Distance-vector QoS-based Routing with Three Metrics *

Luís Henrique M. K. Costa[1,2], Serge Fdida[1], and Otto Carlos M. B. Duarte[2]

[1] LIP6 – Université Pierre et Marie Curie
4, place Jussieu - 75252 - Paris Cedex 05 - France
{Luis.Costa,Serge.Fdida}@lip6.fr
[2] Grupo de Teleinformática e Automação – GTA
Universidade Federal do Rio de Janeiro – COPPE/EE
P.O. Box 68504 - 21945-970 - Rio de Janeiro - RJ - Brasil
otto@gta.ufrj.br

Abstract Quality of Service (QoS) based routing provides QoS guarantees to multimedia applications and an efficient utilization of the network resources. Nevertheless, QoS routing is likely to be a costly process that does not scale when the number of nodes increases. Thus, the routing algorithm must be simple and a class-of-service routing is an alternative to provide QoS guarantees instead of per-flow routing. This paper proposes and analyzes the performance of a distance-vector QoS routing algorithm that takes into account three metrics: propagation delay, available bandwidth, and loss probability. Classes of service and metric-combination are used to turn the algorithm scalable and as complex as a two-metric one. The processing requirements of the proposed algorithm and those of an optimal approach are compared and the results show an improvement up to 50%.

1 Introduction

To facilitate the use of multimedia applications in the Internet, new characteristics and services must be added. New classes of service should be offered to provide better guarantees of Quality of Service (QoS). An important issue for that purpose is the definition of a routing architecture that considers the QoS requirements of applications. In traditional routing, packets are delivered using a route based on their source and destination addresses while in QoS-based routing the traffic requirements are also taken into account by the routing algorithm.

In classical routing, the number of hops is the metric used to make route decisions. QoS-based routing is more complex because the routing decisions are based on n metrics. Scalability is a main issue of QoS-based routing [1,2]. The routing algorithm must be simple. Some authors consider the use of precomputed paths to minimize the setup overhead [2,3]. Scalability also relates to the size of routing tables, important even in traditional routing. The uncertainty of QoS

* Sponsors: UFRJ, CNPq, FAPERJ, CAPES, REENGE, and COFECUB.

parameters may also affect QoS routing, e.g., in a link-state protocol the network topology database may be out of date [4,5]. The routing algorithm may take into account these uncertainties. Setting the period of update messages is a tradeoff between the accuracy of QoS parameters and the protocol overhead [2,6].

QoS routing is likely to be an expensive process [2] – it should not keep per-flow state in routers, as this approach is unscalable [7,8]. A more suitable solution is to use per-flow state only for special flows (flows that have stringent QoS requirements which demand a value-added service). Less demanding flows can be satisfied with a connectionless service, where routing is done per class of service and not per flow. In this work a class of service is defined by specific limits on a set of QoS requirements, e.g., end-to-end delay and available bandwidth. The application chooses the class of service that fits the best its requirements.

This paper discusses the approaches found in the literature and proposes an algorithm for unicast route computation based on distance-vectors. The algorithm is based on the SMM approach first presented in [9], where the source routing version was proposed. This paper presents the distance-vector algorithm (SMM-DV) and results obtained from simulation. SMM-DV was implemented as an additional module to NS-2 (Network Simulator) [10]. The performance analysis compares SMM-DV with an optimal algorithm for three-metric route computation. The objective is to evaluate the processing requirements of the proposed algorithm. The results show an improvement up to 50% for SMM-DV.

The rest of the paper is organized as follows. Sections 2 and 3 discuss the main issues related to QoS routing. Section 4 presents our SMM-DV proposal. The proposed approach was implemented and tested, as shown in Section 5. Section 6 presents some final considerations.

2 Routing Model

Several routing protocols were developed for packet switched networks [11]. They are classified in distance-vector based (DV) and link-state based (LS), and they use variants of the Bellman-Ford and of the Dijkstra algorithm. There are two approaches for packet forwarding: source routing and hop-by-hop routing. In the first one, the packet header has a list of routers that must be traversed from the source to the destination (the source chooses the route). In hop-by-hop routing, each router chooses the next hop that a packet will follow. The routing model impacts QoS-based routing, as discussed in the next two sections.

2.1 Source routing versus hop-by-hop routing

Since QoS requirements are diverse, it is difficult to specify a set of requirements that applies to most applications. Therefore source routing, which computes forwarding paths on-demand on a per-flow basis, fits well. QoS-based source routing can be thought as a way to implement QoS paths for specific applications in enterprise networks. The source-routing concept is present in explicit routing [12], which provides fixed paths to applications.

On the other hand, hop-by-hop routing allows distributed computation, produces smaller overhead, and reduces setup delays. It is currently the most common forwarding paradigm whereas source routing is only used for specific applications. This is due also to security issues, since to setup a route at the source one must know the entire topology. In general this information is not available, as many organizations may not want to divulge their internal topology.

2.2 DV versus LS routing

In DV routing each node knows its direct neighbors and keeps a database with the distance (hop count) to reach destinations in the network. Each node constructs distance-vectors and sends it to his neighbors. Routes are calculated by examining the vectors received and choosing the shortest path. DV protocols have problems such as slow convergence and creation of temporary forwarding loops. Several improvements were proposed to solve these problems [11]. RIP (Routing Information Protocol) [13] is nowadays the most used protocol in the Internet for intra-domain routing despite the growth of OSPF (Open Shortest-Path First) [14], a LS protocol.

In the LS model each router maintains a database with the complete network topology where each link is associated with a state. Link-state information is exchanged between routers. Since each node knows the entire topology, loops are easily avoided, the routing convergence is faster than with DV protocols, and source routing is easily implemented. However, LS protocols are inadequate for inter-domain routing because of the scalability and security issues involved. For inter-domain routing, protocols are currently being deployed based on variations of the DV algorithm, such as the path-vector approach [15]. Since in the LS model routers know the topology, it is easy to do per-flow routing [16]. On the other hand, for DV protocols the class-of-service approach is better suited.

3 Multi-constrained Routing

Route computation constrained by multiple QoS restrictions is complex [17,18]. The algorithms may treat the metrics in different ways. The first difference is related to the metric correlation. End-to-end transmission delay depends on the link capacity, the reserved bandwidth, the propagation delay, the link loss probability, and the traffic model employed. Ma and Steenkiste [17] proposed some algorithms based on dependent metrics but defining the metric relationship is a major problem.

Another way is to use metrics that are independent from each other. For instance, propagation delay is independent from the available bandwidth. In this case, the computational complexity of the algorithm is the main problem. Wang and Crowcroft [18] showed that for certain combinations of QoS requirements the problem is NP-complete.

This work considers independent metrics. The proposals found in the literature treat the metrics in one of three ways: individually, as an overall cost, or by means of heuristics. The next sections discuss these approaches.

3.1 Multiple metrics

The complexity of the algorithm is function of the number of metrics treated and of their composition rules. The most usual requirements are delay, jitter, cost, packet loss probability, and bandwidth consumption. These metrics have different composition rules. If the metric of a path is the minimum value between all link metrics, the composition rule is concave. The additive and multiplicative cases are straightforward. Available bandwidth is concave. Delay, jitter, and cost are additive. The composition rule of loss probability is more complex. Let p be a path, $lp(p)$ the path metric, and $lp(i,j)$ the metric associated with the link between node i and node j. The composition rule for loss probability is given by

$$lp(p) = 1 - ((1 - lp(i,j)) \times (1 - lp(j,k)) \times ... \times (1 - lp(q,r))). \quad (1)$$

However, if the transmission-success probability is used instead of the loss probability, the composition rule turns multiplicative.

Wang and Crowcroft [18] proved that finding a path subject to constraints on n additive and m multiplicative metrics is NP-complete if $n + m >= 2$. Since bandwidth is concave, pruning infeasible links may treat it. Therefore, algorithms of reasonable complexity can only handle bandwidth and another metric.

3.2 Single mixed metric

Since algorithms based on a single metric are well known, a direct approach is to map the user's QoS requirements into a single measure and use it as metric. Given a path p, a mathematical function proportional to the bandwidth ($B(p)$) and inversely proportional to the delay ($D(p)$) and to the loss probability ($L(p)$) can be defined (Equation 2). The path with the greater $f(p)$ is the best route.

$$f(p) = \frac{B(p)}{D(p) \times L(p)}. \quad (2)$$

The single mixed metric has two problems. First, maximizing $f(p)$ does not guarantee each of the QoS requirements. Second, the composition rule of this function is hard to define since the composition rules of its parameters are different. Nevertheless, the single mixed metric is used in many proposals [16,19]. This approach does not guarantee each QoS requirement individually, but adjusting the metric weights in the overall cost may increase the importance of a particular requirement.

3.3 Heuristics

There are different heuristic proposals for QoS routing [15,20,21]. The algorithm in [19,20] uses one metric to find a path, and then it tests if this path respects the other QoS requirements. If not, it uses another metric as the routing decision factor, and so on. When a feasible path is found the algorithm stops.

In general, these approaches may be based on two hypotheses. First, they may suppose centralized routing, i.e., the router knows the entire topology [19,20]. This is true for LS routing but not for DV routing. Second, some approaches

try to find a feasible path: this may lead to problems in hop-by-hop routing as route decisions made by all routers must be coherent at the risk of not being able to avoid loops. The use of feasible paths supposes some kind of connection-oriented behavior or source routing [15,21]. These hypotheses are not valid for hop-by-hop packet forwarding and DV routing. This paper proposes a routing algorithm adapted to hop-by-hop routing based on distance-vectors.

4 The SMM-DV Algorithm

In a companion paper, we proposed the Single Mixed Metric (SMM) routing [9]. This proposal is a simple source-routing algorithm that finds routes subject to three constraints: bandwidth, delay, and loss probability. The key idea is to execute a search in the network graph to eliminate all arcs that do not meet the bandwidth restriction and then to run a modified version of Dijkstra's algorithm. In this scheme, the source knows the network topology.

In this paper we propose and analyze the performance of the Single Mixed Metric - Distance Vector (SMM-DV) algorithm for hop-by-hop routing. SMM-DV uses a single mixed metric that combines delay and loss probability. Instead of loss probability, the algorithm uses the logarithm of transmission-success probability function ($slog$) to avoid complex composition rules. Furthermore, we assume that routes can not have more than 90% of loss probability, that is, $0 < |slog| < 1$. As ms unit is likely to be appropriate [22] to represent delay, we assume that delay is an integer. As a consequence, we have a simple single-metric representation of delay and loss, where the integer part is delay and the decimal part represents loss.

The usual approach in current hop-by-hop routing algorithms is to compute the best path to every destination. With a single metric, the best path is easily defined. With multiple metrics, however, the best path with optimal values for all parameters may not exist at all.

Actually, the objective of QoS routing is not to find the optimal path, but a feasible path. Nevertheless finding a feasible path is inadequate for distributed route computation based on distance-vectors. Permanent loops may be formed since routers do not know the entire topology. To avoid this, we adopted the approach of [18]. The shortest-widest path is loopless [18]. In fact, the "shortest" property avoids loops. This design decision guarantees the absence of loops at the expense of not finding some feasible paths.[1] In the algorithm that follows, the bandwidth precedes the delay metric. This centralized version of the algorithm is based on the Bellman-Ford's algorithm for distance-vector route computation.

Let $G = (N, A)$ be a network with N nodes and A arcs. Each arc (i, j) is assigned two real numbers: b_{ij} is the available bandwidth and $u_{i,j} = slog_{i,j} + d_{i,j}$ is the single mixed metric, where $slog_{i,j}$ is the logarithmic transmission-success probability and $d_{i,j}$ is the propagation delay.

[1] E.g., the shortest-widest route may not meet the delay constraint, while another route with less bandwidth (although still enough) meets the delay requirement.

When arc (i,j) is inexistent $b_{ij} = 0$ and $u_{ij} = \infty$. Given a path $p = (i,j,k,...,q,r)$, the path width, $w(p)$, is $min[b_{ij}, b_{jk}, ..., b_{qr}]$ and its length is $l(p) = u_{ij} + u_{jk} + ... + u_{qr}$. Given three constants, $\mathbf{B}$ the minimum bandwidth, $\mathbf{D}$ the maximum delay, and $\mathbf{P}$ the maximum logarithmic transmission-success probability ($\mathbf{U} = \mathbf{D} + \mathbf{P}$ is the maximum single mixed metric), the problem is to find a path, p, between i and r such that $w(p) \geq \mathbf{B}$ and $l(p) \leq \mathbf{U}$: in our case, the shortest-widest path with three constraints.

Suppose node 1 is the source and h the number of arcs away from this node. Let $B_i^{(h)}$ and $U_i^{(h)}$ be the width and the length of the shortest-widest path from node 1 to node i within h hops. By convention, $B_1^{(h)} = \infty$ and $U_1^{(h)} = 0$ for all h. Adding a length checking, when multiple widest paths are found, produces the shortest-widest path algorithms. The SMM-DV algorithm is as follows:

`Step 1:` $\forall i \neq 1 : h = 0$ `and` $B_i^{(0)} = 0$
`Step 2:` $\forall i \neq 1$, `find`
$K = \{k \mid w(1,...,k,i) = \max_{1 \leq j \leq N}[\min[B_j^{(h)}, bji]] \wedge w(1,...,k,i) \geq \mathbf{B}\}$.
`Step 3:` $\forall i \neq 1$, `if` $\#K > 1$, `find` $k \in K \mid$
$l(1,...,k,i) = \min_{1 \leq j \leq N}[U_j^{(h)} + u_{ji}] \wedge l(1,...,k,i) \leq \mathbf{U} \wedge slog(1,...,k,i) \leq \mathbf{P}$.
`Step 4:` $B_i^{(h+1)} = w(1,...,k,i)$ `and` $U_i^{(h+1)} = l(1,...,k,i)$.
`Step 5: If` $h \geq A$, `the algorithm stops.`
`If not,` $h = h + 1$ `and go to step 2.`

Step 2 finds the maximum-bandwidth paths from node 1 to node i within h hops (and tests if the bandwidth is enough). When multiple widest paths are found, the minimum-length path is chosen (step 3). Step 3 also checks if the chosen path has a $slog$ value smaller than $\mathbf{P}$, where $slog_{ij}$ is the decimal part of u_{ij}. Thus the loss probability is guaranteed without any additional search. Step 4 updates width and length of the path from node 1 to node i.

The optimal approach takes into account the metrics separately. This leads to a similar algorithm, where step 2 chooses the widest path, step 3 chooses the shortest path (minimum delay) and an additional step chooses the path with minimum loss probability. This additional search turns the approach unscalable. The optimal algorithm has two path lengths instead of one, so $U_i^{(h)}$ is replaced by $D_i^{(h)}$ and $Slog_i^{(h)}$. The optimal algorithm is as follows:

`Step 1:` $\forall i \neq 1 : h = 0$ `and` $B_i^{(0)} = 0$
`Step 2:` $\forall i \neq 1$, `find`
$K = \{k \mid w(1,...,k,i) = \max_{1 \leq j \leq N}[\min[B_j^{(h)}, bji]] \wedge w(1,...,k,i) \geq \mathbf{B}\}$.
`Step 3:` $\forall i \neq 1$, `if` $\#K > 1$, `find`
$Y = \{y \in K \mid d(1,...,y,i) = \min_{1 \leq j \leq N}[D_j^{(h)} + d_{ji}] \wedge d(1,...,y,i) \leq \mathbf{D}\}$.
`Step 4:` $\forall i \neq 1$, `if` $\#Y > 1$, `find` $y \in Y \mid$
$slog(1,...,y,i) = \min_{1 \leq j \leq N}[Slog_j^{(h)} + slog_{ji}] \wedge slog(1,...,y,i) \leq \mathbf{P}$.
`Step 5:` $B_i^{(h+1)} = w(1,...,y,i)$, $D_i^{(h+1)} = d(1,...,y,i)$ `and` $Slog_i^{(h+1)} = slog(1,...,y,i)$.
`Step 6: If` $h \geq A$, `the algorithm stops.`

`If not,` $h = h + 1$ `and go to step 2.`

The distributed version of these algorithms is simple. Each router periodically receives distance-vectors from its neighbors and chooses the best route among the possible options. SMM-DV takes advantage of the proposed single metric to find the route consuming fewer steps.

5 SMM-DV Simulation

We implemented the SMM-DV algorithm in NS-2[10]. NS-2 has a dynamic implementation of the Bellman-Ford routing algorithm that is the base of our work. Our implementation consists of two new routing modules and a modified link interface that admits three metrics: propagation delay, available bandwidth, and packet loss probability. The packet classifier takes into account the packet type and maps it onto three classes of service.

Table1. Classes of Service.

	QoS guarantees		
CoS	Delay	Bandwidth	Loss probability
0	**10** ms	1 $Mbps$	0.04
1	40 ms	**5** $Mbps$	0.04
2	40 ms	1 $Mbps$	**0.01**

Each class of service (CoS) is characterized by QoS guarantees on bandwidth, delay, and loss probability. Within a specific class of service one requirement is more important than the others.[2] Table 1 presents the classes of service used in our implementation. The QoS values are not based on real applications but chosen to differentiate the classes of service. Each service has one strong guarantee: delay (CoS 0), bandwidth (CoS 1), or loss probability (CoS 2).

The optimal algorithm for CoS 1 is exactly the one presented in Section 4. Bandwidth is optimized, then delay and loss probability are minimized. The optimal algorithm for CoS 0 is obtained by inverting the order of steps 2 and 3 (so delay is first minimized); for CoS 2 step 4 precedes step 2 (so loss probability is first optimized).

The SMM-DV algorithm for CoS 1 is the one presented in Section 4, where bandwidth is first optimized, then the single mixed metric is minimized. CoS 0's algorithm first finds the paths that minimize the single mixed metric (they also minimize delay) and among these paths it chooses the one with maximum bandwidth. SMM-DV for CoS 2 finds the paths with minimum loss probability (the rational part of the single mixed metric) and afterwards the widest path among these. Delay is not minimized but the path is delay-constrained.

[2] For example, an application may have a strong requirement on the end-to-end delay but low bandwidth consumption (e.g., voice traffic). We can also imagine that for some applications the most important is to have high bandwidth available (e.g., video transmission), and real-time applications that demand high reliability (e.g., telemedicine) for which the loss probability must be minimized.

Our proposal was evaluated through the implementation of two protocols. The first one (3M) uses the optimal algorithm that searches the route using the metrics one by one. The second protocol uses the SMM-DV algorithm. SMM-DV does not find the route with optimal values for all three metrics, as does 3M, but finds a route constrained by the QoS requirements defined for each class of service. Therefore the routes may be different for some source-destination pairs.

The performance metric is the time complexity [23] expressed in steps: when a node receives routing update messages it recomputes its routes. A meter counts the number of loops needed by the routing algorithm to compute routes. This measure reflects the processing requirements of the algorithm. In the experiments link failures produced routing updates.

5.1 Topologies

Several topologies were produced by the GT-ITM topology generator [24]. The flat model was used. Nodes are placed on a plane with a grid: with a probability p a link is put between a pair of nodes; link delay is proportional to the distance between the nodes in this grid. The generator was modified to produce values for link delay, bandwidth, and loss probability. These values are used just for route computation. Physically, all links have the same bandwidth, delay, and loss probability. The objective is to measure the algorithm complexity. We generated topologies with 10, 15, 20, 25, 30, and 35 nodes. The parameter p was decreased for the 35 node topology due to limitations in NS.

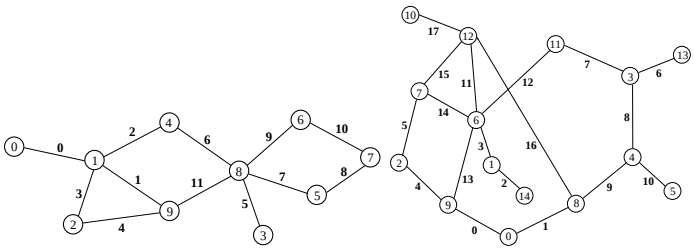


Figure1. Two of the test topologies.

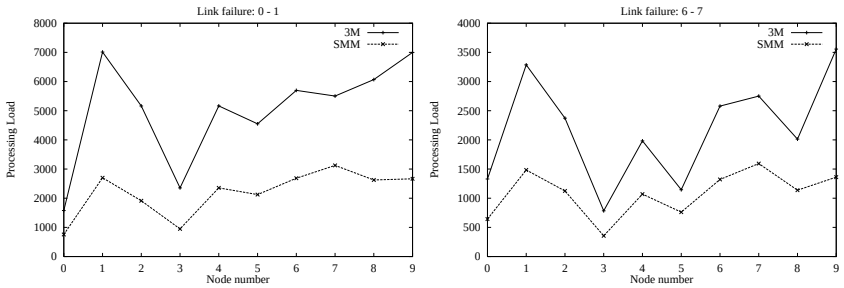
The model used does not represent well the Internet structure since it is purely random. Nevertheless it can model one routing domain: it is the approach used in [24], where Internet topology generation is studied. The model can give some clues on the issues involved in the deployment of inter-domain QoS routing. Despite its limitations, this model was useful in analyzing the complexity of the algorithm regarding routing dynamics.

5.2 Experiments

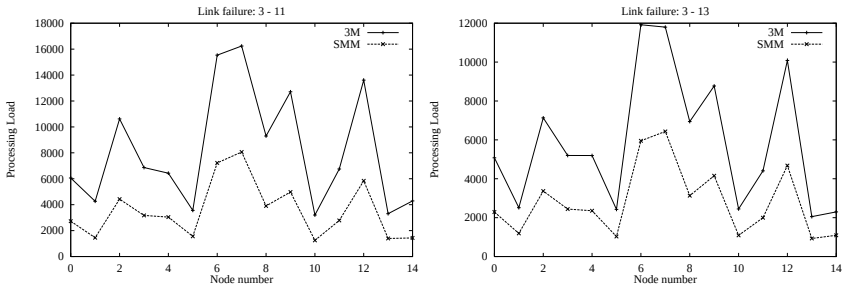
The objective of our experiments is to measure the processing requirements of QoS routing. This is done through the analysis of the trace files produced by the

meter mentioned before. Each simulation lasts for 2.5 seconds. During the first moments, all routers exchange messages that enable them to compute routes for every destination in the network. Then, at $t = 1.0s$, one link goes down, it goes up at $t = 2.0s$. The experiment is repeated for each link in the topology.

The Bellman-Ford algorithm has a problem of convergence: counting to infinity [11]. Protocol configuration limits "infinity" to a finite number (16 is usual with RIP [13]). For metrics other than the hop-count (integer) the definition of infinity is not so evident. Our first experiments presented problems due to too large values of "infinity". With slow convergence, the nodes exchange a huge number of messages, this leads to congestion which causes routing messages losses, slower convergence and the production of more routing messages; this leads to more congestion and so on. The choices of the infinity value and of the metric granularity define the convergence speed. Therefore infinity was empirically chosen: 50 for delay, 0 for bandwidth and 0.05 for loss probability. With this approach it was possible to obtain simulations with topologies up to 25 nodes without routing-packet losses. With 30 nodes, there were some losses; with 35 nodes, as "infinity" became short, some source-destination pairs became unreachable. An option to setting various infinity values is to add the hop-count metric just for counting to infinity. Metric granularity influences also link-state QoS routing since it affects the amount of update messages produced [2,6].



(a) 10-node topology.



(b) 15-node topology.

Figure2. Processing requirements at each node.

5.3 Analysis

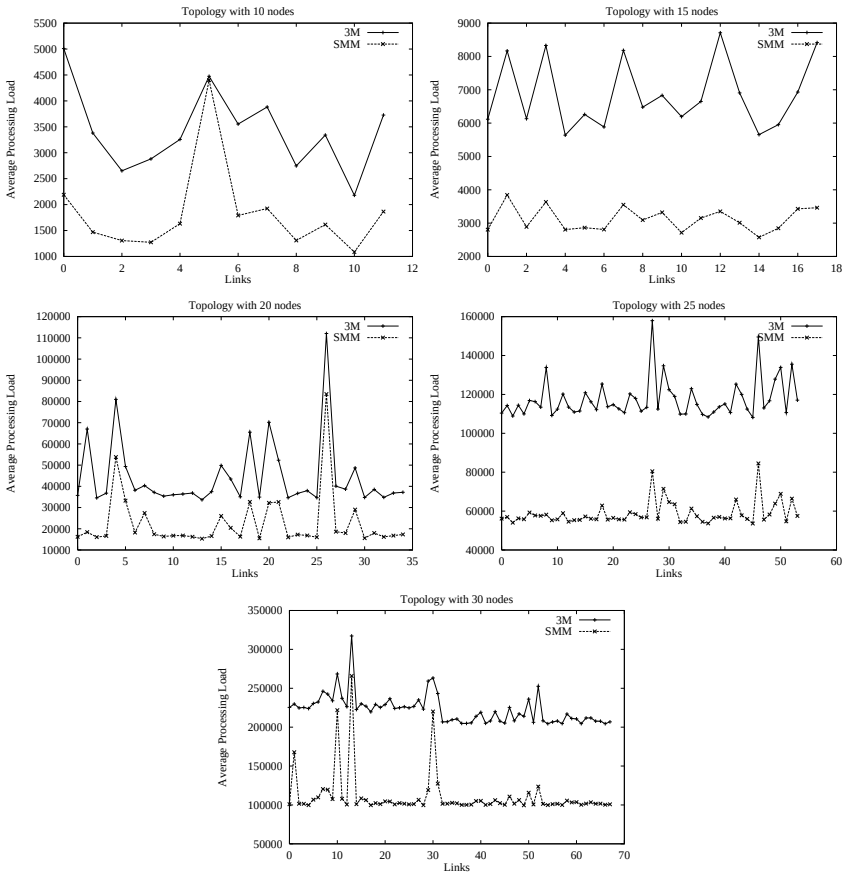


Figure3. Global processing requirements.

Figure 2 shows experiments made for the ten and fifteen-node topologies presented in Figure 1. Each graph corresponds to a specific link failure. Nodes that have more neighbors experience a higher processing charge. Intuitively, these nodes have more options in choosing the next hop to a destination (and receive more routing messages than nodes with just one neighbor). This is the case of nodes **8**, **1**, and **9** in the 10-node topology and of nodes **6**, **7**, and **12** in the 15-node topology. On the other hand, nodes that have just one neighbor have the lightest load: **0** and **3** in Figure 1 and **5**, **10**, **13**, and **14**, as well as **1** (that has two neighbors, but one of them is **14**) in Figure 1.

The failure of link 0_1 produces a higher processing charge than that of link 6_7 in the 10-node topology (the graphs in Figure 2(a) have different scales). In the first case the network is disconnected: node **0** becomes unreachable forcing the other nodes to count to infinity.

Figure 3 has one graph per topology, each one present the average processing load (on all nodes in the topology) versus the link failure.The effect of different link failures is observed. The higher processing requirements are observed:

- when the topology is disconnected (links 0 and 5 in the 10-node topology, and links 3 and 17 in the 15-node topology);
- when the link failure adds to the graph diameter, slowing the routing convergence (links 7 and 9 in the 10-node topology and links 1, 7, and 12 in the 15-node topology).

SMM-DV may perform worse than 3M - see, e.g., when link 5 in the 10-node topology fails. Although SMM-DV discovers routes more quickly, it may present a slower convergence if 3M uses the delay metric for counting to infinity and SMM-DV uses the single mixed metric which has a finer granularity. Nevertheless, this is not likely to happen as our experiments showed. The global processing requirements increase with the topology size. The advantage of SMM-DV over 3M, the optimal algorithm, is as large as 50%.

6 Conclusion

QoS routing provides better QoS guarantees to applications and improve the network resource utilization. Nevertheless QoS routing is a costly mechanism. Improved scalability can be achieved by class-of-service based routing.

Scalability and security issues arise in the use of link-state protocols for QoS routing. Storing the complete network topology does not scale and knowing the entire topology is not always the case. Therefore a distance-vector like protocol is more adapted for QoS routing in some cases.

SMM-DV is an algorithm for distance-vector QoS routing that computes routes per class of service taking into account three metrics. SMM-DV is as complex as the two-metric algorithm proposed in [18]. This paper compared the processing requirements of SMM-DV with that of an optimal algorithm, 3M. Simulations used randomly generated topologies. The average performance gain of SMM-DV is as large as 50%. We also observed that the network topology influences the processing requirements of DV QoS routing. Nodes that have more neighbors experience bigger processing charge. On the other hand, the link failure is more critical when the network is disconnected and when the network diameter is increased. Therefore DV QoS routing takes advantage of highly connected networks.

References

1. E. Crawley, R. Nair, B. Rajagopalan, and H. Sandick, *RFC2386: A Framework for QoS-Based Routing*. Network Working Group, Aug. 1998.
2. G. Apostolopoulos, R. Guerin, S. Kamat, and S. K. Tripathi, "Quality of service based routing: A performance perspective," in *ACM SIGCOMM'98*, pp. 17–28, Sept. 1998.

3. N. Traft-Plotkin, B. Bellur, and R. Ogier, "Quality-of-Service routing using maximally disjoint paths," in *IEEE/IFIP IWQoS'99*, June 1999.
4. D. H. Lorenz and A. Orda, "Qos routing in networks with uncertain parameters," in *IEEE INFOCOM'98*, 1998.
5. V. Fayet, D. A. Khotimsky, and T. Przygienda, "Hop-by-hop routing with node-dependent topology information," in *IEEE INFOCOM'99*, Mar. 1999.
6. A. Shaikh, J. Rexford, and K. G. Shin, "Evaluating the overheads of source-directed quality-of-service routing," in *International Conference on Network Protocols*, 1998.
7. R. Braden, D. Clark, and S. Shenker, *RFC1633: Integrated Services in the Internet Architecture: An Overview*. Network Working Group, June 1994.
8. S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, *RFC2475: An Architecture for Differentiated Services*. Network Working Group, Dec. 1998.
9. L. H. M. K. Costa and O. C. M. B. Duarte, "A scalable QoS-based routing mechanism for supporting multimedia applications," in *IEEE International Conference on Multimedia Computing and Systems - ICMCS'99*, vol. 2, pp. 346–351, June 1999.
10. K. Fall and K. Varadhan, *NS notes and documentation*. University of California at Berkeley/Lawrence Berkeley National Laboratory, Mar. 1999. Available at http://www-mash.cs.berkeley.edu/ns.
11. C. Huitema, *Routing in the Internet*. Prentice Hall, 1996.
12. B. Davie and J. Gibson, "Enabling explicit routing in IP networks," in *IEEE GLOBECOM'98 - Internet Mini-Conference*, pp. 125–130, 1998.
13. G. Malkin, *RFC1721: RIP Version 2 Protocol Analysis*. Network Working Group, Nov. 1994.
14. J. Moy, *RFC1583: OSPF version 2*. Network Working Group, Mar. 1994.
15. R. Vogel, R. G. Herrtwich, W. Kalfa, H. Wittig, and L. C. Wolf, "QoS-based routing of multimedia streams in computer networks," *IEEE Journal on Selected Areas in Communications*, vol. 14, no. 7, pp. 1235–1244, Sept. 1996.
16. R. Guerin, A. Orda, and D. Williams, "QoS routing mechanisms and OSPF extensions," in *IEEE GLOBECOM'97*, Nov. 1997.
17. Q. Ma and P. Steenkiste, "Quality of service routing for traffic with performance guarantees," in *IFIP Fifth International Workshop on Quality of Service*, (New York), pp. 115–126, May 1997.
18. Z. Wang and J. Crowcroft, "Quality of service routing for supporting multimedia applications," *IEEE Journal on Selected Areas in Communications*, vol. 14, no. 7, pp. 1228–1234, Sept. 1996.
19. L. L. H. Andrew and A. A. N. A. Kusuma, "Generalised analysis of a QoS-aware routing algorithm," in *IEEE GLOBECOM'98*, pp. 118–123, 1998.
20. W. C. Lee, M. G. Hluchyj, and P. A. Humblet, "Routing subject to quality of service constraints in integrated communication networks," *IEEE Network Magazine*, vol. 9, no. 4, pp. 46–55, July-Aug. 1995.
21. H. F. Salama, D. S. Reeves, and Y. Viniotis, "A distributed algorithm for delay-constrained unicast routing," in *IEEE INFOCOM'97*, 1997.
22. A. Fei, G. Pei, R. Liu, and L. Zhang, "Measurements on delay and hop-count of the Internet," in *IEEE GLOBECOM'98 - Internet Mini-Conference*, 1998.
23. J. J. Garcia-Luna-Aceves and J. Behrens, "Distributed, scalable routing based on link-state vectors," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 8, pp. 1383–1395, Oct. 1995.
24. E. W. Zegura, K. Calvert, and S. Bhattacharjee, "How to model an internetwork," in *IEEE INFOCOM'96*, vol. 2, pp. 594–602, Mar. 1996.