Admission of Packet Flows in a Self-Aware Network

Georgia Sakellari
Imperial College London
g.sakellari@imperial.ac.uk

Erol Gelenbe
Imperial College London
e.gelenbe@imperial.ac.uk

Maurizio D'Arienzo
Seconda Università
degli studi di Napoli
maudarie@unina.it

Abstract

The demand for stable and reliable networks which can offer packet delivery under Quality of Service constraints led to the development of autonomous networks that use adaptive packet routing in order to provide the best possible QoS. A mechanism which takes networks like these a step further in guaranteeing packet delivery, even under strict QoS constraints, is Admission Control (AC). In this paper we describe a measurement-based admission control algorithm which tries to control the ingress traffic of a network by not allowing the entrance of connections which would negatively affect the QoS of the existing users of the network. It is a multiple QoS mechanism in which the users are the ones that specify the QoS levels they need in order to function properly. The impact that this new call will have on the QoS of the existing users, is based on measurements of probe traffic and monitoring of the network . The decision of whether to accept a new call or not is made using a novel algebra of QoS metrics, inspired by Warshall's algorithm, which searches whether there is a feasible path with enough resources to accommodate the new flow, without affecting the ongoing traffic.

1 Introduction

The current "best effort" Internet architecture does not secure the Quality of Service (QoS) that multimedia traffic and real-time applications, such as video on demand, Internet telephony (Voice-over-IP), remote medical diagnosis and treatment, and online trading systems require in order to function properly. This need led to the development of Self-Aware Networks [11] that use adaptive packet routing protocols, like the Cognitive Packet Network (CPN) [16], to address QoS and provide reliable service to their users. But even in these networks, congestion is a factor that can lead to unstable and unreliable situations which will affect the service quality. This is a situation which can be improved by Admission Control (AC) which controls the incoming traffic of a network so that it can guarantee QoS throughout the lifetime of each of the network's connections. A proposal of a measurement-based AC, which exploits the ability of CPN to collect QoS information on all links, is the purpose of the work presented in this paper. Our scheme bases its decision of whether to accept a new connection in the network by estimating both the resources it will need and the impact it will have on the ongoing connections.

1-4244-1455-5/07/$25.00©2007 IEEE

2 Types of AC Algorithms

According to whether the traffic parameters are specified a priori or whether the admission decisions rely on measurements of the actual traffic load, Admission Control Algorithms can be classified in two categories: parameter-based and measurement-based.

Parameter-based AC algorithms compute the amount of network resources required to support new flow by given a priori flow characteristics. They can be further classified as non-statistical and statistical allocation algorithms. The **Non-statistical allocation or deterministic** is the simplest form among all admission control algorithms. The only knowledge it uses is the peak rate and it tries to ensure that the sum of requested resources and the existing connections is bounded by the physical link capacity. This type assumes that connections transmit at their peak rates all the time, thus they allocate more bandwidth than it is required to provide QoS guarantees for the existing connections. In **Statistical allocation** the bandwidth for a connection is assigned at less than the peak bandwidth of the connection, depending on the statistical distribution of the arriving cells in the connection [22]. Statistical allocation results in statistical multiplexing gain, when dealing with sources that arrive in "bursts", since it assumes sharing bandwidth resource with other connections and thus the sum of all peak rates may be greater than the capacity of the output link. This type is difficult to be implemented effectively because of the uncertainty in the distribution of the incoming traffic and the inaccurate and difficult-to-calculate statistical information of the traffic arrival process.

The most well-known parameter based algorithms are *Rate or Simple Sum* [21], *Equivalent Bandwidth* [18, 19, 6],

and *Diffusion based statistical AC* [14].

Measurement-based AC (MBAC) algorithms rely on measurements of actual traffic load in making admission decisions. They use these network measurements to estimate the current load of existing traffic, instead of computing the traffic characteristics out of the user specified connection's parameters. They have no prior knowledge of the traffic characteristics and make the admission decisions based only on the current state of the network. The measurement-based schemes alleviate the burden on the users to accurately specify the parameters for their traffic flow, and thus is a more practical approach for achieving statistical multiplexing gain with variable-rate traffic.

The major existing MBAC mechanisms which have been proposed for conventional networks are *Measured Sum* [21], *Measurement Based Admission Control with Delay and Bandwidth Constraint* [20], and the most frequently used *Endpoint Admission Control* [20, 2, 1, 4, 17, 3, 7].

Measurement-based AC algorithms are shown to achieve much higher utilisation than parameter-based [20], and the more accurate and up-to-date the measurements the better the algorithm. So the last few years the research is focused on accurate network monitoring tools and as admission control is concerned it is turned towards measurement-based approaches.

3 Self Aware Networks

Self Aware Networks (SAN) is a proposal of QoS enabled networks with enhanced monitoring and self improvement capabilities that use adaptive packet routing protocols, such as Cognitive Packet Network (CPN) ([16, 12, 10, 9, 13]) and address QoS by using adaptive techniques based on on-line measurements.

In CPN, it is the users that declare their QoS requirements (QoS Goals) such as minimum delay, maximum bandwidth, minimum cost, etc, or a combination of those. It is designed to perform Self-Improvement by learning from the experience of smart packets, using random neural networks (RNN) [8] with reinforcement learning (RL), and genetic algorithms. RL is carried out using a QoS Goal defined by the user, who generated a request for the connection, or by the network itself.

CPN makes use of three types of packets: smart packets (SP) for discovery, source routed dumb packets (DP) to carry payload, and acknowledgements (ACK) to bring back information that has been discovered by SPs which are used in nodes to train neural networks. SPs discover routes by using random neural networks (RNN) [8] with reinforcement learning (RL). RL is carried out using a QoS Goal which is defined by the user who generated a request for the connection, or by the network itself. The decisional weights of a RNN are increased or decreased based on the observed success or failure of subsequent SPs to achieve the Goal. Thus RL will tend to prefer better routing schemes, more reliable access paths to data objects, and better QoS.

When a Smart Packet arrives to its destination, an ACK is generated and heads back to the source of the request, following the reversed path of the SP. It updates mailboxes (MBs) in the CPN nodes it visits with the information which has discovered, and provides the source node with the successful path to the node. The route brought back by an ACK is used as a source route by subsequent DPs of the same QoS class having the same destination, until a newer and/or better route is brought back by another ACK.

Each node stores a specific RNN for each active source-destination pair, and each QoS goal. The number of neurons in an RNN corresponds to the number of routing outputs of a node. Each output link of a node is represented by a neuron in the RNN. The arrival of Smart Packets(SPs) triggers the execution of RNN and the routing decision is the output link corresponding to the most excited neuron. CPN reinforcement learning changes neuron weights to reward or punish a neuron according to the level of goal satisfaction measured on the corresponding output.

The level of goal satisfaction is expressed by a reward. Given some goal G that a packet has to minimize, the reward R is formulated simply as $R = 1/G$. The state q_i of ith neuron in the network is the probability that it is excited. The $q_i, 1 < i < n$ satisfy the following system of nonlinear equations:

$$q_i = \frac{\lambda^+(i)}{r(i) + \lambda^-(i)} \tag{1}$$

where

$$\lambda^+(i) = \sum_j q_j w^+_{ji} + \Lambda_i \;\; and \;\; \lambda_-(i) = \sum_j q_j w^+_{ji} + \lambda_i \tag{2}$$

w^+_{ji} is the rate at which neuron j sends "excitation spikes" to neuron i when j is excited, w^-_{ji} is the rate at which neuron j sends "inhibition spikes" to neuron i when j is excited, and $r(i)$ is the total firing rate from the neuron i. For an n neuron network, the network parameters are these n by n "weight matrices" $W^+ = \{w^+(i,j)\}$ and $W^- = \{w^-(i,j)\}$ which need to be "learned" from input data.

The RNN weights are updated based on a threshold T:

$$T_k = \alpha T_{k-1} + (1 - \alpha) R_k \tag{3}$$

where R_k, k = 1, 2, ... are successive measured values of reward R and α is some constant ($0 < \alpha < 1$) that is used to tune the responsiveness of the algorithm: for instance $\alpha = 0.8$ means that on the average five past values of R are being taken into account. Neurons are rewarded or punished based on the difference between R_k, the current reward, and T_{k-1}, the last threshold.

4 Our proposed multiple criterion AC algorithm

The measurement-based AC algorithm we propose [23, 15] is based on measurements of the QoS metrics on each

link of the network . This does not require any special monitoring mechanism since the CPN already collects QoS information on all links and paths that the SPs have explored and on all paths that any user is using in the network. Our scheme is a centralized algorithm since the QoS data mentioned above is gathered in one or more locations in the network, where the decision of whether to accept a new call is also taken.

The proposed AC scheme consists of two stages. The Probing Stage, which is the stage where the estimated QoS values of the new flow and the impact that it will have to the existing users is estimated, by probing the network, and the Decision Stage in which the decision on whether to accept the new call into the network is taken, based on whether there is a feasible path which can accommodate the new call without affecting the quality of formerly accepted flows.

4.1 Probing Stage - Estimation of the impact of a new flow

Every QoS metric can be considered as a value which increases as the "traffic load" increases. A new connection will increase the load of the paths it may be using so it is assumed that the value taken by the QoS metrics will increase. For example, delay increases as the network traffic load increases

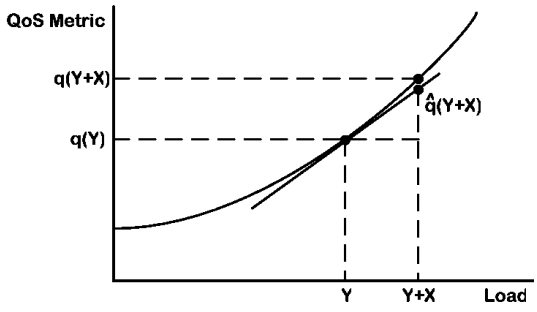

Figure 1. QoS metric vs Load

Let as consider some link (i, j). A small increase x in the load that is obtained in a controlled manner, e.g. by sending probe packets at rate x, generates an estimate of the manner in which the QoS metric q varies around the current load point Y:

$$\hat{q}' = \frac{q(Y+x) - q(Y)}{x}. \quad (4)$$

The impact of a new flow with total traffic rate X can then be evaluated by using the estimate and the measured derivative from (4):

$$\hat{q}(Y+X) = q(Y) + \hat{q}'X, \quad (5)$$

without having to know the initial load Y. This estimate may be optimistic or pessimistic. However it is likely that the path that CPN will select for the probe traffic, because it provides the most favorable impact on current flows and because it satisfies the QoS needs of the new flow. It is also likely that this path is also the best path in terms of actual observed QoS after the new user's full traffic is inserted.

Contrary to the existing measurement-based AC schemes that use probing, in our scheme, it is not required to send the probe packets at the same rate as the new call's requested rate. Instead we can send them at a much lower rate and still have an accurate estimation. This is a major advantage since this way the probing process has no significant impact on the network's congestion.

4.2 Decision Stage

Let us assume that the users may be concerned with m distinct QoS metrics $q_v \in R, v = 1, ...m$ that are specified in terms of QoS constraints [$q_v \in C_v(u)$ for each user u], where $C_v(u) \subset R$ is typically an interval of acceptable values of the QoS metric v for user u (a "contract" between the user and the network which must not be violated). We will detail the AC algorithm in terms of forwarding packets from some source s to a destination d. However this approach can be generalised to the case where u is requesting some service S.

A network can be considered as a network graph $G(N, E)$ with nodes $N, n = |N|$, and a set E of directional links (i, j), where $i, j \in N$. The CPN algorithm explores $G(N, E)$ and collects QoS data about the parts of the network that are being currently used, or which have been explored by SPs. We assume that this data is available in one or more locations in the form of $n \times n$ *link QoS matrices* Q_v with elements:

- $Q_v(i, j) = r$ where $r \geq 0$ is a real number representing the QoS of link (i, j) which has been measured at some recent enough time, and
- $Q_v(i, j) = unknown$ if i and j are not directly connected or if either a SP has not explored the link for QoS metric v or if this happened so long ago that the value could be inaccurate.

From the link matrices Q_v we can compute:

- The set of known (explored) paths $P(s, d)$ from s to d, and
- The *path QoS matrices* K_v, where $K_v(s, d)$ is the *known best value* of the QoS metric v for any *path* going from s to d if such a path exists and if the links on the path have known entries in the link QoS matrices. Other entries in K_v are set to the value "unknown".

By "best value" we mean that several paths may exist for the source-destination pair (s, d), but $K_v(s, d)$ will store, for instance, the smallest known delay for all paths going from s to d if q_v is the delay metric. We will discuss below how the path QoS matrices are computed from the link matrices.

4.3 The AC Algorithm

Let us assume that the network is currently carrying z users, any one of which will be generically represented by some QoS constraint $q_w(z)$ and a new user u requests admission for a connection from source s to destination d carrying a traffic rate X and with QoS constraint $q_v(u)$. The proposed AC algorithm proceeds as follows:

- Find the set $P(s,d)$. If it is empty, send SPs to discover paths. If unsuccessful, reject the request. Otherwise monitor the current network, create the $Q_w(i,j)$ matrices for all discovered links and all QoS metrics (including $w = v$), and then send probe traffic at rate x along the network.
- Use the probe traffic to obtain $\hat{q}'_w(i,j)$ for each QoS metric w of interest, including $w = v$, and for all links (i,j). Note that some links may not be concerned by the probe traffic so for that links we take $\hat{q}'_w(i,j) = 0$. The path that the probe packets will follow, will be the one that the SPs have chosen as more appropriate so that it satisfy the QoS needs of the new flow, so, it is very likely to also be the path that will be followed after the new user's full traffic is inserted.
- Afterwards compute the estimation
$$\hat{Q}_w(i,j) = Q_w(i,j) + X\hat{q}'_w(i,j) \quad (6)$$
for all concerned links and all QoS metrics. For unconcerned links we take $\hat{Q}_w(i,j) = Q_w(i,j)$.
- Compute $\hat{K}_w$ from $\hat{Q}_w$ (to be detailed below) for all the QoS metrics of interest, including v.
- Finally, if $\hat{K}_v(s,d) \in C_v(u)$ AND $\hat{K}_w(s',d') \in C_w(z)$ for all other current users z with source-destination pair (s',d') and QoS metric $q_w \in C_w(z)$, then accept u; else reject the request.

4.4 Computing the QoS matrices

For each $i,j \in N$ of a network graph $G(N,E)$ with nodes N, $n = |N|$, and a set E of directional links (i,j). the well known "Warshall's algorithm" [24] determines whether there is a path from node i to node j by computing the Boolean matrix K, the transitive closure of the graph's adjacency matrix Q, in less than n^3 Boolean operations.

$$K^n[i,j] = K^{n-1}[i,j] \bigvee \left(K^{n-1}[i,n] \bigwedge K^{n-1}[n,j] \right) \quad (7)$$

where $K^1[i,j] = Q[i,j]$ and the matrix elements are treated as boolean values with $\bigvee$ being the logical "OR" and $\bigwedge$ the logical "AND".

Floyd's algorithm [5] extends Warshall's algorithm to obtain the cost of the "smallest cost path" between any pair of vertices in the form of a real-valued matrix.

$$K^n[i,j] = min\Big\{ K^{n-1}[i,j], \left(K^{n-1}[i,n] + K^{n-1}[n,j] \right) \Big\} \quad (8)$$

Thus, our algorithm can use Floyd-Warshall's technique to construct K_v from Q_v, and hence $\hat{K}_v$ from $\hat{Q}_v$ if the QoS metric q_v is additive, so that $K_v(i,j)$ is the smallest value of the QoS metric among all known paths from i to j. Note that delay and the variance of delay, are both additive metrics. Although loss rate is not additive , the number of lost packets is an additive metric.

For non-additive metrics we have developed a generalisation of the Floyd-Warshall, which is described next.

4.5 Generalisation of the Floyd-Warshall algorithm to non-additive QoS metrics

Consider the matrix Q_v mentioned above, whose entries are the measured QoS values $r \geq 0$ over links (i,j) whenever such a link exists, or otherwise have the value "unknown".

The matrix K_v which is calculated as shown below, provides us with the "best QoS value" for every path between every pair of vertices (i,j).

$$K_v^n[i,j] = K_v^{n-1}[i,j] \otimes \left(K_v^{n-1}[i,n] \oplus K_v^{n-1}[n,j] \right) \quad (9)$$

where $K_v^1 = Q_v$. The operator $\oplus$ between two QoS parameters depends on the QoS metric that is being considered and can be the addition (+) for delay and variance, the minimum (min) for bandwidth etc. The $\otimes$ is also an operator that depends on the specific QoS metric q, and selects the "best value" among the elements on which it operates, e.g. in case of the delay or variance metric it will obtain the minimum value, while for bandwidth or security it will select the maximum value for all paths going from i to j.

5 Experimental results

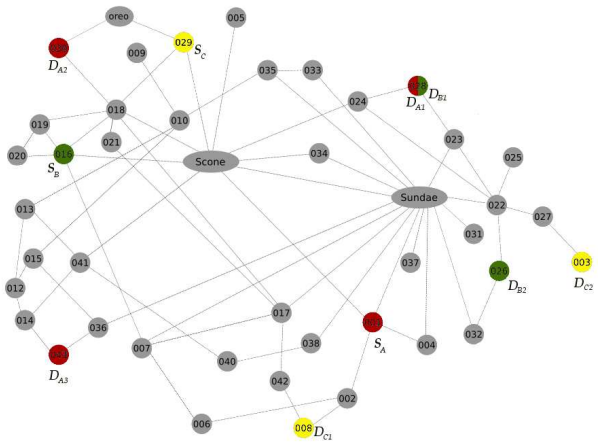

Figure 2. The CPN testbed used in our experiments

The experiments where conducted in a 44-node test-bed representing the SwitchLAN network topology[1]. All links have the same capacity ($10\,Mbits/s$) and all users have the same QoS requirements: $delay \leq 50\,ms$, $jitter \leq 5\,ms$, $bandwidth \geq 3\,Mbits/s$, and $packetloss \leq 5\,\%$. These values are quite fastidious (for high-quality video the delay should be only less than $150ms$, 150 ms is also the maximum desired one-way latency to achieve high-quality voice) and are chosen in order to evaluate our algorithm for very demanding requests.

There are 7 Source-Destination (S-D) pairs $(S_A - D_{A1}, S_A - D_{A2}, S_A - D_{A3}, S_B - D_{B1}, S_B - D_{B2}, S_C - D_{C1}, S_C - D_{C2})$, that correspond to 7 users $(A1, A2, A3, B1, B2, C1, C2)$.

In order to avoid having more than one users requesting to enter the network at the same time, which could lead to multiple flows of probe traffic and misleading measurements, each user enters a queue ("request queue") at the data gathering point. The users that are denied access, instead of being considered rejected, they enter another queue ("reject queue") and request to enter again. This process continues until they are finally accepted or a specific period of time ("user lifetime") has past. In our experiments the user lifetime of each user is $150\,s$, meaning that every user will wait to be served for at most $150\,s$ otherwise the user will be considered rejected. The "reject queue" has bigger priority than the "request queue" and is served first.

After making a request, regardless if the request is satisfied or not, the user waits for a constant time W and then makes another request. In our experiments, W is chosen to be uniformly distributed in a range of values: $(150, 120, 90, 80, 70, 60, 50, 40, 30, 20, 15)\,s$ which correspond to total arrival rate λ for all three users of $(2.8, 3.5, 4.67, 5.25, 6, 7, 8.4, 10.5, 14, 21, 28)\,rqsts/min$ respectively. Thus the load on the system is increased in each of the successive experiments.

Our experiments covered three cases: when the Admission Control is disabled (NO AC), when the AC is enabled (WITH AC), and when the AC is enabled but also the length of the feasible paths is restricted (WITH AC & MPL). The third approach tries to take under consideration the fact that if there is a lot of traffic in the network the algorithm may accept a very long feasible path which CPN would possibly not use for the new traffic. To avoid this conflict of the two intelligent mechanisms, we set a maximum path length (MPL) limit for the length of the feasible path. In our experiments the limit of the feasible path length was set to 6.

In total 99 experiments were conducted (33 for each case), lasting $15\,min$ each. Each experiment was conducted 3 times and the results presented in this paper are the average value of those three runs.

Figure 3 compares the total rejection rate for the connection requests in all three cases, while figure 4 reports the satisfaction rate of a user (here user $A1$) in all three cases described above. By "satisfaction" of a user we mean that all 4 QoS requirements of that user are fulfilled at all times. Of course when the AC is disabled all users are immediately accepted and so the rejection rate is zero.

[1]The Swiss Education & Research Network, http://www.switch.ch/network/

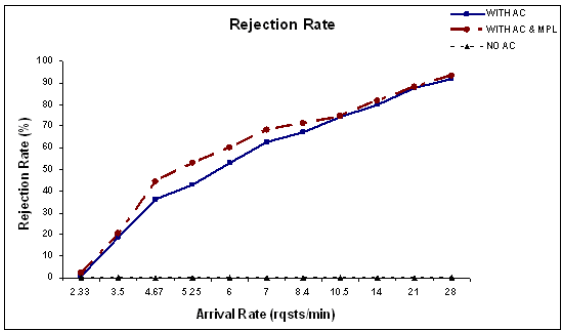

Figure 3. Average rejection rate

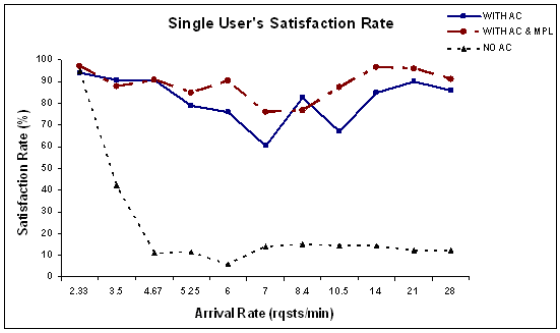

Figure 4. Average user satisfaction

By restricting the length of the feasible path which is used to make the decision, makes the algorithm more strict, and thus the rejection rate should increase, as figure 3 confirms. As it is obvious in figure 4, when the AC algorithm is enabled the satisfaction of user $A1$ is much higher than when there is no AC. When the feasible path's length is limited the percentage of the user traffic that is led through the feasible path increases so the results are more accurate and the satisfaction rate is better than when only the AC is enable without path length restrictions (figure 4). Finding the optimal limit of the feasible paths length is something that could further improve our algorithm and should therefore be further investigated.

6 Conclusions

This paper describes a measurement based AC algorithm which estimates the expected resources that the new user will consume if he/she enters the network and the impact that he/she will have on the QoS the existing users.

A basic difference between our algorithm and other

measurement-based AC schemes that use probing, is that our algorithm estimates the new flow's impact by probing at a small rate, so that probe packets will not contribute noticeably to the network's congestion. Also, a major contribution point is that the users are the ones that specify the QoS constraints they need in order to obtain the network service they require for a successful connection, so, each user can have different QoS requirements. Moreover, the decision of whether to accept a new user is based on a novel, and easily implemented, algebra of QoS metrics which finds whether there is a feasible path which can provide the required resources to accommodate the new request without affecting the service quality of the ongoing connections.

Further work following this paper should be done in order to investigate which is the optimum rate and duration of the probing, which would most probably lead to more accurate estimations and further improve our algorithm. Another issue is on whether the new or existing users will adhere the QoS levels they specified during their entrance, since violating the initial "contracts" could affect the QoS of the other users. One easy way would be to dismiss the non-compliant users (drop all their packets), but this should be studied more carefully. Finally future papers could provide experimental results showing the effectiveness of the algorithm compared to existing approaches of AC.

References

[1] G. Bianchi, A. Capone, and C. Petrioli. Throughput analysis of end-to-end measurement-based admission control in ip. In *Proceedings of IEEE INFOCOM 2000*, pages 1461–1470, Tel Aviv, Israel, Mar. 2000. IEEE.

[2] L. Breslau, E. W. Knightly, S. Shenker, I. Stoica, and H. Zhang. Endpoint admission control: Architectural issues and performance. In *Proceedings of ACM SIGCOMM 2000*, pages 57–70, Stockholm, Sweden, Oct. 2000. Acm.

[3] C. Cetinkaya and E. W. Knightly. Egress admission control. In *Proceedings of IEEE INFOCOM 2000*, volume 1, pages 1471–1480, Tel Aviv, Israel, Mar. 2000. IEEE.

[4] V. Elek, G. Karlsson, and R. Ronngren. Admission control based on end-to-end measurements. In *Proceedings of IEEE INFOCOM 2000*, volume 2, pages 623–630, Tel Aviv, Israel, Mar. 2000.

[5] R. W. Floyd. Algorithm 97: Shortest path. *Communications of ACM*, 5(6):345, June 1962.

[6] S. Floyd. Comments on measurement-based admissions control for controlled-load services. *Submitted to Computer Communication Review*, July 1996.

[7] A. J. Ganesh, P. Key, D. Polis, and R.Srikant. Congestion notification and probing mechanisms for endpoint admission control. *IEEE/ACM Transactions on Networking*, 14(3):568–578, June 2005.

[8] E. Gelenbe. Learning in the recurrent random neural network. *Neural Computation*, 5(1):154–164, Jan. 1993.

[9] E. Gelenbe, M. Gellman, R. Lent, P. Liu, and P. Su. Autonomous smart routing for network qos. In *Proceedings of the First International Conference on Autonomic Computing (ICAC)*, pages 232–239, New York, NY, USA, May 2004.

[10] E. Gelenbe, M. Gellman, and P. Su. Self-awareness and adaptivity for quality of service. In A. Tantawy and K. Inan, editors, *Proceedings of the IEEE International Symposium on Computers and Communications (ISCC'03)*, pages 3–9, Kemer-Antalya, Turkey, June/July 2003. IEEE Computer Society. Invited Paper.

[11] E. Gelenbe, R. Lent, and A. Nunez. Self-aware networks and qos. *Proceedings of the IEEE*, 92(9):1478–1489, Sep. 2004.

[12] E. Gelenbe, R. Lent, and Z. Xu. Design and performance of cognitive packet networks. *Performance Evaluation*, 46(2-3):155–176, Oct. 2001.

[13] E. Gelenbe and P. Liu. Qos and routing in the cognitive packet network. In *Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, pages 517–521, Taormina, Italy, June 2005.

[14] E. Gelenbe, X. Mang, and R. Oenvural. Diffusion based statistical call admission control in atm. *Performance Evaluation*, 27/28(Com):411–436, Oct. 1996.

[15] E. Gelenbe, G. Sakellari, and M. D' Arienzo. Controlling access to preserve qos in a self-aware network. In *Proceedings of the First IEEE International Conference on Self-Adaptive and Self-Orginizing Systems (SASO 2007)*, Boston, MA, USA, Jul. 2007.

[16] E. Gelenbe, Z. Xu, and E. Seref. Cognitive packet networks. In *Proceedings of the 11th International Conference on Tools with Artificial Intelligence (ICTAI '99)*, pages 47–54, Chicago, IL, Nov. 1999. IEEE Computer Society Press.

[17] R. J. Gibbens and F. Kelly. Distributed connection acceptance control for a connectionless network. In *Proceedings of the 16th International Teletraffic Congress (ITC 99)*, volume 2, pages 941–52, Edinburgh, UK, Feb. 1999.

[18] R. Guerin, H. Ahmadi, and M. Naghshineh. Equivalent capacity and its application to bandwidth allocation in high-speed networks. *IEEE Journal on Selected Areas in Communications*, 9(7):968–981, Sep. 1991.

[19] R. Guerin and L. Gun. A unified approach to bandwidth allocation and access control in fast packet-switched networks. In *Proceeding of INFOCOM'92*, volume 1, pages 1–12, Florence, Italy, May 1992.

[20] S. Jamin, P. Danzig, S. Shenker, and L. Zhang. A measurement-based admission control algorithm for integrated services packet networks. *IEEE/ACM Transactions on Networking*, 5(1):56–70, Aug./Sep. 1997.

[21] S. Jamin, S. J. Shenker, and P. B. Danzig. Comparison of measurement-based admission control algorithms for controlled-load service. In *Proceedings of the Conference on Computer Communications (IEEE INFOCOM '97)*, volume 3, pages 973–980, Kobe, Japan, Apr. 1997. IEEE Computer Society Press.

[22] H. G. Perros and K. M. Elsayed. Call admission control schemes: A review. *IEEE Communications Magazine*, 34(11):82–91, Nov. 1996.

[23] G. Sakellari, M. D' Arienzo, and E. Gelenbe. Admission control in self aware networks. In *Proceedings of the 49th annual IEEE Global Telecommunications Conference, GLOBECOM 2006*, San Francisco, CA, USA, Nov./Dec. 2006.

[24] S. Warshall. A theorem on boolean matrices. *Journal of the ACM*, 9(1):11–12, Jan. 1962.