# Connectivity Aware Routing – a Method for Finding Bandwidth Constrained Paths Over a Variety of Network Topologies

Karol Kowalik and Martin Collier
Research Institute for Networks and Communications Engineering (RINCE)
Dublin City University, Dublin 9, Ireland
{kowalikk,collierm}@eeng.dcu.ie

## Abstract

*Multimedia traffic and real-time e-commerce applications can experience quality degradation in traditional networks such as the Internet. These difficulties can be overcome in networks which feature dynamically set up paths with bandwidth and delay guarantees. The problem of selecting such constrained paths is the task of Quality of Service (QoS) routing.*

*Researchers have proposed several ways of implementing QoS routing, preferring either mechanisms which distribute the network load or algorithms which conserve resources. Our previous studies have shown that network connectivity is an important factor when deciding which of these two approaches gives the best performance. In this paper we propose an algorithm, which features both load distribution and resource conservation. It takes a hybrid approach which balances between these two extreme approaches, according to the level of network connectivity. Our simulations indicate that this algorithm offers excellent performance over a wider range of network topologies than existing algorithms.*

## 1. Introduction

Retrofitting the Internet with QoS capabilities is a challenging task. New service models need to be implemented and existing routing protocols must be enhanced or replaced by QoS routing mechanisms to enable the selection of paths with specific QoS requirements. There is a wide variety of proposed solutions to the problem of selecting a path with specific QoS requirements – a comprehensive survey can be found in [5]. All such QoS routing mechanisms aim to compute a QoS providing paths during periods of low traffic level as well as during periods of congestion. This suggest that traffic should be balanced overall network links. It is also commonly agreed that QoS routing algorithms should conserve resources for future connections. However, the two goals of QoS routing: balancing the network load and resource conservation, are contradictory. The first approach may use very long path to setup a new connection; the second approach is usually restricted to the set of minimum-hop paths only. This makes it very difficult to satisfy both goals of QoS routing at the same time.

In [9] we have shown that, the performance of load distributing (LD) approach and approach performing resource conservation (RC) depend strongly on the level of network connectivity. Our findings suggest that QoS routing algorithms should be aware of this connectivity, to provide improvements comparing to static routing over a wide range of topologies.

In this paper we propose an algorithm, which does not perform only load distribution or resource conservation, but rather tries to find a good trade-off between these two mechanisms. Our solution is to use a hybrid approach which balances between these two extremes, according to the level of network connectivity.

## 2. Link-state QoS routing

This paper considers link-state QoS routing, which can be implemented simply by extending existing routing protocols such as OSPF [1]. In such algorithms the source router selects a route based on information about network resources provided by link state advertisements and information about QoS requirements carried within set-up requests. Flooding of link state information is used to ensure that all routers process the same topological and state information.

However, maintaining global up-to-date state information at every router can result in a high overhead. Hence update policies are used, to control the frequency of link state advertisements. Such update policies result in inaccurate state information being maintained by routers. This puts a requirement on QoS routing algorithms to tolerate such inaccuracy [2].

Link state routing is normally based on the computation of least cost path, although the link cost metric can be expressed in many ways. It is also common to use a second cost metric [10, 12, 14], when the least cost path computation results in a set of equal-cost paths. The choice of primary cost metric is determined by the primary goal of the routing algorithm e.g.: the avoidance of congested links, or minimisation of resource consumption.

If the link cost is expressed strictly as a function of its utilisation (or delay) traffic fluctuations are likely to occur due to feedback [4]. Such oscillations are observed when links advertised as having a low utilisation are favoured by many incoming requests (so they attract new connections) and so in the next update interval are highly loaded. After advertising a high utilisation few connections will be routed via these links and the whole cycle repeats.

If the update policy inserts a large time interval between concurrent link state advertisements (i.e. the response of the feedback mechanism is slow), such traffic fluctuations can cause dramatic performance degradation [14]. This often makes link cost expressed strictly as a function of link utilisation less attractive than the constant value used in min-hop path computation. However, as our previous results show [9], for many networks with low connectivity only the load balancing approach of existing algorithms, can fully utilise available resources.

In this paper we show that the primary link cost metric should not depend solely on link utilisation or be a constant. Such definitions result in the pure LD or RC approaches. In [9] we have shown that the performance of these two approaches depends on the level of network connectivity. This suggests that the primary link cost metric should be a combination of these two approaches, with their relative weighting set according to the level of network connectivity.

## 3. Load Distribution (LD) vs. Resource Conservation (RC)

As we have already described, these two goals of QoS routing, are mutually contradictory, and thus it is usually impossible to simultaneously satisfy both. Hence the community of researchers is divided into two groups advocating the first [11–14] or second approach [3, 6, 8]. In Table 1 we show the main differences between these two approaches.

It is a commonly held view, that traffic distribution pays off when the network load is low, while conserving resources performs better at high loads. In [9] we show that this statement is not always true and that for networks with low connectivity, the LD approach is in general much better solution than RC algorithms, even if the network load is high. On the other hand, at the low loads in highly connected networks, the resource conserving approach performs as well as load balancing does.

| | **Load distribution approach (LD)** | **Resource conservation approach (RC)** |
|---|---|---|
| *primary link cost metric* | depends on link utilisation | is a constant |
| *resource utilisation* | all network links are usable | only links belonging to set min-hop paths are usable |
| *resource conservation* | not present | present |
| *vulnerability to inaccuracy of network state information* | high (link cost is a function of link utilisation) | low (link cost is constant) |
| *traffic fluctuations can occur* | over all links | over the set of min-hop paths |

**Table 1. Load distribution vs. Resource conservation**

This suggest that routing mechanisms should operate differently over networks with different level of connectivity. We propose a routing algorithm which reflects this in the next section.

## 4. Routing algorithm aware of the level of network connectivity

The performance of the LD approach suffers a lot from traffic fluctuations. Such oscillations can be damped by adding a positive constant, called *bias*, to the link cost function [4]. It was observed that by adding a large *bias* the routing becomes fairly stable, but it also loses the ability to avoid congestion [4].

We think that the value of *bias* should increase with the level of network connectivity, because LD methods are beneficial for networks with low connectivity, and for highly connected networks they are outperformed by RC algorithms [9].

### 4.1. Assumptions

Let's assume that we operate on a weighted graph model $G = (V, E)$ (each node $n \in V$ and each link $e \in E$). Each link $e$ has assigned capacity $c(e)$ and an amount $r(e)$ of this bandwidth is already reserved and cannot be assigned to new connections. So the link utilisation $u(e)$ can be expressed as $r(e)/c(e)$.

### 4.2. Primary link cost metric

Let us assume that the values of link cost metric values belong to the interval $< MIN; MAX >$. In our experiment we use a range between 1 and 65535, as is used in OSPF. We assume that the routing algorithm performs load distribution using a normalised exponential link cost function [3, 6] with base value $A$ ( [6] presents a method for

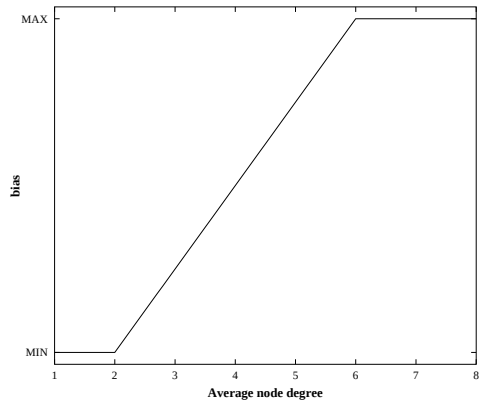


**Figure 1.** *Bias* $\alpha$

calculating $A$):

$$\mathbf{exp}(e) = \frac{A^{u(e)} - 1}{A - 1}$$

which returns values from the interval $< 0; 1 >$. However in our hybrid approach, we add a constant $\alpha$ called *bias*, so that the primary link cost metric of link $e$ is expressed by:

$$\mathbf{cost}(e) = \alpha + \beta\mathbf{exp}(e)$$

where: $\beta = MAX - \alpha$.

As we explained earlier, the value of bias $\alpha$ should increase with the level of connectivity. We decided to vary $\alpha$ as a function of average node degree in the manner shown in Figure 1. This function has been chosen to yield the LD approach for average node degree $< 2$, and the RC approach for average node degree $> 6$. Our previous work [9] has shown that the respective approaches give the best performance in these two situations. We assume that the task of setting the *bias* $\alpha$ according to the connectivity of the network is performed by the administrator. Initially this would be done as per Figure 1. However, the administrator can change $\alpha$ if he decides that more or less load balancing is required. Since the level of network connectivity will not change very often, this task will need to be done only infrequently.

### 4.3. Connectivity aware routing (CAR)

For each link the primary cost is calculated as described above. However, the computation of least cost path can result in a set containing more than one path of equal cost. The second link cost metric is used to spread the traffic between them. For simplicity we decided that the path with the highest residual bandwidth is chosen from the set of least cost paths.

The CAR algorithm may be summarised as follows. For each link, the two values: primary cost and residual bandwidth are advertised using a flooding mechanism. The frequency of such advertisements is controlled by the link state update policy [2] (in our experiment we use a threshold based policy controlled by a *hold-down* timer). Upon receiving a route request the set of least cost paths is computed; if there is more than one such path, the one with the highest residual bandwidth is chosen. If the reservation along a chosen path can be realised, the request is accepted, and otherwise it is rejected.

## 5. Performance evaluation

In this section we compare the performance of the *connectivity aware routing* (CAR) algorithm with a representative of the RC approach – the *widest-shortest path* (WSP) algorithm [10], and with a LD algorithm, specifically one using link cost metrics expressed as an exponential (EXP) function of link utilisation [3, 6].

### 5.1. Network Model

Researchers have evaluated the performance of different routing approaches on simulated networks with various topologies such as: ISP [2], MCI [10], random topologies [13, 14], regular topologies [14], and others [7, 12]. However the topology of the Internet is difficult to characterise because it is constantly changing, so selecting an algorithm on the basis of a limited set of topologies should not be recommended.

In [9] we showed how the routing algorithm performance is affected by the level of network connectivity. We explored this by randomly adding links to a base network with $N$ nodes and $L$ links until the required connectivity was achieved. In this paper we use the same procedure, and as a base network we use the topology shown in Figure 2, which has $N = 21$ nodes and $L = 20$ links (the minimal level of connectivity corresponding to $L = N - 1$), so that the average node degree is equal to $2L/N = 1.9$.

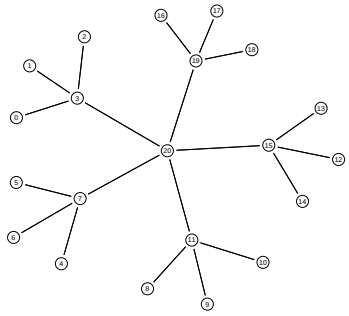

**Figure 2. The base network topology**

We add links randomly to our base network until the number of links is $L = 80$ which gives an average node degree of 7.6. Results obtained by other researchers (for example [13, 14]) usually relate to topologies with a degree of around four, and rarely report results for degrees higher than six. Thus the range of connectivities we study here encompasses those of earlier research. We extend the base network by adding links until rich connectivity is achieved.

Adding one or more links to a network can result in different routing performance, depending on the placement of the newly added links. Therefore this process needs be repeated several times to get a stable and uniform result. In our experiment we create 200 different network topologies with the same connectivity level, and average their performance. All the average results are obtained with a 95% confidence interval.

To show also how the CAR, WSP and EXP algorithms perform under increasing load we use two randomly generated networks each with a different level of connectivity created using the GT-ITM software [15]. Both networks have 50 nodes. The first one has an average node degree of 2.3, while the second network topology has an average node degree of 4.96.

In all the networks we use bidirectional links, each with identical capacity $C$ ($C = 45Mbps$ as in DS-3 links).

### 5.2. Traffic model

The requests arrive at each node independently according to a Poisson distribution with rate $\lambda$ and have exponentially distributed holding times with mean value $1/\mu$. The requested amount of bandwidth is uniformly distributed over the interval: $[64kb/s, 6Mb/s]$, with mean value $B = 3.32Mb/s$. If $N^a$ nodes in the network generate the traffic, the load offered to the network is [14] $\rho = \lambda N^a B h'/\mu LC$, where $h'$ is the average shortest path distance between nodes, calculated over all source-destination pairs (for our simulation: $N^a = 21$, and $h' = 3.04762$ for the base network shown in Figure 2). In our experiment we adjust $\lambda$ to produce the required offered load and fix the mean connection holding time at 180sec. However each time a link is added, the values of $L$ and $h'$ changes. To keep the network load $\rho$ constant, we recalculate the values of $L$, $h'$ and $\lambda$ every time we add a new link.

To model bursty traffic we use: interarrival times following a Weibull distribution with shape parameter $c = 0.7$ and connection durations following a Pareto distribution with shape parameter $a = 2, 5$. In section 5.4 we explicitly state when bursty traffic is used, otherwise we assume that requests arrive according to a Poisson distribution and have exponentially distributed holding times.

### 5.3. Performance metrics

We compare the performance of routing algorithms using:

**call blocking rate** – defined as the number of rejected requests, divided by the number of all requests;

**relative average path length** – defined as the average path length of accepted requests, divided by $h'$ (the average min-hop path distance between nodes, calculated over all source-destination pairs). This allows us to check whether a routing algorithm uses resources excessively (ratio $> 1$) or conserves resources (ratio $\leq 1$);

**variation in link utilisation** – which is the average variation in utilisation of all network links. If its value is small then the load is spread well over all links, and if it is high then the routing algorithm is likely to produce points of congestion, while there are still unused resources.

### 5.4. Results

The goal of our experiment is to show how the CAR algorithm improves on the LD and RC approaches represented respectively by the EXP and WSP algorithms. Our experiment covers a wide range of network connectivities and levels of inaccuracy of state information.

The inaccuracy is introduced through the use of *hold-down* timers which set the minimal time interval between concurrent link state updates (we will denote this interval as $hd$). We assume that each node uses a threshold based update policy [2] with the threshold trigger of $10\%$ controlled by a hold-down timer with value $hd$. We use only one hold-down timer for all links outgoing from the node.

#### 5.4.1. Average results

In this part of our experiment we change the level of connectivity as described in Section 5.1. The use of resources under bursty traffic is illustrated in Figure 3, which shows the *relative average path length* under increasing connectivity. The EXP algorithm, by trying to avoid congested links, uses paths with a non-minimal number of hops. The path length of WSP algorithm asymptotically approaches the value of 1, which confirms that it minimises resource consumption. The hybrid approach used in the CAR algorithm, selects non min-hop paths when the level of connectivity is low, while reducing the use of excessive resources when connectivity is high. All three algorithms start with the value of *relative average path length* below 1, because for networks with very low connectivity short paths are more likely to be successfully established, than longer ones.

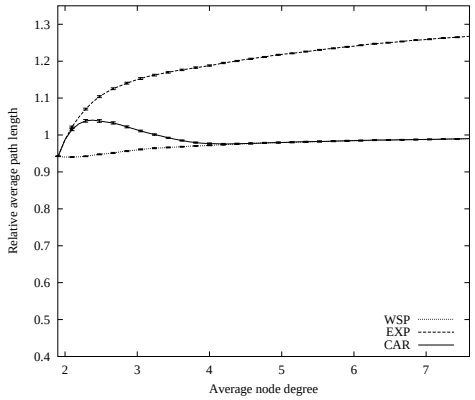


**Figure 3. Relative average path length versus increasing connectivity, $\rho = 0.8$, $hd = 40sec$**

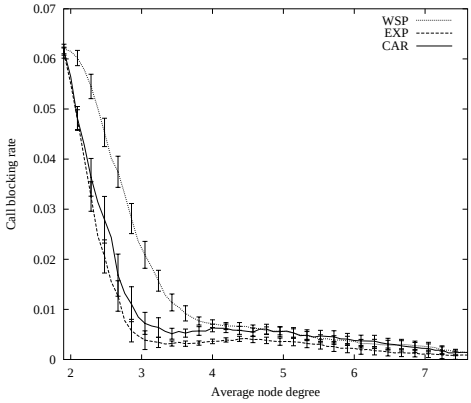


**Figure 5. Call blocking rate versus increasing connectivity, $\rho = 0.3$, $hd = 1sec$**

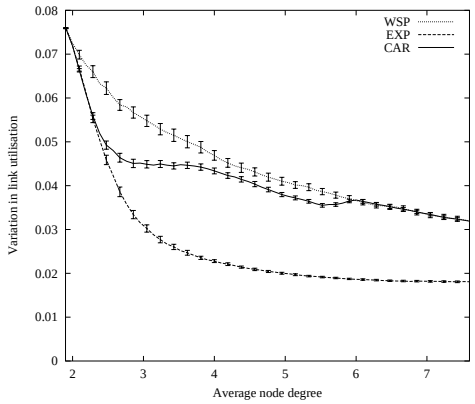


**Figure 4. Variation in link utilisation versus increasing connectivity, $\rho = 0.8$, $hd = 40sec$**

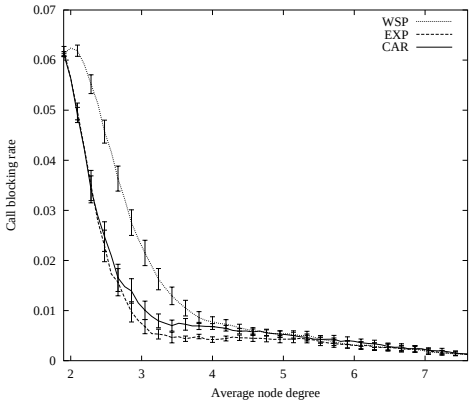


**Figure 6. Call blocking rate versus increasing connectivity, $\rho = 0.3$, $hd = 40sec$**

In Figure 4 we show the variation in link utilisation, which illustrates how well the network traffic is balanced over all links (the smaller the variation, the better load balancing is performed). The EXP algorithm is effective at balancing the load, but by using non min-hop paths it results in a high blocking rate (Figure 8). The WSP performs load distribution poorly, since it is restricted to the use of shortest paths. The CAR algorithm provides a compromise between these two extremes, reducing its load balancing ability for highly connected networks to keep the blocking rate low (see Figure 8).

When the network load is light, such as when $\rho = 0.3$ (see Figures 5 and 6), the EXP algorithm balances the network load well both when there is a low level of inaccuracy (Figure 5) and when the level os inaccuracy is high (Figure 6). The CAR algorithm offers comparable performance in these situations.

When the network load is high, eg. when $\rho = 0.8$ (Figures 7 and 8) the EXP algorithm still performs well if the level of inaccuracy is low (Figure 7), but its performance degrades quickly if the level of inaccuracy increases (Figure 8). The CAR algorithm due to its use of the *bias* factor does not suffer so much from the imprecision in state information.

WSP produces good results for highly connected networks and is robust to imprecise state information (Figures 6 and 8). However if the level of connectivity is low and thus there is usually only one min-hop path between any two nodes, the WSP cannot utilise resources as well as either EXP or CAR.

This average results show that the CAR algorithm, by adapting its link cost metric according to the level of network connectivity, provides a good routing strategy over a

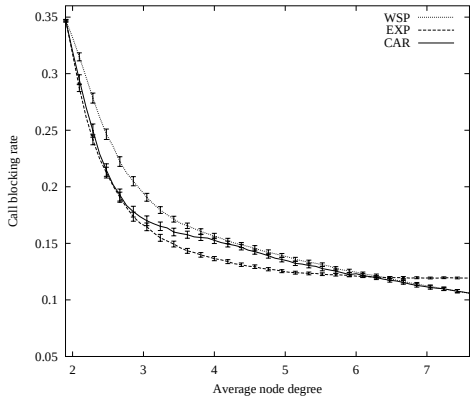


**Figure 7. Call blocking rate versus increasing connectivity, $\rho = 0.8$, $hd = 1sec$**

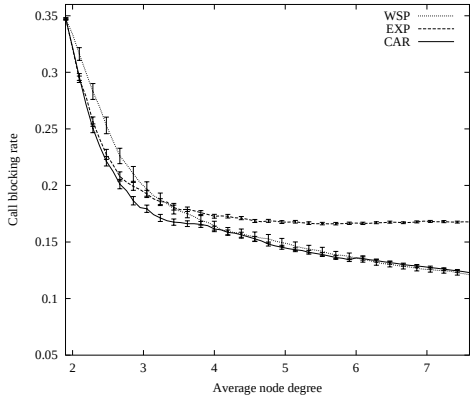


**Figure 8. Call blocking rate versus increasing connectivity, $\rho = 0.8$, $hd = 40sec$**

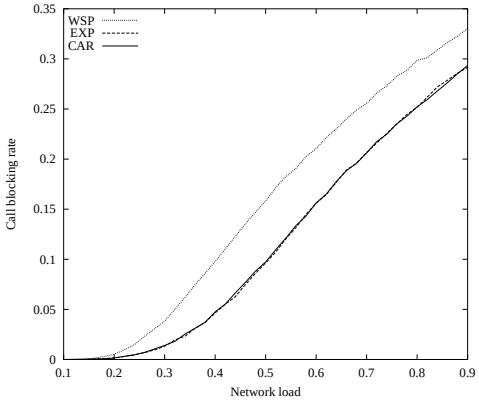


**Figure 9. Network with low connectivity, $hd = 1sec$**

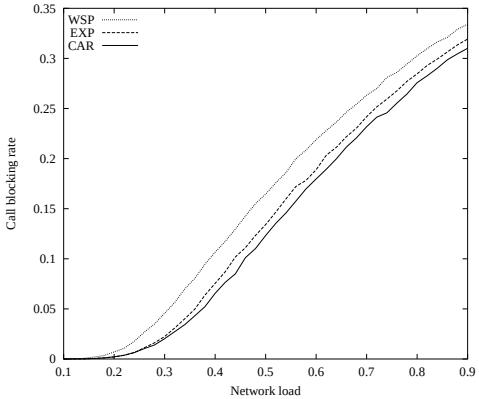


**Figure 10. Network with low connectivity, $hd = 80sec$**

wide range of network topologies.

#### 5.4.2. Case of network with low connectivity

In this section we compare the call blocking probability or CAR, EXP and WSP for an example network with low connectivity, shown in Figures 9 and 10.

In Figure 9 we observe that when state information is up-to-date CAR and EXP perform LD equally well under wide range of network loads. The WSP algorithm, due to its use of only the set of min-hop paths, cannot fully utilise resources, so it gives rise to a higher blocking rate.

If state information is imprecise, as in Figure 10 when $hd = 80sec$, the CAR algorithm outperforms EXP due to its use of the *bias* factor. The EXP algorithm, still gives a lower blocking rate than WSP, which confirms our findings described in [9], that for networks with low connectivity the LD approach outperforms RC.

#### 5.4.3. Case of network with high connectivity

When routing algorithms operate on highly connected networks, our findings from [9] suggest that RC should be preferred. Indeed as shown in Figures 11 and 12, WSP results in a lower blocking rate than EXP over a wide range of loads. The EXP algorithm performs better than WSP only when state information is precise and the load is low (when loads are lower than 0.5 in Figure 12) and the advantage is slight.

The performance of the CAR algorithm for highly connected networks is almost the same as WSP. This is the result of using a large *bias* factor, which limits the influence of the load distributing mechanism. This is beneficial for networks with a high connectivity level.

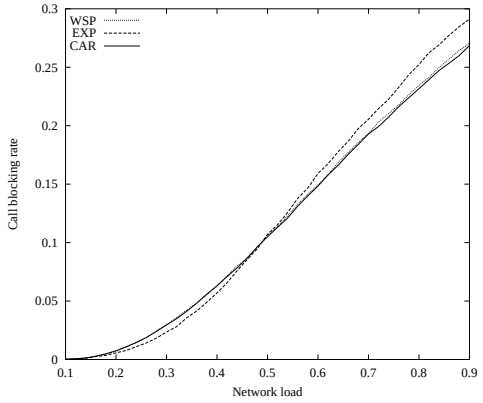


**Figure 11. Network with high connectivity,** $hd = 1sec$

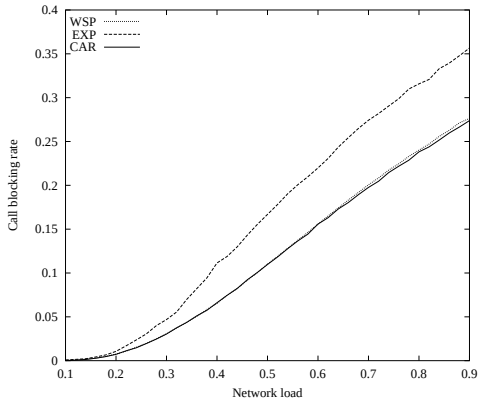


**Figure 12. Network with high connectivity,** $hd = 80sec$

## 6. Conclusions

We have shown in [9], that for networks with low connectivity the load distributing (LD) approach outperforms the resource conserving (RC) approach, and the opposite situation occurs for highly connected networks especially when the state information is not precise. Routing algorithms distributing the network load over all links often perform least cost path computation, when the link cost is expressed as an exponential function of its utilisation. In this paper we show that by specifying the link cost metric as a sum of an exponential function of link utilisation and a positive constant proportional to the level of network connectivity we can obtain a robust routing algorithm, which performs well for various levels of network connectivity. Such an approach favours load distribution for networks with low connectivity and conserves resources when the level of connectivity is high.

Our findings suggest that in general the routing algorithm should modify its load balancing ability according to the level of network connectivity to obtain a good performance for the widest range of network topologies.

## References

[1] G. Apostolopoulos, D. Williams, S. Kamat, R. Guerin, A. Orda, and T. Przygienda. QoS Routing Mechanisms and OSPF Extensions. *RFC 2676*, August 1999.

[2] G. Apostopoulos, R. Guerin, S. Kamat, A. Orda, and S. K. Tripathi. Intradomain QoS Routing in IP Networks: A Feasibility and Cost/Benefit Analysis. *IEEE Network, 13(5):42–54*, Sept./Oct. 1999.

[3] B. Awerbuch, Y. Azar, S. Plotkin, and O. Waarts. Throughput-Competitive On-line Routing. *34th Annual Symposium on Foundations of Computer Science*, November 1993.

[4] D. Bertsekas and R. Gallager. *Data Networks*. Prentice-Hall, Inc., 1987.

[5] S. Chen and K. Nahrstedt. An Overview of Quality of Service Routing for Next-Generation High-Speed Networks: Problems and Solutions. *IEEE Network*, pages 64–79, Nov./Dec 1998.

[6] R. Gawlick, A. Kamath, S. Plotkin, and K. Ramakrishnan. Routing and Admission Control in General Topology Networks. *Technical Report STAN-CS-TR-95-1548*, 1995.

[7] R. Gawlick, A. Kamath, and K. Ramakrishnan. Online routing for virtual private networks. *Computer Communications*, 19:235–244, March 1996.

[8] A. Kamath, O. Palmon, and S. A. Plotkin. Routing and Admission Control in General Topology Networks with Poisson Arrivals. *SODA: ACM-SIAM Symposium on Discrete Algorithms*, 1996.

[9] K. Kowalik and M. Collier. Should QoS routing algorithms prefer shortest paths? *IEEE 2003 International Conference on Communications*, May 2003. Anchorage, Alaska.

[10] Q. Ma and P. Steenkiste. On path selection for traffic with bandwidth guarantees. *In Proceedings of IEEE International Conference on Network Protocols*, October 1997.

[11] Q. Ma and P. Steenkiste. Quality-of-service routing for traffic with performance guarantees. In *Proc. IFIP International Workshop on Quality of Service*, page 115126, Columbia University, New York, May 1997.

[12] I. Matta and A. Shanka. Dynamic Routing of Real-Time Virtual Circuits. In *Proceedings of IEEE International Conference on Network Protocols*, pages 132–139, 1996.

[13] C. Pornavalai, G. Chakraborty, and N. Shiratori. QoS Based Routing Algorithm in Integrated Services Packet Networks. In *Proceedings of International Conference on Network Protocols*, pages 167–175, Atlanta, Georgia, October 1997.

[14] A. Shaikh, J. Rexford, and K. S. Shin. Evaluating the impact of stale link state on quality-of-service routing. *IEEE/ACM Transactions on Networking*, April 2001.

[15] E. W. Zegura, K. L. Calvert, and S. Bhattacharjee. How to Model an Internetwork. In *IEEE Infocom*, volume 2, pages 594–602, San Francisco, CA, March 1996. IEEE.