Enhancing Aggregate QoS through Alternate Routing [*]

Stephen D. Patek [*] *Raja Venkateswaran* [**] *Jörg Liebeherr* [**]

[*] Department of Systems Engineering
University of Virginia
Charlottesville, VA 22903

[**] Department of Computer Science
University of Virginia
Charlottesville, VA 22903

Abstract

Recent work on differentiated services in the Internet has defined new notions of QoS that apply to aggregates of traffic in networks with coarse spatial granularity. Most proposals for differentiated services involve traffic control algorithms for aggregate service levels, packet marking and policing, and preferential treatment of marked packets in the network core. The issue of routing for enhancing aggregate QoS has not received a lot of attention. This study investigates the potential benefit of using alternate routing strategies in support of differentiated services. We propose a traffic control scheme, called *Simple Alternate Routing (SAR)*, wherein portions of marked packet flows can be assigned to alternate paths through a Service Provider Network (SPN) in response to congestion feedback information. The scheme is simple, requiring only minor changes to the SPN border routers so that alternately routed packets can be tunneled via conventional paths to an intermediate border node and then tunneled from there to the original egress border node. We present distributed algorithms for (1) discovering congestion within the SPN, and (2) allocating traffic to alternate paths that are uncongested. We have implemented the scheme in a packet-level simulation, and we have examined the transient response of the algorithm to perturbations in the nominal traffic levels experienced by the SPN. The experimental study of this paper provides some understanding of the scheme's ability to adapt in routing packets around congestion. Our results indicate that the alternate routing framework shows promise and warrants further consideration.

Key Words: Quality-of-Service, Routing, QoS-Routing, Differentiated Services.

[*] This work is supported in part by the NSF through grants ANI-9903001, ECS-9875688 (CAREER), ANI-9730103, and NCR-9624106 (CAREER).

1 Introduction

The need for Quality of Service (QoS) on the Internet to meet the service requirements of new and emerging applications requires fundamental changes to the basic connectionless best-effort architecture of the Internet. The first approaches to introduce QoS in the Internet from the early 1990s have focused on supporting varying service qualities for each individual end-to-end traffic flow. In this *per-flow* model, network resources are reserved separately for each individual flow to support the desired QoS level. In the Internet Engineering Task Force (IETF), the Integrated Services Working Group (IntServ WG) has devised a per-flow QoS service model [42, 37]. However, Internet service providers generally have not embraced the per-flow model, mostly due to the need to maintain state information for each flow at each router on its path.

The gap between the growing need for service differentiation and the inability of the existing per-flow QoS model to serve this need has triggered a rethinking of the basic tenets of QoS on the Internet and has led to a major revision of the approach to implement QoS in the Internet. Starting as early as 1995 [16], a revised QoS notion has emerged [17, 18, 34], and, since November 1997, is being made precise by the *Differentiated Services Working Group* (DiffServ WG) group in the IETF [8, 9, 10]. A main characteristic of the new QoS model is that service guarantees are given to aggregate flows, rather than on a per-flow basis. While proposals vary widely in their specifics, they all share the following characteristics.

- Service providers and users agree upon a hierarchy of service classes defined with respect to a generalized notion of bandwidth consumption.
- The service agreements are enforced at the network boundaries, through a combination of marking, dropping, or shaping of incoming packets.
- Network elements in the core of a network process packets based exclusively on the marking that packets received at the network border.
- Service agreements are made for traffic aggregates as opposed to single traffic flows. Elements in the core of the network do not have any notion of end-to-end flows.

QoS for aggregate traffic is fundamentally different from per-flow QoS. For example, the QoS guarantees for aggregate traffic in a network can have a different geographical scope [9, 17] between a specific source/destination pair, from a specific source to a set of destinations, and from a source to any destination.

In this paper, we consider a network abstraction as depicted in Figure 1. The network is composed of customer networks and Service Provider Networks (SPNs). Each customer network has access to at least one SPN. Customer networks are the ultimate sources and sinks of traffic, so that each SPN must be connected to at least two other networks. Each SPN consists of a set of interconnected routers. Routers which connect to another network are called *border nodes*, the other routers are called *core nodes*. Border nodes that receive incoming traffic are *ingress nodes*, and border nodes that transmit traffic to neighboring SPNs are called *egress* nodes. Any given border node can be both an ingress node and an egress node. We refer to the aggregate traffic for a given ingress/egress node pair as an *aggregate flow*. This is not to be confused with an individual flow in the *per-flow* QoS model.

Network service providers offer customer networks a range of network services. Customers and service providers negotiate a traffic profile which specifies the traffic rate which can be submitted to the network for a given service [9]. A traffic profile is manifested in a so-called Service Level Agreement (SLA) with a corresponding Service Level Specification (SPS). Traffic conditioning at network boundaries is a common denominator in most Internet differentiated services proposals [8, 9, 10, 17, 23, 24, 44]. Traffic conditioning includes metering, marking, dropping, and shaping of traffic. A simple way to condition traffic is to mark packets which comply to the negotiated traffic profile as ‘in-profile’, and to mark all other packets as ‘out-of-profile’, implying that out-of-profile traffic has a higher drop priority. Each traffic conditioner is responsible

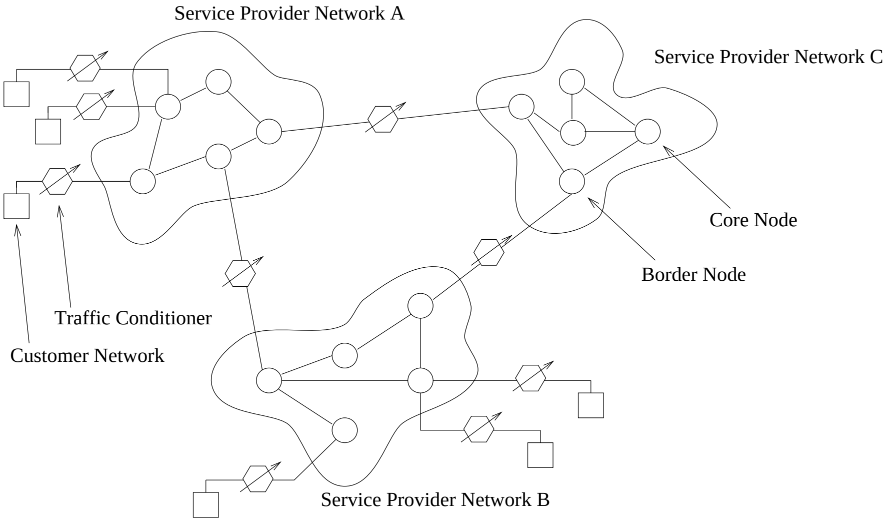

Service Provider Network A
Service Provider Network C
Core Node
Border Node
Traffic Conditioner
Customer Network
Service Provider Network B

Figure 1: Network View.

for maintaining state information for the aggregate flows it monitors. The conditioning of a packet can be different at each network boundary traversed by the packet, based on the SLAs between adjacent networks. As a convention, we refer to in-profile packets as *marked*, and to out-of-profile packets as *unmarked*.

In this paper we focus on traffic control algorithms for a single service provider network. We adopt a service model similar to the *Assured Forwarding Per Hop Behavior* [23], currently proposed by the DiffServ WG. In short, we seek to minimize the loss of in-profile traffic in networks with *coarse spatial granularity* [18], that is, where the service profile is applied to any possible destination in the Internet.

Aggregate QoS through Alternate Routing Without the ability to establish per-flow state in the network, and with limited complexity at core nodes, traffic control algorithms which enable or support differentiated services will be heavily based on algorithms implemented at border nodes. In addition to traffic conditioning we propose assigning two extra responsibilities to the border nodes: (1) congestion discovery and detection of alternate paths, and (2) allocation of traffic along alternate paths. We describe these extra responsibilities in detail in Section 3, while the bulletized description below should convey the main ideas.

- **Congestion Discovery and Detection of Alternate Paths:** Since the directionality and volume of traffic is not specified in advance in networks with aggregate QoS and coarse spatial granularity, traffic control algorithms must rely heavily on feedback from the network. In this paper, we require the border nodes of the network to periodically collect congestion information about the network to facilitate subsequent redirection of traffic flows. Each border node periodically transmits a probe packet to egress nodes to determine the existence of congestion on the prevailing paths with the SPN. If a probe packet encounters a link which is utilized above a given threshold, then the path it is traversing is declared to be congested. More generally, however, probing mechanisms can be used to collect detailed information about the state of the network, such as the amount of bandwidth and buffer space available at each link along a path [11, 25]. Alternatively, feedback information can be obtained by piggybacking state information along the return path for a flow.

- **Allocation of Flow Along Alternate Paths:** Again, without specific prior information about the volume and directionality of traffic in networks with aggregate QoS and coarse spatial granularity, provisioning for QoS guarantees is extremely difficult without suffering from underutilization of network resources. Thus, it is of interest to have mechanisms in place which allow the network to make use of capacity that would otherwise go unused. The proposed algorithm of Section 3 requires border nodes to allocate varying amounts of flow along underutilized paths in response to the probing and feedback mechanism described above. We assume that the network employs an existing, distributed routing algorithm such as OSPF [32]. We allow the possibility that underlying network routes change dynamically in response to congestion, but we assume that these routing updates are infrequent, at least with respect to the time-scale of the rerouting process that we propose. We will employ an alternative technical mechanism, called *alternate routing*, in which we assume that the network has the ability to implement "IP tunneling" between border nodes, i.e., the network has the ability to perform IP-in-IP encapsulation [43]. Thus, we do not require or assume that the algorithms for flow redirection need to cooperate with the underlying (slowly varying) routing protocol.

The purpose of this paper is to introduce and evaluate an alternate routing framework for aggregate QoS and to provide results from an initial simulation study. The layout of the paper is as follows. In Section 2, we describe related work in the area of routing. In Section 3, we provide a complete description of our alternate routing scheme, referred to as *Simple Alternate Routing (SAR)*, and indicate how it applies in various contexts for aggregate QoS. In Section 4, we present simulation results that illustrate the ability of our scheme to reroute flows around congestion. In Section 5, we discuss our results and make brief conclusions.

2 Related Literature

The basic idea in alternate routing has its roots in the dynamic and alternate routing algorithms developed for circuit switched networks in the 1980's and 1990's [4, 5, 36, 20, 30, 29, 6]. The decentralized scheme known as Dynamic Alternative Routing (DAR) introduced by Gibbens, Kelly, et al. [20] is of particular interests. In DAR, assuming a complete graph topology, individual calls, say between nodes i and j, are directly connected whenever enough capacity is available on the link (i, j). As soon as the direct connection becomes unavailable, then an intermediate node k is randomly selected, and if the utilization of each link (i, k) and (k, j) is below trunk reservation thresholds, then the call is routed on the alternative 2-link path (i, k, j). DAR is often referred to as "sticky random routing" because node k defined for calls between i and j is held fixed until the alternative path becomes unavailable due to trunk reservation on each constituent link, at which time a new tandem node is selected. The alternate routing scheme of this paper is similar to DAR in that (1) alternates are constructed by tunneling traffic to intermediate egress nodes and (2) the same alternative path is used for a given ingress/egress pair until it become congested at which time a new alternative path is randomly selected. Of course, our alternate routing scheme is intended for the Internet, which is packet switched where routing decisions apply to aggregates of traffic and not to individual calls. Other researchers have considered interesting variations on DAR, and we cite particularly the Aggregated Least Busy Alternative scheme of Mitra et al. [29], where alternative paths are selected with consideration to the load already being experienced on each candidate. Load dependent alternative path selection is an idea that applies in our alternate routing framework, but we do not consider this possibility further in this paper.

Recently there has been a lot of interest in per-flow QoS routing for the Internet. Here, the focus has been on technical mechanisms including per-flow QoS extensions to OSPF [1, 19, 49], algorithmic considerations (complexity of optimal routing) [12, 14, 22, 35, 39, 46], the issue of imperfect state information [13, 21, 26], and overall practical consideration [2, 3, 27, 28, 41]. Recent work by Nelakuditi, Zhang, and Tsang [33] bears a particularly close relationship to ours. In [33], they propose Adaptive Proportional Routing (APR), a "localized" QoS routing scheme, where ingress nodes use locally available information in selecting paths for individual QoS flows based on the notion of virtual capacity. They describe a simple and robust implementation of their idealized scheme, referring to it as "proportional sticky routing". The alternate routing scheme of this paper is similar in that we attempt to reroute flows on the basis of locally collected information, however, the underlying QoS models are fundamentally different. Other related work is due to Segall et al. [40], who describe a means of reducing the number of blocked sessions in a guaranteed services network by constructing alternate paths for traffic as a sequence of intermediate destinations without requiring full knowledge of the underlying routing structure. Alternate paths are selected on the basis of feedback information about the availability of resources on their constituent links, and the concept applies to unicast and well as multicast. Zappala [48] discusses an alternative path routing mechanism similar to ours for multicast traffic, focusing on issues of path computation and installation.

In studying the literature, we have found very little published research on routing for enhanced aggregate QoS or differentiated services. Stoica and Zhang's recent work on Location Independent Resource Accounting (LIRA) [44] considers economic mechanisms for traffic conditioning and routing without appealing to a per-flow QoS model. LIRA is essentially a pricing-based mechanism for differentiated services, where traffic is marked with respect to link prices that depend on utilization. Each aggregate traffic source is equipped with a leaky bucket traffic conditioner, where (1) tokens flow into the leaky bucket at a prescribed rate according to a service contract between the aggregate user and the SPN and (2) the number of tokens required for a packet to be marked as in-profile depends on the size of the packet and on the sum of the per-bit prices for each link on a given path. Link prices are set as the inverse of available capacity and are computed incrementally (cf. Equation (2) in [44]). One implication of this is that traffic marking in LIRA depends on the state of the network. That is, holding fixed the total volume of traffic produced by an aggregate source, the percentage of marked (in-profile) traffic depends on the level of congestion in the network. Routing in LIRA is accomplished by maintaining a list of minimum cost paths for each ingress/egress pair and then balancing the load assigned to each path in accordance with their prices. LIRA is a relatively complicated scheme for aggregate QoS since source routing is used to assign packets to a given path. In comparison, our alternate routing scheme does not require any interaction with the underlying routing protocol. In comparison to LIRA, the benefits of our schemes are that (1) our use of tunneling introduces less overhead than source routing, and that (2) all of the complexity of the scheme resides at the network's edge.

3 Simple Alternate Routing

The alternate routing scheme of this paper, referred to as *Simple Alternate Routing (SAR)*, has two main components: (1) a feedback mechanism which informs border nodes of congestion within the network and (2) a distributed control mechanism for selecting alternate paths and assigning traffic to alternate paths. We describe these mechanisms in detail in Subsections 3.1 and 3.2, respectively. We assume a form of differentiated services where marked (in-profile) packets receive preferential service within the network. For the purposes of this study this means that marked packets are the ones that get alternately routed.[1] We assume that routes in the SPN are maintained by an underlying routing protocol, such as OSPF [31, 32], which updates routes on a relatively long time scale compared to the rate at which alternate routing operations are performed. The underlying routing protocol defines the *direct paths* for the packets associated with an aggregate flow. In general, SAR seeks to reroute marked traffic away from congested direct paths. Candidate *alternate paths* between two border nodes are those routes which pass through a third border node. This is illustrated in Figure 2, where we depict the direct path and an alternate path for the aggregate flow between border nodes A and B. The alternate path between A and B has two segments, the direct path between A and C and the direct path between B and C. Note that the underlying routing algorithm for selecting direct paths need not be modified. All that is required for establishing alternate path is the ability

[1] This would be in addition to other kinds of special treatment, including favorable scheduling for marked packets.

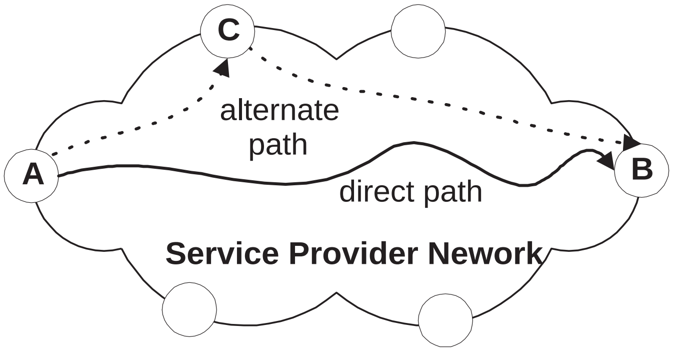


Figure 2: The direct path from A to B is the route determined by an underlying routing protocol. The alternate path through C is comprised of the direct paths from A to C and from C to B.

of border nodes to set-up tunnels, using IP-in-IP encapsulation, between nodes.

3.1 Congestion Discovery

Since the goal in alternate routing is to reroute traffic around congestion, it is essential to have a mechanism in place for discovering congestion at least along direct paths within the SPN. For this paper we adopt a minimalistic congestion discovery method. Specifically, we propose to use a probe-based mechanism that provides binary feedback indicating the existence of congestion along a given direct path. Congestion is defined in terms of buffer occupancy. If a node along a direct path has a buffer which is occupied beyond a given threshold level X, then the entire path is declared to be congested. Binary congestion feedback has been used extensively for flow control in computer networks [15, 38]. Its application here is somewhat different in that we are only interested in routing and do not attempt to change source characteristics through feedback. Congestion information assists in maintaining and allocating flow to alternate paths, as discussed in Section 3.2.

We next specify the congestion discovery mechanism, called *Simple Congestion Discovery (SCD)*, for a given border node. The algorithm is executed once per so-called a *congestion discovery period*. The SCD algorithm is a binary congestion feedback scheme [38], similar to the FECN algorithm used in ATM traffic management for ABR connections [7]. Once per congestion discovery period, each border node sends one probe packet to every other border node.

Probe packets are reflected by the destination back to the sending source node. On its forward path, a probe packet collects congestion information. A congested router will set a dedicated bit in the probe packet, similar to the EFCI bit in [7] If a probe packet is returned by the destination back to the sending node, it carries information on all congested paths of the destination node. Probe packets get the highest possible priority in the network. We assume that probe packets are never dropped and do not experience processing delay.

The main task of the following algorithm is to periodically collect and distribute congestion information on all paths between border nodes. Using the algorithm, each border node learns about the congestion status of all paths between any pair of border nodes in the network.

Algorithm 1 (Simple Congestion Discovery)

- *Each border node maintains congestion information in a congestion vector which contains the most recent information about congestion along direct paths to all border nodes. Congestion information consists of single bit, indicating the presence or absence of congestion.*

- *For each path (A, B) between two border nodes A and B being probed, the core nodes at each link along the path compare their current buffer occupancy levels to the threshold level X. If the buffer occupancy at any node is greater than X, then the entire path is declared to be congested. If congestion is discovered this way, then a bit is set in the probe. The probe continues on to B, where B appends it's congestion vector and then returns the probe to A. Finally, A stores B's congestion vector and notes the existence or absence of congestion on (A, B) in its own congestion vector.*

3.2 Allocation of Traffic to Alternate Paths

Here we describe an algorithm for selecting alternate paths for aggregate flows and allocating traffic to the alternate paths. The general approach is completely decentralized; the control algorithm is realized independently for each (A, B) pair. Decisions to reroute flow along alternate paths occur at the same time scale as the congestion discovery process described above. The allocation method is rather simple: the direct path for a given flow is used exclusively until congestion is first detected. Once this occurs, the algorithm identifies an alternate path to which some fraction of the flow from A to B may be allocated. To define an alternate path all that is required is an intermediate egress node C such that the direct paths from A to C and from C to B are uncongested; alternately routed flow will simply be tunneled to C, and then from C to B. In our scheme, only one alternate path is considered at any time. As in the alternate routing scheme in [40] where alternate paths are constructed on demand for individual QoS flows, our control algorithm does not need to know the actual composition of the alternate path, only that it is uncongested. Once an alternate path has been defined, the main work of the algorithm is (1) to select new alternate paths if congestion encountered, and (2) to adjust the flow amounts according to congestion feedback information. For the latter, the main control variable is the *fraction* of alternately marked routed traffic. We do not assume that we are able to control the absolute amounts of traffic entering the network; in fact we do not even assume that this quantity is directly observable. The only mechanism at our disposal is one where an adjustable fraction of marked packets originating at A destined for B can be shunted through an alternate path, perhaps through randomization. This fraction is adjusted up or down depending on the persistence of congestion along either the primary and alternate paths, as described below.

Algorithm 2 (Alternate Flow Allocation)

Part 1. Find_Alternate_Path

> **Input:** *Three nodes A, B, C, where (A, C, B) is the current alternate path for the aggregate flow between A and B; $C = \emptyset$ if no alternate path exists.*
> **Output:** *New alternative path (A, C', B).*

- *If $C = \emptyset$ or (A, C, B) is congested, then find a node $C' \neq C$ such that (A, C', B) is uncongested ($C' = \emptyset$ if no such node exists).*
- *Otherwise, $C' := C$, that is, the alternative path (A, C, B) is unchanged.*

Part 2. Allocate_Alternate_Flow

> **Input:** *A, B and u_{AB}, the fraction of alternately routed marked traffic from A to B.*
> **Output:** *Updated value u'_{AB}*

- *If (A, B) is uncongested, then $u'_{AB} := \max\{0, u_{AB} - k_a\}$.*
- *If (A, B) was uncongested and is now congested, then $u'_{AB} := k_0$.*

- *If* (A, B) *remains congested and an alternate path exists, then* $u'_{AB} := \min\{u_{AB} + k_a, 1\}$.
- *Otherwise* $u'_{AB} = u_{AB}$ *remains unchanged.*

The procedures *Find_Alternate_Path* and *Allocate_Alternate_Flow* are implemented simultaneously and independently for each pair of border nodes (A, B). *Find_Alternate_Path* finds an uncongested alternate path, if one exists. If an alternate path becomes congested, the algorithm will select a new alternate path. *Allocate_Alternate_Flow* determines the fraction of marked traffic which is sent on the alternate path. Unmarked traffic is never rerouted on an alternate path. The increment and decrement functions for alternately routed traffic, are following a *additive increase*/*additive decrease* using the vocabulary from [15, 38]. If congestion persists, the change to the fraction of alternately routed flow is either a constant amount k_a or the difference between the current allocation and fully alternately routed flow (*additive increase*). However, if no alternate path can be found by *Find_Alternate_Path*, the fraction of marked traffic routed on the alternate path is decreased by k_a.

4 Experimental Results

Here, we present simulation results that illustrate our scheme's ability to enhance aggregate QoS. We are particularly interested in the stability properties and transient characteristics of the scheme as a closed loop feedback control system. Our simulator is adapted from the LIRA simulator used in [44]. In our experiments we simulate our SAR algorithm in a large backbone network subjected to various types of traffic perturbations. We have considered four types of perturbations: uniform step, uniform ramp, uniform impulse train, and non-uniform impulse train. For each perturbation model, we compare the response of SAR to a baseline Internet routing protocol and to a LIRA-type multipath routing protocol. Comparisons are made in terms of aggregate marked packets lost and marked packets delivered.

Service Provider Network Our testbed SPN model is based on the vBNS backbone [45], as shown in Figure 3. Our model consists of 10 border nodes and 12 core nodes. Note that the core nodes in the simulated network are connected exactly as are the main points-of-presence in the vBNS. All links are full duplex with 10 Mbps transmission capacity, each equipped with a 1 Mb buffer. By today's standards 10 Mbps is very slow for a backbone network, however, we choose this number to reduce the overhead associated with our packet-level simulation. Propagation delay between any two nodes is fixed at 10 ms. Each link employs the droptail scheduling policy where (1) unmarked incoming packets are dropped if buffer utilization exceeds 50%, and (2) all incoming packets are dropped if buffer utilization exceeds 95%.

In the simulated traffic scenario, each border node is an ingress point for aggregate traffic destined for exactly two other border nodes, and an egress point for aggregate traffic of exactly two other border nodes, with the traffic matrix shown in Table 1. So, with 10 border nodes, there are a total of 20 aggregate flows being supported by the SPN. Each aggregate flow is comprised of a large number of individual Pareto [47] traffic sources. The nominal traffic load of each aggregate flow is defined by 400 Pareto sources. Each source starts generating traffic at simulation time $t = 30$ seconds. We set the parameters for each Pareto source as follows: (1) each source draws ON and OFF interval sizes τ from a Pareto probability density function $f(\tau) = \beta a^{\beta} \tau^{-\beta-1}$ with $\tau \geq a$, where $a \geq 0$ is a constant set so that sources transmit between 800 and 8000 bytes during ON periods and the power factor $\beta = 1.2$. Routes in the SPN are selected on the basis of minimum hop count.

Parameters for the SAR Algorithm: In the simulation runs of this section, we used the following parameters and setting for the SAR described in Section 3. First, for each aggregate flow (A, B), whenever

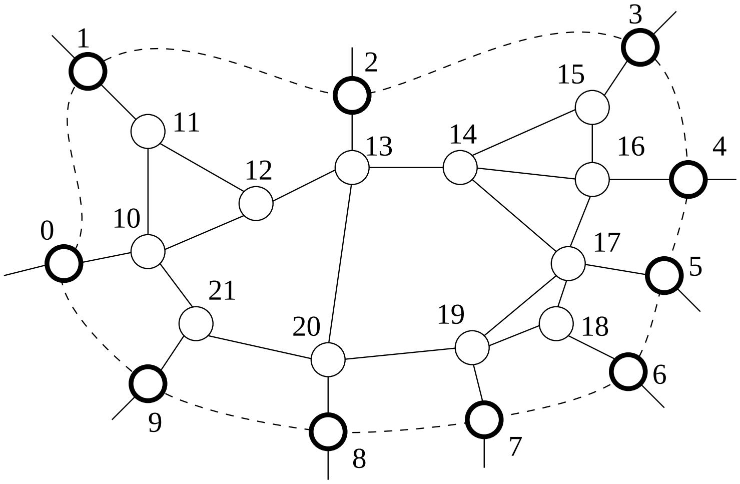


Figure 3: Simulated network topology. The thick circles depict border nodes, and all remaining nodes (inside the dashed line) are core nodes.

		TO									
		0	1	2	3	4	5	6	7	8	9
FROM	0							X	X		
	1					X				X	
	2									X	X
	3	X									X
	4	X							X		
	5		X	X							
	6			X	X						
	7				X	X					
	8		X				X				
	9						X	X			

Table 1: Traffic matrix for the SPN. Each “X”denotes indicates the presence of an aggregate flow.

it is necessary for the ingress node A to select an alternate path, it chooses one randomly out of the set of uncongested paths. Available alternate paths are chosen with equal probability. Congestion is defined by a buffer occupancy threshold of $X = .95$. Updates to the allocation u_{AB} of alternately routed marked traffic are made according to the additive increase and additive decrease rule, where the initial percentage of alternately routed marked traffic k_0 is set to zero, and the additive increase parameter k_a is set to 0.1%. Congestion discovery periods for each ingress node are separated by random intervals chosen uniformly between 1.275 and 1.725 seconds.

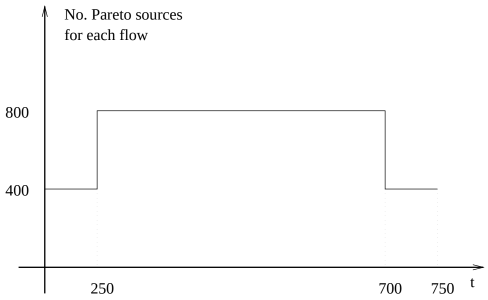


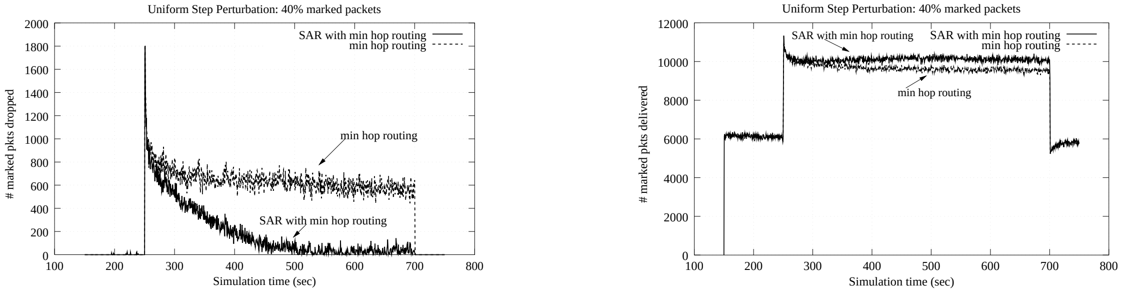
Figure 4: Uniform step perturbation model.

4.1 Uniform Step Perturbation

Here, we examine the performance of SAR when the system is subjected to an overwhelming step increase in the amount of traffic subjected to the network. We perturb the system at $t = 250$ seconds, after the system has almost reached steady state with respect to the nominal traffic load. As shown in Figure 4, the perturbation is accomplished by increasing the number of Pareto sources from 400 to 800 for *each* aggregate flow shown in Table 1; this additional traffic persists up to $t = 700$ seconds, at which time the number of sources reverts back to nominal levels. We refer to this perturbation model as a "uniform step" since the same increase in traffic is experienced in all flows simultaneously, without any particular directionality in the additional traffic. The idea is to capture the effect of a sudden increase in the number of users making use of the network. In running the simulation we collect performance statistics measured over 0.5 second intervals, and these measurements begin at $t = 150$ seconds.

We are interested in the network's response to the perturbation in different aggregate QoS scenarios. First, we examine the case where the percentage of number marked packets being generated is held fixed at 40%. That is, even after the perturbation at time $t = 250$ seconds, the percentage of traffic entering the network that is marked is 40%. Since the number of sources per aggregate flow doubles from 400 to 800, the volume of marked traffic entering the network doubles. The response of the network to this perturbation is shown Figure 5, where we plot both the numbers of marked packets dropped and marked packets delivered as a function of time. The plots show a performance of running SAR on top of a basic underlying routing algorithm, which, in our case, is a min-hop routing protocol. From Figure 5(a), which shows marked packets lost as a function of time, we observe that SAR is able to almost completely eliminate packet loss in response to the perturbation, whereas this is not the case with min-hop routing alone. Figure 5(b), which shows marked packets delivered at a function of time, helps to put this in perspective. The elimination of packet loss amounts to a relatively small percentage improvement in the aggregate number of marked packets delivered.

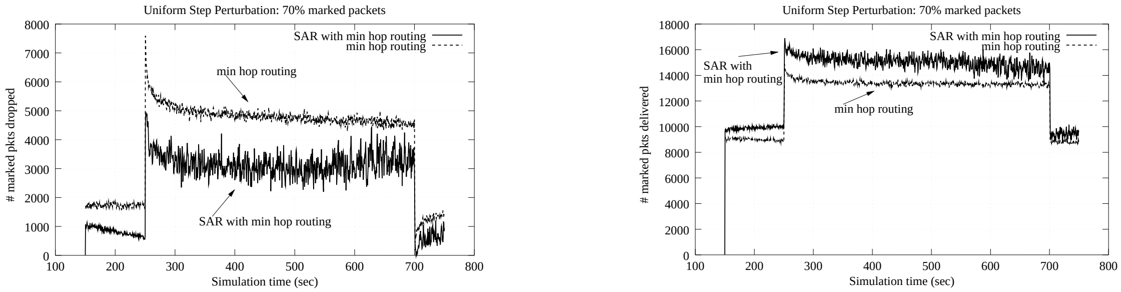
It is interesting to point out some peculiar aspects of the transient responses in Figure 5. First, with regard to marked packets lost, the initial response of SAR is intuitive: the number of marked packets lost rises sharply when the new traffic hits and then rather quickly decays to a very low level. The response of min-hop routing alone is somewhat harder to explain, particularly the apparent decay in the number marked packets lost. Why should the number of marked packets lost diminish when the routes in min-hop routing do not change? The answer is a little subtle and depends on behavior which can only arise in a network setting. We point out that before the perturbation hits at time $t = 250$ seconds, the buffers at the border nodes are not quite full, which means that any marked packet loss is due to congestion at core nodes. Since the buffers in the border nodes are not full when the perturbation starts, there is a short period of time when the network admits a great deal more traffic than it can handle at the new steady state. The traffic admitted during this time leads to an initial positive spike in *both* the number of marked packets lost and the number of marked



(a) Marked packets dropped.

(b) Marked packets delivered.

Figure 5: Sample response to the uniform step perturbation when 40% of all packets are marked.



(a) Marked packets dropped.

(b) Marked packets delivered.

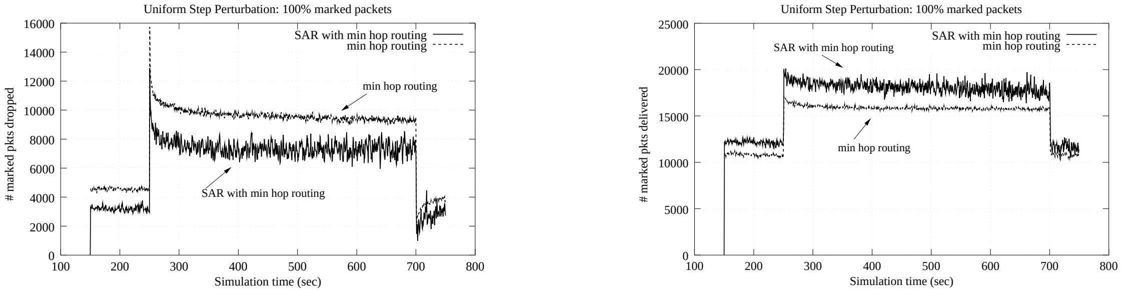
Figure 6: Sample response to the uniform step perturbation when 70% of all packets are marked.

packets delivered. This spike is short lived, and the subsequent decay in the case of min-hop routing alone is not due to route adaptation. Similar reasoning helps to explain the reverse spike which is apparent at the end of the perturbation when the number of Pareto sources per flow drops back to 400. In reverting back to the nominal traffic levels, it takes some time for the buffers at the border nodes to empty back to their nominal levels, causing a temporary shortage in the number of marked packets entering the network.

Figures 6 and 7 illustrate the performance of SAR when the percentages of marked packets arriving at the network are 70% and 100%, respectively. In both cases the network is overwhelmed by the volume of marked traffic and simply does not have the resources to reduce packet loss to zero, even with SAR in effect. We point out that the variability (noise) in the plots of marked packets lost and delivered for SAR is significantly higher than that for min-hop routing alone. This is due to rapid switching of alternate paths. Because of the extreme volume of marked traffic, as soon as an alternate path is established, the additional alternately routed traffic causes the alternate path to become congested, and this forces border nodes to seek new alternative paths. While a rapid switching of routes indicates to a certain degree of instability, these oscillations are only observed under extremely heavy volumes of marked traffic. Note that, even with the rapid switching of alternative paths, the performance of SAR in terms of marked packets lost and delivered is uniformly better than that achieved by min-hop routing alone.

So far we have focused on the performance of SAR in networks where packet marking is determined completely by source characteristics. We now examine its performance in the context of load-dependent packet marking. Specifically, we consider alternate routing as a replacement for the load balancing functionality in LIRA, while keeping the LIRA pricing-based packet marking mechanism.[2] Recall that, in LIRA, each aggregate source is equipped with a leaky bucket traffic conditioner that marks packets entering the net-

[2] In implementing SAR in this context, we were careful to mark packets with respect to congestion levels on both the direct and alternate paths for a given aggregate flow, in the proper proportions.



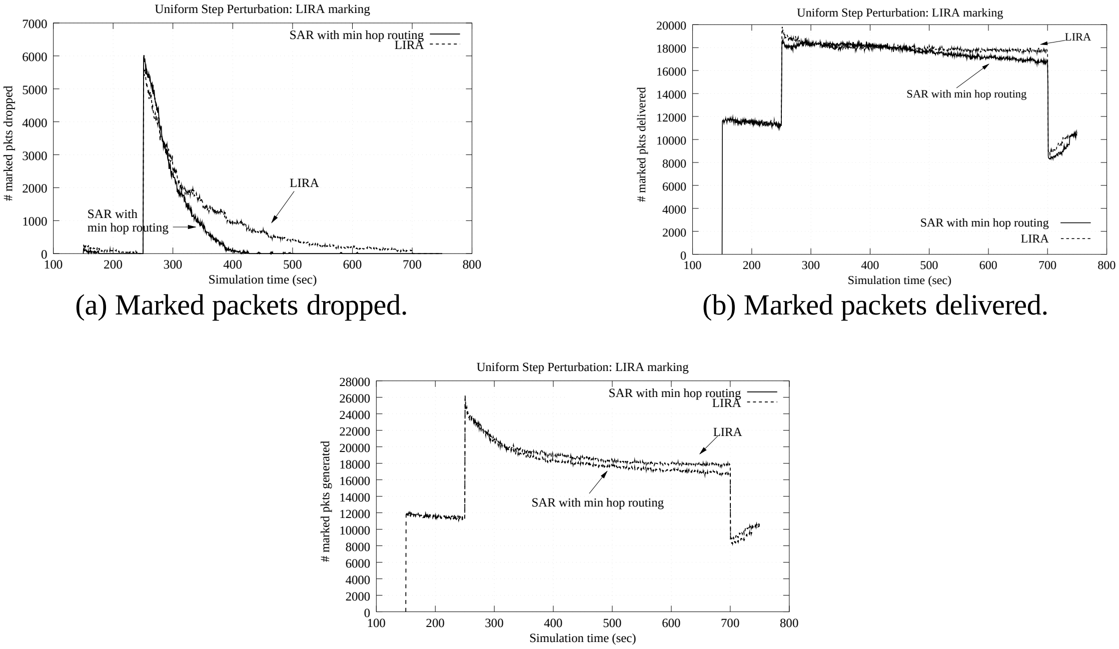
(a) Marked packets dropped.

(b) Marked packets delivered.

Figure 7: Sample response to the uniform step perturbation when 100% of all packets are marked.

work only if enough packets are available in the token buffer. The number of tokens required per packet depends on a congestion-dependent per-bit price associated with each link on the path from ingress to egress. As a result, the percentage of marked traffic entering the network depends on the level of congestion in the network, and with heavy enough load, the percentage of marked traffic could drop well below 40%. Our LIRA-based simulation results[3] are shown in Figure 8. Since the percentage of marked packets is variable, we now plot the number of marked packets being generated as a function of time [cf. Figure 8 (c)] in addition to the numbers of marked packets lost and delivered. The figure compares the performance of SAR to LIRA. Both schemes generate and deliver roughly the same number of marked packets, with the LIRA doing slightly better. With respect to marked packet loss, the SAR seems to respond faster to the perturbation than LIRA, resulting in a smaller total number of marked packets lost. Overall, SAR and LIRA perform comparably in our simulation runs, with no scheme outperforming the other. We point out, however, that SAR is considerably easier to implement. Setting aside the shared complexity of pricing-based packet marking, LIRA uses source routing to assure that traffic follows only the least-cost paths from ingress to egress. On the other hand, our alternate routing scheme is built on top of the routes constructed from an underlying routing protocol and source routing is not required.

[3] In these and all subsequent LIRA-oriented runs, we used the following parameter settings. The fixed congestion-free cost for each link α is set to one token/bit. The leaky bucket traffic conditioner for each aggregate flow has a resource token rate of 50 tokens per microsecond and a bucket size of 500,000 tokens. We limit the number of paths maintained by LIRA for each aggregate flow to two.



(a) Marked packets dropped.

(b) Marked packets delivered.

(c) Marked packets generated.

Figure 8: Sample response to the uniform step perturbation with LIRA-type packet marking.

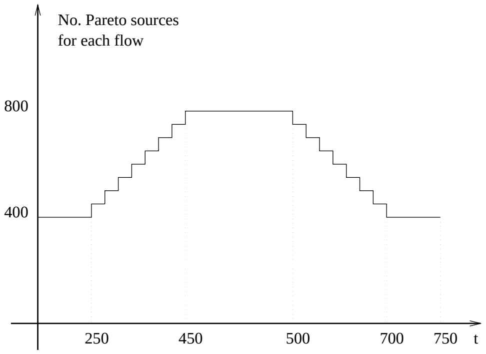
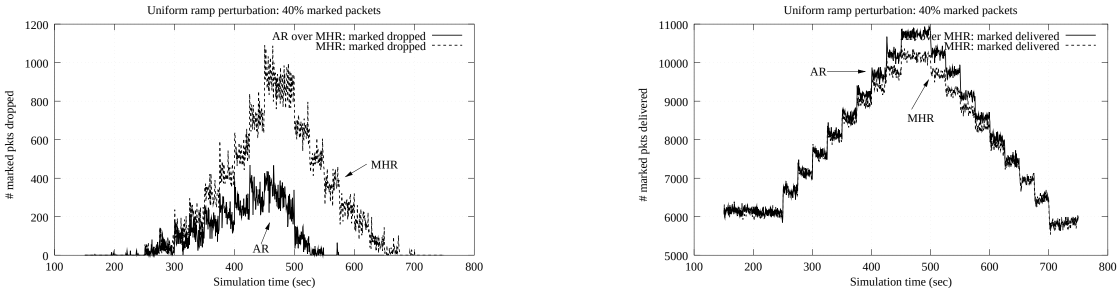


Figure 9: Uniform ramp up/down perturbation model.

4.2 Uniform Ramp Perturbation

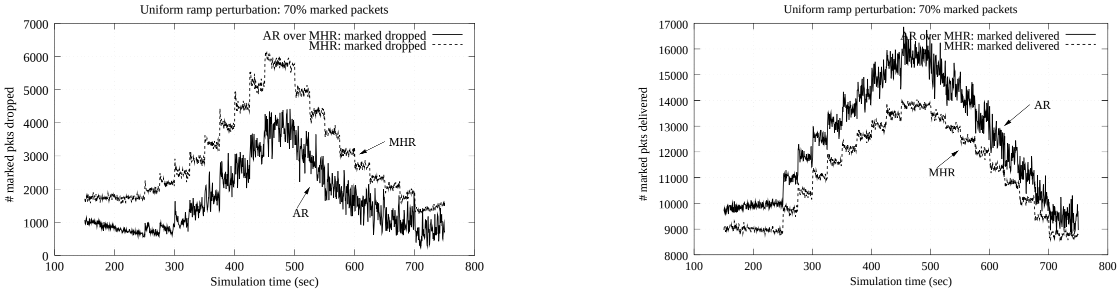
Here, we consider a variation on the uniform step perturbation model of the preceding subsection. We are still interested in the response of the system to an overwhelming increase in the traffic load, however now we slowly ramp up the traffic to its peak levels and then slowly ramp it back down by the end of the simulation, as shown in Figure 9. As before, the perturbation begins at $t = 250$ seconds, and the change in the amount of traffic is accomplished by increasing/decreasing the number of Pareto sources per aggregate flow. The peak traffic level persists up to $t = 500$ seconds, at which time the number of sources slowly steps down to nominal levels. We refer to this perturbation model as a "uniform ramp" since the same increase in traffic is experienced in all flows simultaneously, without any particular directionality in the additional traffic. The idea is to capture the effect of a slow increase in the number of users making use of the network.

Figures 10 through 12 compare the performance of SAR to min hop routing alone with 40%, 70%, and 100% packets marked. Figure 13 compares the performance of SAR to LIRA. The results are presented in exactly the same format as in the preceding subsection, the only difference being the nature of the perturbation to nominal traffic. Generally speaking, SAR performance compares favorably to min hop routing alone, again eliminating marked packet loss in the 40% case. Many of the same comments from the preceding subsection apply here. For example, in looking at the performance of SAR when 70% and 100% packets are marked, we see that the traces for marked packets dropped and delivered are considerably more noisy than for min hop routing alone. With respect to LIRA's packet-marking scheme, SAR results in fewer marked packets dropped. On the other hand, because of its optimal choice of multiple routes LIRA generates and delivers slightly more marked packets. Overall, SAR and LIRA perform similarly.



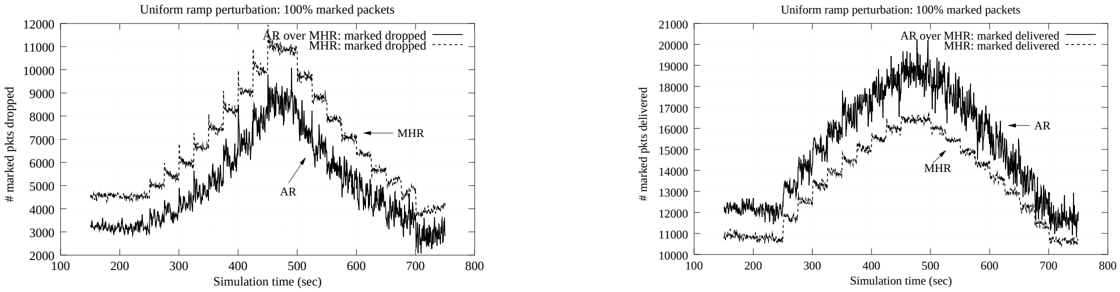
(a) Marked packets dropped. (b) Marked packets delivered.

Figure 10: Sample response to the uniform ramp perturbation when 40% of all packets are marked.



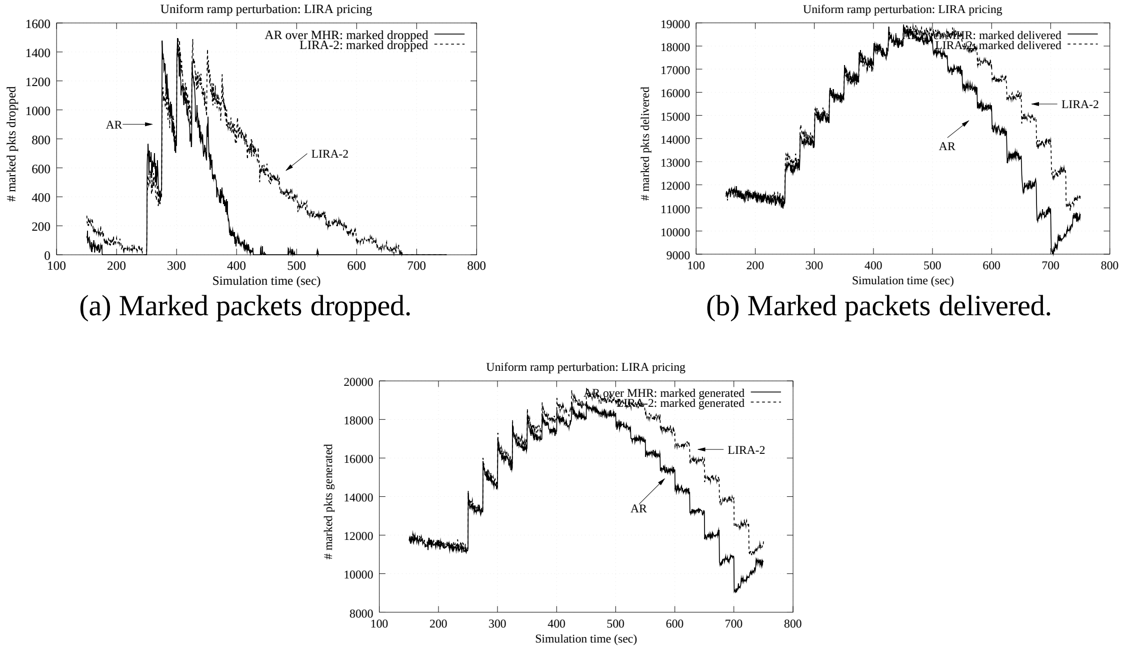
(a) Marked packets dropped. (b) Marked packets delivered.

Figure 11: Sample response to the uniform ramp perturbation when 70% of all packets are marked.



(a) Marked packets dropped. (b) Marked packets delivered.

Figure 12: Sample response to the uniform ramp perturbation when 100% of all packets are marked.



(a) Marked packets dropped.

(b) Marked packets delivered.

(c) Marked packets generated.

Figure 13: Sample response to the uniform ramp perturbation with LIRA-type packet marking.

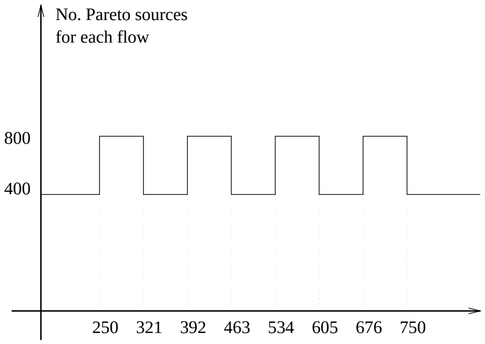
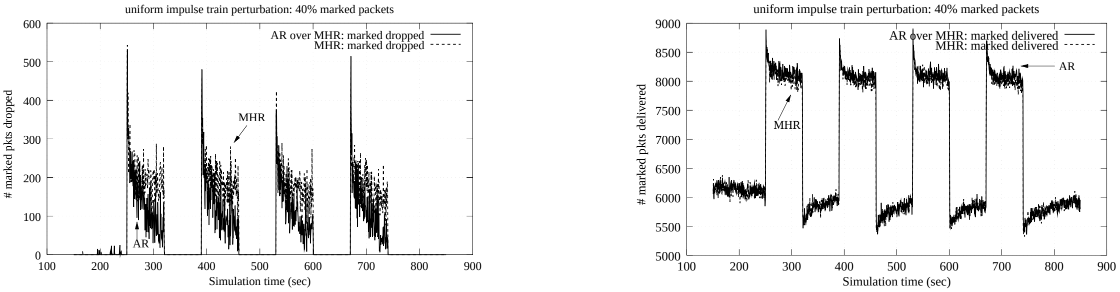


Figure 14: Uniform impulse train perturbation model.

4.3 Uniform Impulse Train

Here, we consider another overwhelming perturbation that tests the the network's ability to respond to sudden spikes in traffic levels. This time the perturbation comes as a sequence of synchronized impulses, as shown in Figure 14. Each aggregate flow experiences a periodic increase in the number of Pareto sources from 400 to 800 and back, evenly spaced in time from $t = 250$ to $t = 750$. We refer to the perturbation model as a uniform impulse train because the same change in load is experienced for each aggregate flow simultaneously. There is no particular directionality to the increase in traffic.

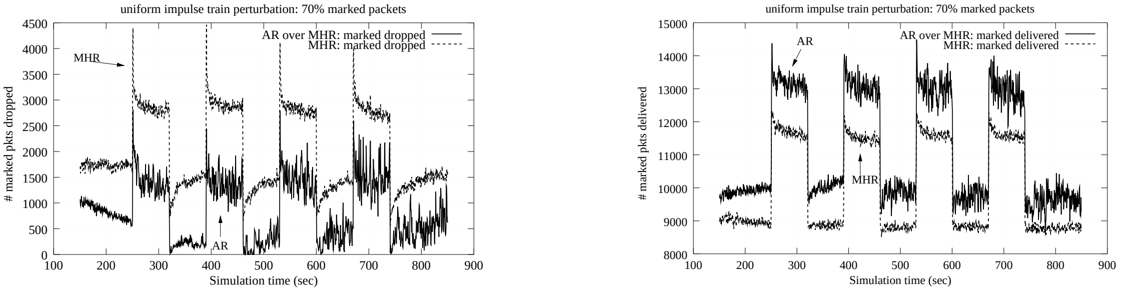
Figures 15 through 17 compare the performance of SAR to min hop routing alone with 40%, 70%, and 100% packets marked. Figure 18 compares the performance of SAR to LIRA. The results are presented in exactly the same format as in the preceding subsections. Generally speaking, SAR with min hop routing outperforms min hop routing alone. The fact that the perturbation comes as a sequence of spikes doesn't seem to cause SAR to behave erratically. As observed with the uniform step and ramp models, the plots of marked packets lost and marked packets delivered with 70% and 100% packets marked are more "noisy" than the plots for min hop routing. With respect to LIRA's packet-marking scheme, SAR results in slightly fewer marked packets dropped. On the other hand, because of its choice of multiple routes LIRA generates and delivers slightly more marked packets. Overall, SAR and LIRA perform similarly.



(a) Marked packets dropped.

(b) Marked packets delivered.

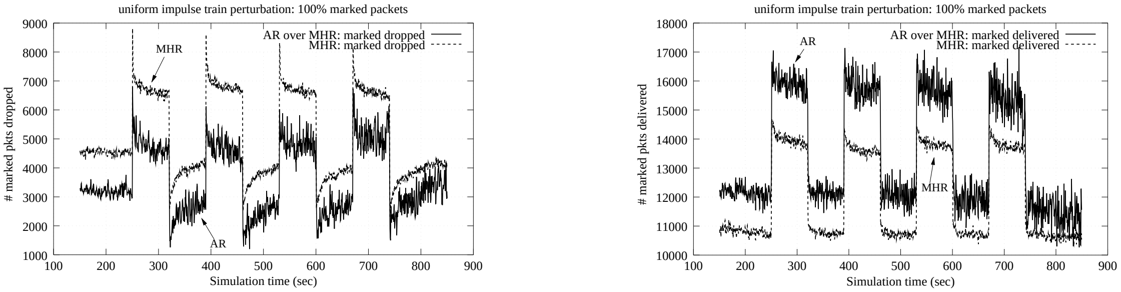
Figure 15: Sample response to the uniform impulse train perturbation when 40% of all packets are marked.



(a) Marked packets dropped.

(b) Marked packets delivered.

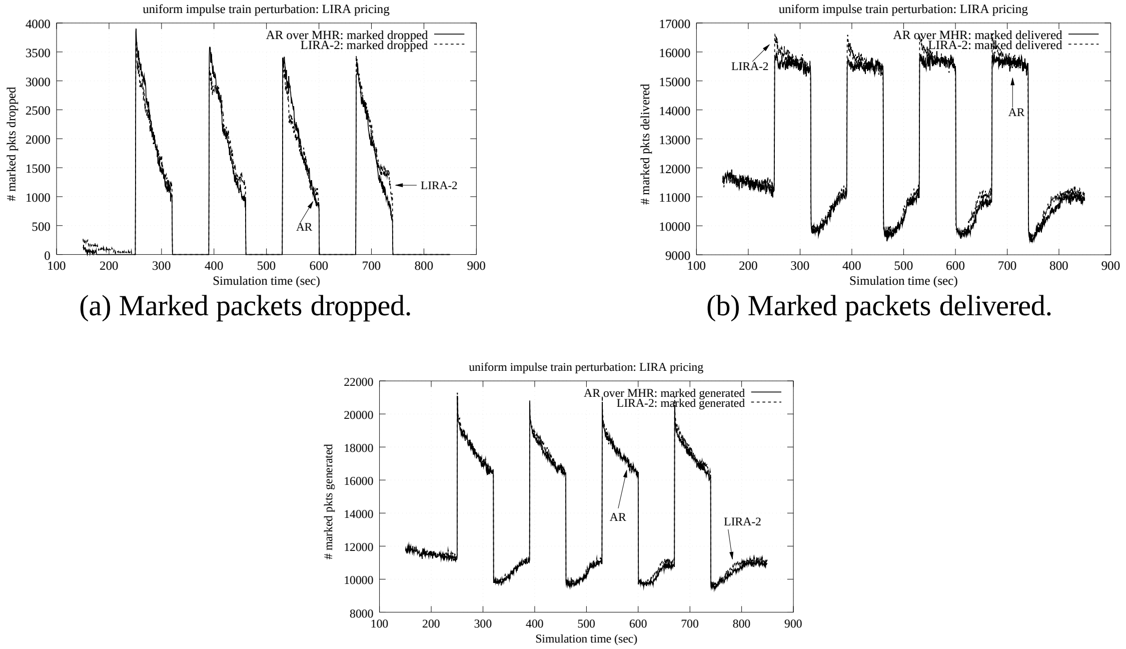
Figure 16: Sample response to the uniform impulse train perturbation when 70% of all packets are marked.



(a) Marked packets dropped.

(b) Marked packets delivered.

Figure 17: Sample response to the uniform impulse train perturbation when 100% of all packets are marked.



(a) Marked packets dropped.

(b) Marked packets delivered.

(c) Marked packets generated.

Figure 18: Sample response to the uniform impulse train perturbation with LIRA-type packet marking.

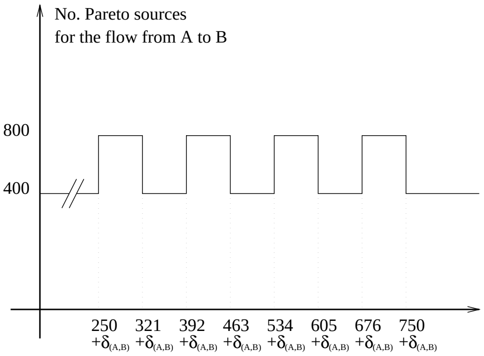
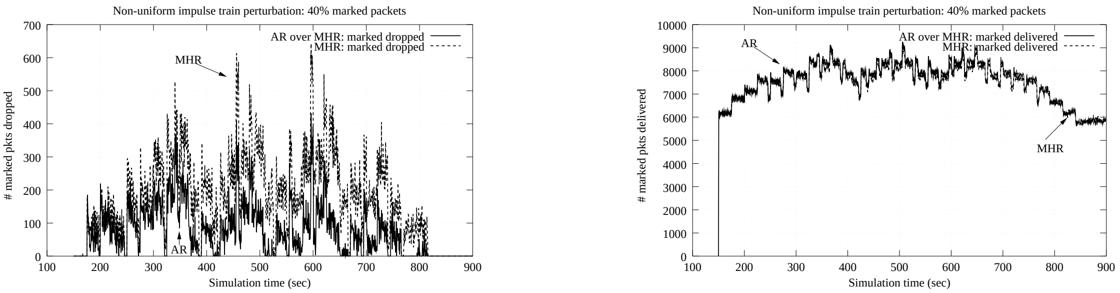


Figure 19: Nonuniform impulse train perturbation model.

4.4 Nonuniform Impulse Train

Here we consider a variation on the impulse train model from the preceding subsection. As before, each aggregate flow experiences a periodic sequence of sudden jumps from 400 to 800 Pareto sources and back again. This time, however, the timing of the jumps is staggered across aggregate flows. This is illustrated for the aggregate flow from A to B in Figure 19. Specifically, the sequence of traffic spikes begins at time $t = 250+\delta_{(A,B)}$, where $\delta_{(A,B)}$ is chosen randomly from the set of offset values $\{-75, -50, -25, 0, 25, 50, 75, 100\}$ independently of the offsets for the remaining aggregate flows. By choosing offset values this way we introduce directionality in the traffic perturbations, and for this reason we refer to the perturbation model as a nonuniform impulse train.

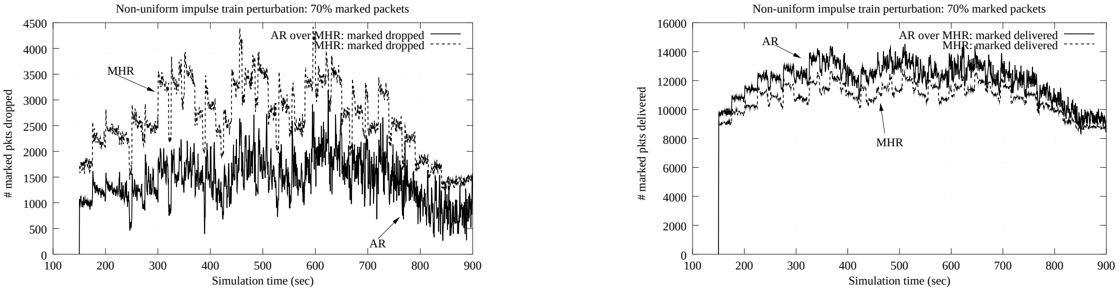
Figures 20 through 22 compare the performance of SAR to min hop routing alone with 40%, 70%, and 100% packets marked. Figure 23 compares the performance of SAR to LIRA. The results are presented in exactly the same format as in the preceding subsections. Generally speaking, SAR with min hop routing outperforms min hop routing alone. The fact that the perturbation comes as a nonuniform sequence of spikes doesn't seem to cause SAR to behave erratically. As observed with the uniform step and ramp models, the plots of marked packets lost and marked packets delivered with 70% and 100% packets marked are more "noisy" than the plots for min hop routing. With respect to LIRA's packet-marking scheme, SAR results in slightly fewer marked packets dropped. On the other hand, because of its choice of multiple routes LIRA generates and delivers slightly more marked packets. Overall, SAR and LIRA perform similarly.



(a) Marked packets dropped.

(b) Marked packets delivered.

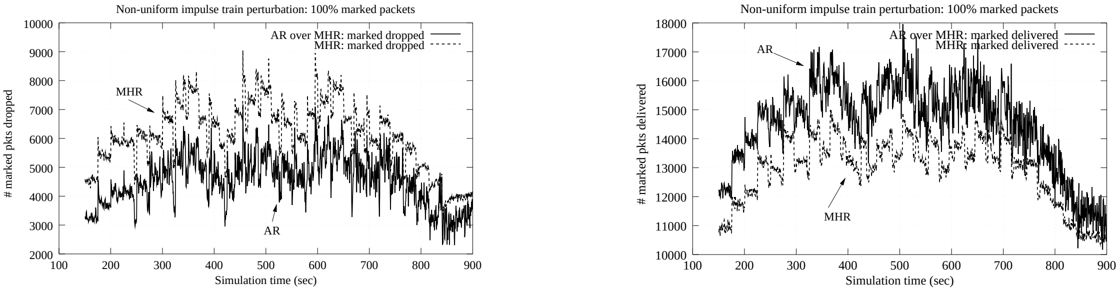
Figure 20: Sample response to the nonuniform impulse train perturbation when 40% of all packets are marked.



(a) Marked packets dropped.

(b) Marked packets delivered.

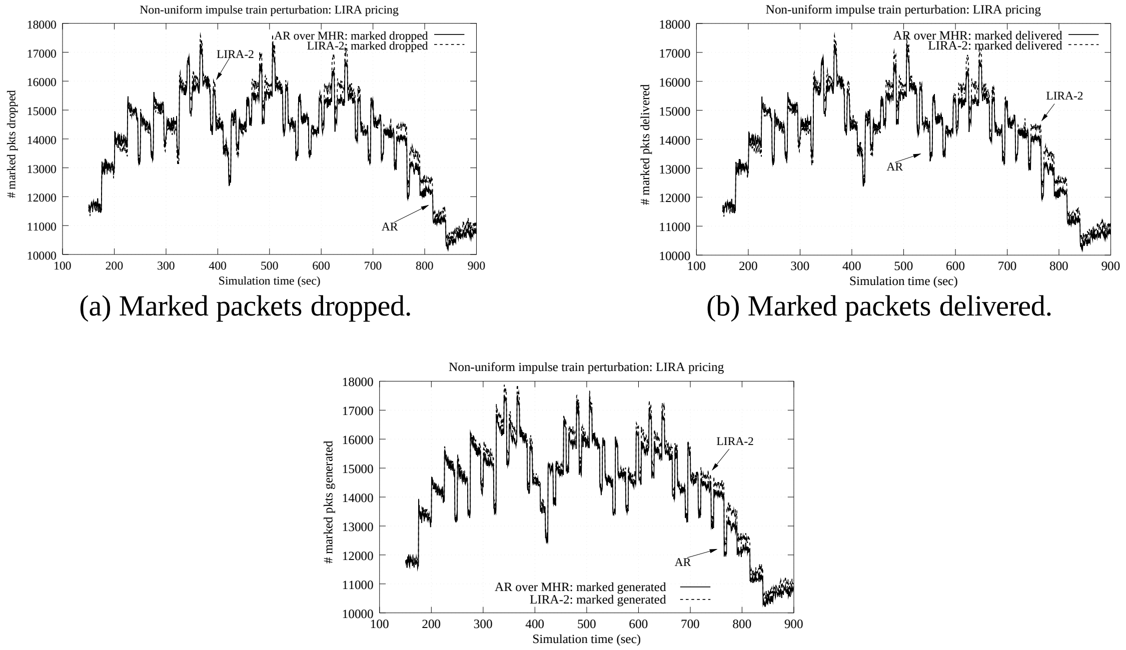
Figure 21: Sample response to the nonuniform impulse train perturbation when 70% of all packets are marked.



(a) Marked packets dropped.

(b) Marked packets delivered.

Figure 22: Sample response to the nonuniform impulse train perturbation when 100% of all packets are marked.



(a) Marked packets dropped.

(b) Marked packets delivered.

(c) Marked packets generated.

Figure 23: Sample response to the nonuniform impulse train perturbation with LIRA-type packet marking.

5 Discussion and Conclusions

Our simulation results indicate that a simple alternate routing scheme like ours can have a positive impact on the performance of aggregate QoS networks. We have tested the alternate routing scheme under a wide variety of perturbation models, and we have observed an improvement in the performance of the network at least with regard to packet loss. Because of this, we think that alternate routing holds out the promise of significantly enhancing the performance of networks with aggregate QoS.

We have uncovered a number of important issues that require further consideration. As with any feedback control system, oscillations can result in responding aggressively to congestion. Even with the very mild feedback gains used in Section 4 (i.e. $k_0 = 0$ and $k_a = .1\%$) oscillations arose in situations with very large amounts of marked traffic flow. Another question is how alternate routing will perform when used in conjunction with an underlying routing protocol with congestion-sensitive metrics. In our preliminary simulation runs, we have seen that interactions can arise between alternate routing and the underlying state-dependent routing protocol, and generally these interactions serve to degrade performance. This tricky issue is really one of coordinating routing decisions on multiple time scales, with alternate routing decisions occurring frequently and underlying routing table updates occurring infrequently. We point out, however, that in practice, OSPF is typically not set to respond to congestion-sensitive metrics.

References

[1] G. Apostopoulos et al. QoS Routing Mechanisms and OSPF Extensions, August 1999. Internet RFC 1633.

[2] G. Apostopoulos, R. Guérin, S. Kamat, A. Orda, and S. K. Tripathi. Quality of Service Based Routing: A Performance Perspective. In *Proc. SIGCOMM*, 1998.

[3] G. Apostopoulos, R. Guérin, S. Kamat, A. Orda, and S. K. Tripathi. Intradomain QoS Routing in IP Networks: A Feasibility and Cost/Benefit Analysis. *IEEE Network*, 13(5):42–54, Sept./Oct. 1999.

[4] G. R. Ash, R. H. Cardwell, and R. P. Murray. Design and Optimization of Networks with Dynamic Routing. *Bell System Technical Journal*, 60(8):1787–1820, 1981.

[5] G. R. Ash, R. H. Cardwell, and R. P. Murray. Servicing and Real-Time Control of Networks with Dynamic Routing. *Bell System Technical Journal*, 60(8), 1981.

[6] G. R. Ash and B. D. Huang. An Analytical Model for Adaptive Routing Networks. *IEEE Transacations on Communications*, 41(11):1748–1759, 1993.

[7] ATM Forum, ATM Forum Traffic Management Specification Version 4.0, April 1996.

[8] Y. Bernet et al. A Conceptual Model for DiffServ Routers. IETF Internet Draft, Diffserv Working Group, <draft-itef-diffserv-model-01.txt>, October 1999.

[9] Y. Bernet et al. A Framework for Differentiated Services. IETF Internet Draft, Diffserv Working Group, <draft-itef-diffserv-framework-02.txt>, February 1999.

[10] S. Blake et al. An Architecture for Differentiated Services. IETF Internet Draft, Diffserv Working Group, <draft-itef-diffserv-arch-02.txt>, December 1998.

[11] R. L. Carter and M.E. Crovella. Measuring Bottleneck Link Speed in Packet-Switched Networks. Technical Report BU-CS-96-006, Boston University, Computer Science Department, March 1986.

[12] S. Chen and K. Nahrstedt. An Overview of Quality-of-Service Routing for the Next Generation High Speed Networks: Problems and Solutions. *IEEE Network*, 12(6):64–79, Nov./Dec. 1998.

[13] S. Chen and K. Nahrstedt. Distributed Quality-of-Service Routing in High-Speed Networks Based on Selective Probing. In *23rd Annual Conference on Local Computer Networks, LCN'98*, 1998.

[14] S. Chen and K. Nahrstedt. On Finding Multi-constrained Paths. In *Proceedings of the International Conference on Communications, IEEE ICC*, 1998.

[15] D. Chiu and R. Jain. Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks. *Computer Networks and ISDN Systems*, 17:1–14, 1989.

[16] D. D. Clark. Adding Service discrimination to the Internet, 1995.

[17] D. D. Clark and W. Fang. Explicit Allocaiton of Best Effort Packet Delivery Service. *IEEE/ACM Transactions on Neworking*, 6(4):362–373, August 1998.

[18] D. D. Clark and J. Wroclawski. An Approach to Service Allocation on the Internet. IETF Internet Draft, Diffserv Working Group, October 1997.

[19] E. Crawley et al. A Framework for QoS-based Routing in the Internet, August 1998. Internet RFC 2386.

[20] R. J. Gibbens, F. P. Kelly, and P. B. Key. Dynamic Alternative Routing - Modeling and Behaviour. In *TELETRAFFIC SCIENCE for New Cost-Effective Systems, Networksand Services, ITC-12*, pages 1019–1025. Elsevier Science Publishers B. V. (North-Holland), 1989.

[21] R. Guérin and A. Orda. QoS Routing in Networks with Inaccurate Information: Theory and Algorithms. In *IEEE Infocom*, 1997.

[22] R. Guérin and A. Orda. Networks with Advance Reservations: The Routing Perspective. In *IEEE Infocom*, 2000.

[23] J. Heinanen et al. Assured Forwarding PHB Group, June 1999. Internet RFC 2597.

[24] V. Jacobson et al. An Expedited Forwarding PHB, June 1999. Internet RFC 2598.

[25] S. Keshav. *Congestion Control in Computer Networks*. PhD dissertation, University of California at Berkeley, September 1992.

[26] D. H. Lorenz and A. Orda. QoS Routing in Networks with Uncertain Parameters. In *IEEE Infocom*, 1998.

[27] Q. Ma and P. Steenkiste. On Path Selection for Traffic with Bandwidth Guarantees. In *IEEE International Conference on Network Protocols, ICNP*, 1997.

[28] Q. Ma and P. Steenkiste. Supporting Dynamic Interclass Resource Sharing: a Multi-Class QoS Routing Algorithm. In *IEEE Infocom*, 1999.

[29] D. Mitra, R. J. Gibbens, and B. D. Huang. Analysis and Optimal Design of Aggregated-Least-Busy-Alternative Routing on Symmetric Loss Networks. In *TELETRAFFIC AND DATATRAFFIC in a Period of Change, ITC-13*, pages 477–482. Elsevier Science Publishers B. V. (North-Holland), 1991.

[30] D. Mitra and J. B. Seery. Comprative Evaluation of Randomzied and Dynamic Routing Strategies for Circuit-Switched Networks. *IEEE Transactions on Communications*, 39(1):102–116, 1991.

[31] J. Moy. OSPF Version 2, April 1998. Internet RFC 2328.

[32] J. T. Moy. *OSPF - Anatomy of an Internet Routing Protocol*. Addison-Wesley, 1998.

[33] S. Nelakuditi, Zhi-Li Zhang, and R. P. Tsang. Adaptive Proportional Routing: A Localized QoS Routing Approach. In *IEEE Infocom*, 2000.

[34] K. Nichols, V. Jacobson, and L. Zhang. A Two-bit Differentiated Service Architecture. IETF Internet Draft, Diffserv Working Group, <draft-itef-diffserv-arch-02.txt>, October 1998.

[35] A. Orda and R. Guérin. QoS Routing: the Precomputation Perspective. In *IEEE Infocom*, 2000.

[36] T. J. Ott and K. R. Krishnan. State Dependent Routing of Telephone Traffic and the Use of Separable Routing Schemes. In *TELETRAFFIC SCIENCE in an Advanced Information Society, ITC-11*, pages 867–872. Elsevier Science Publishers B. V. (North-Holland), 1985.

[37] S. Shenker R. Braden, D. Clark. Integrated services in the internet architecture: an overview, June 1994. Internet RFC 1633.

[38] K. K. Ramakrishnan and R. Jain. A Binary Feedback Scheme for Congestion Avoidance in Computer Networks. *ACM Transactions on Computer Systems*, 8(2):158–181, 1990.

[39] H. F. Salama, D. S. Reeves, and Y. Vinotis. A Distributed Algorithm for Delay Constrained Unicast Routing. In *IEEE Infocom*, 1997.

[40] A. Segall, P. Bhagwat, and A. Krishna. QoS Routing Using Alternate Paths. *Journal of High Speed Networks*, 7(2):141–158, 1998.

[41] A. Shaikh, J. Rexford, and K. Shin. Evaluating the overheads of source-directed quality-of-service routing. In *IEEE International Conference on Network Protocols, ICNP*, 1998.

[42] S. Shenker and C. Partridge. Specification of guaranteed quality of service, July 1995. Internet Draft, IETF Integrated Services WG.

[43] W. Simpson. IP in IP Tunneling, October 1995. Internet RFC 1853.

[44] I. Stoica and H. Zhang. LIRA: A Model for Service Differentiation in the Internet. In *NOSSDAV'98*, 1998.

[45] MCI WorldCom and NSF's very High Speed Backbone Network Service: http://www.vbns.net/.

[46] Z. Wang and J. Crowcroft. Quality-of-Service Routing for Supporting Multimedia Applications. *IEEE Journal on Selected Areas in Communications*, 14(7):1228–1234, 1996.

[47] W. Willinger, M. S. Taqqu, R. Sherman, and D. V. Wilson. Self-Similarity through High-Variability: Statistical Analysis of Ethernet LAN Traffic at the Source Level. *IEEE/ACM Transactions on Networking*, 5(1):71–86, 1997.

[48] Daniel Zappala. Alternate Path Routing for Multicast. In *IEEE Infocom*, 2000.

[49] Zhi-Li Zhang et al. Quality of Service Extension to OSPF. IETF Internet Draft, <draft-zhang-qos-ospf-01.txt>, September 1997.