Available online at www.sciencedirect.com
SCIENCE DIRECT®
Performance Evaluation 57 (2004) 105–119
PERFORMANCE EVALUATION
An International Journal
www.elsevier.com/locate/peva

Using adaptive routing to achieve Quality of Service☆

Pu Su*, Michael Gellman

School of Electrical Engineering and Computer Science, University of Central Florida, Orlando, FL 32816, USA

Received 15 April 2003; received in revised form 27 September 2003

Abstract

Self-monitoring allows a network to observe its own behavior via probing and measurement mechanisms. This can then be exploited by the system to take autonomous decisions for the purpose of system management, performance management and user Quality of Service (QoS). In this paper, we experimentally explore how QoS goals that are externally set by network users, can then be explicitly exploited by a self-aware network to control its own behavior to attain these goals. The experiments we report are conducted in two distinct Cognitive Packet Network (CPN) test-beds which use probing to select the routes which best satisfy the QoS goal. Our experiments validate this concept with QoS goals which include end-to-end delay, packet loss, and a mixture of these two metrics. We observe that using only delay in the QoS goal is a good way to reduce delay and loss if losses result only from congestion. However, using loss in the QoS goal is seen to be useful if the paths that are adaptively selected avoid nodes where packet losses occur for reasons other than congestion. In general we observe that CPN networks effectively adapt routing behavior to the QoS goal that is specified.

Keywords: Quality of Service; Routing; Self-monitoring; Loss; Delay; Cognitive packet networks

1. Introduction

Broadly speaking, Quality of Service (QoS) is the performance of a network as perceived by some specific user or by a class of users [5]. Many authors have developed techniques for estimating QoS from given traffic characteristics (e.g. [3,6,7]), while others have studied routing schemes which try to achieve desired QoS objectives [8,18–21].

This paper describes experiments which investigate how self observation and on-line adaptation can be used to satisfy user specified QoS. The experiments are conducted in the "Cognitive Packet Network" (CPN) framework which has been described in several recent papers [9,11,13,14]. In [23], it is shown that the CPN can work seamlessly with the standard IP protocol, and that it can be used for traffic engineering and load balancing within an IP network.

☆ This work was supported by US Army Stricom via NAWC under Contract N61339-02-C0117 and by the National Science Foundation under Grant EIA0203446.
* Corresponding author.
E-mail addresses: psu@cs.ucf.edu (P. Su), michaelg@cs.ucf.edu (M. Gellman).

0166-5316/$ – see front matter © 2003 Published by Elsevier B.V.
doi:10.1016/j.peva.2003.10.002

The work presented here has been conducted on experimental test-beds which have been described in [16,22]. The concepts behind CPN were first developed to implement a large scale agent-based simulation system [12]. CPN uses neural network learning based on the Random Neural Network model discussed in [4,10,17].

CPN uses smart packets (SPs) and acknowledgement packets (ACKs) to continuously collect and store distributed state information in a packet switching network. Mailboxes in routers store data about paths in the network which has been collected by SPs and brought back by ACK packets, and subsequent SPs then use this information to search for better paths based on user specified QoS goals. Dumb packets (DPs) carry the user's payload packets from source to destination along paths that SPs have discovered.

This paper briefly describes the operating principles of CPN, and then develops QoS goals which combine both packet loss and delay. Experiments are then reported on two CPN test-beds in which QoS goals are used by SPs to discover paths which satisfy the QoS goal. Path discovery and adaptation is continuously carried out during a connection to best satisfy the user's QoS requirement.

Our measurements basically show that the CPN routing protocol allows the users' payload traffic, carried by DPs, to experience QoS specified by the users. We use two different CPN test-beds, one small and one medium sized, to make separate sets of measurements so as to strengthen our point.

2. CPN routing

The CPN protocol has been described in detail in [14,16,22]. In this section, we briefly summarize its principles. CPN includes three types of packets which play different roles:

- Smart or cognitive packets (SPs) are used to discover routes for connections; they are routed using a reinforcement learning (RL) algorithm [2] based on a QoS "goal". We use the term "goal" to indicate that there are no QoS guarantees, and that CPN provides best effort to satisfy the users' desired QoS. They do not carry payload.
- When a smart packet arrives to its destination, an acknowledgement (ACK) packet is generated and the ACK stores the route followed by the original packet and the measurement data it collected and will return along the "reverse route" of the SP. ACKs deposit information in the mailboxes (MBs) [13] of the nodes they visit.
- Dumb packets (DPs) carry payload and use source routing. Dumb packets also collect measurements at nodes. The route brought back to a source node by an ACK of an SP is used as a source route by subsequent dumb packets of the same QoS class having the same destination, until a new route is brought back by another ACK.

When an ACK for a packet which was going from S to D and was of class K enters some node N from node M, the following operation will be carried out: the difference between the local time-stamp and the time-stamp stored in the ACK for this particular node is computed and divided by 2. The resulting time is stored in the mailbox as the value $W(K, D, M)$—it is an estimate of the forward delay for a packet of QoS class K going from node N to D and exiting node N via the port leading to M. Note that the identity of the local node N is obvious and need not be stored. The source node S is also not relevant since the $W(K, D, M)$ refers to the time to go from N to D using the next node M. The QoS class K is needed since

the decision at each node, and the resulting observed delay, will depend on the requirements expressed by the QoS class K. The quantity $W(K, D, M)$ is inserted in the goal function (see Eq. (1) for the delay value W).

2.1. Reinforcement learning and routing

Different approaches to learning could in principle be used to discover good routes in a network, including Hebbian learning, back-propagation [1], and reinforcement learning (RL) [2]. Hebbian learning was proved to be slow in the simulation studies that were conducted at the beginning of the CPN project [9,11,12] and was excluded from further consideration. Simulation experiments conducted on a 100 node network [12,13] showed that RL is far more effective, and that it provides significantly better QoS than shortest-path routing [16]. In order to guarantee convergence of the RL algorithm to a single decision (i.e., selecting an output link for a given smart packet), CPN uses the Random Neural Network (RNN) [4,10,17] which has a unique solution to its internal state for any set of "weights" and input variables. CPN's RL algorithm uses the observed outcome of a decision to "reward" or "punish" the routing decision, so that future decisions are more likely to meet the desired QoS goal. The "goal" is the metric which characterizes the success of the outcome, such as packet delay, loss, jitter, or some composite metric.

As an example, the QoS goal G that SPs pursue may be formulated as minimizing Transit Delay W, or Loss Probability L, jitter, or some weighted combination captured in the numerical goal function G.

From G we construct the reward function $R = 1/G$. Then, successive values of R denoted by R_l, $l = 1, 2, \ldots$, are computed from the measured values of G (e.g. delay, loss, etc.): see Section 3), and are used to update the neural network weights. In [13,16] the RL algorithm that is used in CPN is described in more detail.

The performance of the CPN test-beds we use in this paper is presented in [22], while the use of CPN-based routing for traffic engineering for a web server is described in [23]. Some of the theoretical foundations of QoS driven routing are discussed in [24].

3. Constructing composite QoS goals

For an application which has QoS needs that can include both loss and delay, the QoS goal that may be used to route packets will have to combine in one single goal function both the loss and delay incurred from source to destination. In this case, we can form the goal function G as follows:

$$G = (1 - \bar{L}_{\mathrm{f}})\bar{W} + \bar{L}_{\mathrm{f}}(T + G), \tag{1}$$

where $\bar{W}$ is an estimate of the forward delay, $\bar{L}_{\mathrm{f}}$ an estimate of the forward packet loss ratio, and T is the additional time incurred by a packet which is retransmitted after a loss, including the time-out delay before a non-acknowledged (and presumably lost) packet is retransmitted, and any additional overhead resulting from the retransmission of the lost packet. The expression (1) is based on the idea that if a loss occurs, with probability $\bar{L}_{\mathrm{f}}$, then the resulting cost is the delay T until the packet is retransmitted, and this will be followed by the same equivalent total delay G incurred by the freshly retransmitted packet. If on the other hand a packet is not lost with probability $[1 - \bar{L}_{\mathrm{f}}]$, then the cost is simply the delay $\bar{W}$ that will be incurred by a packet as it traverses the network to reach its destination. Note that G appearing on both

sides of (1) is written under the assumption that the subsequent packet sent out to replace the lost packet will on the average incur the same total cost G, since it also may be lost and could be retransmitted. This expression simplifies to yield the reward $R = 1/G$:

$$R = \frac{1}{T(\bar{L}_f/(1-\bar{L}_f)) + \bar{W}}. \tag{2}$$

In order to use R we must obviously be able to estimate $\bar{W}$ and $\bar{L}_f$. In Section 2, we describe how ACK packets deposit an estimate of "delay to the destination" into the MBs of nodes that they visit. In order to select a particular path in the network based on composite path QoS metrics, CPN also needs to estimate path packet loss ratios defined as the number of packets sent but not received, divided by the number of packets which have been sent. We will discuss how to estimate link loss and path loss in the following section.

3.1. Estimating link loss and path loss

Packet loss ratios are simply the ratio of number of lost packets to the number of packets sent. The link loss ratio refers to the corresponding quantity measured over a single link connecting two nodes. Path loss ratio on the other hand refers to the quantity measured over a path, from a source to a destination. We will use the terms "cumulative" or "path" loss interchangeably. In CPN, we estimate the link loss ratio by forwarding, over the link and back to the predecessor node, the number of packets that have been received by the next node on the link. This information is in fact stored or "piggy-backed" in ACK packets. If N packets have been sent over a link and R packets have been received at the next node, then the loss ratio is

$$L = 1 - \frac{R}{N}. \tag{3}$$

To estimate the path loss ratio, we use ACKs coming back from the destination node. The source is able to estimate the round-trip loss ratio by keeping track of the number of packets sent and the number of ACKs it receives. However, in addition the destination can keep track of the number of packets which are received at the destination, and this number can be piggy-backed inside ACK packets and returned to the source. The time-stamp at the destination which is carried by the ACK, will allow the sender to estimate forward loss rates over a given period of time. Thus even if some ACK packets are lost, it is still possible to have a fairly accurate estimate at the source of forward (source to destination) path losses, and not of just round-trip losses. However, the loss ratio estimates at the source can be insensitive to *short-term* changes which are important to QoS driven adaptive routing. We address this problem in CPN by using the following scheme:

- The sender maintains a smoothed average of the packet loss ratio: $\bar{L} \leftarrow (1-a)\bar{L} + aL$.
- The receiver modifies R as follows for some threshold value of $R_{\max}$:

  if $R > R_{\max}$ then $R \leftarrow 0$.

- If R_l is lth value of R received at the sender, the sender carries out the following operation:

  if $R_{i+1} < R_i$ then $N \leftarrow R_{i+1}$.

As a result, large values of R are eliminated, while $\bar{L}$ preserves an accurate estimate of the loss ratio over the link from the sender's perspective.

3.2. Simplifying the cumulative loss

Since it is impractical to have the destination nodes keep a count of the number of packets received for each possible route from every possible source, we need to find a scheme that will reduce the amount of data that is stored. This requires us to make a simplifying assumption based on the idea that forward and reverse routes generally use the same set of nodes and links. Thus, we assume that the DP loss ratio L_f from the source S to the destination D is proportional to the ACK loss ratio L_b in the opposite direction or $L_\mathrm{b} = \alpha L_\mathrm{f}$. Let N be the number of DPs sent from S to D, and A be the corresponding number of ACKs received by S. We can write

$$\frac{A}{N} = 1 - L_\mathrm{f} L_\mathrm{b} = 1 - \alpha (L_\mathrm{f})^2, \tag{4}$$

so that

$$L_\mathrm{f} = \sqrt{\frac{1}{\alpha}\left(1 - \frac{A}{N}\right)}. \tag{5}$$

The source S therefore stores separate (N, A) values for each of its destinations. Assuming that forward and reverse loss rates are identical, we set $\alpha = 1$ and $L_\mathrm{f} = \sqrt{1 - A/N}$.

If routing only selects paths which offer the lowest packet loss, there are several ways in which we can construct the reward R. One approach is to set $\bar{W} = 0$ in the expression (2), obtaining

$$R = \frac{1 - \bar{L}_\mathrm{f}}{T\bar{L}_\mathrm{f}}, \tag{6}$$

so that $1/T$ acts as a constant multiplier. In practice, since we do not want R to be infinite when $L_\mathrm{f} = 0$, we set

$$R = \frac{1 - \bar{L}_\mathrm{f}}{T(\bar{L}_\mathrm{f} + \epsilon)}, \tag{7}$$

where ϵ is a constant representing some minimal value for the loss. A simpler approach is to use R of the form:

$$R = \frac{\beta}{\bar{L}_\mathrm{f} + \epsilon}, \tag{8}$$

which relates loss directly to the reward. This is the approach we have taken in our experiments when we just deal with loss (rather than loss and delay). In the experiments we report, we have used the following numerical values of the constants: $\epsilon = 10^{-5}$ and $\beta = 0.5$.

4. Measurement results

The measurements that we report in this section were performed under a variety of conditions on the two network test-beds of Figs. 1 and 5. All tests were conducted using a flow of UDP packets entering

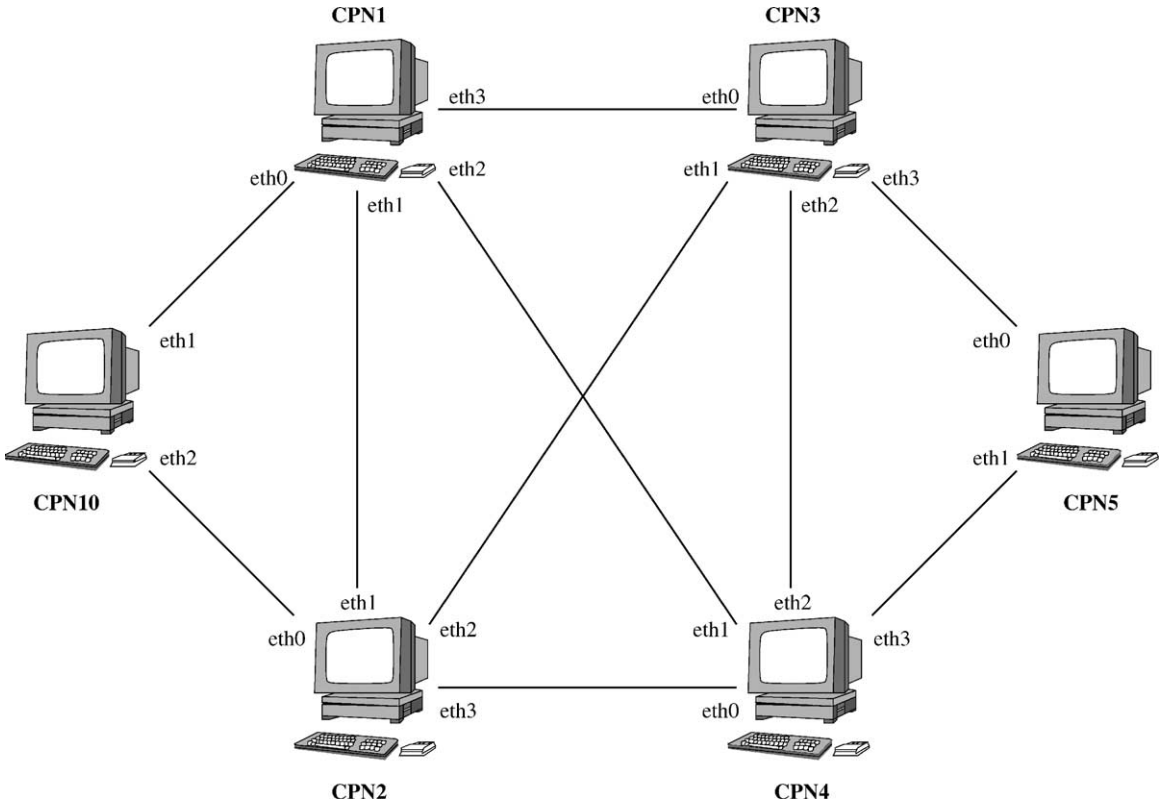

Fig. 1. Small CPN test-bed.

the CPN network with constant bit rate (CBR) traffic and a packet size of 1024 bytes. All CPN links used 10 Mbps point-to-point Ethernet. In wired networks, loss is typically correlated with congestion while congestion can be observed via packet forwarding delays which grow with congestion, so that using delay as a QoS goal can serve to minimize loss as well as delay. However, in an environment where loss may be unrelated to congestion, such as in a wireless setting, minimizing delay will not necessarily minimize loss. To study the effect of packet loss when it is not just a consequence of congestion, an "artificial loss ratio" was introduced in one of the nodes (Node 1, that is CPN1) in the network of Fig. 1, with the possibility of varying the artificial loss rate.

We first conducted experiments with "zero artificial loss rate". In Fig. 2, we can see that both delay and loss performance based on only using delay as the QoS goal is better than the performance based on using loss *and* delay, either at the link level or over whole paths (cumulative). Thus routing based on delay appears to minimize *both* observed loss and delay. We were surprised to observe that the end-to-end performance for routing based on link loss was better than that for the routing using cumulative loss.

When we introduce a very small (1%) artificial packet loss rate (see Fig. 3) delay does not seem to change, but the loss performance curves that use cumulative loss in the goal function are clearly the best as long as the network is not heavily loaded (i.e., link traffic under 8 Mbps). When traffic is close to saturation levels, losses due to congestion dominate and (as one would expect) the results become similar to those of Fig. 2. Finally, in Fig. 4, we increase the artificial packet loss ratio to 5% and observe

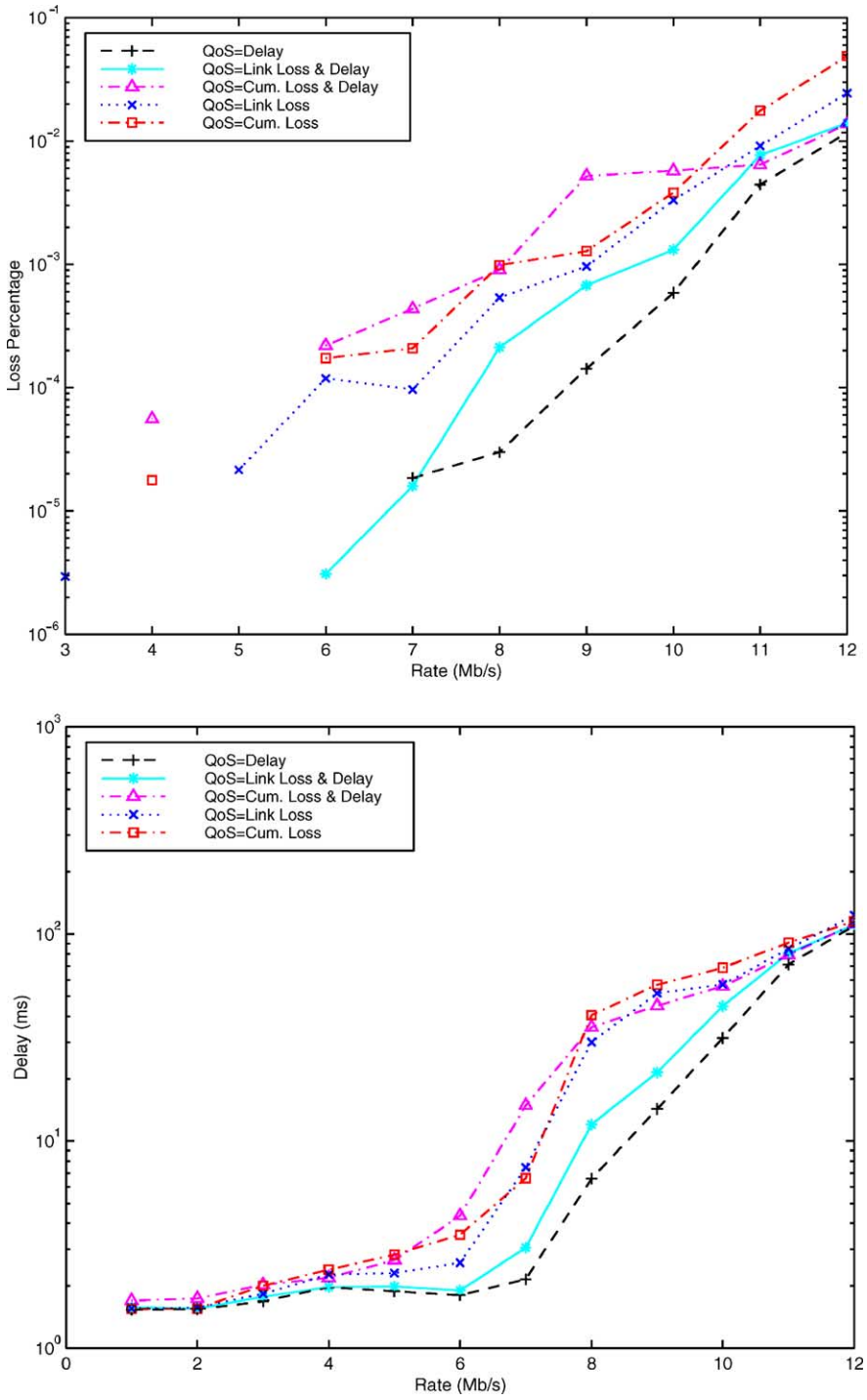

Fig. 2. Loss (top) and delay (bottom) in the small test-bed with 0% artificial loss rate at Node 1.

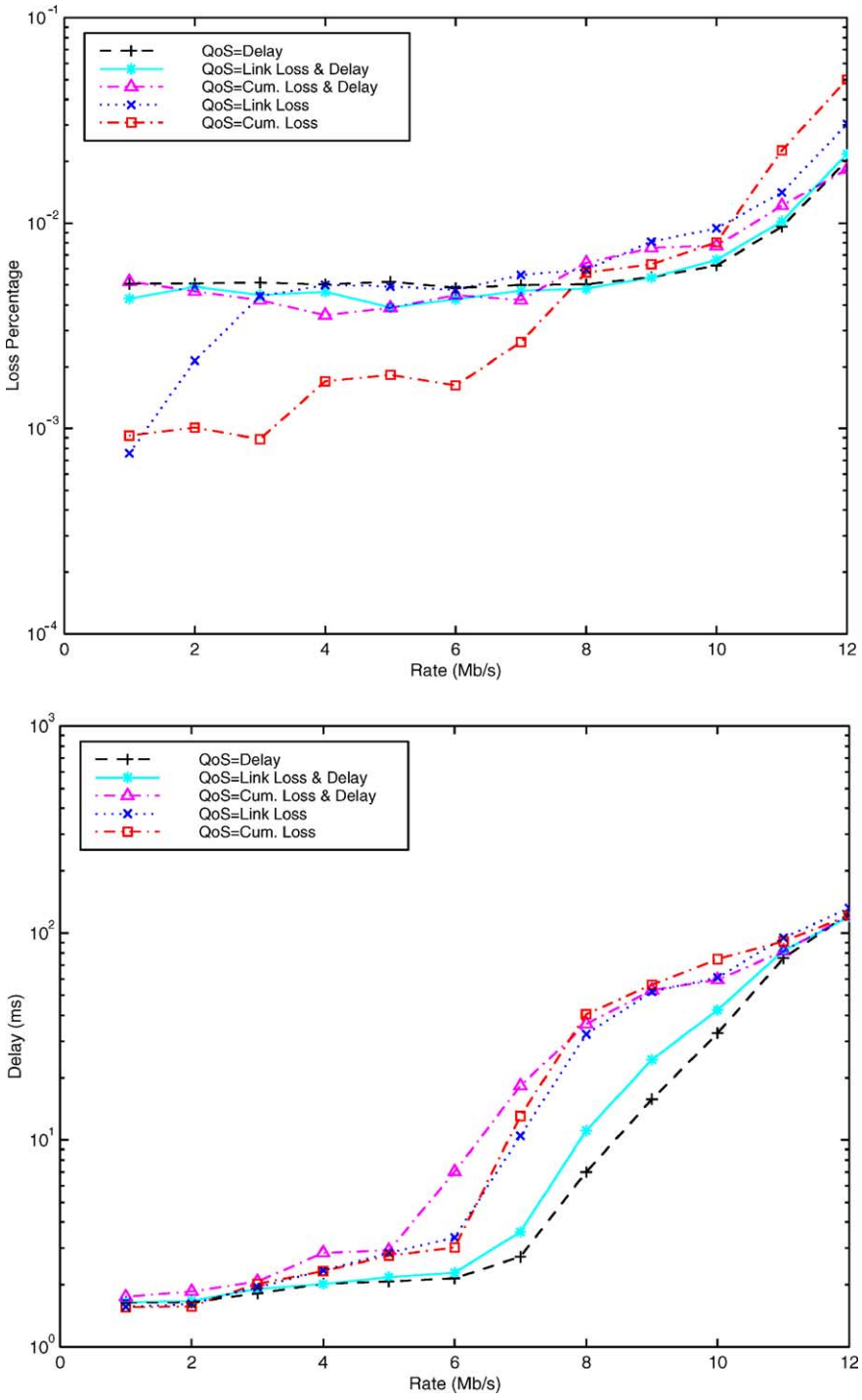

Fig. 3. Loss (top) and delay (bottom) in the small test-bed with 1% artificial loss rate at Node 1.

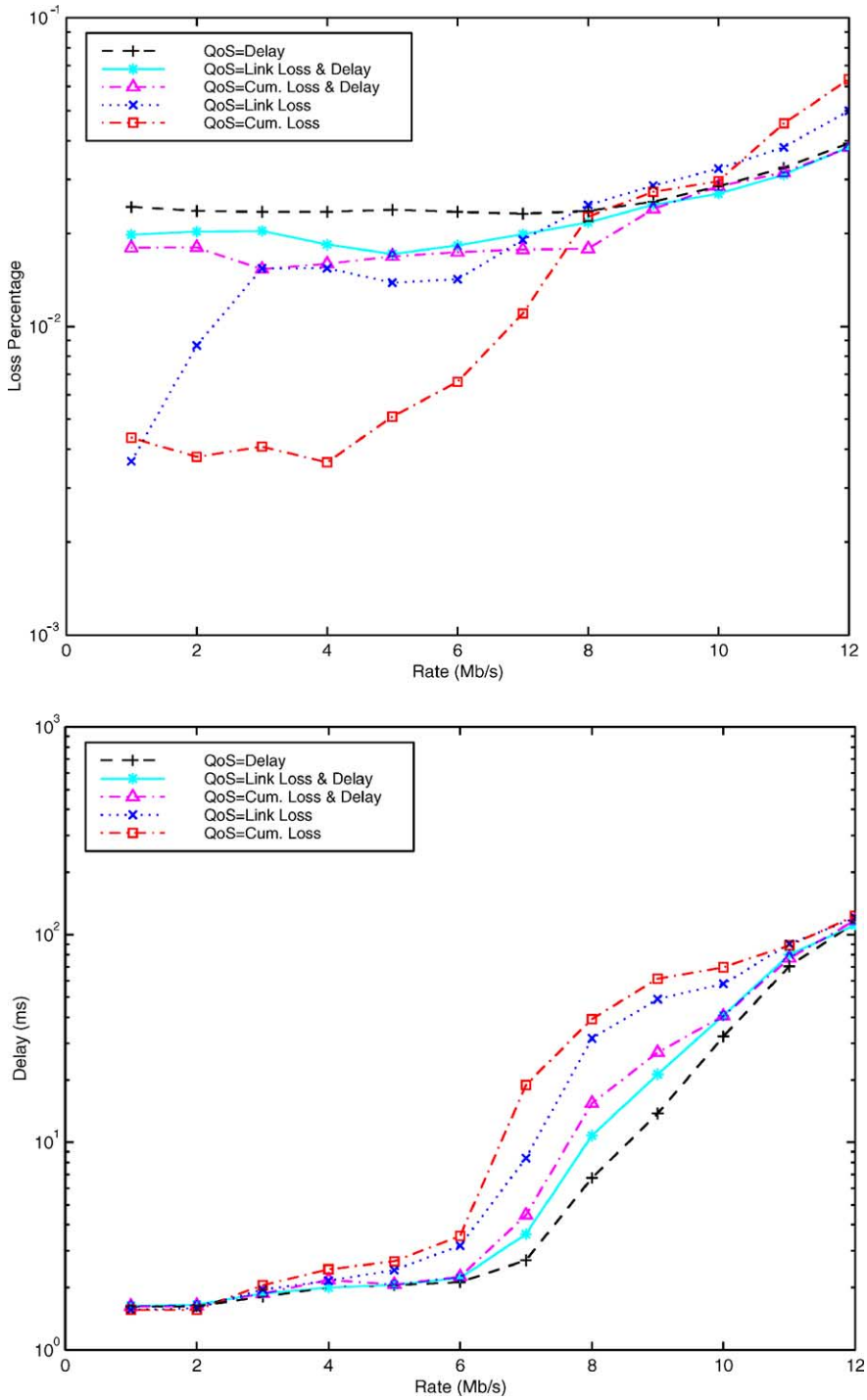

Fig. 4. Loss (top) and delay (bottom) in the small test-bed with 5% artificial loss rate at Node 1.

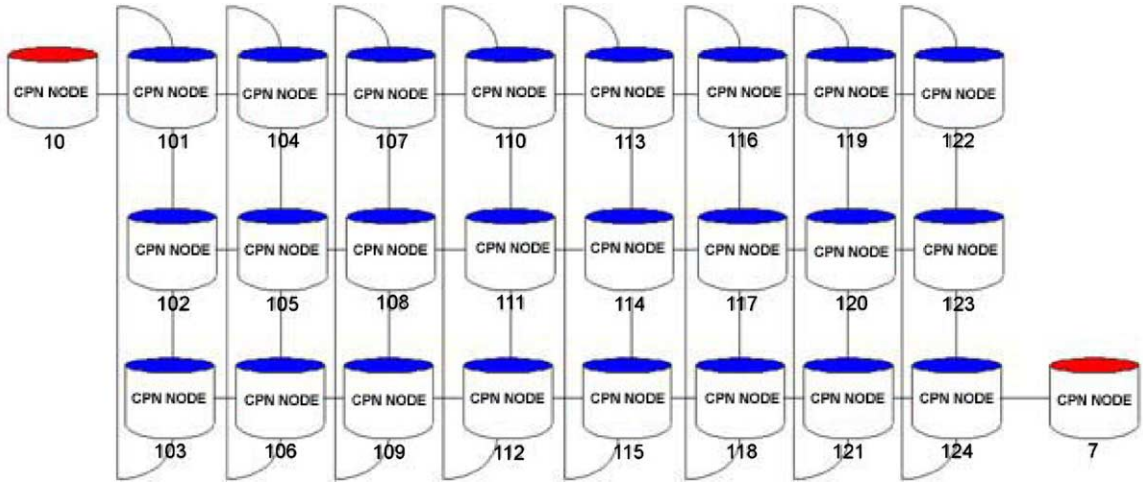

Fig. 5. Larger CPN test-bed.

once again that using cumulative loss as the goal in the routing algorithm provides the best overall results.

To provide further evaluation of composite goal functions, we conducted another set of measurements on the larger CPN test-bed consisting of 26 nodes shown in Fig. 5. The same UDP packet stream with CBR, as before, were sent into the larger CPN test-bed into Node 10 as the source, for forwarding to Node 7 as the destination. The CPN protocol was run in the usual manner with paths being discovered by SPs, while DPs were used to carry and source route the payload UDP packets. No artificial losses were introduced. As shown in the top figure of Fig. 6, the reduction in packet loss rate due to the use of cumulative loss alone, appears even more significantly in the larger test-bed. Here, using the goal function based only on cumulative loss results in the lowest observed loss rates. The curves in the bottom figure show that the lowest delay is obtained by using only delay *or loss and delay* in the goal function. The linear scale used for delay (y-axis) in this figure clearly shows a peak for traffic in the range of 11–12 Mb/s, with a reduction in delay above that value, when cumulative loss or only delay are used in the routing goal function. The drop in delay at higher traffic values is presumably due to the significant loss of packets which results in lower congestion.

Fig. 7 summarizes measurements on the larger test-bed, with packet loss only resulting from congestion (i.e. no artificial packet loss) from the perspective of QoS perceived by the *user*. The purpose of these figures is to see whether the end user is indeed obtaining the outcome in QoS that it is requesting. On the y-axis of the three figures we show the reward function value at the *end user* (and not the numerical value that is used by the SPs, which operate at the lower network level). The reward is computed using the formulas given in Eqs. (2) and (8), using the *measured* values of loss and delay. In the upper curves, we show the 'loss-based" reward for the user, as a function of user traffic. We see very clearly that the highest (therefore the best) reward is obtained if we use *only loss* in the goal function, while using *delay* and *loss and delay* are worse but equivalent from the user's perspective. In the middle and lower figures, we see that using delay as the QoS goal will provide the user with the best reward at low traffic; all three goal functions provide equivalent QoS at higher traffic values. The two lower figures are nearly identical.

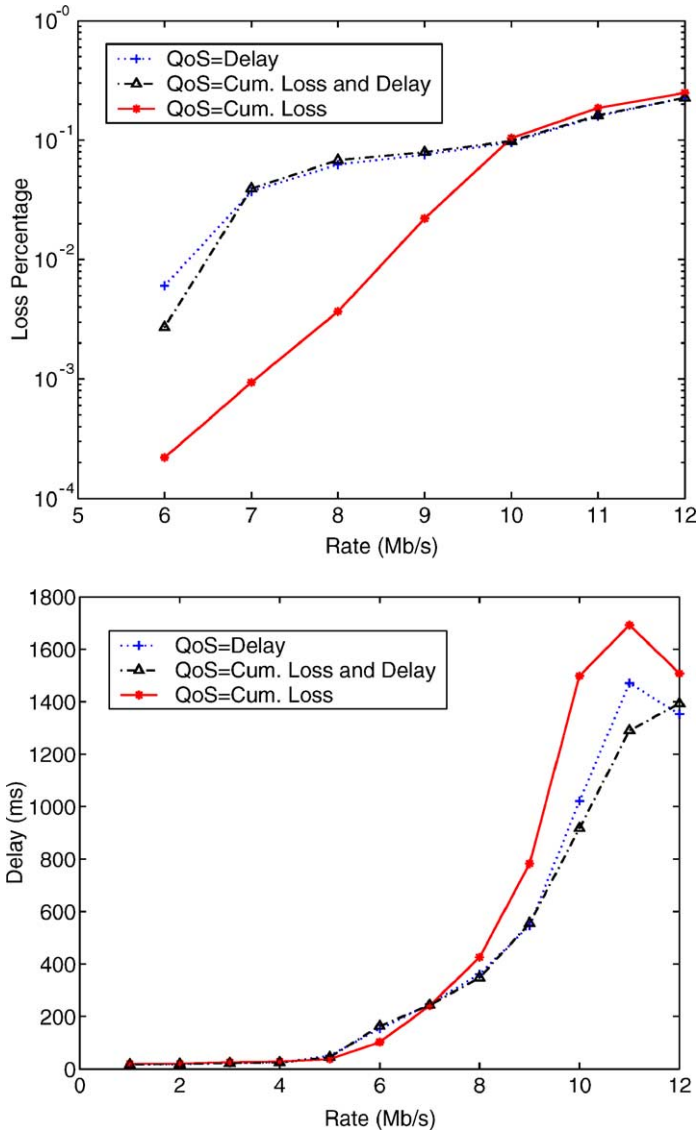

Fig. 6. Loss (top) and delay (bottom) in the large test-bed with 0% artificial loss rate.

The reason is that when we combine loss and delay to compute the reward function, the effect of the loss rate for: (1) low loss values (i.e. less than 10%) is negligible in the reward function, while for (2) high loss values all of the congestion avoidance methods embodied in the three goal functions provide equivalent performance and QoS. However, in view of the upper set of curves, we do see that using a goal function which is specifically tuned to the user's needs is justified since in some cases it will have a definite effect, while in other cases it will not be detrimental to user perceived QoS. One more set of measurements on the larger test-bed are reported in Fig. 8. The purpose is to evaluate the impact of the parameter T used in the goal functions. The curve shows that varying T between 10^2 and 10^5 has little impact on the values

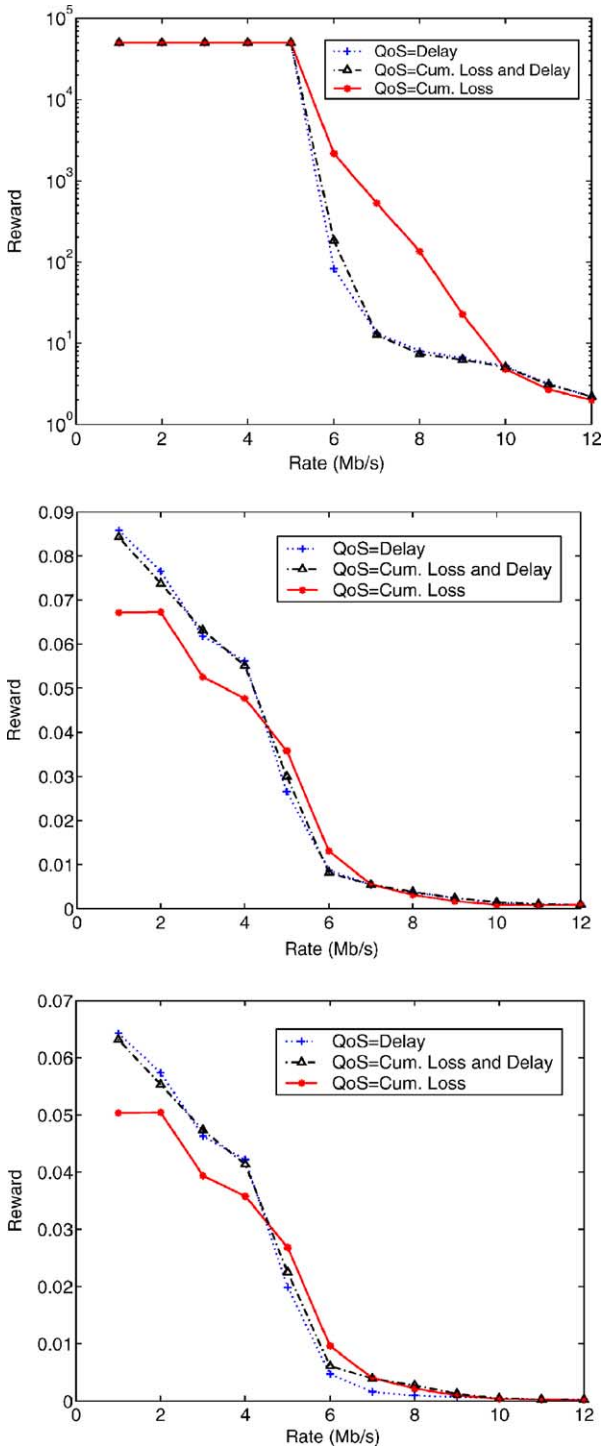

Fig. 7. Measured reward value at user level when RL is based on loss (upper) and delay (middle), and combined loss and delay (lower); large test-bed with 0% artificial loss.

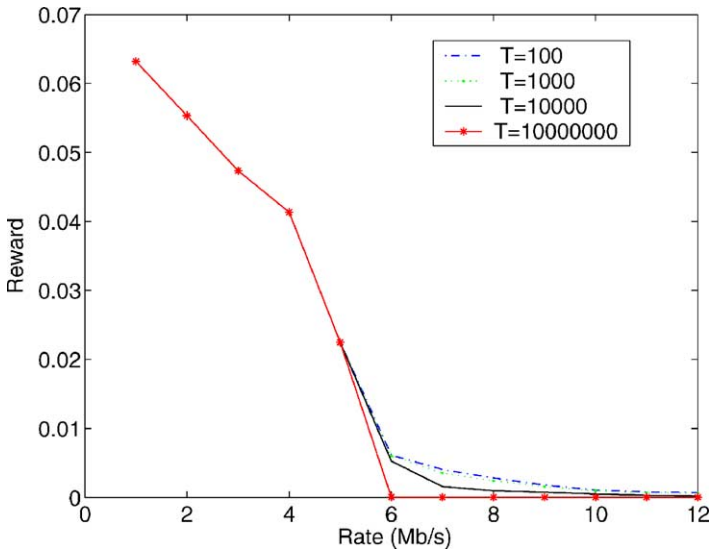

Fig. 8. Observed reward for different values of T.

of the reward experienced by the user; a more significant difference is only observed at high traffic rates (hence high loss rates) when $T = 10^7$.

In the measurements on the small test-bed, we have observed that when packet loss is only due to congestion, the use of delay by itself in the routing goal function results *both* in the smallest delay and the smallest loss. This is a strong indication that measuring delay is a good sufficient indicator for congestion in the small test-bed. However, when non-congestion related losses are introduced in a node, using loss alone results in the smallest loss. Nonetheless, when the network is saturated, we see that all goal functions (with or without both metrics) essentially result in the same delay and loss. Also we observe using link loss or cumulative path loss in the goal function does not seem to have a significant difference under low to moderate load conditions, but cumulative loss is generally the better choice. In the larger test-bed, we observe the use of *delay* or *loss and delay* in the goal function results in better delay characteristics for DPs. Similarly, using loss alone in the goal function provides the smallest loss rate for DPs. Thus our experiments support the claim that there is a good correspondence between the QoS goal that the SPs use to find paths, and the resulting QoS observed by the users' payload.

5. Future work

The CPN protocol suggests many further avenues of investigation. One interesting direction is the study of multi-dimensional QoS goals. For instance, certain network users may require that QoS be maintained within a predefined boundary of values for several different QoS metrics such as loss, delay or jitter. Also, since QoS is increasingly concerned by power limitations which are particularly relevant to wireless networks [15], it will be very interesting to see how power limitations, together with other metrics, can be used to control the network.

It would also be very useful to combine priority schemes for packets within routers, together with routing schemes for packets through the network. The combination of dynamic priorities and dynamically changing routes constitutes yet another interesting challenge in the design of self-adaptive networks.

All these extensions can provide a broad framework within which network self-monitoring will be exploited dynamically to offer best-effort QoS to users.

Acknowledgements

The authors are grateful to E. Gelenbe for guidance during this research, and to R. Lent and J.A. Nunez with help in preparing and conducting the experiments.

References

[1] D.E. Rumelhart, J.L. McClelland, Parallel Distributed Processing, vols. I and II, Bradford Books/MIT Press, 1986.
[2] R.S. Sutton, Learning to predict the methods of temporal difference, Mach. Learn. 3 (1988) 9–44.
[3] H. Ahmadi, R. Guerin, M. Nagshineh, Equivalent capacity and its application to bandwidth allocation in high speed networks, IEEE J. Sel. Area Commun. 9 (1991) 968–981.
[4] E. Gelenbe, Learning in the recurrent random neural network, Neural Comput. 5 (1) (1993) 154–164.
[5] H. Zhang, Service disciplines for guaranteed performance service in packet switching networks, Proc. IEEE 83 (10) (1995) 1374–1396.
[6] E. Gelenbe, X. Mang, Y. Feng, Diffusion cell loss estimates for ATM with multiclass bursty traffic, Comput. Syst.—Sci. Eng. 11 (6) (1996) 325–334 (special issue on ATM networks).
[7] V. Srinivasan, A. Ghanwani, E. Gelenbe, Block cell loss reduction in ATM systems, Comput. Commun. 19 (1996) 1077–1091.
[8] S. Chen, K. Nahrstedt, Distributed quality-of-service routing in ad-hoc networks, IEEE J. Sel. Area Commun. 17 (8) (1999) 1–19.
[9] E. Gelenbe, E. Seref, Z. Xu, Towards networks with intelligent packets, in: Proceedings of the IEEE-ICTAI Conference on Tools for Artificial Intelligence, Chicago, November 9–11, 1999.
[10] E. Gelenbe, Z.-H. Mao, Y. Da-Li, Function approximation with spiked random networks, IEEE Trans. Neural Networks 10 (1) (1999) 3–9.
[11] E. Gelenbe, R. Lent, Z. Xu, Towards networks with cognitive packets, in: Proceedings of the IEEE MASCOTS Conference, San Francisco, CA, August 29–September 1, 2000, pp. 3–12.
[12] E. Gelenbe, E. Şeref, Z. Xu, Simulation with learning agents, Proc. IEEE 89 (2) (2001) 148–157.
[13] E. Gelenbe, R. Lent, Z. Xu, Measurement and performance of cognitive packet networks, Comput. Networks 37 (2001) 691–701.
[14] E. Gelenbe, R. Lent, Z. Xu, Design and performance of cognitive packet networks, Perform. Eval. 46 (2001) 155–176.
[15] E. Gelenbe, R. Lent, Mobile ad-hoc cognitive packet networks, in: Proceedings of the IEEE ASWN, Paris, July 2–4, 2002.
[16] E. Gelenbe, R. Lent, A. Montuori, Z. Xu, Cognitive packet petworks: QoS and performance, in: Proceedings of the IEEE MASCOTS Conference, San Antonio, TX, October 14–16, 2002 (Keynote Paper).
[17] E. Gelenbe, K. Hussain, Learning in the multiple class random neural network, IEEE Trans. Neural Networks 13 (6) (2002) 1257–1267.
[18] F. Hao, E.W. Zegura, M.H. Ammar, QoS routing for anycast communications: motivation and an architecture for Diffserv networks, IEEE Commun. Mag. 46 (2) (2002) 48–56.
[19] Y.-D. Lin, N.-B. Hsu, R.-H. Hwang, QoS routing granularity in MPLS networks, IEEE Commun. Mag. 46 (2) (2002) 58–65.
[20] S. Nelakuditi, Z.-L. Zhang, A localized adaptive proportioning approach to QoS routing granularity, IEEE Commun. Mag. 46 (2) (2002) 66–71.
[21] M. Kodialam, T.V. Lakshman, Restorable quality of service routing, IEEE Commun. Mag. 46 (2) (2002) 72–81.

[22] E. Gelenbe, R. Lent, A. Nunez, Self-aware networks and quality of service, in: Proceedings of the IEEE Symposium on Computer Communications'03 (ISCC'03), Antalya, Turkey, June 29–July 3, 2003.
[23] E. Gelenbe, A. Nunez, R. Lent, Smart WWW traffic balancing, in: Proceedings of the IEEE Workshop on Internet Performance and IP Operations Management, Kansas City, MO, October 1–3, 2003.
[24] E. Gelenbe, Sensible decisions based on QoS, Comput. Manage. Sci. 1 (1) (2004) 1–14.

Pu Su received her B.E. degree in Computer Science and Technology from Tsinghua University, Beijing, China in 2001, and M.S. degree in Computer Science from University of Central Florida, Orlando, USA in 2003. Currently she is a Ph.D. student in University of Central Florida. Her current research interests include network performance, QoS routing and transport control protocol.

Michael Gellman received his B.S. in Computer Science from the University of Central Florida, Orlando in the Spring of 2002. Currently, he is attending Imperial College, London for his Ph.D. in Electrical Engineering. His research interests include network security, specifically focusing on Denial of Service Attacks.