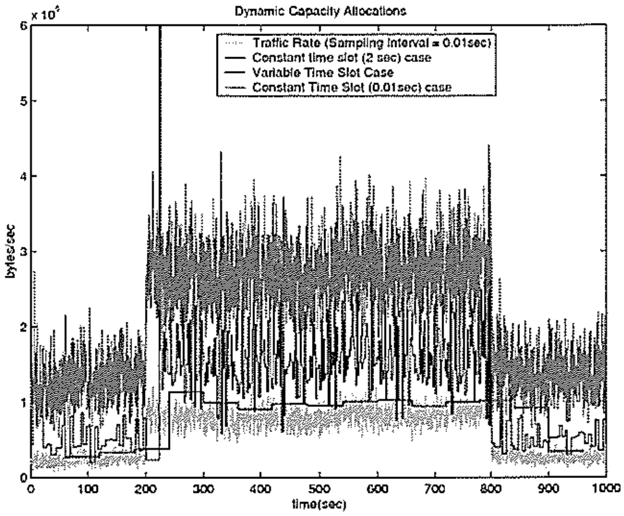

Figure 4. Capacity allocations vs. traffic mean rate.

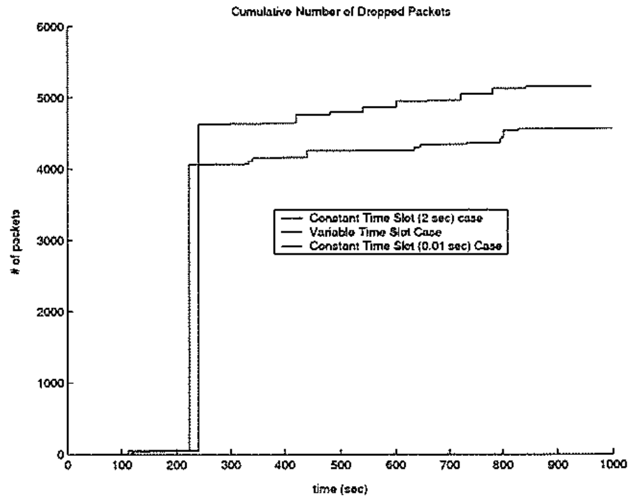

Figure 5. Cumulative Number of Dropped Packets vs. time.

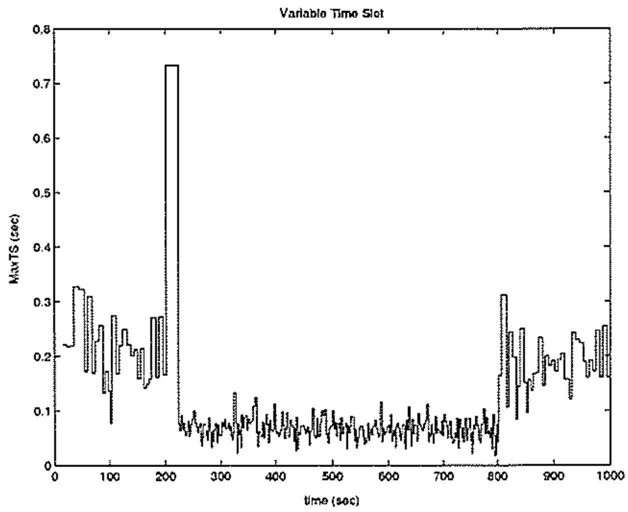

Figure 6. The Plot of Dynamic Time Scale Parameter t^* vs. time.

8. ACKNOWLEDGMENTS

This research was partly supported by the Center for Advanced Computing and Communication - North Carolina State University, and by Alcatel (Plano, TX). The authors thank Aziz Mohammed and Chao Kan of Alcatel, Plano, for their comments and assistance.

REFERENCES

[1] Q. Hao, S. Tartarelli, and M. Devetsikiotis, "Self-sizing and optimization of high-speed multiservice networks," in *Proc. of IEEE Globecom '00*, vol. 3, San Francisco, Nov. 2000, pp. 1818–1823.

[2] J. Yang and M. Devetsikiotis, "On-line estimation, network design and performance analysis with effective bandwidths," in *Proc. of 17th ITC*, Salvador Da Bahia, Brazil, Dec. 2001.

[3] D. Morato, J. Aracil, L. A. Diez, M. Izal, and E. Magana, "On linear prediction of internet traffic for packet and burst switching networks," in *Proc. of ICCCN '01*, Scottsdale, Arizona, Oct. 2001, pp. 138–143.

[4] S. Ostring, H. Sirisena, and I. Hudson, "Rate control of elastic connections competing with long-range dependent network traffic," *IEEE Transactions on Communications*, vol. 49, no. 6, pp. 1092–1101, June 2001.

[5] C. Huang, I. Lambadaris, M. Devetsikiotis, P.W.Gylnn, and A. Kaye, "DTMW; a new congestion control scheme for long-range dependent traffic," in *Proc. of 15th ITC*, Washington, D.C., June 1997.

[6] M. Grossglauser, S. Keshav, and D. N. C. Tse, "RCBR: A simple and efficient service for multiple time-scale traffic," *IEEE/ ACM Transactions on Networking*, vol. 5, no. 6, pp. 741–755, 1997.

[7] L. Breslau, S. Jamin, and S. Shenker, "Comments on the performance of measurement-based admission control algorithms," in *Proc. of IEEE Infocom '00*, vol. 3, Tel-Aviv, Israel, Mar. 2000, pp. 1233–1242.

[8] P. Droz, "Wavelet-based resource allocation in atm networks," in *Proc. of IEEE Globecom '97*, vol. 2, Phoenix, Arizona, November 1997, pp. 833–837.

[9] J. Qiu and E. W. Knightly, "Measurement-based admission control with aggregate traffic envelopes," *IEEE/ACM Transactions on Networking*, vol. 9, no. 2, pp. 199–210, 2001.

[10] M. Grossglauser and D. N. C. Tse, "A framework for robust measurement-based admission control," *IEEE/ACM Transactions on Networking*, vol. 7, no. 3, pp. 293–309, 1999.

[11] F. Kelly, *Stochastic Networks: Theory and Applications*. Oxford University Press, 1996, pp. 141–168.

[12] G. Kesidis, J. Walrand, and C. S. Chang, "Effective bandwidths for multiclass markov fluids and other atm sources," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 424–428, Aug. 1993.

[13] S. Tartarelli, M. Falkner, M. Devetsikiotis, I. Lambadaris, and S. Giordano, "Empirical effective bandwidths," in *Proc. of IEEE Globecom '00*, San Francisco, Nov. 2000.

[14] C. Courcoubetis and R. Weber, "Buffer overflow asymptotics for a buffer handling many traffic sources," *Journal of Applied Probability*, vol. 33, pp. 886–903, 1996.

[15] C. Courcoubetis, V. A. Siris, and G. Stamoulis, "Application of the many sources asymptotic and effective bandwidths to traffic engineering," *Telecommunication Systems*, vol. 12, no. 2-3, pp. 167–191, 1999.

[16] R. Gibbens and P. Hunt, "Effective bandwidths for the multi-type uas channel," *Queueing Systems*, vol. 9, pp. 17–28, 1991.

[17] S. Li, S. Park, and D. Arifler, "SMAQ: A measurement-based tool for traffic modeling and queueing analysis, part I - design methodologies and software architecture," *IEEE Communications*, vol. 36, pp. 56–65, Aug. 1998.

[18] I. Norros, "On the use of fractional brownian motion in the theory of connectionless networks," *IEEE Journal of Selected Areas in Communications*, vol. 13, no. 6, pp. 953–962, 1995.

[19] S. Molnar and T. D. Dang, "Pitfalls in long range dependence testing and estimation," in *Proc. of IEEE Globecom '00*, San Francisco, Nov. 2000.

[20] D. Veitch and P. Abry, "A wavelet based joint estimator of the parameters of long-range dependence," *IEEE Transactions on Information Theory*, vol. 45, no. 3, pp. 878–897, 1999.

[21] W. Shen, "On-line measurement of effective bandwidth and a hierarchical self-sizing framework," Master's thesis, Institute for Electrical and Computer Engineering, Carleton University, Apr. 2002.

[22] R. Guerin, H. Ahmadi, and M. Naghshineh, "Equivalent capacity and its application to bandwidth allocation in high-speed networks," *IEEE Journal of Selected Areas in Communications*, vol. 7, no. 7, pp. 968–981, 1991.

[23] B. K. Ryu and S. B. Lowen, "Point process approaches to the modeling and analysis of self-similar traffic, I. model construction," in *Proc. of IEEE Infocom '96*, vol. 3, San Francisco, Mar. 1996, pp. 1468–1475.

[24] M. Roughan, D. Veitch, and P. Abry, "Real-time estimation of the parameters of long-range dependence," *IEEE/ACM Transactions on Networking*, vol. 8, no. 4, pp. 467–478, 2000.

[25] N. Fonseca, G. Mayor, and C. Neto, "On the equivalent bandwidth of self-similar sources," *ACM TOMACS*, vol. 10, no. 2, pp. 104–124, Apr. 2000.

Fatih Haciömeroğlu (fatih61@ieee.org) received his B.S. degree in electrical and electronics engineering from the Middle East Technical University, Turkey, and his M.Sc. degree in electrical and computer engineering from the North Carolina State University in 2000 and 2003, respectively. His acceptance to the Ph.D. program in the Institut National Des Télécommunications, France is pending. His research interests include dynamic measurement and resource allocation in high speed networks.

Michael Devetsikiotis (S 1985, M 1994, SM 2003) was born in Thessaloniki, Greece, in 1964. He received the Diploma degree in Electrical Engineering from the Aristotle University of Thessaloniki, Thessaloniki, Greece, in 1988, and the M.Sc. and Ph.D. degrees in Electrical Engineering from North Carolina State University, Raleigh, in 1990 and 1993, respectively. In October 1993 he joined the Broadband Networks Laboratory at Carleton University, Ottawa, Canada, as a Research Associate. He became an Adjunct Professor the Systems and Computer Engineering Department, Carleton University, in 1995, an Assistant Professor in 1996 and an Associate Professor in 1998. Since November 2000 he has been with the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, as an Associate Professor. His research work has been in the areas of telecommunication systems modeling, performance evaluation, and efficient simulation; traffic characterization and management; and optimization techniques applied to the analysis and design of communication systems. His present focus is on the performance and nonlinear behavior of communication networks as they become larger in size, and more complex in topology and traffic. He currently serves as the chairman of the IEEE Communication Society committee on communication systems integration and modeling (CSIM), as an Associate Editor for the ACM Transactions on Modeling and Computer Simulation (TOMACS), and as an Associate Editor for the IEEE Communications Letters.