Procedures and tools for analysis of network traffic measurements

Costas Courcoubetis [a,b] and Vasilios A. Siris [a,⋆]

[a] *Institute of Computer Science (ICS)*
Foundation for Research and Technology - Hellas (FORTH)
P.O. Box 1385, GR 711 10 Heraklion, Crete, Greece

[b] *Department of Informatics, Athens University of Economics and Business*

Abstract

We present procedures and tools for the analysis of network traffic measurements. The tools consist of stand-alone modules that implement advanced statistical analysis procedures and a flexible web-based interface through which a user can create, modify, save, and execute experiments using the statistical analysis modules. The tools do not assume a specific source traffic model, but rather process actual measurements of network traffic. Indeed, the theory that the tools are based on can identify the time-scales that affect a link's performance, hence suggest the appropriate time granularity of traffic measurements. We present and discuss case studies that demonstrate the application of the tools for answering questions related to network management and dimensioning, such as the maximum link utilization when some quality of service is guaranteed, how this utilization is affected by the link buffer and traffic shaping, the acceptance region and the effects of the scheduling discipline, and the token (or leaky) bucket parameters of a traffic stream. The case studies involve actual traffic measurements obtained by a high performance measurement platform that we have deployed at the University of Crete network.

Key words: Traffic analysis, traffic burstiness, resource dimensioning, token bucket, web-based tool

⋆ Corresponding author. E-mail: vsiris@ics.forth.gr, tel.: +30 81 391726, fax: +30 81 391601.

Submitted: 29 September, 2000. Revised: 11 May, 2001.

The stand-alone software modules can be downloaded from `http://www.ics.forth.gr/netgroup/msa`; the Web interface is available at `http://trace.ucnet.uoc.gr`

1 Introduction

Recent advances in protocol development and technology are creating the necessary building blocks for transforming the Internet from a pure best-effort services network into a network capable of supporting quality of service (QoS) guarantees. The QoS guarantees along with bounds to the traffic that a user is allowed to send into the network form the service level agreement (SLA) between the user and the network provider. The aforementioned capabilities increase the flexibility of networks in handling traffic with different performance requirements, but they also increase the complexity of network management and dimensioning.

Important questions related to the management and dimensioning of next generation networks that support QoS guarantees include the following: What combinations of users can a network provider accept while ensuring some target level of performance? How many more users can the provider accept if he increases the capacity of his network, introduces priority scheduling, or shapes incoming traffic? How can a user select the parameters of his traffic contract or service level agreement (SLA) with his provider?

An objective in managing next generation networks is the efficient use of network resources shared through statistical multiplexing, while guaranteeing some QoS. Traditional approaches to network management such as measuring the average load in intervals of the order of minutes are not adequate, since they fail to capture traffic burstiness, which is important when some QoS is guaranteed. Approaches based on queueing theory are also inadequate, since they require elaborate traffic models and cannot be effectively applied in the context of large multi-service networks. Furthermore, evidence of self-similar or fractal behavior of network traffic [10,14], as well as the fast growing and changing nature of network applications, has rendered traditional traffic source models inadequate.

In addition to providing the necessary building blocks for supporting QoS guarantees, advances of technology have enabled the collection of detailed traffic statistics at very high speeds: 155 Mbps, 622 Mbps, and higher [4]. We have deployed such a platform at the University of Crete network (UCnet) and used it to obtain the network traffic traces used in our case studies. At very high speeds the amount of data that is collected is huge. Hence, inevitable questions posed by both researchers and engineers is whether all this data is required and, if not, which measurements are most important and how can they be used for network management [1] .

[1] See presentations at the Internet Statistics and Metrics Analysis (ISMA) workshop on passive measurement data and analysis at `http://www.caida.org/outreach/isma/9901/`

Some recent work on measurement and analysis of IP traffic is reported in [3]. The measurement infrastructure, which involves both active and passive measurements, continuously feeds measurements to a data repository. The different forms of the data in the repository, along with the ability to correlate different sets of data, enable the characterization of network usage and behavior. The authors of [11] present a tool, called SMAQ, that integrates traffic modelling and queueing analysis. The tool has components for identifying traffic statistics that are important for queueing analysis, for constructing a stochastic model of the traffic, and for computing the queue length and loss in a finite buffer system.

In this paper we present procedures and tools for advanced traffic analysis of real network traffic measurements. The tools consist of the following:

- Software modules implementing advanced statistical analysis procedures.
- A web-based user interface.

The software modules are two: the `msa` and `lb` modules. The first module is a continuation of our work in [6]. Here we present additional procedures implemented by the tool and further discuss efficient numerical methods for implementing these procedures. The approach does not involve traffic models, such as the work of [11], but relies on actual measurements of real network traffic from which the effective bandwidth of a stream can be estimated. The second module computes the token bucket parameters of a stream based on the equivalence of the token bucket to a queue with finite buffer. The above two software modules can be used to investigate important questions related to network management and dimensioning, and can clarify and demonstrate the effects that traffic statistics, such as bursts over various time-scales, and control mechanisms, such as priority scheduling and traffic smoothing, have on the statistical multiplexing that occurs in a network link. In addition, our approach can suggest the appropriate time granularity that traffic measurements must have, in order to capture the statistics that are important for evaluating the performance of a link.

In addition to stand-alone software modules, we have developed a web-based user interface that provides a flexible and interactive environment through which a user with a Java enabled web browser can create, modify, save, and execute experiments; these experiments are performed by the modules presented above. The results of the experiments are displayed graphically. Moreover, it is possible to present results of two or more experiments in the same graph, thus allowing the straightforward visualization of the effects of network control mechanisms.

The rest of this paper is structured as follows. In Section 2 we discuss typical questions related to the management and dimensioning of networks that

support QoS guarantees. In Section 3 we describe the functionality and procedures implemented by the two statistical analysis modules, and in Section 4 we present case studies demonstrating how the tools can be used to answer questions related to network management and dimensioning. In Section 5 we describe the functionality and architecture of the web-based tool through which users can access the traffic analysis modules, and in Section 6 we present some concluding remarks and identify related research activities we are currently pursuing.

2 Questions concerning network management

Next we discuss some typical questions concerning the management and dimensioning of networks that carry bursty traffic and guarantee a target QoS. Our discussion considers a single link that is statistically shared by a number of bursty traffic streams of various types (e.g., web, video, and voice traffic). The QoS measure that we consider is the probability of traffic being delayed more than some maximum value [2]. Finally, the link has some amount of resources (capacity and buffer), and services packets according to some scheduling discipline (e.g., First In First Out - FIFO, priority scheduling).

Question 1: Resource usage

What is the amount of resources used by each traffic stream? For an aggregate traffic flow, what is the contribution of each individual flow (e.g., the traffic from a subnetwork or protocol) to the amount of resources required for the aggregate flow?

Traditionally, the above questions are answered by measuring the average load in intervals having duration of the order of minutes. Such an approach is appropriate for best-effort networks, but is inaccurate for networks with QoS guarantees. For such networks, a stream's resource usage depends not only on its statistics, but also on the multiplexing that occurs at the link.

As an example, consider a link that multiplexes three types of traffic streams while ensuring a target QoS, Figure 1. All three stream types have the same average rate. However, type (a) traffic is smooth (constant rate) whereas the traffic of the other two types is bursty with different duration of "on" and "off" periods. What amount of resources does each stream type require? Both theory [9,7] and experimentation [6] indicate that the answer to this question depends on the link resources, the QoS guarantees, and the characteristics of

[2] The proportion of time a buffer of size B is full (probability of overflow) is equivalent to the probability of traffic being delayed more than $D = B/C$ in an infinite buffer served at rate C.

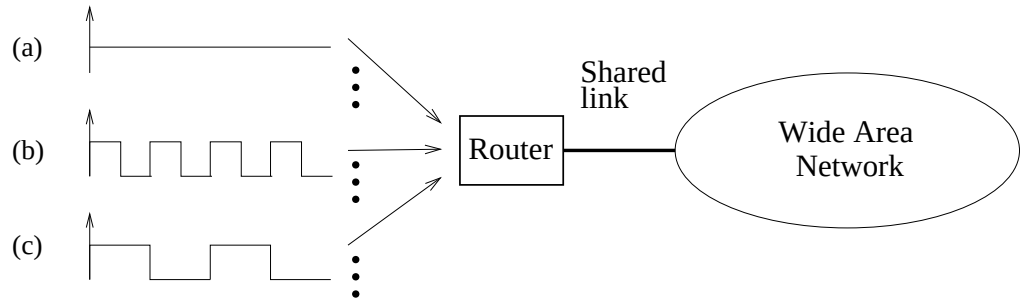


Fig. 1. A stream's resource usage depends, in addition to its statistics, on the link resources, the QoS guarantees, and the characteristics of the other multiplexed traffic.

the multiplexed traffic. In particular, when the link's capacity is small, it might be the case that resource usage for stream type (a) is smaller than that for type (b), which in turn is smaller than that for type (c). For a larger capacity, it might be the case that resource usage for stream types (a) and (b) are the same, and smaller than that for type (c). Finally, for even larger capacities or when the QoS guarantees are loose, it might be the case that all three stream types have the same resource usage.

Question 2: Traffic measurements

What is the size of the time interval for traffic measurements (number of bytes or cells) so that all the statistics that affect a link's performance are captured?

At one extreme, one can capture traces of individual packets. In addition to producing a very large amount of data, particularly for high speed links, such fine statistics can contain more information than what is necessary for evaluating a link's performance. An alternative is to measure the load in consecutive intervals of the same duration. A longer duration of the measurement interval results in less amount of data, but also in less statistical information, since fluctuations are averaged within each interval. The question posed above refers to the maximum duration of the measurement interval so that all the statistics that affect a link's performance are captured.

Question 3: Acceptance region and link utilization

Given the link resources (capacity and buffer), what combinations of traffic types can be multiplexed while ensuring a target QoS? These combinations of traffic types form the *acceptance region*.

A related question is the following: Given the link resources and the traffic mix (i.e., traffic types and percentage of each type), what is the maximum link utilization that can be achieved (equivalently, the maximum number of streams that can be multiplexed), while ensuring a target QoS? How much can the link's utilization increase when its resources increase by some amount?

Question 4: Quality of Service

Given the link resources, the traffic mix, and the link utilization (number of multiplexed streams), what is the level of QoS offered? How much does the QoS improve when the link's resources increase by some amount?

Question 5: Resource dimensioning

Given the number and combination of traffic streams, what is the minimum amount of resources (capacity and buffer) required to guarantee a target QoS? This question applies both to the dimensioning of link resources within a domain (intra-domain), and to the dimensioning of service level agreements (SLAs) between neighboring domains (inter-domain).

Question 6: Traffic shaping

What effect does traffic shaping have on the link's multiplexing capability? How does the effect of traffic shaping depend on the link's buffer size?

Question 7: Scheduling discipline

What effect does the scheduling discipline have on the link's multiplexing capability and on the shape of the acceptance region?

Question 8: Token bucket parameters

What token (or leaky) bucket parameters should be selected for a particular flow? How does traffic shaping affect the value of these parameters? For the wide area link connecting a large organization to the Internet, how do the token bucket parameters change throughout the day?

3 Advanced statistical analysis modules

The analysis of traffic measurements is performed primarily by two modules: The first, called `msa`, implements the analysis based on the effective bandwidths and many sources asymptotic, and the second, called `lb`, computes the token (or leaky) bucket parameters. Both modules take input from a trace file that contains the load (number of bytes or cells) in consecutive intervals (epochs) of fixed duration. A sample trace file is shown below:

```
# epoch_in_msecs = 10
# bits_per_info_unit = 8
737
584
551
657
...
```

The first two lines define the epoch duration and size (in bits) of the unit of load. Hence, the stream corresponding to the above trace produced 737 bytes in the first epoch ($0 - 10$ msec), 584 bytes in the second ($10 - 20$ msec), etc.

Both modules, along with a detailed manual and other related information, are publicly available as stand-alone programs at [16]. In addition to these two modules, we have written a number of supportive programs (e.g., for traffic smoothing and policing) and scripts for executing the same function for a range of link parameters. Using the scripts we can, e.g., run the function for computing the maximum utilization for a range of buffer sizes, thus producing the data for plotting the maximum utilization as a function of the buffer size; such results are presented in Section 4.

In the next two subsections we describe the procedures and functionality of the `msa` and `lb` modules.

3.1 The `msa` *module*

The `msa` module is based on the theory of effective bandwidths and many sources asymptotic [9,7,5], which can be applied to cases where a large number of bursty traffic streams are multiplexed while guaranteeing a target QoS. The effective bandwidth of a stream is a measure of the resources required by the stream. In addition to the statistical characteristics of the stream, the effective bandwidth depends on the link resources and the characteristics of the other traffic it is multiplexed with, which we will refer to as the context of the stream. The context of a stream can be encoded in just two parameters s, t, which are taken to characterize the operating point of a link.

The `msa` module has the following five functions:

(1) Calculates the effective bandwidth (measure of resource usage) for a given link operating point (expressed through parameters s, t) and traffic mix (source types and percentage of each type).
(2) Calculates the buffer overflow probability (BOP) for a given link capacity, buffer, number of sources, and traffic mix.
(3) Calculates the maximum load (equivalently, the maximum number of sources) for a link of given capacity, buffer, and traffic mix, while ensuring a maximum buffer overflow probability.
(4) Calculates the minimum buffer size for a link with a given capacity, number of sources and traffic mix, such that a maximum buffer overflow probability is guaranteed.
(5) Calculates the minimum capacity for a link with a given buffer size, number of sources and traffic mix, such that a maximum buffer overflow probability is guaranteed.

By running the above functions for a range of link parameters we can compute, for example, the buffer overflow probability (BOP) as a function of buffer size (for fixed capacity, traffic mix and load), the maximum load as a function of buffer size (for fixed capacity, overflow probability, and traffic mix), and the acceptance region (for fixed capacity, buffer, and BOP). Examples of such investigations will be presented and discussed in Section 4.

3.1.1 Effective bandwidths and many sources asymptotic: theory and application

Next we present the procedures implemented by the `msa` module, which, as mentioned above, are based on the theory of the effective bandwidths and many sources asymptotic [9,7,5,6]. The effective bandwidth of a stream of type j that produces load $X_j[0,t]$ in a time interval t is defined as [9]

$$\alpha_j(s,t) = \frac{1}{st} \log E\left[e^{sX_j[0,t]}\right], \tag{1}$$

where the parameters s and t characterize a link's operating point and depend on the context of the stream, i.e., the link resources and the characteristics of the multiplexed traffic. Specifically, the space parameter s (measured in, e.g., Mbit^{-1}) indicates the degree of multiplexing and depends, among others, on the size of the peak rate of the multiplexed streams relative to the link capacity: For links with capacity much larger than the peak rate of the multiplexed streams, s tends to zero and the effective bandwidth approaches the mean rate, while for links with capacity not much larger than the peak of the streams, s is large and the effective bandwidth approaches the peak rate $X[0,t]/t$, measured over an interval of duration t. On the other hand, the time parameter t (measured in, e.g., seconds) corresponds to the most probable duration of the buffer busy period prior to overflow.

Investigations [6] have shown that the parameters s, t are to a large extend insensitive to small variations of the traffic mix. This suggests that particular pairs of these parameters can characterize periods of the day during which the traffic composition remains relatively constant.

Computation of the buffer overflow probability (BOP)

If N streams are multiplexed in a link with capacity C and buffer B, and ρ_j is the percentage of streams of type j, then the parameters s, t can be computed from the following formula

$$NI = \inf_t \sup_s F(s,t), \tag{2}$$

where

$$F(s,t) = s(B+Ct) - stN\sum_j \rho_j \alpha_j(s,t)\,.$$

The buffer overflow probability (BOP) can then be approximated by

$$BOP \approx e^{-NI}\,. \tag{3}$$

The above has been proved for discrete time in [7] and continuous time in [2].

The many sources asymptotic can be improved using the Bahadur-Rao theorem [12]. With the Bahadur-Rao improvement the buffer overflow probability is approximated by [13]

$$BOP \approx e^{-NI-\frac{1}{2}\log(4\pi NI)}\,,$$

where NI is computed as before using (2).

Let T be the total duration of the trace. For simplicity, assume that T is integer multiples of t. The expectation in (1) can be approximated by the empirical average, hence the effective bandwidth can be approximated by

$$\tilde{\alpha}_j(s,t) = \frac{1}{st}\log\left[\frac{1}{T/t}\sum_{i=1}^{T/t} e^{sX_j[(i-1)t,it]}\right]\,.$$

The solution of (2) involves two optimization procedures: The first consists of finding, for a fixed value t, the maximum $F^*(t) = \max_s F(s,t)$, and the second consists of finding the minimum $NI = \min_t F^*(t)$.

The maximization $F^*(t) = \max_s F(s,t)$ can be numerically solved in an efficient manner by taking into account that the logarithmic moment generating function $st\alpha_j(s,t) = \log E[e^{sX_j[0,t]}]$ is convex in s, whereas $s(B+Ct)$ is linear in s. Due to this, $F_t(s) = F(s,t)$ is a unimodal function of s and the maximizer is unique. Hence, to find $F^*(t) = \max_s F(s,t) = \max_s F_t(s)$ one can start from an initial "uncertainty" interval $[s_a, s_b]$ that contains the maximum (for this to be the case it is sufficient that for some $x \in [s_a, s_b]$ we have $F_t(x) > F_t(s_a)$ and $F_t(x) > F_t(s_b)$), and decrease the uncertainty interval using a *golden section* search as follows:

(1) Given the interval $[s_a, s_b]$, two trial points s_l, s_r are selected such that $s_r - s_a = s_b - s_l = g(s_b - s_a)$, where g is the *golden ratio*, which is equal to $\frac{\sqrt{5}-1}{2} \approx 0.618$.

(2) We evaluate $F_t(s_l)$ and $F_t(s_r)$, and identify three cases:
   (a) if $F_t(s_l) > F_t(s_r)$ the interval becomes $[s_a, s_r]$,
   (b) if $F_t(s_l) < F_t(s_r)$ the interval becomes $[s_l, s_b]$, and
   (c) if $F_t(s_l) = F_t(s_r)$ the interval becomes $[s_l, s_r]$.

(3) Steps 1 and 2 are repeated until the uncertainty interval has length less than some small value ϵ.

The golden section search is the limit (for a large number of steps) of the *Fibonacci search*, which minimizes the maximum number of steps needed to reduce the uncertainty interval to some prescribed length.

Unlike the function $F(s,t)$, there is no general property for $F^*(t)$ that we can take advantage of in order to perform the minimization $\min_t F^*(t)$ in an efficient manner. Indeed, $F^*(t)$ can have more than one local minimum. For this reason, the minimization is solved by linearly searching the values of t in the interval $[0, \kappa\tau]$ with granularity equal to one epoch τ. The value of κ is determined empirically and depends on the buffer size: The extremising value of t is larger for larger buffer sizes.

To select the time granularity of traffic measurements, i.e., the duration of the consecutive intervals in which the traffic volume is measured, in order to capture the traffic statistics that are important for buffer overflow one proceeds as follows. The minimizing value of t in (2) is computed as described above. Obtaining the value $t = \tau$ (i.e., the duration of one epoch) indicates that buffer overflow occurs on time scales less than or equal to τ. This results indicates that the load measurements are too coarse, hence the measurement interval must decrease in order for the minimizing value of t to be a few times τ. Of course, since $F^*(t)$ can have more than one local maxima, such a procedure does not guarantee that the global maximum is found. However, empirical results indicate that for Internet traffic, when the buffer size is larger than 10,000 bytes, i.e., 10 packets assuming a packet size of 1000 bytes, the minimizing value of t is typically larger than 10 milliseconds.

The run-time of the above procedure for solving (2) depends on the size (number of epochs) of the trace file, the number of values of t that are processed, and the number of different stream types. On the other hand, the run-time is independent of the number of multiplexed streams. Some illustrative numbers for the implementation (in C) of the `msa` module running on a FreeBSD 4.0 workstation with a Pentium III processor at 550 MHz are the following: For one stream type, given by a trace file containing 40,000 epochs, and when 50 values of t are searched, the run-time is approximately 4.3 seconds. When there are two and three different stream types, the run-time is 9.2 seconds and 13.1 seconds, respectively.

Computation of the maximum number of multiplexed streams

Next we describe the implementation of the third function of the `msa` module presented in subsection 3.1. The maximum utilization, or equivalently the maximum number of streams, for a given amount of link resources (capacity C and buffer B), traffic mix $\{\rho_1, \rho_2, \ldots\}$, and target overflow probability $e^{-\gamma}$ can be computed by solving the following equation

$$N^* = \inf_t \sup_s G(s,t)\,, \tag{4}$$

where

$$G(s,t) = \frac{s(B+Ct)-\gamma}{st\sum_j \rho_j \alpha_j(s,t)}\,.$$

The above, as well as the relations for the next two functionalities of the `msa` module, can be proved in a similar way as the proof of (2) and (3) in [7]. To use the Bahadur-Rao improvement, the maximum number of multiplexed sources is computed using (4), after replacing γ with

$$\gamma_{\text{B-R}} = \gamma - \frac{1/2\log(4\pi\gamma)}{1+1/(2\gamma)}\,. \tag{5}$$

As was the case for $F(s,t)$ in (2), $G(s,t)$ is also a unimodal function of s with a unique maximizer, hence $G^*(t) = \max_s G(s,t)$ can be solved using a golden section search.

Computation of the minimum buffer size

The fourth function of the `msa` module, i.e., the computation of the minimum buffer size such that a maximum buffer overflow probability is guaranteed, can be computed by solving the following equation:

$$B^* = \sup_t \inf_s K(s,t)\,, \tag{6}$$

where

$$K(s,t) = \frac{stN\sum_j \rho_j\alpha_j + \gamma}{s} - Ct\,.$$

As was the case for $F(s,t)$ in (2), $K(s,t)$ is also a unimodal function of s with a unique minimizer, hence $K^*(t) = \min_s K(s,t)$ can be solved using a golden

section search. To use the Bahadur-Rao improvement, in (6) we replace γ with $\gamma_{\text{B-R}}$ given in (5).

Computation of the minimum capacity

Finally, the fifth function of the `msa` module, i.e., the computation of the minimum capacity such that a maximum buffer overflow probability is guaranteed, can be computed by solving the following equation:

$$C^* = \sup_t \inf_s R(s,t)\,, \tag{7}$$

where

$$R(s,t) = \frac{stN\sum_j \rho_j\alpha_j + \gamma}{st} - \frac{B}{t}\,.$$

As was the case for $F(s,t)$ in (2), $R(s,t)$ is also a unimodal function of s with a unique minimizer, hence $R^*(t) = \min_s R(s,t)$ can be solved using a golden section search. Finally, to use the Bahadur-Rao improvement, in (7) we replace γ with $\gamma_{\text{B-R}}$ given in (5).

3.2 The `lb` *module*

The `lb` module computes the token (or leaky) bucket parameters for a particular trace file when all traffic is to be conforming or when some percentage of the traffic is allowed to be non-conforming.

A token bucket, Figure 2, consists of a token pool of size b (bucket size) that fills at rate r (token rate), measured in tokens per second. The token bucket is used to police a traffic stream in the following way: While there are enough tokens, a number of tokens equal to the size of the packet are removed from the token pool. In this case, the packet is said to be conforming. On the other hand, if the number of tokens in the pool is less than the size of the packet, then the packet is non-conforming.

For a particular trace, there is not a single pair of token bucket parameters, but a set of such pairs that form the indifference curve, Figure 3. The indifference curve of a stream is convex, intersects the rate axis (horizontal axis in Figure 3) at the peak rate p of the stream, and increases abruptly as the token rate approaches the mean rate m of the stream.

The procedure for computing the token bucket parameters is based on the equivalence of the token bucket to a finite buffer of size equal to the bucket

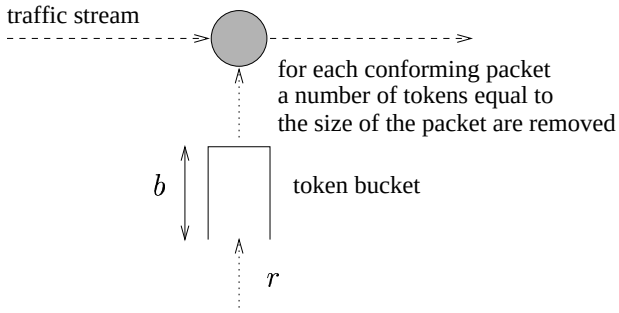


Fig. 2. Token bucket policing. A packet is conforming if when it arrives the number of tokens in the token bucket is more than the packet size, else it is non-conforming.

size, which is served at rate equal to the token rate. Non-conforming packets of the token bucket correspond to lost packets in the equivalent buffer.

If all traffic of the stream is to be conforming, then for a particular value of r, the bucket size is equal to the maximum backlog b of a queue with infinite buffer serviced at rate r, that is fed with the stream's traffic. The maximum backlog can be found using the following procedure: For each epoch of the trace file, the queue length q is increased by the traffic volume produced in that epoch and decreased by $r\tau$, where τ is the duration of one epoch. The maximum backlog will equal the maximum value of q.

The above procedure applies to the case where all traffic is conforming. On the other hand, when some percentage of the traffic is allowed to be non-conforming, the bucket size corresponding to the token rate r is the minimum value of b' such that the percentage of lost traffic in a finite buffer of size b', that is fed with the stream's traffic and is serviced at rate r, is less than the target percentage of non-conforming traffic. Calculating this percentage requires one pass of the trace file for each value of b' that is processed. Because

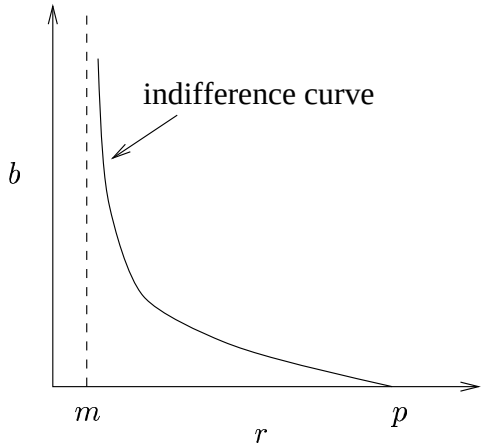


Fig. 3. The indifference curve is convex, intersects the rate axis at the stream's peak rate p, and increases abruptly as the token rate approaches the mean rate m.

the percentage of non-conforming traffic is monotone with the bucket size, the minimum buffer b can be found using a binary search or, more efficiently, using a golden section search similar to the procedure for finding the optimal value of s in the `msa` module (Subsection 3.1.1). The initial search interval for the golden section search can be set to $(0, b_c(r))$, where $b_c(r)$ is the bucket size corresponding to the token rate r when all traffic is conforming.

4 Case studies

In this section we present case studies demonstrating the application of our analysis tools to answer questions such as those discussed in Section 2. In particular, we present experimental results investigating the following issues:

- Effect of link resources (in particular, the buffer size) and traffic shaping on the multiplexing capability of a link.
- Required or minimum capacity such that a target QoS is guaranteed.
- Acceptance region for the case of two traffic types, and effect of the scheduling discipline.
- Token bucket parameters for a traffic stream, and effect of traffic shaping.

The case studies where performed using the Bellcore Ethernet WAN trace [3] [10], MPEG-1 sequences made available [4] by O. Rose [15], and traces of incoming IP traffic over the University of Crete's wide area link.

4.1 Maximum link utilization and effect of traffic shaping

Comparison of the three curves in Figure 4 shows the effect of traffic shaping on the maximum link utilization, when a maximum overflow probability is guaranteed. Shaping [5] is performed by evenly spacing traffic in consecutive time intervals of length d, referred to as shaping delay. Each of the three curves in Figure 4 was obtained by shaping the same traffic trace, but with a different shaping delay.

First, observe that the same increase of the buffer size does not produce the

[3] Available from the Internet Traffic Archive at `http://www.acm.org/sigcomm/ITA/`

[4] Available at ftp://ftp-info3.informatik.uni-wuerzburg.de/pub/MPEG/

[5] The method described is just one way for performing traffic shaping, and is not necessarily the best. Moreover, the value of d represents an upper bound on the delay. The average delay depends on the traffic burstiness, and can be much smaller than the maximum.

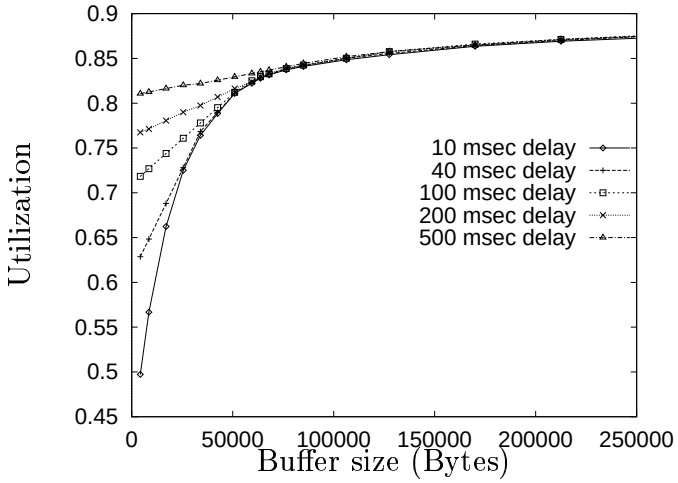


Fig. 4. Utilization as a function of buffer for various shaping delays. Traffic shaping affects the maximum utilization only for small buffer sizes. $C = 34$ Mbps, $BOP \leq 10^{-6}$, Bellcore trace.

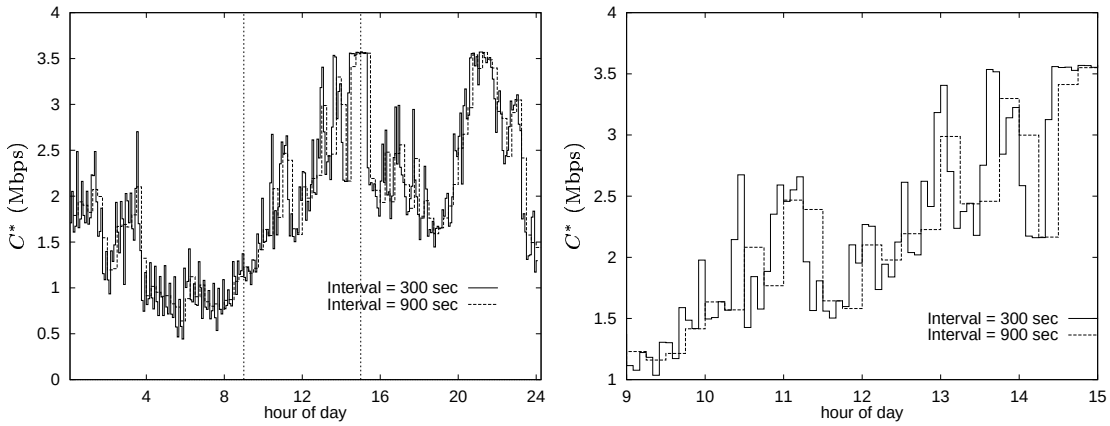
same increase of the maximum utilization. In particular, an increase of the buffer has a larger effect for smaller buffer sizes. The reason behind this is that for small buffer sizes, the fast time-scales dominate buffer overflows, hence increasing the buffer can be used to smooth out these time-scales and as a result increase the maximum utilization. On the other hand, for large buffer sizes the slow time-scales dominate buffer overflows, hence the utilization is affected only with large increases of the buffer size.

Second, Figure 4 shows that traffic shaping affects the maximum link utilization for small buffer sizes. On the other hand, for large buffer sizes there is practically no effect. As with the previous observation, the reason is that the slow time-scales that govern buffer overflows for small buffer sizes can be smoothed out with traffic shaping. Such smoothing does not occur for large buffer sizes, where overflow is dominated by slow time-scales.

4.2 Required capacity

Next we consider measurements of network traffic for a whole day and investigate how the minimum capacity, C^*, that is required to guarantee a target QoS changes throughout the day.

The traffic measurements are taken as follows: In periods of duration T, the traffic produced in consecutive epochs of duration τ (= 10 msec) is measured. For each period, the minimum capacity is estimated with the fifth function of the `msa` module, which is based on (7). This estimation of minimum capacity is repeated for consecutive periods, all of duration T. The results, for two values of T, are shown in Figure 5.



(a) For whole day [00:00-24:00] (b) For time period [9:00-15:00]

Fig. 5. $B = 20000$ Bytes, $BOP \leq 10^{-5}$. The traffic is that of 8 organizations, each producing traffic similar to that of the University of Crete.

Observe in Figure 5(b) that a larger value of T results in a smoother variation of the required capacity estimate, but tracks the required capacity with some delay; the latter is because measurements for duration T are needed before the required capacity can be computed. The appropriate value of T will depend on the smallest interval between traffic control actions, such as redimensioning of link capacity or renegotiation of SLAs [8,17].

Furthermore, Figure 5(a) shows that the minimum capacity exhibits daily variations. Indeed, an area for further work is to investigate procedures for tracking the daily variations, as well as changes due to trend. Such procedures can be based on forecasting techniques from time series analysis.

4.3 Acceptance region

Figure 6 shows the acceptance region for the case of two source types under two scheduling disciplines: First In First Out (FIFO) and priority scheduling [9,1]. With the latter, the high priority class is guaranteed a maximum delay that is lower than that guaranteed by the low priority class. For simplicity, we have assumed that all sources of type 1 are of high priority, and those of type 2 are of low priority. Each priority has a different set of operating point parameters s, t, computed using (4) [9]. The region between the two lines represents the gains from using priority scheduling.

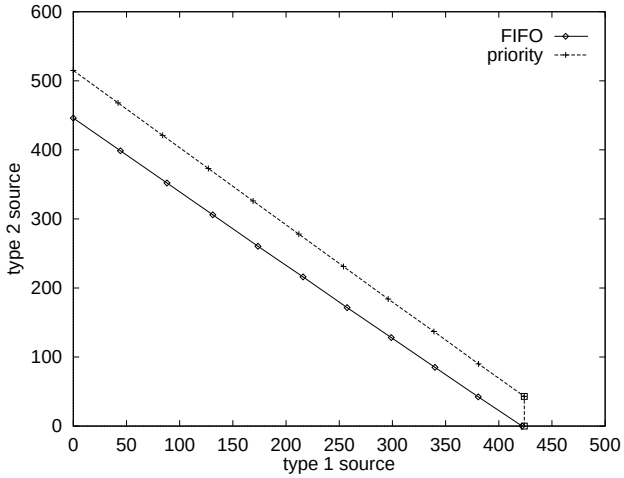


Fig. 6. Acceptance region for two traffic types ("Terminator" and "Star Wars"). In priority scheduling, the high priority class has maximum delay 4 ms with probability 10^{-7}, and the low priority class has maximum delay 16 ms with probability 10^{-5}.

4.4 Indifference curve and effect of traffic shaping

Figure 7 shows the indifference curve for traffic taken from the University of Crete at different periods of the day. As expected, periods of the day with higher traffic have higher values of token bucket parameters.

Figure 8(a) shows the indifference curve when some percentage of the traffic is non-conforming. Observe that there are large gains in allowing some percentage of the traffic to be non-conforming.

Finally, Figure 8(b) shows the indifference curve for various shaping delays. Observe that traffic shaping affects mostly the lower-right portion of the indifference curve, hence the peak rate.

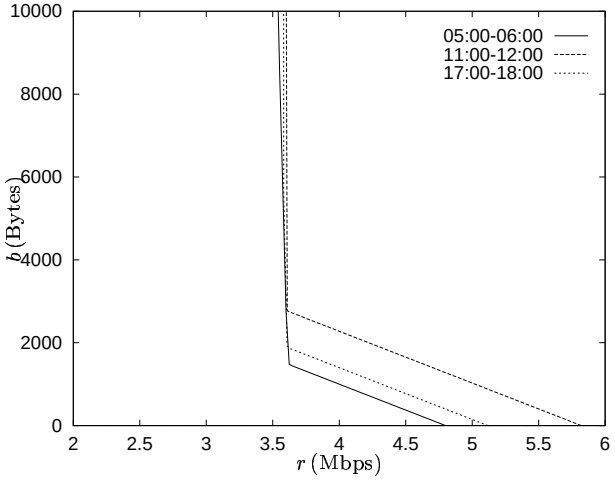
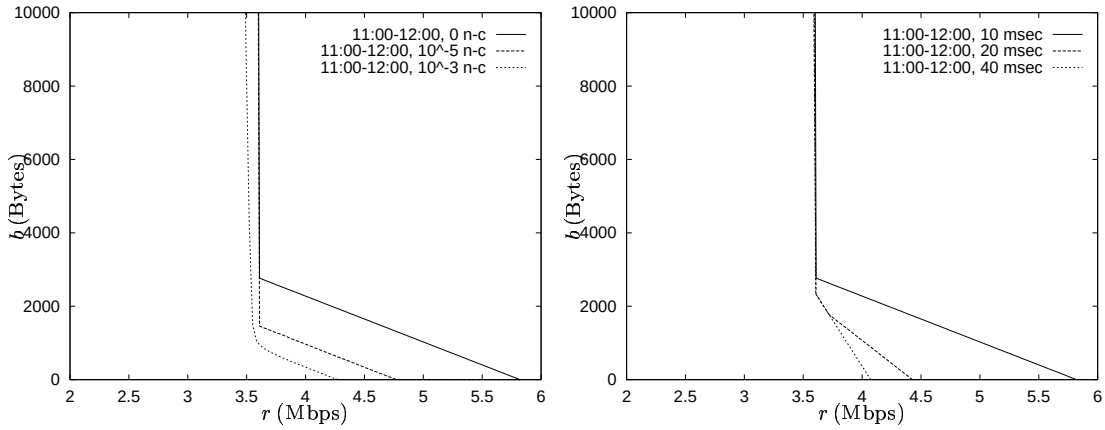


Fig. 7. Indifference curve for various hours of the day.



(a) allowing non-conformance (b) effect of shaping

Fig. 8. Indifference curve.

5 Web-based traffic analysis tool

In this section we discuss the functionality and architecture of the web-based tool through which users can use the statistical analysis modules presented in the previous sections. The interface, Figure 9, provides a flexible environment through which a user, using a Java enabled web browser, can create, modify, save, and execute *experiments*. Experiments can be of the following type:

(1) Buffer Overflow Probability (BOP) of a link for a specific traffic mix.
(2) Maximum link utilization for a specific traffic mix, while satisfying a target BOP.
(3) Combinations of two types of traffic streams that satisfy a target BOP (these combinations form the acceptance region).
(4) Token bucket's leak rate vs. bucket size trade-off for a traffic stream (these pairs form the indifference curve). The indifference curve can be calculated both for the case where all traffic is conforming and for the case where some percentage of the traffic can be non-conforming.

The link model considered is that of a single buffer of size B serviced First In First Out (FIFO) at rate C, Figure 10. The buffer is fed by a number of bursty traffic streams, which are given by actual traffic traces. Before entering the buffer, the traffic streams can go through one or more filters ("Fs" in Figure 10). Possible filters include traffic shaping, peak rate policing, and leaky bucket policing.

The tool currently contains traces of actual MPEG compressed video traffic, traces obtained from the Internet Traffic Archive, as well as measurements from the UoC's traffic measurement platform.

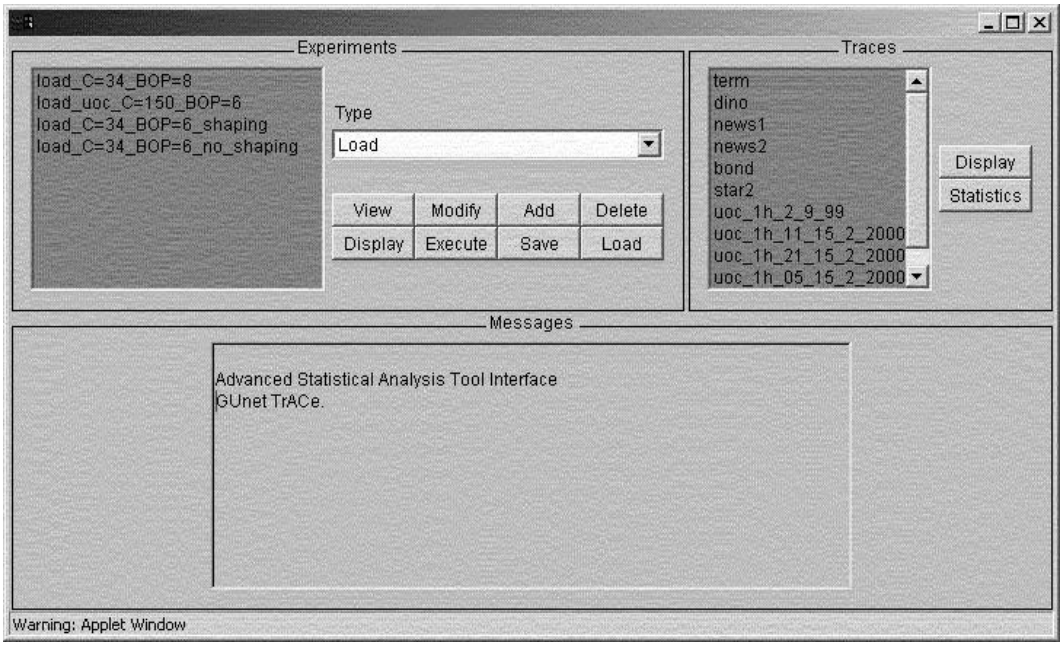


Fig. 9. The web-based tool's interface. The upper left box is the experiments box from which the user can create, save, and execute experiments. The upper right box shows the available traffic traces. The lower box displays status messages.

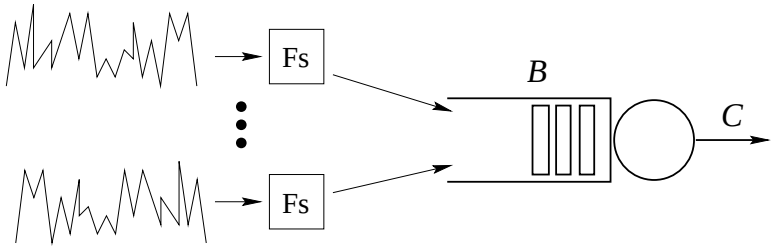


Fig. 10. Link model. A buffer of size B serviced FIFO at rate C is fed by a number of bursty traffic streams, which can go through one or more filters ("Fs").

To facilitate the initial use of the tool, we have created a number of predefined experiments for each experiment type. Hence, the user can start by executing these predefined experiments, and subsequently modify the parameters to create new experiments. An example definition of an experiment is shown in Table 1.

5.1 Graphical display of results

The results of the experiments are displayed graphically. Moreover, the tool can display the results from two experiments of the same type in the same

Table 1
Example definition of an experiment. The experiment produced the top curve in Figure 11: the maximum load was estimated for a particular trace and parameters ("Star Wars" stream, BOP $= 10^{-6}$, etc.), and for 8 different buffer sizes with corresponding delay from 5 msec to 40 msec. The bottom curve was produced for the same traffic stream and parameters, but without shaping.

Experiment type	load (maximum utilization)
Traffic stream	star2
Filters	shaping (80 msec)
C	34 Mbps
D	from 5 msec to 40 msec, step 5 msec
$-\log_{10}(BOP)$	6
Method	Many sources

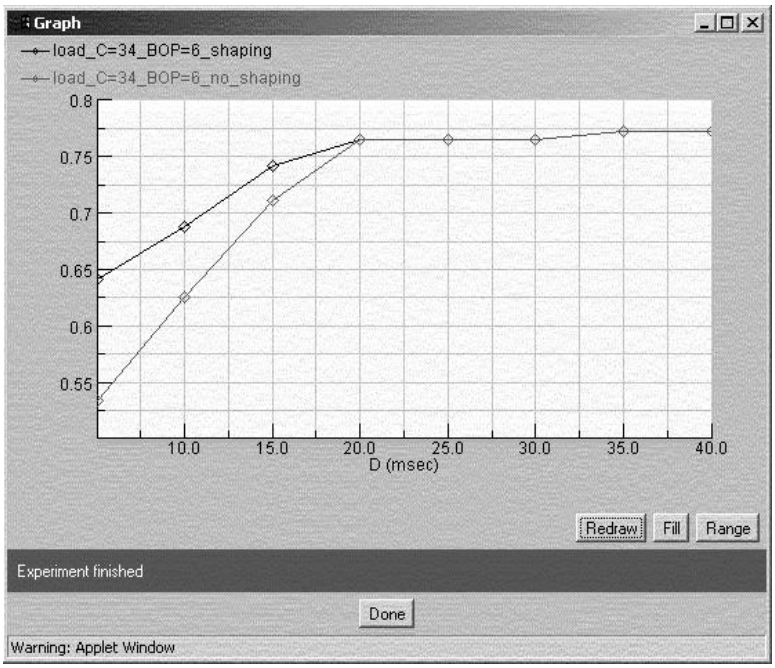


Fig. 11. Maximum link utilization experiment: effect of traffic shaping on the maximum utilization. $C = 34$ Mbps, $BOP = 10^{-6}$, "Star Wars" traffic. The top curve corresponds to a smoothed traffic stream obtained by averaging the original stream in consecutive time intervals of duration 80 msec, whereas the lower curve is for the same traffic without shaping. Similar results were shown in Figure 4.

graph. This capability can be used to view, for example, the effects of traffic shaping on the maximum link utilization, as shown in Figure 11 whose results are similar to those of Figure 4.

The architecture of the web-based tool is shown in Figure 12. The client interface (written in Java) runs on the user's browser, and sends requests to a server (also written in Java), that runs on a high performance workstation, running FreeBSD. The requests can involve saving, loading, or executing experiments. In the latter case, the server launches the execution of shell scripts that in turn run the appropriate software module, namely the `msa` or `lb` module (both written in C). When the execution of an experiment is completed, the server sends the results back to the client interface, which displays them in a graph contained in a new window (i.e., different from the user interface window).

In Figure 12 both the server and the statistical analysis modules are shown to run on the same workstation. Our implementation however supports a rudimentary form of load balancing, which allows the even distribution of experiments on more than one workstations.

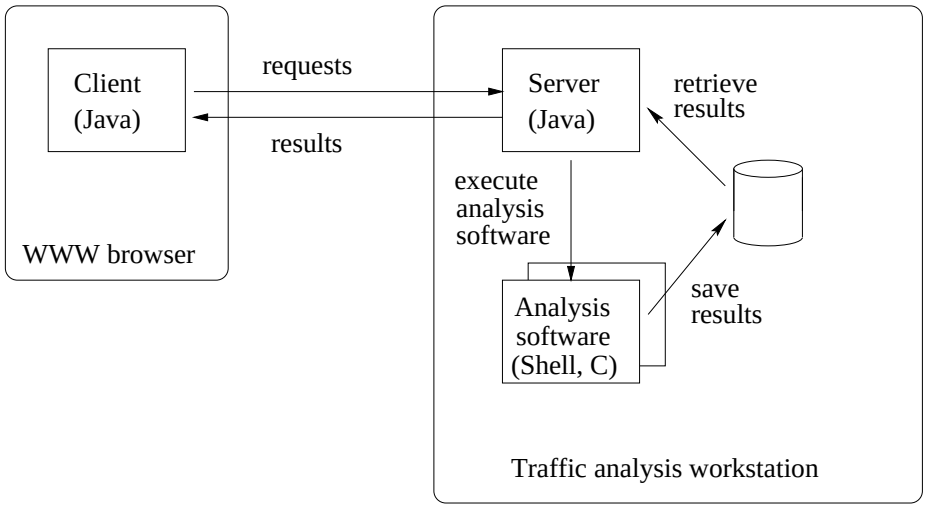


Fig. 12. Architecture of the web-based traffic analysis tool. The client sends requests to the server. If the request is for the execution of an experiment, the corresponding software module is launched. The results are returned to the client.

6 Concluding remarks

We have presented a set of tools for advanced statistical analysis of network traffic measurements. The tools consist of freely available stand-alone statistical analysis modules (written in C), namely the `msa` and `lb` modules, and a flexible user interface (written in Java), that allows users with a Java enabled browser to create, modify, save, and execute experiments. The tools can help

network engineers understand how the various traffic characteristics, such as burstiness of various time-scales, and network control mechanisms, such as traffic shaping and scheduling, can affect the performance of a network, hence assist them in operating and dimensioning networks more efficiently. Moreover, the tools have been used in graduate-level networks courses at the University of Crete for giving students hands-on experience with traffic analysis methods.

We are currently using the tools to analyze traffic from an operational Frame Relay network of a large Internet Service Provider. Other research directions we are investigating include the application of our tools, together with policy-based mechanisms, for controlling network resource usage, and measurement-based techniques for forecasting bandwidth requirements and trigger the resizing of network link capacity (intra-domain resource dimensioning) and of service level agreements (inter-domain resource dimensioning).

The work presented in this paper is part of our activities in traffic measurement and analysis, which are being performed in collaboration with the Communication and Networks Center at the University of Crete. [6] The objectives of this work, in addition to the development of the tools presented in this paper, includes the deployment of a flexible platform for passive measurement of network traffic from an operational network and the development of web-based tools for requesting and displaying on-line traffic statistics, such as the composition (e.g., application, protocol), origin (e.g., national or international link), and destination (e.g., department, subnetwork) of incoming IP traffic.

Acknowledgements

The authors thank GUnet (Greek Universities Network), which funded the traffic measurement platform deployed at the University of Crete, and Antonis Dimakis, who wrote the initial version of the user interface, and the anonymous reviewers, whose comments have helped improve the paper.

References

[1] A. W. Berger and W. Whitt. Extending the effective bandwidth concept to networks with priority classes. *IEEE Commun. Mag.*, pages 78–83, August 1998.

[6] More information on these activities and other web-based tools is available at `http://trace.ucnet.uoc.gr`.

[2] D. D. Botvich and N. G. Duffield. Large deviations, the shape of the loss curve, and economies of scale in large multiplexers. *Queueing Systems*, 20:293–320, 1995.

[3] R. Cacere, N. Duffield, and A. Feldmann *et. al.* Measurement and analysis of IP network usage and behavior. *IEEE Commun. Mag.*, pages 144–151, May 2000.

[4] CoralReef. `http://www.caida.org/tools/measurement/coralreef/`.

[5] C. Courcoubetis, F. P. Kelly, and R. Weber. Measurement-based usage charges in communications networks. Technical Report 1997-19, Statistical Laboratory, University of Cambridge, 1997. To appear in *Operations Research*.

[6] C. Courcoubetis, V. A. Siris, and G. D. Stamoulis. Application of the many sources asymptotic and effective bandwidths to traffic engineering. *Telecommunication Systems*, 12(2-3):167–191, 1999. A shorter version appeared in *Proc. of ACM SIGMETRICS'98/PERFORMANCE'98*.

[7] C. Courcoubetis and R. Weber. Buffer overflow asymptotics for a switch handling many traffic sources. *J. Appl. Prob.*, 33:886–903, 1996.

[8] N. G. Duffield, P. Goyal, A. G. Greenberg, P. P. Mishra, K. K. Ramakrishnan, and J. E. van der Merwe. A flexible model for resource management in virtual private networks. In *Proc. of ACM SIGCOMM'99*, pages 95–108, 1999.

[9] F. P. Kelly. Notes on effective bandwidths. In F. P. Kelly, S. Zachary, and I. Zeidins, editors, *Stochastic Networks: Theory and Applications*, pages 141–168. Oxford University Press, 1996.

[10] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson. On the self-similar nature of ethernet traffic. In *Proc. of ACM SIGCOMM'93*, pages 183–193, Ithaca, NY, USA, September 1993.

[11] S. L. Li, S. Park, and D. Arifler. SMAQ: A measurement-based tool for traffic modeling and queuing analysis Part I: Design methodologies and software architecture. *IEEE Commun. Mag.*, pages 56–77, August 1998.

[12] N. Likhanov and R. R. Mazumdar. Cell loss asymptotics for buffers fed with a large number of independent stationary sources. In *Proc. of IEEE INFOCOM'98*, San Fransisco, CA, USA, April 1998.

[13] M. Montgomery and G. de Veciana. On the relevance of time scales in performance oriented traffic characterizations. In *Proc. of IEEE INFOCOM'96*, pages 513–520, April 1996.

[14] V. Paxson and S. Floyd. Wide-area traffic: The failure of Poisson modeling. *IEEE/ACM Trans. on Networking*, 3(3):226–244, June 1995.

[15] O. Rose. Statistical properties of MPEG video traffic and their impact on traffic modeling in ATM systems. Technical Report 101, University of Wuerzburg, February 1995.

[16] V. A. Siris. Large deviation techniques for traffic engineering. `http://www.ics.forth.gr/netgroup/msa/`.

[17] A. Terzis, L. Wang, J. Ogawa, and L. Zhang. A two-tier resource management model for the Internet. In *Proc. of Global Internet 99*, December 1999.