Available at
www.ComputerScienceWeb.com
POWERED BY SCIENCE DIRECT

Computer Communications 26 (2003) 834–844

computer communications

www.elsevier.com/locate/comcom

NetLets: measurement-based routing daemons for low end-to-end delays over networks☆

Nageswara S.V. Rao[a,1], Young-Cheol Bang[b,*], Sridhar Radhakrishnan[c], Qishi Wu[d], S. Sitharama Iyengar[d], Hyunseung Choo[e]

[a]*Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831, USA*
[b]*Routing Protocol Team, Network Technology Laboratory, EIRI, Daejun], South Korea*
[c]*School of Computer Science, University of Oklahoma, Norman, OK 73019, USA*
[d]*Department of Computer Science, Louisiana State University, Baton Rouge, LA 70803, USA*
[e]*School of ICE, Sungkyunkwan University, Suwon, South Korea*

Received 7 August 2002; accepted 7 August 2002

Abstract

Routing in the Internet is based on the best-effort mechanism, wherein the routers generally forward packets to minimize the number of hops to the destination. Furthermore, all packets of a type are treated the same independent of their size. We propose the framework of NetLets to enable the applications to send data packets to the destination with certain guarantees on the end-to-end delay. NetLets employ in situ instruments to measure the effective bandwidth and propagation delays on the links, and compute the paths with minimum measured end-to-end delay for data packets of various sizes. Based on experiments over local area networks, the paths selected by NetLets indeed achieve the minimum end-to-end delay, and our method outperformed the best-effort mechanism based on the hop count. We also describe an implementation of NetLets over the Internet to illustrate their viability for wide-area networks.

Keywords: End-to-end delays; Network instrumentation and measurements; Quality of service; Performance evaluation; NetLets

1. Introduction

Routing is a critical component in determining the end-to-end performance of messages in distributed computing applications over the Internet or a specialized wide-area network. In the current operational networks, routing protocols embedded in routers support various metrics such as link-delay and link-bandwidth to construct the routing table. However, they generally attempt to minimize the number of hops to the destination by statically setting link-delay as 1 for the fast computation of the routing table and the easy management, and do not take advantage of viable alternate routes with higher bandwidths or less congestion when the routing tables are computed. Instead, alternate paths are provided relying on routing polices or network administrator for the purpose of load balance in common. Furthermore, one of the reasons for the unsatisfactory end-to-end path performance of the present Internet is the inadequate measurement infrastructure. Paxson [9] points out the need for infrastructure to diagnose performance problems, measure the performance of network paths, and to asses performance of different Internet service providers. In this paper, we employ in situ measurements that do not require the infrastructure but are sufficient for computing the minimum end-to-end delay paths.

Quality-of-Service (QoS) guarantees are currently needed by a variety of network applications such as video conferencing, real-time control of instruments over the networks, and distributed simulations. To facilitate the growing number of such applications, new protocols and

☆ A preliminary version of this paper with a coarser bandwidth estimation method and no Internet implementation has been presented at the International Conference on Networking, Colmar, France, 2001.

* Corresponding author.
E-mail addresses: ybang@kpu.ac.kr (Y.C. Bang), raons@ornl.gov (N.S.V. Rao).

[1] Research of Rao is sponsored by Defense Advanced Research Projects Agency under MIPR No. K153, and by the Laboratory Director's Research and Development Program of Oak Ridge National Laboratory, managed by UT-Battelle, LLC for the US Department of Energy. In addition, the research of Rao Radhakrishnan, Wu and Iyengar is sponsored by the Engineering Research program of the Office of Science, US Department of Energy.

0140-3664/03/$ - see front matter © 2002 Elsevier Science B.V. All rights reserved.
PII: S0140-3664(02)00217-7

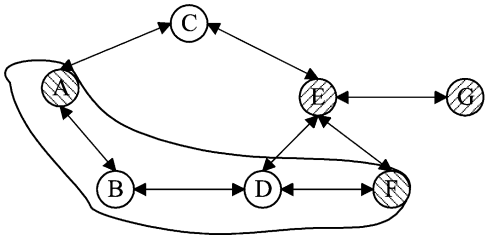


Fig. 1. The shaded nodes house NetLets. A virtual link from node A to node F corresponds to the path A–B–D–F in the underlying network.

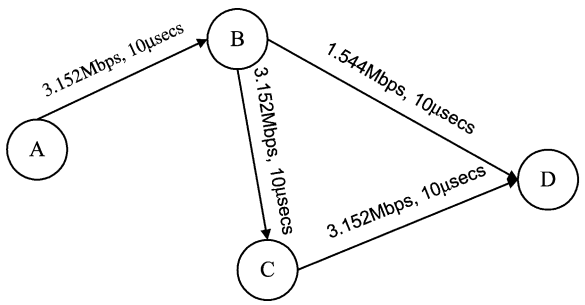
infrastructures are being developed (Integrated Services, Differentiated Services, and Multi Protocol Label Switching [6,7,20]) for ensuring certain types of end-to-end performance. Despite the proliferation of such technologies, it is pragmatic to assume that not all parts of the wide-area networks will be replaced by these newer methods. Thus, there is a need for methods that provide QoS without requiring significant infrastructure enhancements in the near term but are upward compatible to the next generation technologies. We propose such method by incorporating daemons, called the *NetLets*, above the existing network layer to achieve low end-to-end delays compared to what is possible in the current Internet. NetLets will be deployed on certain intermediate sites such as free telnet sites and routers (when possible) and also on host machines. In grid and distributed computing environments, NetLets will be deployed on various processes and grid points, respectively. The NetLets form a virtual network such that any two of them are connected via a virtual link if they are connected by an Internet Protocol (IP) path containing zero or more routers of the underlying network. In Fig. 1 an example of the NetLets virtual network overlaid on a real network is shown.

The end-to-end delay experienced by a packet over a network depends on its size, and but the best effort routing methods based on minimum number of hops are insensitive to the packet size. We now illustrate that the packet size has a strong influence on the end-to-end delay of routing paths. It is indeed possible for the same source and destination pairs to have different minimum end-to-end delay paths for packets of different size. The size of an IP packet can vary from a minimum of 41 bytes (20 bytes of TCP header, 20 bytes of IP header and 1 byte of data) to a maximum of 65,535 bytes (40 bytes of header and 65,515 bytes of data). Consider the network shown in Fig. 2. The nodes A and D are the source and destination, respectively. The bandwidths on the links have been chosen to be either DS1 or DS1C and all links are assumed to have a propagation delay of 10 μs. Then, to a first order of approximation, the end-to-end delay of a link is $\sigma/B + d$, where σ is the size of the data packet, B is the bandwidth on the link and d is the propagation delay of the link. In the table in Fig. 1, the end-to-end delay of the routes for $A–D$ for various packet sizes are shown. Clearly, for a minimum sized packet one should choose the path $A–B–D$, while for the maximum sized packet one should choose the path $A–B–C–D$. At the intermediate nodes, we have not included the queuing and processing delays, collectively called the node delays. The maximum sized packet will choose $A–B–D$ if the node delay at C is greater than 7 ms. The node delays are often much harder to estimate unlike the bandwidth and link delays. The node delays can be handled by utilizing the probabilistic components to the end-to-end delays to derive similar conclusions but with certain probability [13,16].

As evident from the above example, routing methods that ignore the available bandwidth and propagation delay will produce paths with sub-optimal end-to-end delays in general. We present the NetLet framework that accounts for the bandwidth and propagation delays, wherein:

(a) software in situ instruments are used to measure the bandwidth and propagation delay on the links;
(b) bandwidth and delay estimates are used to compute the minimum end-to-end delay paths; and
(c) interfaces and routing daemons are provided for the applications to route data packets along the computed paths.

Analytical basis for this approach is provided in Refs. [13,17], where it was shown that measurements are sufficient (with a specified probability) to compute the paths with the minimum end-to-end delay, irrespective of delay distributions. It is important to note that it is not necessary to derive delay distributions for providing this type of end-to-end delay guarantees, and one can achieve



	End-To-End Delay from Node A to D	
Packet Size (bytes)	**A-B-D Path**	**A-B-C-D Path**
21	182μsecs	190μsecs
65535	506msecs	499msecs

Fig. 2. A network with A and D as the source and destination, respectively. The bandwidth and propagation delays on the links are indicated. The table indicates the end-to-end delays for minimum and maximum sized IP packet using different paths.

useful performances based on simple in situ measurements alone.

We compare the paths chosen by NetLets with those chosen by the best-effort mechanism in an experimental setup of workstations that route data packets using the Internet Protocol. In the experiments, the paths selected NetLets based on the measurements indeed achieve the minimum end-to-end delay. Our method outperformed the best-effort mechanism and in this sense, our results provide an experimental evidence for the analytical results in Refs. [13,17]. Furthermore, NetLets installed on the nodes of the network, provide an environment for applications such as distributed simulation and grid computation, wherein the end-to-end delay minimization is performed in a transparent manner. In the experimental setup, the workstations are located over a local-area network to enable controlled experimentation. We also present an implementation over the Internet to illustrate that the NetLets represent a viable technology for wide-area networks as well.

In Section 2, we briefly compare our approach to existing ones. Estimation of bandwidth and propagations delays is described in Section 3. The overall framework of NetLets is described in Section 4. Experimental results are described in Section 5, and the details of Internet implementation are provided in Section 6.

2. Related research

Research on the end-to-end performance of path selection protocols has received much attention in recent years. For example, Savage et al. [18] in their Detour system point out various performance problems that the Internet suffers including the inefficiencies both in the routing and transport layer protocols. Collins [5] also suggests the formation of a virtual network similar to the one proposed in this paper. There are many differences between our work and the Detour system. First, the Detour system does not explicitly minimize end-to-end delay between source and destination pairs. Second, the measurements mechanism suggested by the Detour system involves the use of utilities such as *traceroute* and *ping*. Our measurements of link bandwidth and link propagation delay are obtained using the actual transport layer segments, namely Transport Control Protocol (TCP) segments. That is, our measurements correspond to the transport layer units that are actually routed on the network rather than control data packets sent by traceroute and ping programs. With the deployment of firewalls, the Internet Control Message Protocol (ICMP) traffic is often treated differently from TCP or User Datagram Protocol (UDP) traffic; for example, some firewalls disable response to ping and provide no or misleading response to traceroute. The TCP-based measurement scheme used here is vital to proving the analytical guarantees of Ref. [17], and no such guarantees can be given if only traceroute and ping measurements are used. Third, unlike the Detour system we do not modify the existing IP layer but direct the IP layer to forward the packets via the other NetLets.

Paxson [9] has developed measurement tools for determining the end-to-end delay, packet loss, and actual routing paths on the Internet. The RFC 2330 [8] examines the framework for measuring IP performance metrics. Most of this research points out the sensitivity of the measurements with respect to the changes in clock. For example, in a time synchronized distributed system, the one-way link time can be measured by time-stamping a packet before sending and time-stamping it after it reaches the destination. Since such assumptions are non-plausible in a distributed system, Paxson [9] suggest techniques to adopt to arrive at a reasonably accurate packet delay measurements. Savage et al. [19] introduce mechanisms to find the best alternative paths that is the most closely related to our work. Their protocols are based on measurements of path quality, such as round-trip time, loss rate, and bandwidth between pairs of Internet hosts, and also use the traceroute for measurements. For the minimum delay routes, their works do not consider the packet size that is a very important factor for the end-to-end delay as shown in Fig. 2. While measurements in general are motivated by several purposes, such as traffic modeling and fault diagnosis, the goal of our measurements is limited to reducing the end-to-end delays with respect to packet sizes.

IP routing involves designating groups of nodes (routers and hosts) to form an Autonomous System (AS). The entire system of routers and hosts are grouped into different ASs and communication between any two nodes within an AS is carried out using the Interior Gateway Protocol (IGP). The Border Gateway Protocol (BGP) treats each AS as a single node and provides the routing protocols between two nodes that are in different ASs. Both IGP and BGP [10] use the hop count as a performance measure to determine end-to-end path; that is, given two paths between source and destination pairs, a path that has fewer links will be chosen.

3. Estimation of bandwidth and propagation delay

There are two basic routing mechanisms: circuit switching (also called *pipelining*) and packet switching based on *store-and-forward* method. In circuit switching, data streams are transferred at a fixed rate from the source to destination without buffering. Over the packet switched networks, entire data is stored at every intermediate node before forwarding to next node. The telephone networks belong to circuit switching, and the IP networks belong to packet switching paradigm. For path P, the *path-delay* $D(P)$ is the sum of all propagation delays of links along the path. Let the *end-to-end delay* of a path P can be computed by the formula $T = \sigma/B(P) + D(P)$ in the circuit switching, where $B(P)$ is the *path-bandwidth*

which is appropriately computed based on the routing mechanism. Since in circuit switching the data transfers along the route is with a fixed rate, $B(P)$ is the minimum bandwidth of link along the path. In the case of packet switching, since the incoming data is stored temporarily at each node and then transmitted to outgoing link, transmission time of $\sigma/B(e)$ is required at each node v, where e is the outgoing link of v. Thus, for packet switching mechanism, the path-bandwidth $B(P)$ is $1/\sum_{e\in P} 1/B(e)$, where P is the routing path and e is the link on P [15]. The routing mechanism has a significant effect on the end-to-end delay and must be considered in computing the optimal end-to-end delay paths.

The well-known *quickest path problem* deals with computing a path in a network G to minimize the end-to-end delay to send σ units of message from a source to a destination. Chen and Chin [4], Rosen et al. [11], Rao and Batsell [12–14], and Bang et al. [2] studied the quickest path problem using the circuit-switching mode. However, since the store-and-forward transfer mode is used to send message in the Internet, classical quickest path algorithm cannot be adapted to an IP network. To facilitate the notion of the quickest path to packet switching, we apply the idea of classical quickest path to each link to minimize the end-to-end delay of the routing path. Thus, we take into account both bandwidth and propagation delay of links to minimize the end-to-end delay. We compare the method based on the minimum number of hops with our method based on minimum end-to-end delay by utilizing the estimated bandwidths and propagation times of links.

We measured the bandwidth of a link (a, b), which can be a virtual link or a physical link in the underlying network, using the following steps:

1. Generate a set of messages with various sizes for NetLet measurement at node a;
2. For each size s, construct a number of TCP segments of Maximum Segment Size (MSS) and send them to node b. Normally, the segmentation of MSS and fragmentation of MTU is internally performed by TCP layer and IP layer, respectively.
3. Node b upon receiving each TCP segment simply echo's it back to node a.
4. The round-trip delay is measured for each message size for a couple of times and the average is computed. The end-to-end delay is the round-trip delay divided by two. We also measure the one-way segment delay by utilizing the difference in the clocks of machines a and b as discussed at the end of this section.

Once the average end-to-end delay is determined for messages of different sizes, the linear regression (as suggested in Refs. [13,16], where other regression estimation methods were also studied) is applied to determine a first-order approximation of the bandwidth. The slope of the line, that fits the measured pairs of message size and end-to-end delay, yields the bandwidth estimator. The regression estimate for bandwidth is given by the Lemma 1, whose proof is given in Appendix A.

Lemma 1. *Given a set of test messages with various sizes* $S = \{s_i | i = 1, 2, \ldots, k\}$, *the corresponding end-to-end delays (over one virtual or physical link) are measured at times* $T = \{t_i | i = 1, 2, \ldots, k\}$. *The following formula gives the coefficient vector of a polynomial regression estimate in the Least Squares sense.*

$$\vec{a} = (X^{\mathrm{T}}X)^{-1}(X^{\mathrm{T}}\vec{y})$$

where, $\vec{a}$ *is the coefficient vector of a polynomial regression estimate:* $t = a_{n-1}s^{n-1} + a_{n-2}s^{n-2} + \ldots + a_1 s + a_0$. *Column vector* $\vec{y} = T$, *and matrix*X is constructed as follows:

$$X = \begin{bmatrix} s_1^{n-1} & s_1^{n-2} & \cdots & s_1 & 1 \\ s_2^{n-1} & s_2^{n-2} & \cdots & s_2 & 1 \\ \vdots & & & & \\ s_k^{n-1} & s_k^{n-2} & \cdots & s_k & 1 \end{bmatrix}$$

Fig. 3 illustrates the end-to-end delay measurements between two NetLets deployed at the Louisiana State University (LSU) and Oak Ridge National Laboratory (ORNL), respectively as well as the corresponding regression estimate. The round trip delay measurement for each message size is carried out three times and the end-to-end delay plotted in the figure is the half of the average value. From this computation, we obtain the 'effective bandwidth' B of this link as 1.038 Mbps. To determine the propagation delay (the time for a minimum size message to travel along the link (a, b)), we sent a minimum size message containing just 1 byte several times and averaged it. The actual size delivered between two nodes is 41 bytes to account for both 20 bytes of the TCP header and 20 bytes of the IP header attached to

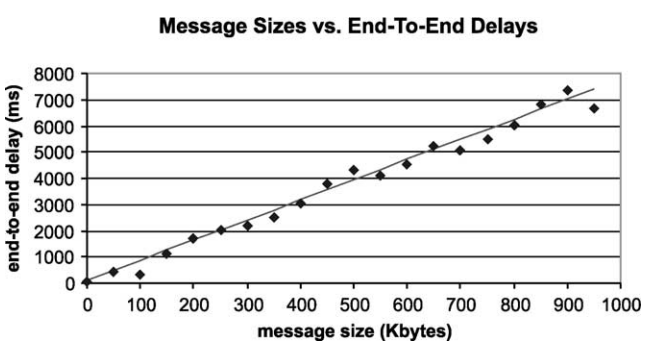


Fig. 3. Determination of the end-to-end delay experienced by transport layer segments of various sizes.

the message. However, it should not make too much difference as long as the actual size is less than the path Maximum Transmission Unit (MTU), which is the smallest MTU in the path between two nodes.

In the measurements we addressed the issues relating to clock and round-trip delay calculation as follows. It has been observed by several researches the end-to-end time cannot be measured accurately by determining the round-trip time, as the forward and return paths could be different. One-way packet delay is the time required for a packet to reach the destination and it is measured by comparing the time reported by a clock at the source node and the time it reached the destination node. If the clocks are not synchronized, knowing the difference in the timings between the two clocks will enable one to determine the correct one-way round trip delay. Yet another issue relates to the time interval and duration during which the measurements are obtained. All our measurements were carried out at regular intervals of time during 24 h of a day and results were aggregated. In order to use the NetLets, the measurements were collected at regular intervals and the bandwidths and propagation delays were updated from time to time.

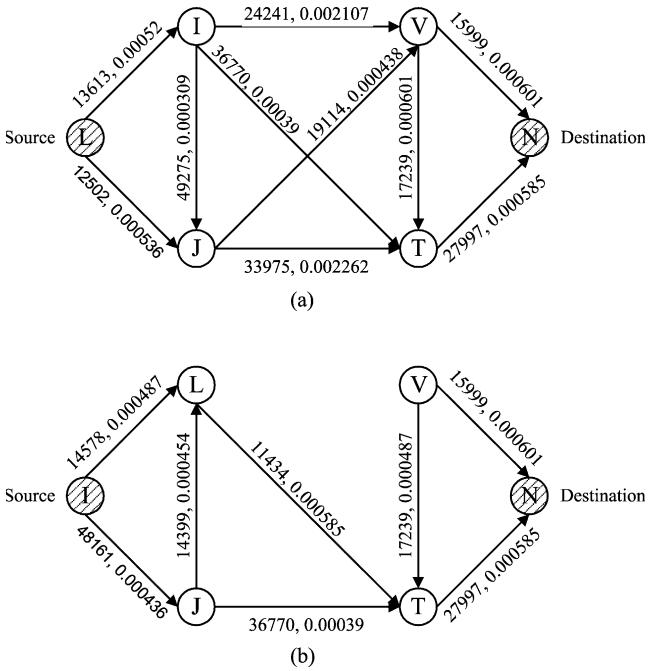


Fig. 4. (a) Virtual topology 1 containing six nodes with bandwidth in bytes per second and propagation delay in seconds indicated on the links. (b) Virtual topology 2 containing six nodes with bandwidth in bytes per second and propagation delay in seconds indicated on the links.

4. NetLets framework

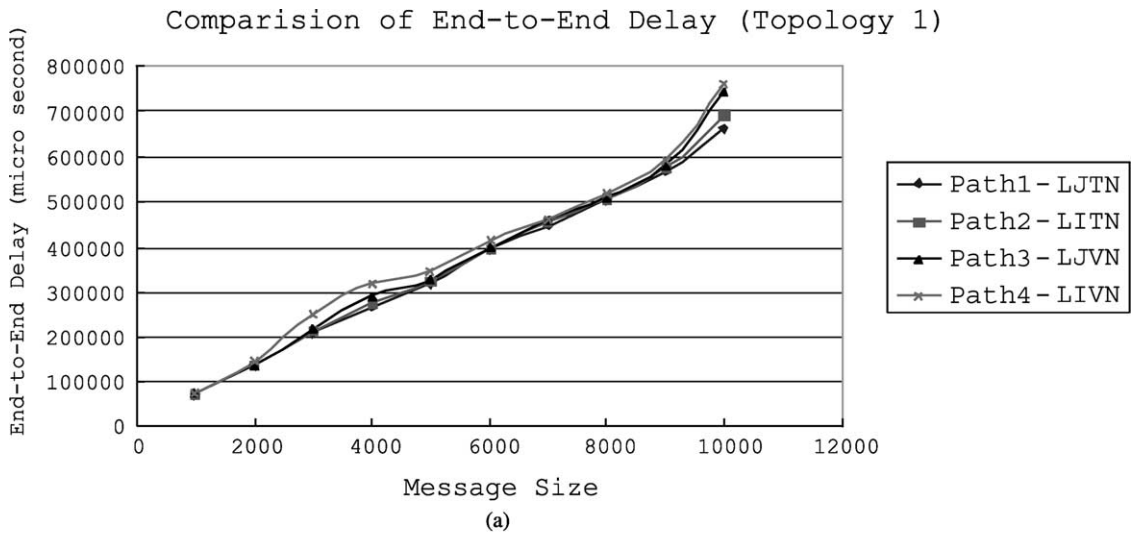
NetLets provide a software interface that the applications use to route messages via the minimum end-to-end paths (more details can be found in Ref. [15] on the overall framework of NetLets). Based on the bandwidth B and the propagation delay D of each link in the virtual network, the minimum end-to-end delay path for a message of particular size σ is determined by assigning to each link a weight of $(\sigma/B) + d$, and computing the shortest path in the resulting network using the one-to-all Dijkstra's shortest path algorithm [3,14]. Clearly, given any source and a set of messages of sizes $\sigma_1, \sigma_2, \ldots, \sigma_k$ we can construct the shortest path trees with respect to each message size and if two trees are the same we can eliminate one of them. Once the shortest path trees are known, the routing tables are constructed. We perform this operation for all nodes that house NetLets in the network.

The routing table contains for each destination the next hop IP address of the node containing the NetLets software. All NetLets software modules communicates using predetermined port numbers. The NetLet after receiving a message from local application sends the datagram using the designated port number and the IP address in the routing table. Of course, we can also establish a TCP connection instead of a UDP connection whenever reliability is an important factor that dictated by the application. NetLet upon receiving the message determines if the packet is destined for it or for any nodes that it can route and performs the necessary actions.

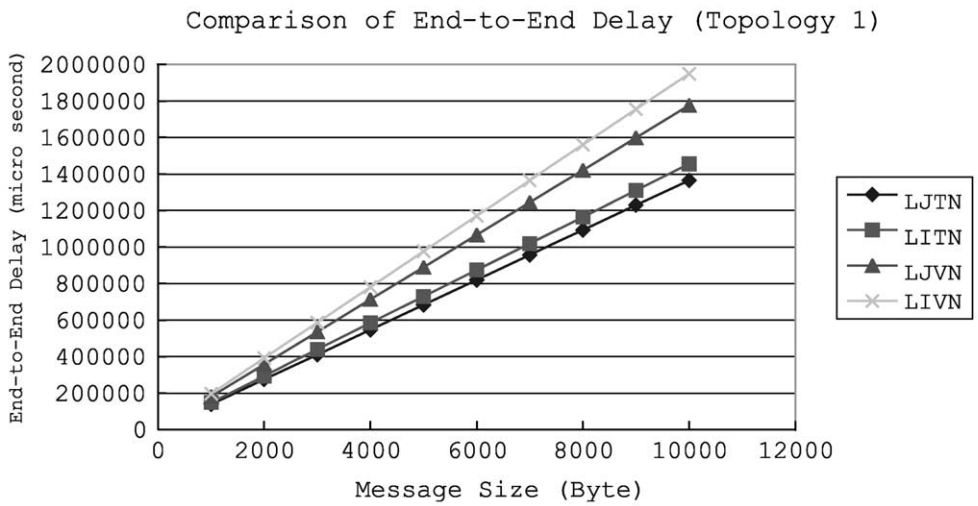
5. Experimental results for local area networks

For the purposes of experimentation we selected a number of nodes in the Internet and constructed two virtual network topologies as shown in Fig. 4(a) and (b). As evident in topology 1 (Fig. 4(a)), there are four different paths from source to destination node with the minimum number of hops, namely 4. Using the link weight of $((\sigma/B) + D)$ one each link, the minimum end-to-end delay path from source L to the destination N is $L \rightarrow J \rightarrow T \rightarrow N$. Also, the paths, $L \rightarrow J \rightarrow T \rightarrow N$, $L \rightarrow I \rightarrow T \rightarrow N$, $L \rightarrow J \rightarrow V \rightarrow N$, and $L \rightarrow I \rightarrow V \rightarrow N$, are paths with minimum number of hops. For topology 2 (Fig. 4(b)), paths with minimum number of hops are $I \rightarrow J \rightarrow T \rightarrow N$, $I \rightarrow J \rightarrow V \rightarrow N$. and $I \rightarrow L \rightarrow T \rightarrow N$. where I and N are source and destination, respectively. The minimum delay path is $I \rightarrow J \rightarrow T \rightarrow N$ which is also has a minimum number of hops.

With respect to the network topology 1, Fig. 5(a) represents the observed end-to-end delay for paths computed by sending messages of different sizes, and Fig. 5(b) shows end-to-end delay obtained by our method (namely, the summation of the link weights on each of the paths). Even though the observed end-to-end delay is



(a)



(b)

Fig. 5. (a) For virtual topology 1 this graph contains the observed end-to-end delay for various paths from source to destination. (b) For virtual topology 1 this graph contains the calculated end-to-end delay for various paths from source to destination.

different in magnitude from the calculated end-to-end delay, the two methods have chosen the same path from the source to the destination as the minimum end-to-end delay path. It is an important observation, since it shows that a complete traffic characterization is not necessary to compute paths with end-to-end delay guarantees (as is also analytically shown in Refs. [5,16]). The differences between the times are attributed to the intermediate node delays. For network 2, Fig. 6(a) and (b) represent the observed and computed end-to-end delay, respectively. Again, while the computed and observed delays are different, both methods have chosen the same minimum end-to-end delay path.

In Fig. 7 we show the observed and computed end-to-end delay. While there are clear differences in delay times, the path chosen by computation and experimentation are exactly the same minimum end-to-end delay path.

6. NetLet implementations over the internet

We installed the NetLet daemons on 5 sites over the Internet at LSU, ORNL, Purdue University, University of Florida (UFL), and University of Oklahoma (OU). The network topology of the deployed NetLet daemons is shown in Fig. 8. Essentially, any two computers on the Internet are connected via a certain number of intermediate routers, so that we abstract the real network connection as a complete graph of 5 nodes and 10 edges.

A NetLet daemon is implemented in C++ in Unix/Linux. In Fig. 9, we show the main framework of

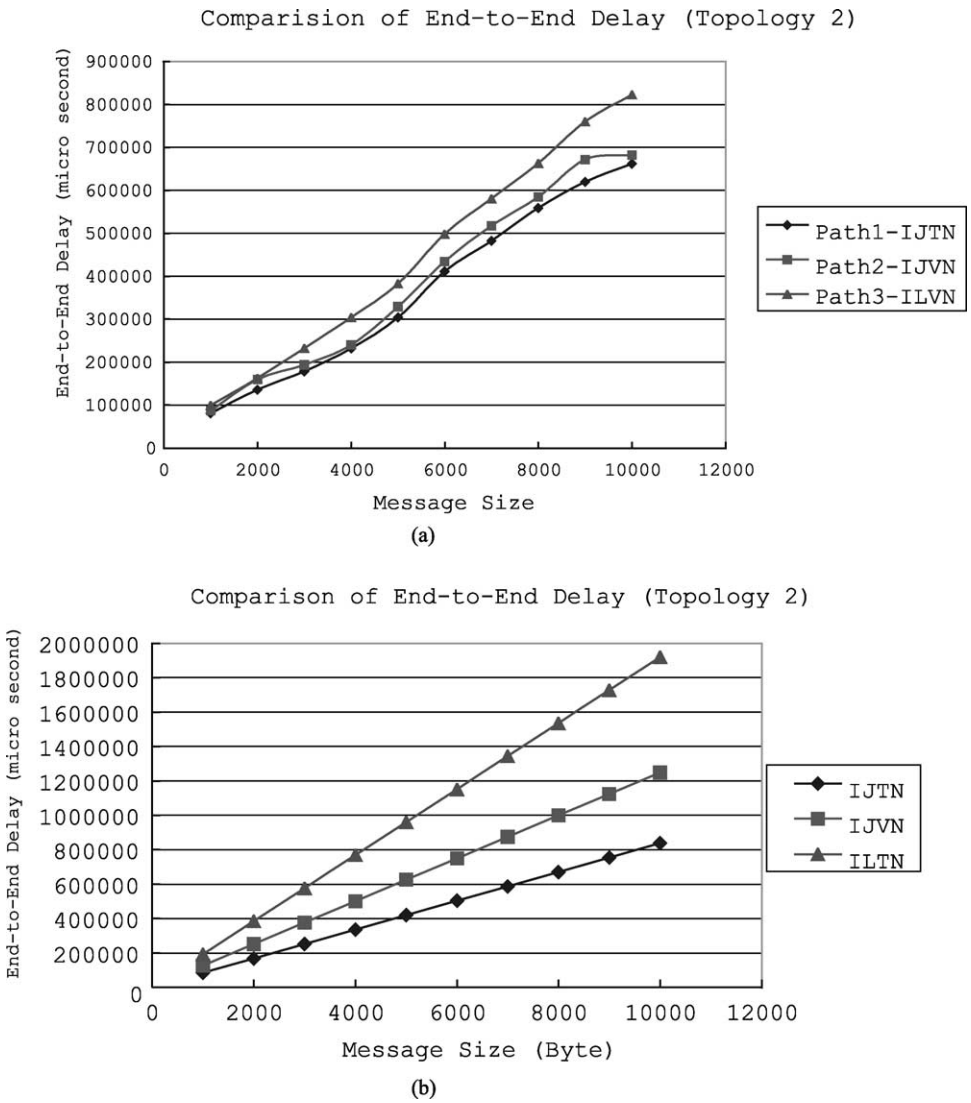


Fig. 6. (a) For virtual topology 2 this graph contains the observed end-to-end delay for various paths from source to destination. (b) For virtual topology 2 this graph contains the calculated end-to-end delay for various paths from source to destination.

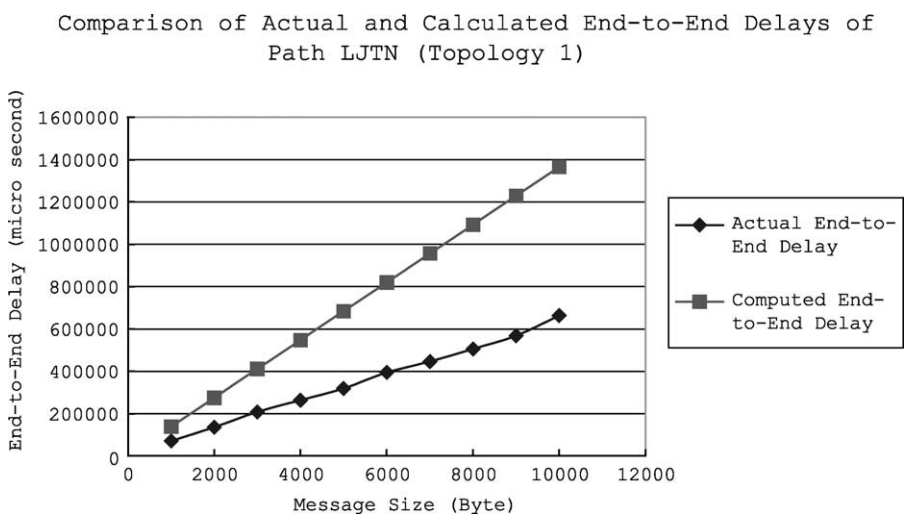


Fig. 7. Comparison of observed (actual) and computed end-to-end delay of the path LJTN in topology 1.

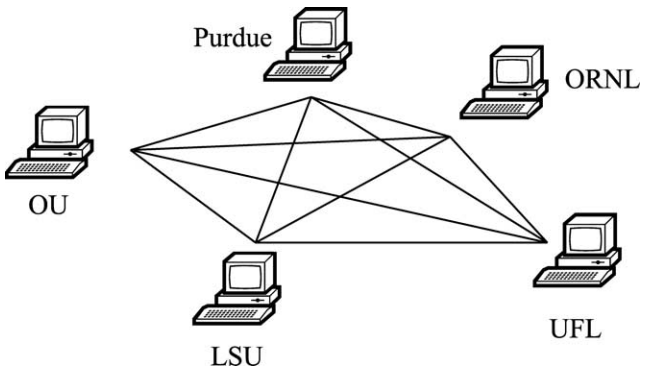


Fig. 8. The Network topology of the NetLet Daemons deployed over Internet.

the functionalities of a NetLet daemon. The NetLet daemon has one interface with the host process for data delivery and three interfaces with the other active NetLet daemons for end-to-end delay measurement, link information exchange as well as data receive/forward. Based upon the measurement minimization results, we activate a selected set of the deployed daemons while keep the rest asleep in order to reduce the computation and communication traffic. Once activated, the NetLet daemons perform end-to-end delay measurements at the optimized rates for a set of various message sizes. The 'effective bandwidth' of each link is determined by the regression estimator as discussed in Section 3. The measured and estimated link information like bandwidth, link propagation delay, and queuing delay is then broadcasted to all the other NetLet daemons in order to build a complete topology of the whole daemon network on each local node. Finally, the quickest path algorithm is applied to compute the routes for a given data size to be transmitted by the host process.

The NetLet we implemented above is programmed to have the ability of adjusting the socket options such as send/receive buffer and TCP MSS, etc. in order to achieve better networking performance. Some of the socket options may have significant impact on the networking performance, depending upon the transmission distance, message size, and effective bandwidth

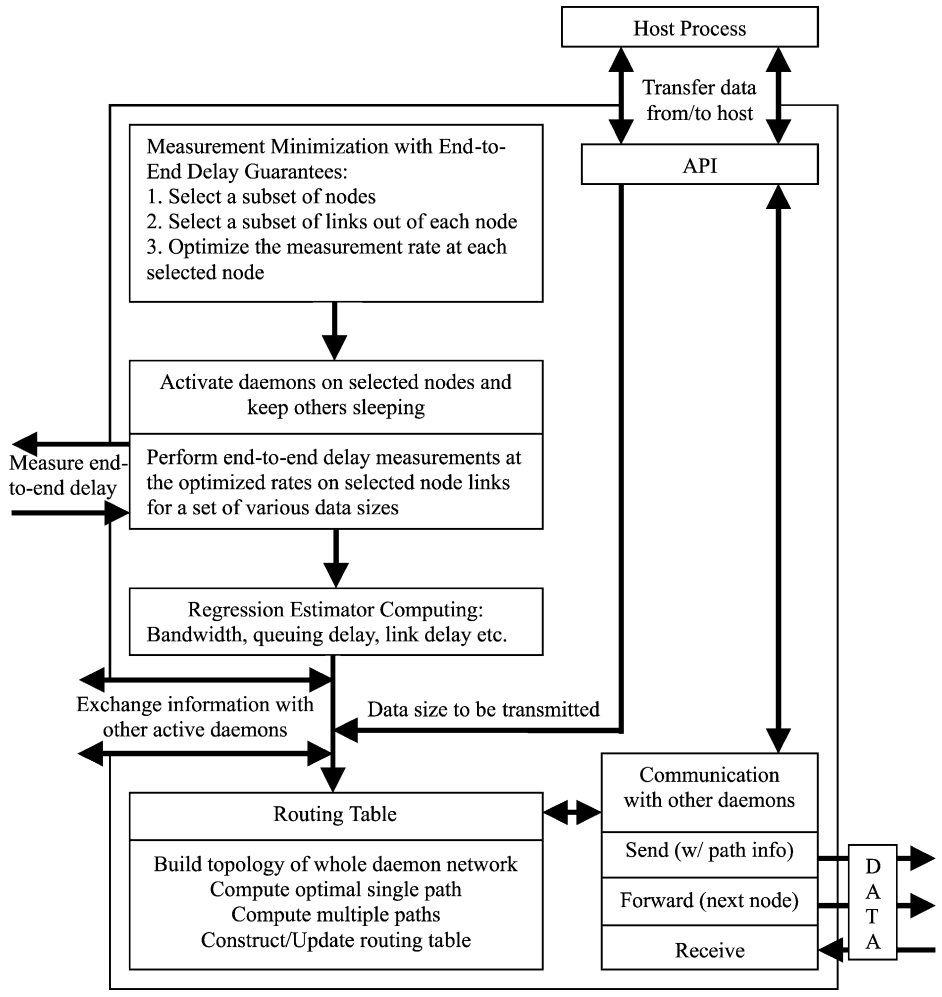


Fig. 9. The main framework of the functionalities of a NetLet daemon.

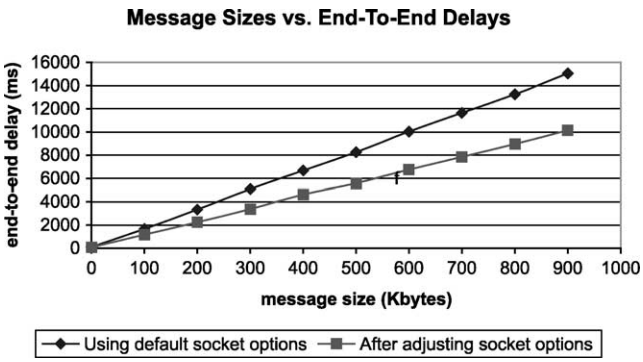


Fig. 10. The impact of socket options on the networking performance.

etc. The performance comparison between the NetLet of the default socket options and the one of the adjusted socket options is shown in Fig. 10, which demonstrates that a properly tuned socket send/receive buffer has successfully reduced the end-to-end delay by about one third. All the measurements participating in the comparison are collected from the NetLets deployed at LSU and ORNL.

7. Conclusion

We presented experimental results on the end-to-end delays of network paths, comparing the default method based on minimum hops with our measurement-based method that computes minimum end-to-end delay paths. Our results showed that the minimum end-to-end delay path computed based on the estimated bandwidth and delays outperformed the path with minimum number of hops. This is because the end-to-end delay depends on the message size, propagation delay, and bandwidth, which are not accounted for by the minimum hop method We also proposed a software framework based on NetLets that can be used for routing with lower end-to-end delays.

Future work could involve utilizing other regression estimation methods such as Nadaraya-Watson estimator, and feedforward sigmoidal network, which were analytically validated in Refs. [13,17]. Also, in terms of experimentation using Internet, NetLets were used in Ref. [16] for realizing two-paths between nodes that are geographically separated by thousands of miles. Our system can be expanded to incorporate more nodes and also by distributing the nodes more widely over the Internet. NetLets are complementary and upward-compatible with QoS mechanisms such as MPLS, DiffServe, and IntServe as well as adapted and/or optimized versions of transport mechanisms such as auto-tuned TCP [21]. It would be interesting to see if existing NetLets can be enhanced to exploit these mechanisms to provide end-to-end performance beyond what is possible in current IP networks. Also, active networks [1] can be exploited to provide more information to the NetLets by attaching measurement and routing code to messages.

Appendix A

Proof of Lemma 1. For the given observations, test message sizes S and measured end-to-end delays T, the coefficients of a polynomial regression estimate $t = a_{n-1}s^{n-1} + a_{n-2}s^{n-2} + \ldots + a_1 s + a_0$ are determined by the Least Squares method such that:

$$\varphi = \min\left(\sum_{i=1}^{k} r_i^2\right)$$

where the residual $r_i = a_{n-1}s_i^{n-1} + a_{n-2}s_i^{n-2} + \ldots + a_1 s_i + a_0 - t_i$, $(i = 1, 2, \ldots, k)$. By taking partial derivative with respect to the unknown coefficients, we have:

$$\begin{cases} \dfrac{\partial \varphi}{\partial a_0} = \displaystyle\sum_{i=1}^{k} 2(a_{n-1}s_i^{n-1} + a_{n-2}s_i^{n-2} + \cdots + a_1 s_i + a_0 - t_i) = 0 \\ \dfrac{\partial \varphi}{\partial a_1} = \displaystyle\sum_{i=1}^{k} 2(a_{n-1}s_i^{n-1} + a_{n-2}s_i^{n-2} + \cdots + a_1 s_i + a_0 - t_i)s_i = 0 \\ \vdots \\ \dfrac{\partial \varphi}{\partial a_{n-1}} = \displaystyle\sum_{i=1}^{k} 2(a_{n-1}s_i^{n-1} + a_{n-2}s_i^{n-2} + \cdots \\ \qquad + a_1 s_i + a_0 - t_i)s_i^{n-1} = 0 \end{cases}$$

By separating the unknowns and the observations on each side of the equation, the above equation group is rearranged as follows:

$$\begin{cases} a_{n-1}\displaystyle\sum_{i=1}^{k} s_i^{n-1} + a_{n-2}\sum_{i=1}^{k} s_i^{n-2} + \cdots + a_1\sum_{i=1}^{k} s_i + a_0\sum_{i=1}^{k} 1 = \sum_{i=1}^{k} t_i \\ a_{n-1}\displaystyle\sum_{i=1}^{k} (s_i^{n-1}s_i) + a_{n-2}\sum_{i=1}^{k} (s_i^{n-2}s_i) + \cdots + a_1\sum_{i=1}^{k} (s_i s_i) + \\ \quad a_0\displaystyle\sum_{i=1}^{k} (1 \cdot s_i) = \sum_{i=1}^{k} (t_i s_i) \\ \vdots \\ a_{n-1}\displaystyle\sum_{i=1}^{k} (s_i^{n-1}s_i^{n-1}) + a_{n-2}\sum_{i=1}^{k} (s_i^{n-2}s_i^{n-1}) + \cdots + a_1\sum_{i=1}^{k} (s_i s_i^{n-1}) \\ \quad + a_0\displaystyle\sum_{i=1}^{k} (1 \cdot s_i^{n-1}) = \sum_{i=1}^{k} (t_i s_i^{n-1}) \end{cases}$$

Then, we rewrite the above equation group in the form of matrix as follows:

$$\begin{bmatrix} \sum_{i=1}^{k}(s_i^{n-1}s_i^{n-1}) & \sum_{i=1}^{k}(s_i^{n-2}s_i^{n-1}) & \cdots & \sum_{i=1}^{k}(s_i s_i^{n-1}) & \sum_{i=1}^{k}(1\cdot s_i^{n-1}) \\ \vdots & & & & \\ \sum_{i=1}^{k}(s_i^{n-1}s_i) & \sum_{i=1}^{k}(s_i^{n-2}s_i) & \cdots & \sum_{i=1}^{k}(s_i s_i) & \sum_{i=1}^{k}(1\cdot s_i) \\ \sum_{i=1}^{k}s_i^{n-1} & \sum_{i=1}^{k}s_i^{n-2} & \cdots & \sum_{i=1}^{k}s_i & \sum_{i=1}^{k}1 \end{bmatrix} \cdot \begin{bmatrix} a_{n-1} \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ a_1 \\ a_0 \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{k}(t_i s_i^{n-1}) \\ \vdots \\ \sum_{i=1}^{k}(t_i s_i) \\ \sum_{i=1}^{k}t_i \end{bmatrix}$$

Finally, the coefficient vector $\vec{a}$ is obtained by multiplying the inverse of the above left-most squared matrix on both sides of the equation:

$$\vec{a} = \begin{bmatrix} a_{n-1} \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ a_1 \\ a_0 \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{k}(s_i^{n-1}s_i^{n-1}) & \sum_{i=1}^{k}(s_i^{n-2}s_i^{n-1}) & \cdots & \sum_{i=1}^{k}(s_i s_i^{n-1}) & \sum_{i=1}^{k}(1\cdot s_i^{n-1}) \\ \vdots & & & & \\ \sum_{i=1}^{k}(s_i^{n-1}s_i) & \sum_{i=1}^{k}(s_i^{n-2}s_i) & \cdots & \sum_{i=1}^{k}(s_i s_i) & \sum_{i=1}^{k}(1\cdot s_i) \\ \sum_{i=1}^{k}s_i^{n-1} & \sum_{i=1}^{k}s_i^{n-2} & \cdots & \sum_{i=1}^{k}s_i & \sum_{i=1}^{k}1 \end{bmatrix}^{-1} \cdot \begin{bmatrix} \sum_{i=1}^{k}(t_i s_i^{n-1}) \\ \vdots \\ \sum_{i=1}^{k}(t_i s_i) \\ \sum_{i=1}^{k}t_i \end{bmatrix}$$

$$= \left(\begin{bmatrix} s_1^{n-1} & s_1^{n-2} & \cdots & s_1 & 1 \\ s_2^{n-1} & s_2^{n-2} & \cdots & s_2 & 1 \\ \vdots & & & & \\ s_k^{n-1} & s_k^{n-2} & \cdots & s_k & 1 \end{bmatrix}^{\mathrm{T}} \cdot \begin{bmatrix} s_1^{n-1} & s_1^{n-2} & \cdots & s_1 & 1 \\ s_2^{n-1} & s_2^{n-2} & \cdots & s_2 & 1 \\ \vdots & & & & \\ s_k^{n-1} & s_k^{n-2} & \cdots & s_k & 1 \end{bmatrix} \right)^{-1} \left(\begin{bmatrix} s_1^{n-1} & s_1^{n-2} & \cdots & s_1 & 1 \\ s_2^{n-1} & s_2^{n-2} & \cdots & s_2 & 1 \\ \vdots & & & & \\ s_k^{n-1} & s_k^{n-2} & \cdots & s_k & 1 \end{bmatrix}^{\mathrm{T}} \cdot \begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_k \end{bmatrix} \right) = (X^{\mathrm{T}}X)^{-1}(X^{\mathrm{T}}\vec{y})$$

□

References

[1] Active networks, http://www.sds.lcs.mit.edu/activeware.

[2] Y.C. Bang, S. Radhakrishnan, N.S.V. Rao, S.G. Batsell, on update algorithms for quickest paths, Computer Communications 23 (11) (2000) 1064–1068.

[3] B.E. Carpenter, D.D. Kandlur, Diversifying Internet delivery, IEEE Spectrum November (1999) 57–61.

[4] Y.L. Chen, Y.H. Chin, The quickest path problem, Computers and Operations Research 17 (2) (1990) 153–161.

[5] A. Collins, The Detour Framework for Packet Rerouting, Technical Report, University of Washington, October, 1998.

[6] A. Dutta-Roy, The cost of quality in Internet-style networks, IEEE Spectrum September (2000) 57–62.

[7] G. Huston, Internet Performance Survival Guide, QoS Strategies for Multiservice Networks, Wiley, New York, 2001.

[8] V. Paxson, G. Almes, J. Mahdavi, M. Mathis, Framework for IP Performance Metrics, Network Working Group, RFC 2330, May, 1998.

[9] V. Paxson, End-to-End Routing Behavior in the Internet, IEEE/ACM Transactions on Networking 5 (5) (1997) 601–615.

[10] L. Peterson, B. Davie, Computer Networks: A Systems Approach, second ed., Morgan Kaufmann, Los Altos, CA, 2000.

[11] J.B. Rosen, S.Z. Sun, G.L. Xue, Algorithms for the quickest path problem and the enumeration of quickest paths, Computers and Operations Research 18 (6) (1991) 579–584.

[12] N.S.V. Rao, S.G. Batsell, Algorithm for minimum end-to-end delay paths, IEEE Communication Letters 1 (5) (1997) 152–154.

[13] N.S.V. Rao, S.G. Batsell, On routing algorithms with end-to-end delay guarantees, Proceedings of International Conference on Computer Communications and Networks (1998) 162–167.

[14] N.S.V. Rao, S.G. Batsell, QoS routing via multiple paths using bandwidth reservation, Proceedings of IEEE INFOCOM98: The Conference on Computer Communications 1 (1998) 11–18.

[15] N.S.V. Rao, W.C. Grimmell, S. Radhakrishan, Y.C. Bang, N. Manickam, Quickest paths for different network router mechanisms,

Proceedings of ninth International Conference on Advanced Computing and Communications, 2001.

[16] N.S.V. Rao, NetLets: End-to-end QoS mechanisms for distributed computing in wide-area networks using two-paths, Proceedings of International Conference on Internet Computing, pp. 475–478, ORNL Draft, 2001, http://saturn.epm.ornl.gov/~nrao.

[17] N.S.V. Rao, Probabilistic end-to-end delay guarantees in a class of networks, ORNL Draft, 2002, http://saturn.epm.ornl.gov/~nrao.

[18] S. Savage, T. Anderson, A. Aggarwal, D. Becker, N. Cardwell, A. Collins, E. Hoffman, J. Snell, A. Vahdat, G. Voelker, J. Zahorjan, Detour: Informed Internet Routing and Transport, IEEE Micro 19 (1) (1999) 50–59.

[19] S. Savage, A. Collins, E. Hoffman, J. Snell, T. Anderson, The end-to-end effects of Internet path selection, Proceedings of the 1999 AcM SIGCOMM Conference, Cambridge, MA, September, 1999, pp. 289–299.

[20] Z. Wang, Internet QoS: Architectures and Mechanisms for Quality of Service, Morgan Kaufmann, Los Altos, CA, 2001.

[21] Web100 concept paper, 1999, http://www.web100.org.