Design and Implementation of Measurement-Based Resource Allocation Schemes Within The Realtime Traffic Flow Measurement Architecture

Robert D. Callaway and Michael Devetsikiotis
Department of Electrical and Computer Engineering
North Carolina State University
Raleigh, NC, 27695-7911 USA
{rdcallaw, mdevets}@eos.ncsu.edu

Chao Kan
Alcatel Research & Innovation Center
Alcatel USA, Inc.
Plano, TX 45045 USA
chao.kan@alcatel.com

***Abstract*— The concept of effective bandwidth can be utilized to estimate the amount of bandwidth that should be allocated to a source in order to meet a QoS requirement. Several different effective bandwidth estimators have been defined in literature; however it is necessary to ensure that these estimators are practically *implementable and feasible* in *realistic* network environments. This necessity serves as our motivation to implement several estimators in a realistic network in order to evaluate the use of online measurement-based resource allocation schemes.**

In this paper, we describe our implementation of three resource allocation schemes within the Realtime Traffic Flow Measurement architecture. We compare our results of emulation to previous simulation results in order to compare the accuracy and performance of the schemes. Finally, we demonstrate that these schemes are feasible to be implemented in network hardware to be utilized in self-sizing high-speed networks.

I. Introduction and Motivation

The goal of self-sizing multiclass networks is to provide efficient utilization of network resources while ensuring appropriate quality of service (QoS) for each class of traffic. The use of effective bandwidths, which estimate the amount of bandwidth that should be allocated to a class of traffic in order to meet a QoS requirement, is a natural fit into the self-sizing architecture. This allows for maximum utilization of available resources while still guaranteeing that QoS requirements are met.

Since arriving traffic is not known *a priori*, a measurement-based resource allocation scheme must be utilized. This allows the network to follow the transient nature of the traffic in a realtime manner, adapting to its changing characteristics. It is also imperative that this scheme be practically implementable and scalable within a high-speed network with a large number of concurrent classes. These factors serve as our motivation to investigate the viability of using effective bandwidth estimators within a self-sizing framework.

We emulate a realistic network by implementing our resource allocation schemes within the Realtime Traffic Flow Measurement (RTFM) architecture, which is defined in [1] as an IETF standard for traffic accounting and measurement. Since previous work is limited to simulations only, and of a single queue model at that, we study here the viability of these resource allocation schemes in a real network which utilizes IETF-standardized measurement tools. By abstracting away the details of the measurement architecture, we are able to independently evaluate the ability of the allocation schemes to conserve network resources while still meeting the QoS constraints of the traffic.

The paper is organized as follows: In Section II, we study the realization and feasibility of three different effective bandwidth estimators. In Section III, we discuss our modifications to an implementation of the Realtime Traffic Flow Measurement architecture. In Section IV, we provide a description of our tests along with the results, which demonstrate the ability to provide adequate quality of service to network traffic while still conserving available bandwidth. We summarize our findings and discuss future work in Section V.

II. Effective Bandwidth Estimators

Effective bandwidth is generally defined in [2] by:

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

where $X[0,t]$ represents the amount of work that arrives from a source in the interval $[0,t]$.

The s parameter in (1) cannot be directly estimated from measurements. It must be calculated using the Large Deviations Theory and making a large buffer assumption. Therefore, the direct application of (1) in an online measurement resource allocation scheme is not practical. Numerous ways of evaluating (1) have been defined in literature; however, most of the approaches rely on unrealistic assumptions or invalid approximations. Therefore, without loss of generality, we limit our analysis to empirical approaches to estimating the effective bandwidth, which are discussed in [3].

A comparison of several different empirical estimators is described in [4]. Three algorithms investigated in [4] are chosen for further analysis because of their computational complexity, performance, and memory requirements. The Gaussian, Courcoubetis, and Norros allocation algorithms are briefly defined

below, and in greater detail in [5], [6], and [7], respectively. Note that all of the reviewed formulae are derived independent of the effective bandwidth formula (1).

A. Gaussian Approximation

The Gaussian Approximation is defined in [5] as:

$$C_{EB} = \mu + \sigma\sqrt{-2\ln\epsilon - \ln 2\pi} \tag{2}$$

where μ is the mean arrival rate of the traffic, σ is the standard deviation of the arriving traffic, and ϵ is the QoS parameter (packet loss percentage).

The Gaussian Approximation assumes a bufferless link, and is therefore an appropriate *upper bound*. It is computationally simple and easy to implement.

B. Courcoubetis Approximation

The Courcoubetis Approximation is defined in [6] as:

$$C_{EB} = \mu + \frac{IDs}{2B} \tag{3}$$

where μ is the mean arrival rate of the traffic, ID is the index of dispersion, s is the space parameter, and B is the buffer size of the queue. The s parameter is calculated from an asymptotically exponential decrease assumption:

$$P(B < Q) = e^{-sB} \tag{4}$$

The index of dispersion is defined in [6] as:

$$ID = \lim_{n\to\infty} \frac{1}{n}\mathbf{E}\left[\left(\sum_{i=1}^{n} X_i\right)^2\right] \tag{5}$$

A limitation of the Courcoubetis Approximation is that it does not address long range dependant traffic.

C. Norros Approximation

The Norros Approximation is defined in [7] as:

$$C_{EB} = \mu + \left[B^{H-1}\kappa(H)\sqrt{-2a\mu\ln\epsilon}\right]^{\frac{1}{H}} \tag{6}$$

where $\kappa(H) = H^H(1-H)^{1-H}$, μ is the mean arrival rate of the traffic, B is the buffer size of the queue, H is the Hurst parameter of the traffic , a is the coefficient of variation of the traffic, and ϵ is the QoS parameter (packet loss percentage) of the traffic flow. The coefficient of variation is approximated by (5); this approximation is only valid when the arriving traffic is short range dependant.

The Norros Approximation is the only formula we considered that uses the Hurst parameter in its calculations; therefore, it is the only formula that takes self-similarity into consideration. It is also the only formula that addresses long range dependant traffic.

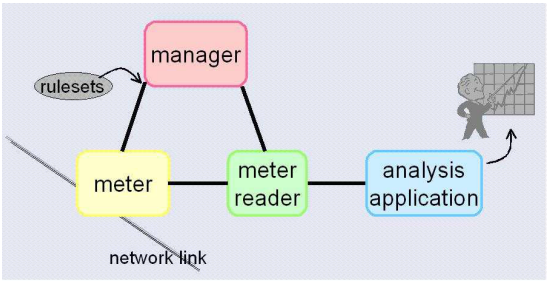

Fig. 1. Logical Diagram of RTFM Architecture

III. RTFM Architecture

A. Architecture Overview

There are 3 main components within the RTFM architecture: the meter, reader, and manager. Figure 1 shows a logical diagram depicting the RTFM architecture. The meter serves to collect statistics on network flows that pass through links that are connected to it. The reader retrieves the statistics from the meter at a regular interval via SNMP. The manager controls which flows are monitored by downloading a *ruleset* to the meter. There can be many meters and readers under the control of a single manager.

The EB functionality will be added to the meter component. Figure 2 shows the details of a meter within the RTFM architecture.

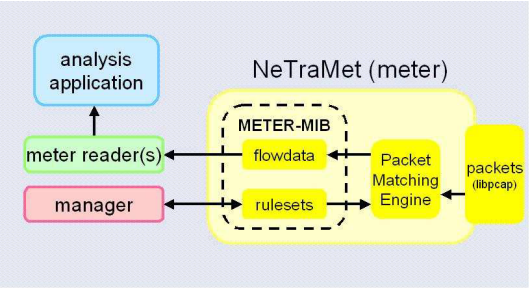

Fig. 2. Logical Diagram of Meter within RTFM Architecture

The meter is the only element of the RTFM architecture that actually handles packets. When a packet is received on the wire, a copy of the packet is passed to the meter application via a low-level packet library (the *libpcap* library is used for this purpose on the Linux operating system). The packet then goes into the packet matching engine, which determines whether the packet is of interest to the meter. This is determined by comparing the packet's properties to the ruleset that the manager has downloaded to the meter. If the packet does not match a valid rule within the ruleset, the packet is ignored and the meter waits for another packet. However, if the packet does match a rule within the current ruleset, the meter attempts to look up an entry in the flow table, which keeps statistics for flows that are specified within the ruleset. Appropriate fields are stored or updated, and the packet is discarded.

B. Modifications to RTFM Architecture

In our experiments, we are interested in the number of arriving bytes in a given time period (t_{slot}) for a particular traffic flow. This value is provided to us by the *toOctets* field defined in [1]. We utilize a sliding window system in our *online* implementation in order to keep the calculations as efficient as possible. The size of the sliding window is defined as N slots. The program computes the change in the *toOctets* field for the flow between the last value of t_{slot}, since the *toOctets* field is a running 64-bit counter. This value, ($\Delta toOctets$), is then used to recompute the mean and variance, and if necessary, the index of dispersion. A flow chart describing the effective bandwidth calculation is seen in Figure 3.

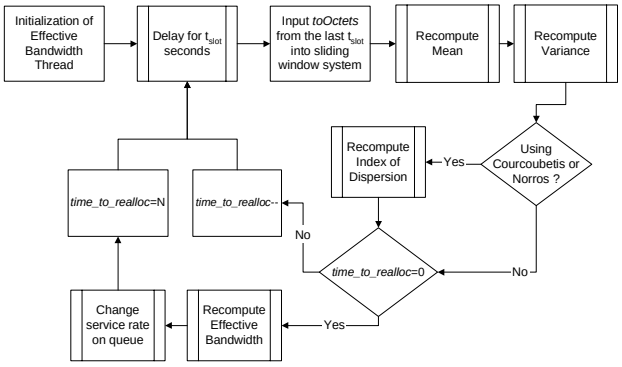

Fig. 3. Flowchart Describing Effective Bandwidth Calculation Process

The mean, variance, and index of dispersion are recalculated after every t_{slot}. Since the sliding window size is known, we can remove the data received N slots ago, and replace it with the newly measured data ($\Delta toOctets$). We then recompute the mean, variance, and index of dispersion of the data stored within the sliding window system.

The variable $time\_to\_realloc$ represents the number of measurements that need to be taken before the effective bandwidth is recalculated. Since $time\_to\_realloc$ is decremented each time a entry is made into the sliding window system, we should recompute the effective bandwidth according to the specified method if $time\_to\_realloc = 0$. We then reset the value of $time\_to\_realloc$ to N. Otherwise, $time\_to\_realloc$ is decremented, and control proceeds back to the t_{slot} delay block.

Once the new effective bandwidth has been computed, the program utilizes the *tc* utility under Linux in order to change the bandwidth allocated to that particular flow. When the network flow is detected in the meter, a queue is created for the network flow. This allows us to set the service rate individually for each flow Also, since some of the approximation formulae require a buffer size parameter, we set the length of the queue to be 150,000 bytes. Once the reallocation is complete, control defaults back to the t_{slot} delay block.

IV. Experiments

A. Overview

We used an implementation of the RTFM architecture called NeTraMet, which is available at `http://www2.auckland.ac.nz/net/NeTraMet/`. The network environment can be seen in Figure 4. The computers *carolina*, *ncstate*, and *wolfpack* represent three separate networks. Each of the computers is running Linux, and all links in the network are 100mbps Ethernet. *ingress* is the router, which is serving as the meter and manager/reader. The NeTraMet code running on *ingress* is dynamically allocating bandwidth for flows originating from *carolina, ncstate,* and *wolfpack* which are destined for the *core1* computer, which represents a destination network.

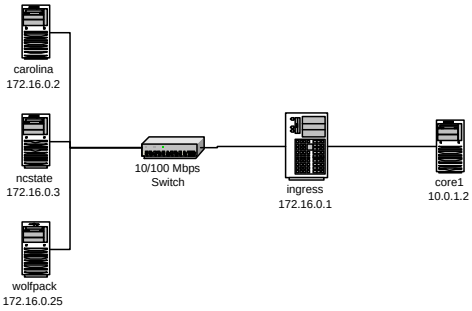

Fig. 4. Network Diagram of Simulation Environment

We used the Sup-FRP traffic model defined in [8] to generate the traffic which is sent through the network. This model generates traffic from the aggregation of ON-OFF sources. This model is widely accepted for use as a self-similar traffic generator.

A logical diagram depicting the traffic flow through the *ingress* router can be seen in Figure 5. We measure the QoS

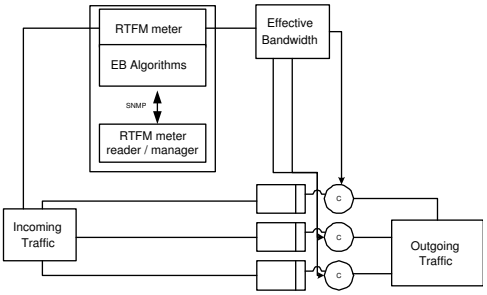

Fig. 5. Logical Diagram of Experimental Environment

parameter, packet loss probability, at the *core1* computer. The traffic sent across the network in all tests last 1000 seconds.

The goal of these experiments is to investigate the feasibility of a *realistic* implementation of effective bandwidth estimators within an online measurement system. Previous work is limited to simulations of the environment shown in Figure 4; therefore, we consider the viability of these resource allocation schemes in a real network which utilizes the IETF-standardized RTFM architecture.

Figures 6, 8, and 10 show a plot of the traffic trace sent across the network along with the actual bandwidth that was allocated to that network stream. These plots, along with their corresponding packet loss probability, shows that we can allocate bandwidths which are less than the peak rate while still ensuring that our QoS constraints are met.

B. Single Flow Experiments

In these tests, we sent a single network flow from the *carolina* computer to the *core1* computer in order to test the functionality of each approximation algorithm. Figure 6 shows a plot of the traffic trace sent, along with the computed effective bandwidth for each algorithm.

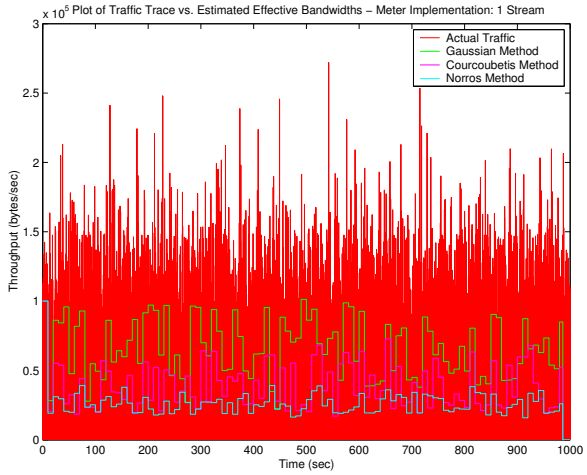

Fig. 6. Throughput vs. Effective Bandwidth - Single Flow

Figure 7 shows that the effective bandwidth allocates the appropriate amount of bandwidth to ensure that the QoS constraint is met, while still preserving bandwidth over a peak-rate allocation scheme.

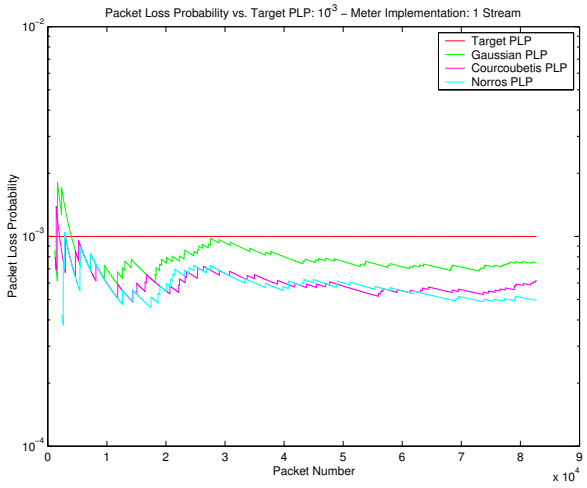

Fig. 7. Packet Loss Probability - Single Flow

The values shown in Figure 6 appear to match the results seen in simulation trials described in [4]. This confirms that our implementation is operating as expected.

We repeat the above tests, changing only the traffic trace, in which the mean arrival rate is doubled between the interval $(400, 800)$ seconds. The purpose of this test is to observe the ability of the online measurement system to track significant changes in the traffic and act accordingly to ensure that the QoS constraint is met. Figure 8 shows a plot of the traffic trace sent, along with the computed effective bandwidth for each algorithm.

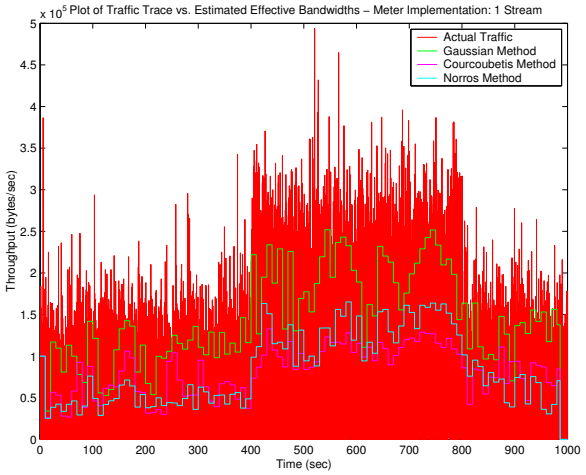

Fig. 8. Throughput vs. Effective Bandwidth - Single Flow with Increased Mean Arrival Rate

Figure 9 shows that when the mean arrival rate is doubled, only the Gaussian approximation of the effective bandwidth allocates the appropriate amount of bandwidth to ensure that the QoS constraint is met, while still preserving bandwidth over a peak-rate allocation scheme. Both of the Courcoubetis and Norros Approximations experienced significant packet loss when the mean arrival rate of the traffic doubled.

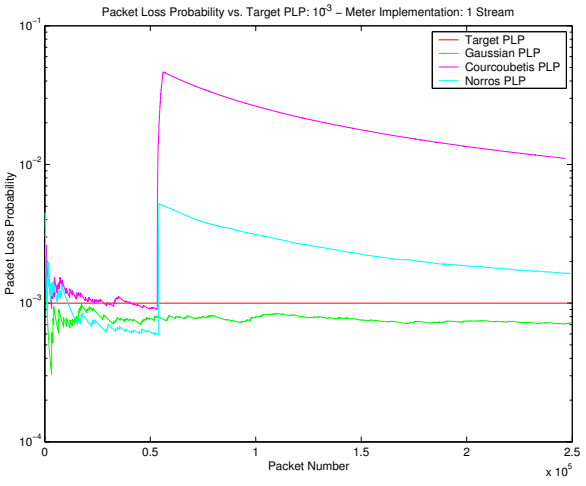

Fig. 9. Packet Loss Probability - Single Flow with Increased Mean Arrival Rate

The measurement time scale is relative to the traffic characteristics; therefore, a static t_{slot} value fails to accurately

capture the characteristics of the traffic. If t_{slot} is too small, then the confidence intervals of the measured statistics of the traffic become larger, and the system tends to over allocate resources. In the opposite case, when t_{slot} is too large, the estimated EB will tend towards the mean arrival rate, due to the law of large numbers. This causes in an under-allocation of resources, which causes the QoS of the traffic to degrade. This is the phenomenon observed in Figure 9; the t_{slot} value is too large for the Courcoubetis and Norros algorithms to estimate the effective bandwidth of the changing traffic which still maintains the required QoS.

The use of a dynamic time scale within these algorithms is discussed in [4], [9], and [10]. Simulation results have shown that dynamically changing the length of t_{slot} at the completion of N window slots allows the system to accurately track large changes in the characteristics of the traffic. With a dynamic t_{slot}, QoS constraints can be met even when dramatic changes in the traffic characteristics are observed.

C. Multiple Flow Experiments

In these tests, we sent three network flows from the *carolina*, *ncstate*, and *wolfpack* computers to the *core1* computer in order to test the scalability of our implementation. Figure 10 shows a plot of the traffic trace sent, along with the computed effective bandwidth for each flow for the Gaussian Approximation.

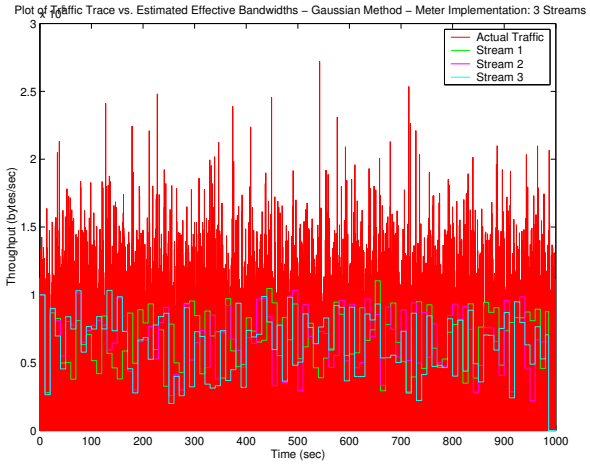

Fig. 10. Throughput vs. Effective Bandwidth - Multiple Flows, Gaussian Approximation

Figure 11 shows that the effective bandwidth allocates the appropriate amount of bandwidth to ensure that the QoS constraint is met, while still preserving bandwidth over a peak-rate allocation scheme.

The tests using the Courcoubetis and Norros approximations showed the same results as the Gaussian Approximation: the effective bandwidth allocated the appropriate amount of bandwidth to ensure that the QoS constraint is met, while still preserving bandwidth over a peak-rate allocation scheme.

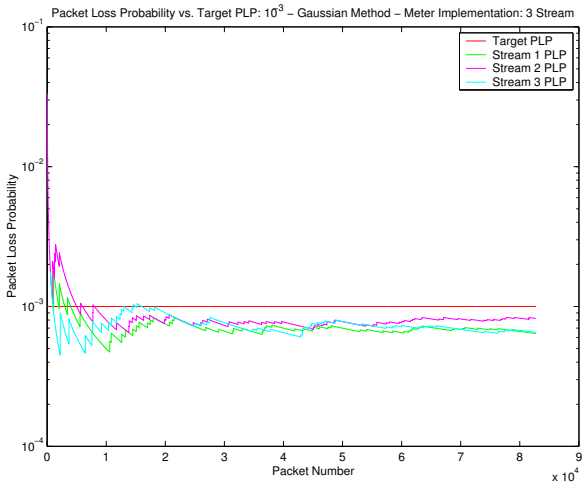

Fig. 11. Packet Loss Probability - Multiple Flows, Gaussian Approximation

V. Conclusions

In this paper, we described our implementation of three online measurement-based resource allocation schemes for use in self-sizing multiclass networks. We discussed the Realtime Traffic Flow Measurement architecture, and how we utilized it in our effective bandwidth estimation. We presented a description of our testing environment, as well as results from multiple tests showing the robustness of our implementation.

We are currently working on adding the ability to utilize a dynamic time scale in our implementation. The objective of this work is to prove the robustness of a self-sizing network which can provide differentiated services to multiple classes of network traffic.

References

[1] N. Brownlee, C. Mills, and G. Ruth, *RFC 2722 - Traffic Flow Measurement: Architecture*, IETF, October 1999.

[2] F. Kelly, *Stochastic Networks: Theory and Applications*. Oxford University Press, 1996, ch. Notes on effective bandwidths, pp. 141–168.

[3] M. Falkner, M. Devetsikiotis, I. Lambadaris, S. Tartarelli, and S. Giordano, "Empirical effective bandwidths," in *Proc. IEEE Global Telecommunications Conference (GLOBECOM 2000)*, 2000, pp. 672–678.

[4] F. Hacıömeroğlu and M. Devetsikiotis, "A comparative analysis of online measurement based capacity allocation schemes," in *Proceedings of the 7th WSEAS International Conference on Communications*, 2002.

[5] R. Guérin, H. Ahmadi and M. Naghshineh, "Equivalent Capacity and Its Application to Bandwidth Allocation in High-Speed Networks," *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 7, pp. 968–981, Sept. 1991.

[6] C. Courcoubetis, G. Fouskas, and R. Weber, "On the performance of an effective bandwidth formula," *Proceedings of 14th International Teletraffic Congress (ITC14)*, vol. 1, pp. 201–212, 1994.

[7] 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.

[8] B. K. Ryu and S. B. Lowen, "Point process approaches to modeling and analysis of self-similar traffic - part 1: Model construction," in *INFOCOM (3)*, 1996, pp. 1468–1475.

[9] F. Hacıömeroğlu, "Online measurement based capacity allocation schemes," Master's thesis, North Carolina State University, 2003.

[10] F. Hacıömeroğlu and M. Devetsikiotis, "A dynamic time scale approach for online measurement based capacity allocation," *Submitted to Proceedings of Brazilian Journal on Telecommunications, Special Issue on Computer Networks*, 2003.