Internet QoS Routing with IP Telephony and TCP Traffic

Alex Dubrovsky, Mario Gerla, Scott Seongwook Lee, Dirceu Cavendish
Computer Science Department
University of California, Los Angeles
405 Hilgard Avenue, Los Angeles, CA USA 90024
Corresponding Author: gerla@cs.ucla.edu

Abstract—

In this paper, we propose the use of QoS routing to enhance the support of IP Telephony. Our proposed scheme is based on QoS intradomain OSPF routing, an extension of the conventional OSPF routing protocol. A DiffServ model is used (no per flow signaling, nor per flow accounting at intermediate nodes). Processing O/H is shifted from core to edge routers, which compute routes, monitor QoS path quality and enforce Call Acceptance Control (CAC) using the link state information advertised by OSPF.

Via simulation, we show significant delay and throughput improvement over IP telephony strategies currently used in the Internet. In particular, hot spots and focussed congestion points are easily avoided. Moreover, the ability to control voice via QoS routing and CAC permits us to adjust the capacity sharing between voice traffic and TCP traffic, by reserving a fraction of link bandwidth to TCP data traffic. We also show that the added control and processing overhead is quite manageable, even in fairly large networks.

Keywords—

Open Shortest Path First, Class Based Queues, Quality of Service, Call Acceptance Control, routing, IP telephony.

I. Introduction & Related Work

QoS support for internet multimedia applications is necessary to meet stringent end-to-end requirements such as bandwidth, delay, and packet loss. The support of QoS applications in a packet network requires a broad range of functions such as priority mechanisms, scheduling disciplines, traffic shaping schemes, and routing algorithms.

QoS Routing consists in finding a path which complies with the end-to-end constraints of the specific application. This task translates into finding routes in a multiple metric scenario. In general, optimal routing with multiple metrics is known to be an NP-complete problem [7]. To avoid the complexity of exact, combinatorial solutions, recent papers have addressed the QoS constrained routing problem using a variety of heuristic strategies. Some researchers have proposed approximate solutions based on minimizing one objective function at a time (i.e., defining a hierarchy of metrics) [8], for a generic multiple metric routing problem. Others have succeeded in reducing the problem complexity to a polynomial degree by assuming a particular scheduling discipline and traffic shaping policy at each router ([10] [16]). More recently, [5] has designed a QoS routing algorithm based on a variant of the Bellman-Ford algorithm. The QoS, Multiple Constraints (MC) Bellman-Ford algorithm finds optimal solutions in selected cases and good approximate solutions otherwise.

This research was supported by Cisco, NASA (NAG2-1249), and NSF (ANI-9805436).

This paper is organized as follows. In Section II we review the problem of routing with multiple constraints. We also present the Multiple Constraints Bellman-Ford algorithm, used in conjunction with OSPF to generate routes for IP Telephony streams. In section III, we define the network models used in our experiments. In Section IV we evaluate the OSPF scheme equipped with the MC-Bellman-Ford route computation by simulation.

II. QoS Routing Algorithm

The Bellman-Ford (BF) algorithm can potentially solve a two metric routing problem in polynomial time, when one of the metrics is the hop count [8]. This fact favors the Bellman-Ford algorithm when compared with the Dijkstra algorithm, which lacks this capability [5].

A. MC-Bellman-Ford Routing Algorithm

Consider the network shown in Fig. 1. Labels represent link delays. Let us assume we wish to compute a path between nodes 1 and 4, subject to delay bound D. We look for the minimum hop path with delay smaller or equal to D. Our choice of path will obviously depend on the value of D. If $D > 11$, the minimum hop path $(1, 2, 4)$ shown with a solid line should be used. If $7 \leq D < 11$, path $(1, 3, 5, 4)$ should be used. If $D = 6$, the dashed path $(1, 2, 3, 5, 4)$ is the only alternative. For delay requirements smaller than $D = 6$, no path exists. Note that delay decreases at each hop iteration, as shown in Fig. 2.

The original Bellman-Ford (BF) algorithm was designed to compute shortest paths between a given vertex s and multiple destinations. It did not include multiple metrics. Moreover, path cost function was always considered to be of additive nature, as the delay metric above. In [5], we show that we can extend the BF algorithm to handle multiple metrics of the types previously defined. More precisely, we can handle multiple "additive" constraints, such as delay, loss, jitter. The "max" or "min" constraints, such as the bandwidth constraint, are easily dealt with in the obvious way, by pruning the links without enough resource.

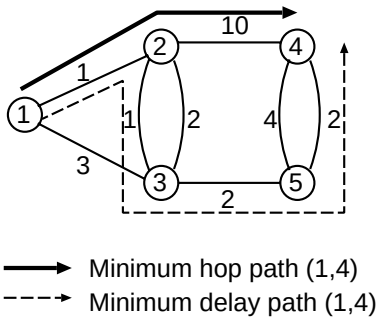

Fig. 1. Delay-hop routing example

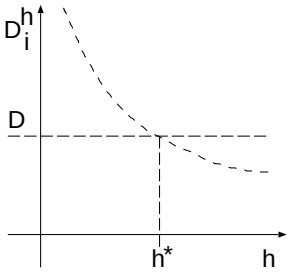

Fig. 2. Non-increasing cost function

Without loss of generality, we assume vertex 1 be the vertex from which we compute distances to all other vertices. As usual in BF algorithm, we define D_i^h as the minimum distance with respect to some metric d from vertex i to vertex 1 with at most h hops. The Bellman-Ford equation [6] is:

$$D_i^{h+1} = \min_{j \in N(i)} [d(i,j) + D_j^h], \qquad \forall i \neq 1 \tag{1}$$

where $N(i)$ is the set of neighbors of vertex i.

We start with initial conditions: $D_i^0 = \infty$ and $D_1^h = 0$. BF algorithm iteratively applies eq. (1) until a predefined number of hops H_{max} has been reached, $H_{max} \leq N$.

One of the claims in [5] is that the BF algorithm can compute the minimum hop path among all paths which satisfy a specific delay bound D. Namely:

Theorem 1: The BF algorithm outputs the minimum-hop path with $cost \leq D$ the very first time $D_i^h \leq D$.

Proof of Theorem 1: We run BF algorithm, until D_i^h falls below D, say at h^* for the first time. Since the number of hops always increases as BF progresses, and by assumption h^* is the point at which D_i^h falls below D for the first time, then h^* is the minimum hop length for a path satisfying D.

Fig. 3 illustrates successive iterations of the Bellman-Ford algorithm for the example shown above.

B. Q-OSPF Implementation

In order to extend OSPF to operate in a QoS mode and thus upgrade it to Q-OSPF, we must first enable it to compute QoS

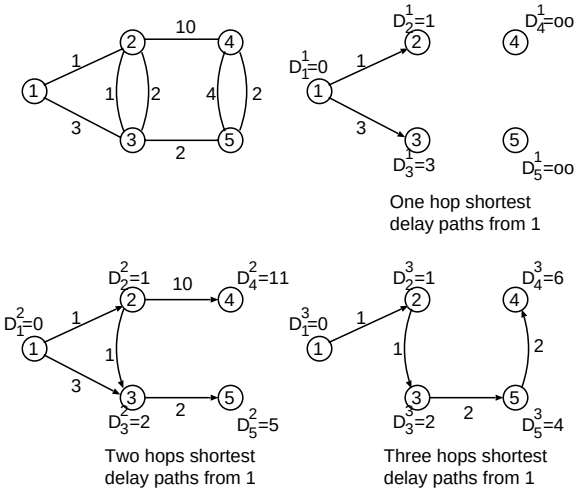

Fig. 3. Successive iterations of Bellman-Ford Algorithm

routes. This implies piggybacking link quality metrics on the link state updates. To this end, we augment the OSPF link state dissemination protocol to include available link bandwidth, and packet delay. Routers measure link utilization for each class over a history window of 10 seconds, and compute available bandwidth using exponential averaging. Likewise, routers keep track of outgoing queue sizes for the various classes, over a 10 second window and derive average delay. Link state advertisements with residual bandwidth and average delay are thus constructed.

In OSPF, a router periodically floods link state updates to all other routers in the routing domain. We have modified the OSPF protocol so that link state updates are sent frequently enough so as to track rapidly changing link parameters, such as bandwidth and delay. Namely, we have changed the update interval from 30 minutes to 2 seconds. Moreover, we have suppressed acknowledgments. The disseminated link state information is eventually collected at each router and entered in a database. This database represents a network topology map available at each router. Using the local data base, routers compute, in a decentralized way, constrained shortest paths to each destination using the MC Bellman Ford algorithm.

Another important implementation issue is the forwarding of packets on the newly computed QoS route. Our simulation is source routing, which is supported in IPv4 as well as IPv6.

C. Admission Control

The edge router performs Call Admission Control (CAC) on the incoming voice calls based on the bandwidth and delay information provided by OSPF. More precisely, upon receiving a call request from the subscriber network, the router invokes a QoS route computation. If a feasible path is found, the call is forwarded onto such path using source routing. Else, the call is rejected. A path is feasible if it satisfies the end-to-end delay constraint, and if it has residual bandwidth $\geq BR$.

The choice of the residual bandwidth BR is dictated by the need to guarantee adequate performance to the new coming connection and at the same time protect the performance of ongoing connections.

III. Network Models

Our simulation environment has three levels of non-preemptive priority. Priority 1 handles QoS sensitive traffic (such as IP telephony) and signaling (OSPF). Priority 2 is reserved for future use, and finally Priority 3 handles best effort traffic, such as FTP, telnet etc. Each queue has a 1 MB buffer.

As for traffic source models, voice connection requests arrive according to a Poisson process. Once a connection is established, the voice source is modeled as 2 state Markov chain with one of the states representing a silent state and the other state representing the "talk spurt" state [15]. The "talk spurt" state duration is exponentially distributed with average of 352 ms. The silent state is also exponentially distributed with average of 650 ms. Each voice call uses PCM encoding, transmitting data at 64 Kbps in talk spurt state. The voice stream is packetized into 180 byte packets, with 160 byte payload and 20 byte IP overhead. Hence 50 packets per second are generated during the talk spurt state. The average voice connection bit rate is 25.3 Kbps in each direction.

In some experiments, TCP traffic is also present. TCP sources correspond to persistent FTP connections, with infinite backlog and data rate limited only by link rates and window size.

IV. Simulation Results

The simulation environment was written in PARSEC, a parallel simulation language developed at UCLA, which allows for simple message passing and message scheduling [2]. The objective of the simulation experiments is to show how a typical Administrative Domain can benefit from QoS OSPF in the support of IP Telephony. Throughout the experiments, we chose a highly connected topology (Fig. 4, which offers potential for bottlenecks and for challenging alternate routes. Finding these routes will represent a good test for our QoS routing algorithm. All links are assumed to be 15 Mbps. Propagation delay varies from link to link, reflecting a regional network scenario.

A. Homogeneous Traffic - IP Telephony only

In this first set of experiments, we use only voice sources (no TCP connections). In the first run, aggregate voice call generation interval is 150 milliseconds. A new voice connection request is generated between a randomly chosen source and destination pair within a predefined set of candidates. The candidate source destination pairs are: $(8, 20), (0, 34), (3, 32), (4, 32), (5, 32), (10, 32), (16, 32)$. The experiment duration is 10 minutes of simulated time. The first 3 minutes of the simulation experiment are not considered in the collection of results in order to allow the system to reach steady state. The QoS constraint for the voice sources was 100 ms end-to-end delay and $BR = 3$ Mbps minimum residual bandwidth requirement. In parallel with the QoS routing algorithm, also the minhop routing algorithm is simulated, for the sake of comparison with current IP Telephony practices. In the minhop case, OSPF updates were sent every 30 minutes, according to OSPF specifications [11]

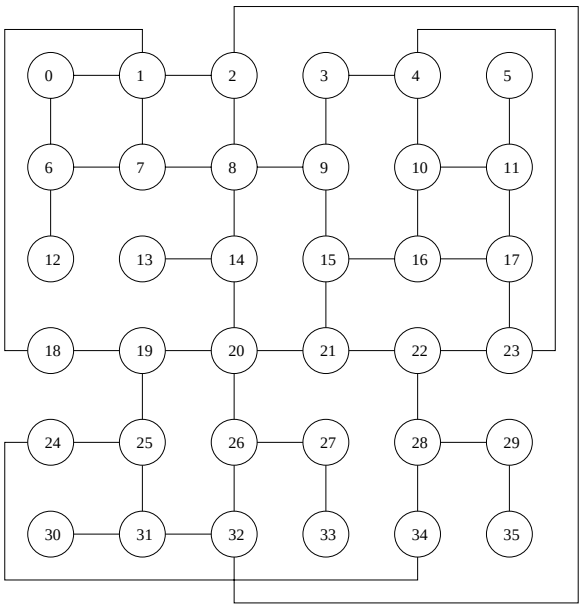

Fig. 4. Simulation topology

Before presenting the simulation results, we compute some key parameters which will be useful for the interpretation of the numbers to be presented. From Little's results [9] we find that, with the minhop routing strategy, a 15 Mbps link can process at saturation at most C_{mh} calls/sec, where $C_{mh} = (15.000/25.3)/180 = 3.2$ calls/sec. With the QoS routing policy, i.e., the MC Bellman-Ford algorithm, the corresponding maximum processing rate is $C_{mh} = (12.000/25.3)/180 = 2.6$ calls/sec. Also, note that in the minhop routing strategy, when the link load approaches 3.2 calls/sec, voice packets are dropped with increasing rate, since voice calls are never denied service (no CAC). In the MC Bellman-Ford strategy, CAC is exercised. With adequate bandwidth and buffer margins, no packets are dropped. Indeed, calls are rejected at the source.

We note that the rate from the 5 sources $(3, 4, 5, 10, 16)$ into destination 32 is 4.76 calls/sec, exceeding the processing capacity of a single path which is 3.2 in minhop and 2.6 in MC BF. The results in Table I show that the MC-Bellman-Ford algorithm can accept all the offered calls by spreading the load on alternate paths. On the other hand, the minhop scheme experiences packet loss due to hot spots created by the routing strategy. Notice that the QoS Routing strategy complies with the delay constraint despite the overhead of transmitting a much larger number of LSA packets (once every 2 secs, against once every 30 minutes in the minhop strategy). Also in this experiment we try using the minhop scheme with Call Acceptance

TABLE I
HIGHLY CONGESTED NETWORK (.15 SEC INTERARRIVAL)

	QoS Routing	Minhop	Minhop w/ CAC
# voice calls attempted in steady state	2762	2762	2790
# voice calls accepted in steady state	2762	2762	1875
%of packets lost	0.0%	11.78%	0.0%
%of packets above 100 ms threshold	0.0%	51.34%	0.0%
%of packets below 100 ms threshold	100.0%	36.88%	100.0%

Minhop OSPF vs. MC Bellman-Ford

TABLE II
OVERLOADED NETWORK (.050 SEC INTERARRIVAL)

Band. Margin	3 Mbps	1 Mbps	0.5 Mbps
# voice calls attempted in steady state	8309	8589	8185
# voice calls accepted in steady state	5473	6417	6388
fraction of accepted calls	0.066	0.75	0.78
%of packets lost	0%	0.08%	0.43%
%of packets above 100 ms threshold	0%	22.17%	46.32%
%of packets below 100 ms threshold	100%	77.75%	53.24%

Effect of reduced bandwidth margin

Control, and even though all packets are below 100 ms delay threshold, there is less calls accepted than in MC Bellman-Ford scheme because MC Bellman-Ford scheme uses alternate path routing which provides additional capacity for the voice connections.

We have also evaluated, for the overload case, the impact of bandwidth margin BR on two key performance measures: number of accepted calls, and; packet loss rate. We recall that the margin BR is essential in keeping packet loss in check. Table II reports fraction of accepted calls and packet loss rate as BR ranges from 0 to 3 Mbps. As expected, we note that the fraction of accepted calls increases and packet loss increases as the margin BR is decreased. It should be noted that when the margin is low, in the limit zero, the bottleneck queue increases and the end to end average delay constraint (in our case 100 ms) is violated. Thus, CAC intervenes and calls are rejected at the source alleviating packet loss.

The previous experiments have focussed on the benefits of Q-OSPF in providing alternate paths in worst case traffic patterns when hot spots are likely. When the traffic pattern is well distributed (eg, uniform), one might expect that the alternate path advantage of Q-OSPF is much reduced since hot spots are less frequent and the minhop routing is quasi-optimal. To verify this property, we have generated a uniform traffic pattern with an offered load high enough to cause overload. For each pattern we have compared two routing strategies: Q-OSPF with multiple path routing, and; Q-OSPF with only the minhop routing enabled. The results are shown in Table III. We note that minhop accepts more calls than QoS Multipath routing. Moreover, the minhop delay performance is slightly better than QoS Multipath. Clearly, this more than proves our hypothesis that QoS routing is not much more efficient than minhop in uniform traffic patterns (when CAC is exercised in both). The fact that QoS Multipath performs **worse** than minhop is however quite surprising at first. Indeed, this can be easily explained by observing that the average hop count for the calls accepted by QoS Multipath is much larger than that of minhop (4.49 vs. 3.66). This is because the QoS Multipath scheme will accept calls (which would be normally dropped by minhop) and reroutes them on longer paths. This causes a somewhat less efficient usage of link resources and leads to a lower throughput in terms of number of accepted calls.

TABLE III
UNIFORMLY DISTRIBUTED TRAFFIC - OVERLOADED NETWORK (.010 SEC INTERARRIVAL)

	QoS Routing	Minhop w/ CAC
# voice calls attempted in steady state	40871	40860
# voice calls accepted in steady state	18245	21410
# average hop count per voice call path	4.49	3.66
%of packets lost	0.0%	0.0%
%of packets above 100 ms threshold	0.21%	0.0%
%of packets below 100 ms threshold	99.79%	100.0%

Uniformly distributed voice call generation

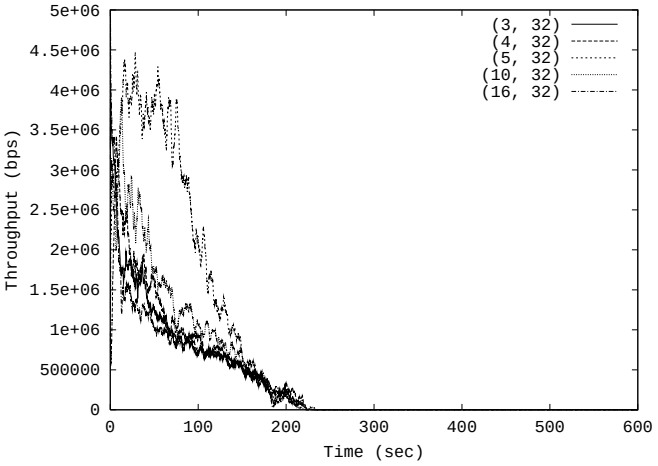

Fig. 5. FTP throughput with the Minhop routing

B. *Integrated Traffic - IP Telephony and FTP*

In this set of experiments, TCP traffic is present in the form of FTP connections. The same seven source destination pairs now generate both voice calls as well as TCP traffic. The voice call generation interval is 150 ms.

In the first TCP experiment, we assume conventional OSPF without QoS support. Thus, the minhop routing is used. Fig. 5 shows the throughput achieved by the FTP sources with the minhop routing, as a function of time.

When the minhop routing is used, voice flows $(8, 20)$ and $(0, 34)$ account for 4.3 Mbps (180×25.344) each along their respective routes, leaving 10.7 Mbps available. Low priority FTP traffic sources manage to acquire that bandwidth over the

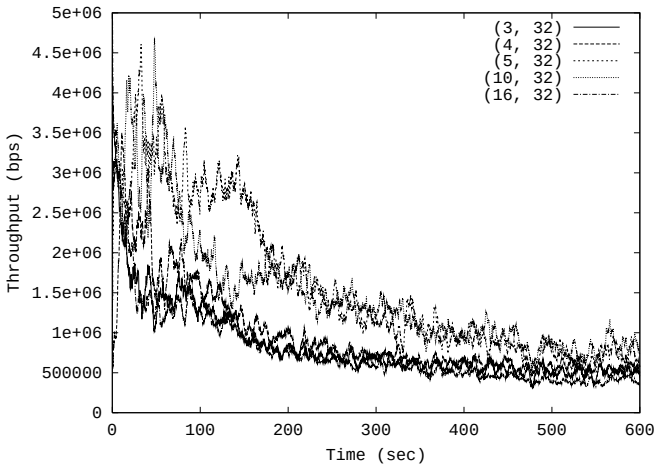

Fig. 6. FTP throughput with MC BF routing

TABLE IV
FTP THROUGHPUT OF THE FIVE CONNECTIONS OVER THE LAST 7 MIN

(src, dst)	Throughput
$(3, 32)$	497522 bps $\simeq$ 498 Kbps
$(4, 32)$	581776 bps $\simeq$ 582 Kbps
$(5, 32)$	753366 bps $\simeq$ 753 Kbps
$(10, 32)$	855171 bps $\simeq$ 855 Kbps
$(16, 32)$	418255 bps $\simeq$ 418 Kbps
Sum	3106090 bps $\simeq$ 3.1 Mbps

same routes. The remaining five source/destination pairs all share a common bottleneck link. Each pair offers a voice load of 4.3 Mbps. Thus, the bottleneck capacity of 15 Mbps is completely saturated, leaving no room for low priority FTP traffic. As expected, FTP throughput on such pairs drops to zero at steady state, as shown in Fig. 5.

Fig. 6 shows FTP throughput results when the Q-OSPF strategy with the MC-Bellman Ford routing algorithm is used. In this case, once the available bandwidth on the bottleneck shared by the five voice connections falls below 3 Mbps, the routing algorithm finds other, less congested paths for the voice flows. The FTP flows which share the bottleneck now get a chance of transmitting some traffic (as shown in Fig 7), since some of the bottleneck bandwidth (3 Mbps) has been left aside by the CAC algorithm. Table IV reports the throughput of these FTP connections at steady state. Note that the aggregate throughput is 3.1 Mbps, showing that the bandwidth margin (3 Mbps) is fully utilized and even exceeded.

V. CONCLUSION

The experimental results clearly show that QoS OSPF is beneficial to IP Telephony in many ways. It permits to exercise effective Call Admission Control (CAC), thus protecting the quality of ongoing calls. It alleviates hot spots caused by non-uniform traffic patterns by spreading traffic on alternate routes. As a byproduct of CAC, QoS routing can also be used to set aside some amount of resources to best-effort traffic.

The results supported here are preliminary in nature. They serve to illustrate the advantages of QoS routing. Work is in progress in several directions: (a) Implementation of more efficient packet forwarding solutions (i.e., MPLS) than source routing; (b) Scaling of QoS OSPF to large topologies; (c) Extensions to interdomain routing.

REFERENCES

[1] G. Apostolopoulos, R. Guerin, and S. Kamat, *Implementation and Performance Measurements of QoS Routing Extensions to OSPF*, Proceedings of Infocom99, New York, 1999.
[2] R. Bagrodia, R. Meyer, M. Takai, Y. Chen, X. Zeng, J. Martin, B. Park, and H. Song, *Parsec: A Parallel Simulation Environment for Complex Systems*, Computer, Vol. 31(10), October 1998, pp. 77-85.
[3] Y. Bernet, J. Binder, S. Blake, M. Carlson, B. Carpenter, S. Keshav, E. Davies, B. Ohlman, D. Verma, Z. Wang, and W. Weiss, *A Framework for Differentiated Services*, Internet Draft, work in progress.
[4] D. Bertsekas and R. Gallager, *Data Networks*, Prentice-Hall.
[5] D. Cavendish and M. Gerla, *Internet QoS Routing using the Bellman-Ford Algorithm*, Proceedings of IFIP Conference on High Performance Networking, Austria, 1998.
[6] Cormen, Leiserson and Rivest, *Introduction to Algorithms*, McGraw-Hill, 1990.
[7] M. R. Garey and D. S. Johnson, *Computers and Intractability*, Freeman, San Francisco, 1979.
[8] R. Guerin, A. Orda, and D. Williams, *QoS Routing Mechanisms and OSPF Extensions*, In Proceedings of IEEE GLOBECOM97, Vol. 3, pp. 1903-1908, Phoenix, Arizona.
[9] L. Kleinrock, *Queueing Systems Volume I: Theory*, Wiley-Interscience, ISBN 0-471-49110-1.
[10] C. Pornavalai, G. Chakraborty, and N. Shiratori, *QoS Based Routing Algorithm in Integrated Services Packet Networks*, In Proceedings of International Conference on Network Protocols, Atlanta, Georgia, pp. 167-175.
[11] J. Moy, *OSPF Version 2*, RFC 2328, April 1998.
[12] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, *RTP: A Transport Protocol for Real-Time Applications*, RFC 1889, January 1996.
[13] R. Braden, D. Clark, and S. Shenker, *Integrated Services in the Internet Architecture: an Overview*, Internet RFC 1633, June 1994.
[14] E. Rosen, A. Viswanathan, and R. Callon, *Multiprotocol Label Switching Architecture*, Internet Draft, work in progress.
[15] M. Schwartz, *Broadband Integrated Networks*, Prentice Hall, ISBN 0-13-519240-4.
[16] W. Zhao and S. K. Tripathi, *Routing Guaranteed Quality of Service Connections in Integrated Services Packet Networks*, In Proceedings of International Conference on Network Protocols, Atlanta, Georgia, pp. 175-182.