INTERNATIONAL JOURNAL OF COMMUNICATION SYSTEMS
Int. J. Commun. Syst. 2004; **17**:1–26 (DOI: 10.1002/dac.623)

On the effectiveness of alternative paths in QoS routing

Yanxia Jia[†], Ioanis Nikolaidis[‡] and Pawel Gburzynski[§,*]

Department of Computing Science, University of Alberta, Edmonton, AB, Canada T6G 2E8

SUMMARY

We propose a routing strategy in which connection requests with specific bandwidth demands can be assigned to one of several alternative paths connecting the source to the destination. The primary goal of this multiple-path approach is to compensate for the inaccuracy of the knowledge available to routing nodes, caused by the limited frequency of link state (LS) information exchanges. We introduce a collection of K-shortest path routing schemes and investigate their performance under a variety of traffic conditions and network configurations. We subsequently demonstrate that K-shortest path routing offers a lower blocking probability in all scenarios and more balanced link utilization than other routing methods discussed in the literature. With our approach, it is possible to reduce the frequency of link state exchanges, and the incurred bandwidth overhead, without compromising the overall performance of the network. Based on the proposed routing scheme, we investigate different link state dissemination algorithms, which are aimed at reducing the communication overhead by prioritizing the scope and differentiating the qualitative content of LS update messages. Copyright © 2004 John Wiley & Sons, Ltd.

KEY WORDS: QoS; routing; resource allocation; multiple-path routing; link state; flooding

1. INTRODUCTION

1.1. Quality of service routing

With the growing diversification of networking applications and proliferation of integrated services, networks are being forced to cater to a variety of traffic classes with definite and often critical quality of service (QoS) requirements. Nowadays, it becomes more and more evident that the QoS requirements of modern networking applications can no longer be ignored by routing algorithms [1], with their objective being redefined as selecting the optimum path between source and destination that satisfies certain (possibly multiple) QoS constrains. The application of constraint-based routing is seen as central to providing QoS in internetworks, such as the current Internet. Proposed solutions include modifications to the shortest path first (SPF) scheme by adding additional constraints that an edge (link) must satisfy in order to be

[*] Correspondence to: P. Gburzynski, Department of Computing Science, University of Alberta, Edmonton, AB, Canada T6G 2E8.

[§] E-mail: pawel@cs.ualberta.ca

[†] E-mail: yanxia@cs.ualberta.ca

[‡] E-mail: yannis@cs.ualberta.ca

Received 10 July 2003
Revised 20 August 2003
Accepted 12 September 2003

included in the constructed shortest path. Such an approach assumes the existence of reasonable link metrics for the constraints at hand (e.g. per-link residual bandwidth for bandwidth constraints).

Notably, constraint-based routing calls for explicit/source routing to be used as part of the overall scheme. For example, for the same source and destination nodes, different paths may have to be used by different flows at the same point in time. Such routing flexibility is not available with traditional destination address-based routing, which makes no distinction between routing and forwarding.

A decoupling of these two functions has been proposed recently and implemented using multiprotocol label switching (MPLS) [2]. MPLS provides exactly the flexibility necessary to separate the routing decisions (which can subsequently be based on complicated objectives and constraints) from the simple forwarding mechanism (based on 'labels' of local significance, similar to virtual circuit routing). Since MPLS assumes no particular resource reservation scheme, there is a need for a protocol that performs the actual reservation and maps its result into labels. Towards this end, two protocols have been proposed, CR-LDP [3] and RSVP-TE [4], with a subsequent extension by crankback options [5].

In this spirit, we present routing path calculation and selection schemes that attempt to mitigate the lack of accurate link state information by exploiting multiple paths to the same destination. It assumes the existence of an RSVP-TE signaling and reservation scheme, coupled with crankback capabilities, like those of Reference [5], and with an underlying MPLS infrastructure capable of performing the forwarding actions.

QoS routing turns out to be considerably more complex than unconstrained routing. In particular, some combinations of constraints (e.g. multiple additive constraints) result in NP-hard problems [6, 7]. However, as pointed out in References [8, 9], certain QoS metrics (notably, bandwidth, delay and delay jitter) are not independent when specific scheduling policies are used. Consequently, several studies, e.g. Reference [10], consider bandwidth as the sole metric in route computation. As stated in Reference [11], this simplifies to some extent the path computation process, so that tractable solutions can be provided in a number of interesting cases. Following this approach, we also use bandwidth as our sole QoS metric.

Another obvious metric that should be taken into account by a routing algorithm is the length of the connection path expressed as the hop count. The number of hops is a good indicator of the total amount of resources used along the path. Our studies indicate that regardless of the other criteria, it always makes sense to keep the connection path as short as possible, as long as it fulfills the QoS constraints.

1.2. The scalability problem of link state-based QoS routing

Most proposals for QoS routing are based on versions of Link State (LS) routing, e.g. [7, 10, 12–16]. Among the cost factors of QoS routing, the cost of LS exchange is the dominant contributor [12, 14] and can severely limit the scalability of QoS routing. Consequently, frequent exchange of link state information may be prohibitively costly, especially in wide area networks of a realistic size. On the other hand, infrequent exchange, and stale state of links, may severely impair the quality of routing. Therefore, there exists a tradeoff between the accuracy of the link state information and the overhead incurred by exchanging that information. Formulated in this context, the problem of LS-based QoS routing is how to get acceptable performance in networks with inaccurate link state information.

In this paper, we propose a multiple-path scheme to deal with inaccurate link state problem. The majority of studies on QoS routing aimed at producing a single optimum path. With this approach, only a single path is considered between a source–destination pair, even if there exist some alternative, possibly sub-optimal, paths. In case of congestion, the single path approach is likely to aggravate the problem and may even trigger routing oscillations. In contrast, multiple-path routing would allow different sessions between the same source and destination to be assigned to different paths, depending on the dynamic state of their links. Intuitively, this approach will tend to smooth out occasional problems occurring on some paths, including those caused by inaccurate link state information. This was our primary motivation to embark on the present study.

Formally, we address in this paper a bandwidth-constrained multiple-path construction problem described as follows: Given a network represented by a directed acyclic Graph (DAG) $G(V, A)$, the capacity of each link, b_l, where $l \in A$, a source node s, and a bandwidth threshold B, find the best K paths from s to all the other nodes, $\{P_t^k | 0 < k \leqslant K, t \in V, t \neq s\}$, such that $b(P_t^k) \geqslant B, \forall 0 < k \leqslant K$, $t \in V, t \neq s$, where $b(P_t^k) = \min_{l \in P_t^k} b_l$ is called the bottleneck bandwidth of path P_t^k. Besides specifying the optimality criteria that define the 'best' paths, we have to also describe the construction and selection strategies for the K paths.

We develop two categories of K-shortest routing algorithms, hop-based and bandwidth-based and correspondingly five path selection algorithms: Best-K-widest (BKW), random-K-widest (RKW), shortest-K-widest (SKW), best-K-shortest (BKS) and widest-K-shortest (WKS). In order to appreciate the impact of the plurality of paths ($K > 1$), we extensively investigate the performance of the proposed routing scheme in large networks (up to 1000 nodes) and topologies that have been demonstrated to capture the current Internet structure in a realistic way [17].

Besides, based on the proposed multiple-path QoS routing scheme, we revisit the scalability problem from the point of view of link state dissemination mechanisms. In traditional link state routing, routers use flooding, which is redundant and blind. In this paper, we tailor the flooding mechanism by adjusting the scope and frequency of LS updates and by qualitatively separating LS update messages. We propose three schemes for reducing the amount of update information based on the distance of its recipients from the source, namely, deterministic frequency reduction (DFR), probabilistic frequency reduction (PFR) and hop/distance threshold (HDT). We also find that by differentiating the content of update messages, we can further reduce update costs without impairing routing performance.

The remainder of the paper is organized as follows. Section 2 briefly summarizes previous work on QoS routing with inaccurate information and approaches to reducing routing update information. Section 3 describes the routing model in our study, including path construction, selection and link state dissemination schemes. Section 4 describes the simulation model used to produce the results presented and discussed in Section 5. Finally, in Section 6 we sum up the major conclusions from our work.

2. RELATED WORK

2.1. *QoS routing with inaccurate information*

In Reference [18], a classification of QoS routing schemes that tolerate imprecise state information is presented: (a) safety-based routing [19], (b) randomized routing [19], (c) multiple-path routing [1] and (d) localized routing [20]. Safety-based routing treats link state information as 'fuzzy' and determines the path with the highest probability of success. The randomized approach calculates a set of feasible paths and randomly selects one. Multiple-path routing probes, in parallel, several paths to determine the best choice. Localized routing bases its decisions entirely on information available locally, without the assistance of global LS information. In the comparison carried out in Reference [18], it is found that the only class of schemes with a promise of widely acceptable performance appears to be multiple-path routing.

Recent studies on multiple path routing include [21–24]. Work somewhat similar to our study has been reported in Reference [24], where *multiple paths with equal cost* are constructed and used as routing alternatives. However, an 'equal-cost multi-path' does not necessarily exist for all source–destination pairs, which restricts the applicability of that approach. Another solution has been proposed in References [22, 23], whereby all the multiple paths are used in parallel to route a single traffic stream. In contrast, our scheme is essentially sequential, with every single connection being set up along one specific path. The parallel multi-path routing scheme [22, 23] reduces the reservation delay but it suffers from synchronization problems and requires reassembly buffers at the destination to account for traffic arriving out of order [1]. The Bellman–Ford based widest-shortest (WS) algorithm studied in Reference [15] also provides multiple paths between a source–destination pair. However, it ignores the equal-hop-count multiple paths, which property, as we shall see later, impairs the performance of the routing protocol if the link state information is inaccurate. Ma and Steenkiste [9] investigate several routing schemes, including the *dynamic-alternative* (*DA*), and compare their performance under a variety of topologies and traffic conditions, but again they ignore the issue of accuracy of the link state information. None of those studies investigates how the performance of the routing algorithms relates to the accuracy of the link state information. In this paper, we will address this problem and compare our widest-*K*-shortest variant to widest-shortest and DA in both regular and realistic power-law topologies, with the link state information being accurate as well as inaccurate.

2.2. *Reduction of link update information*

Regarding the amount of LS information exchange, [25] gives a taxonomy of scalability techniques, among which *frequency reduction* and *quantity reduction* are the main generic schemes. The frequency reduction approach, which uses update triggers, has been explored in many QoS routing studies [10, 12, 13, 16]. Our techniques, discussed in the present paper, are also based on this mechanism.

The quantity reduction techniques studied in the literature include quantized update content [12], topology aggregation [26–28], and limited update distribution [29]. Quantized update content consists in disseminating some quantized approximation of LS parameters instead of their exact values. In a topology aggregation scheme, large networks are structured hierarchically by recursively grouping nodes into routing domains. The topology information of each domain is represented in a 'compact' manner. Topology aggregation simplifies the network structure, reduces the amount of routing information exchanged in the network, and cuts down on the size of routing tables. But the inherent 'lossy' characteristic of topology aggregation tends to produce suboptimal paths. The idea of limited update distribution is to narrow the range of flooding.

In this paper, we look at two ways of reducing the amount of LS update information in the network. First, we attempt to contain the scope and decrease the frequency of update messages progressively as they move away from the source. Second, we consider the impact of qualitative triggers of update messages in addition to periodic (blind) reports.

3. THE ROUTING MODEL

Our routing model is a link-state one, in which routers periodically exchange LS and conduct routing calculation. We assume that after every update, each node immediately has the full knowledge of the current link states in the whole network. This is justifiable because the propagation delay of an LSA message is generally small compared to the length of the update period and to the duration of a traffic session. However, we do capture the fact that the updates do not occur continuously but at certain intervals. For regular periodic updates, the constant interval between two consecutive updates is denoted by LSUP.

A source–destination path can be computed upon request, i.e. when it is demanded by the source, or precomputed in advance, e.g. periodically. In our study, routes are precomputed periodically, with the computation period being equal to LSUP.

The path selection process is carried out for every connection request and is separated from the path computation algorithm. For a given destination, the source router is presented with K paths to choose from. If the connection cannot be established via one of the paths, e.g. due to the insufficient remaining bottleneck bandwidth, the router picks one of the remaining paths and tries again. The order in which the K paths are attempted depends on the specific scheme being used and will be detailed in subsequent sections. The request is blocked if no feasible path is found after all K of them have been tried.

3.1. Path construction

The multiple paths are constructed by a label-setting algorithm [30] based on the optimality principle and being a generalization of Dijkstra's algorithm [31], which we have modified to find K *one-to-all* loopless paths instead of *one-to-one* non-loopless paths. Its space complexity is $O(Km)$, where K is the number of paths and m is the number of edges. Using a pertinent data structure, its time complexity can be kept at the same level $O(Km)$ [31].

With the hop count and path bottleneck bandwidth used as two separate metrics, our algorithm will generate K paths that are either *shortest* in terms of the number of hops (hop-based algorithms), or *widest* in terms of the bottleneck bandwidth (bandwidth-based algorithms). In order to limit the impact of requests for trivially small bandwidth, a threshold is used to prune unsuitable links right away, during the path construction stage. That is paths that, during construction, are identified to possess less bandwidth than what a typical call would request, are ignored.

Let a DAG (N, A) denote a network with n nodes and m edges, where $N = \{1, \ldots, n\}$, and $A = \{a_{ij} | i, j \in N\}$. The problem is to find the top K paths from source s to all the other nodes. Define a label set X and a one-to-many projection $h: N \rightarrow X$, meaning that each node $i \in N$ corresponds to a set of labels $h(i)$, each element of which represents a path from s to i. Each label/path is associated with a major weight and a minor weight. For the hop-based algorithm, the major weight is the inverse of the number of hops and the minor weight is the bottleneck bandwidth of the path represented by this label. Those weights are interchanged for the bandwidth-based algorithm. The minor weight is not used by the path construction algorithm (except for being computed), but it is needed later for path selection. We introduce the following notation:

s	the source node
X	the label set
b_{vj}	the link bandwidth from v to j
$count[v]$	the number of paths from s to v found so far
$lb0$	the *permanent* label selected from X (such that $lb0.bw \geqslant lb.bw, \forall \ lb \in X$)
$lb0.ver$	the node corresponding to label $lb0$
$lb0.bw$	the bottleneck bandwidth of the path from s to $lb0.ver$
$lb0.parent$	the label that generated $lb0$
$P_v(count[v])$	the $count[v]$-th path from s to v
$lb1$	a new label generated from $lb0$

The following algorithm calculates the K best paths from source s to all destinations according to the major weight:

```
$count[i] = 0, \forall \ i \in N$
$lb0 = 1$
$lb0.ver = s$
$lb0.bw = infinite$
$lb0.hops = 0$
$lb0.parent = NULL$
$X = lb0$
while ( $X \neq \emptyset$ and $\exists \ i$ such that $count[i] < K$,
    where $0 < i < n$)
do begin
  find a permanent label $lb0$ from $X$, such that
    $lb0.bw \geqslant lb.bw, \forall \ lb \in X$
  $X = X - lb0$
  $v = lb0.ver$
  $count[v] = count[v] + 1;$
  if ($count[v] \leqslant K$ and $lb0.hops < Max\_Hop\_Num$)
  then begin
    record the path $P_v(count[v])$ by
      following the $lb0.parent$ link
    for each $a_{vj} \in A$ /* generate new labels */
    do begin
      if (the potential new label does not result in
        a loop and $b_{vj} > bw\_thrsh$
      do begin /*generate this label*/
        $lb1.ver = j$
        $lb1 = lb0 + 1$
        $lb1.bw = min(lb0.bw, b_{vj})$
```

```
            $lb1.hops = lb0.hops + 1$
            $lb1.parent = lb0$;
            $X = X \cup lb1$   /*add it into the label set*/
      end
      end
    end
end
```

Proof

Without loss of generality let us consider the hop-based algorithm, and let the K labels found for node v be $lbl_i, \forall\ 1 \leqslant i \leqslant K$. Suppose that if we continue executing the algorithm, we will find a new path from s to v, with a hop number smaller than in one of the K computed paths. That is, the algorithm will generate a new label lbl_0, such that $\exists\ 1 \leqslant i \leqslant K$: $lbl_0.hops < lbl_i.hops$. Naturally we get $lbl_0.parent.hops < lbl_i.parent.hops$ (because $l.parent.hops = l.hops + 1, \forall\ l \in X$). This implies that $lbl_0.parent$ must have been marked as a permanent label earlier than $lbl_i.parent$, because in each step of the algorithm, only the best label is marked as a permanent label. If this is true, it should be lbl_0, not lbl_i, that was generated earlier—a contradiction. The bandwidth based case can be proven in a similar way. □

3.2. Path selection

Below we list five path selection algorithms resulting from two different major criteria and different ways of applying the minor criteria.

BKW	best-K-widest: from the K widest paths, select the one whose bottleneck bandwidth most tightly fits the request bandwidth (best-fit).
RKW	random-K-widest: from the K widest paths, select one at random.
SKW	shortest-K-widest: from the K widest paths, select the one with the least number of hops.
BKS	best-K-shortest: from the K shortest paths, select the one whose bandwidth most tightly fits the connection request.
WKS	widest-K-shortest: from the K shortest paths, select the one with the largest bandwidth.

Note that, although the names *shortest-K-widest* and *widest-K-shortest* resemble *shortest-widest* from Reference [7] and *widest-shortest* from Reference [15], our algorithms are quite different from those previously proposed solutions. In particular, our SKW finds the top K paths, while the *shortest-widest* of Reference [7] uses the 'shortest-widest' constraint during the (single) path construction phase. That is, in the case of Reference [7], when multiple labels with the same bandwidth are found, only the shortest one is permanently labeled while the rest are deleted. SKW, on the other hand, maintains all the widest labels for later use and is therefore able to find top K paths. It decides on the 'shortest' path during the path selection phase rather than when the path is being computed.

Furthermore, note that although the *widest-shortest* algorithm from Reference [15] also generates multiple paths for a given destination, it is different from WKS in that it provides fewer choices for selecting short paths, and the resulting connection is thus likely to need more resources. In particular, it uses an upper bound for the hop count and for each hop count, it only keeps a single widest path (ignoring multiple paths with the same number of hops). Besides, it requires that the $(i+1)$th path be strictly 'wider' than the i-th path. This makes sense when the link state information is accurate, but if it is not the case, some feasible paths may be incorrectly ignored. Thus, the very nature of the algorithm in Reference [15] impairs its performance when the link state information is inaccurate.

3.3. *Link state dissemination*

The introduction of QoS into routing makes traditional flooding prohibitively expensive. Besides, the blatant redundancy of flooding leaves a lot of room for improvement. It is understood that information about the links adjacent to a given node can be deemed accurate, but should the node assign the same importance and faith to the information about all other links in the network? An alternative approach is one whereby the closer a link is to the node, the more important the value of its state becomes. The intuition supporting this approach has to do both with issues of spatial locality (closer links are more likely to be used by one or more paths that originate at the node) as well as less inaccuracy (information about closer links is more likely to be up to date).

Based on these observations, we consider three update frequency reduction schemes, dubbed DFR, PFR and HDT. According to these schemes, LS information is sent more frequently to closer neighbours. DFR and PFR are based on the same idea. Given an update source s, define the h-hop neighbours of s as the nodes h hops away from s. Supposing a node receives m LS update messages from an upstream node, it only forwards a portion $a \times m$ $(0 \leqslant a \leqslant 1)$ of those messages to each of its direct neighbours, instead of forwarding all the m messages, as would happen with straightforward flooding. That is, if there are m update messages generated from s, then each of its h-hop neighours only receives ma^h messages, where a is called the frequency coefficient (FC). The only difference between PFR and DFR is that DFR decides deterministically which subset of messages should be forwarded, while PFR does it probabilistically. Suppose that $FC = 0.8$. With DFR, for every 5 LS updates received, a node forwards only 4 to its direct neighbours, by skipping every fifth received message. With PFR, every time a node receives an LS update, it forwards the update to each of its direct neighbours with probability 0.8. In the extreme case of $FC = 1$, both schemes operate in exactly the same way, which becomes equivalent to flooding.

The third scheme, HDT, is influenced by the 'Fisheye' routing protocol [32] which was proposed for mobile *ad hoc* networks (MANETs) to cope with the uncertainty caused by the continuous node mobility. In HDT, there exists a single distance threshold and two update frequency levels. If the hop distance from a node i to the update source s is smaller than the hop distance threshold, i will receive the update messages at the maximum frequency, i.e. once every LSUP interval. Otherwise, the source, it will be notified at $\frac{1}{2}$ the maximum frequency, i.e. once every two LSUP intervals. The simple idea is to update near nodes twice as frequently as far away nodes.

In addition to addressing the link state dissemination problem from the point view of update scope and frequency, we also investigate the qualitative impact of link state information on routing performance. We say that an LS dissemination scheme is *qualitative*, if its decisions to send new LS information are triggered on the basis of drastic quantitative changes. In our case, these qualitative changes are drastic increases or decreases in the available link bandwidth, where what is meant by 'drastic' is described as a relative threshold. In our study, this threshold is 30% of the previously reported value. We also consider non-symmetric schemes, whereby 'good' news (i.e. increase) or 'bad news' (decrease) of the link bandwidth is considered important, while a change in the opposite direction triggers no LS message.

In Section 5.3, we review the performance of four LS propagation schemes, dubbed *good news* (GN), *bad news* (BN), *both news* (GBN) and *periodic LS only* (PLS) as they apply to QoS routing using the *K*-path approach. In GN, aside from periodic LS updates, an additional update message is sent whenever the load of a link *decreases* qualitatively (in the sense described above). The behaviour of the remaining three schemes should be obvious. In all cases, we keep track of the cost incurred by the exceptional messages necessary to convey the 'news.'

4. THE SIMULATION MODEL

Our simulation study consists of two parts. We start from investigating the performance of the proposed multiple-path routing scheme under the assumption of inaccuracy of the LS information modelled by different LSUP values between 0 and 100 min. In these experiments, the LS messages are sent periodically at fixed intervals, regardless of the level of change of the link state. In the second part, we investigate the cost and performance of the various link state dissemination schemes comparing them to the simple periodic scheme.

The most representative of our simulation results have been obtained for random network configurations built by the generator of Magoni and Pansiot [17]. Reasonably large networks generated by this program have been demonstrated to obey the most relevant (from the viewpoint of routing) power laws found in existing wide area networks (e.g. sections of the Internet). A random network configuration in our experiments is characterized by two parameters: the number of nodes N, and the average node degree δ. A typical value of δ found in the Internet is 2.5 [17]. We consider sparser networks, with $\delta = 2.0$, as well as denser networks, with $\delta = 2.9$. For reliability, a single experiment has been verified on a number of different topology samples generated for a given pair $\langle N, \delta \rangle$. For a network of a reasonably large size ($\geqslant 500$ nodes), the obtained results are highly consistent across different topology samples (obeying the same power laws).

For reference, we also consider regular torus networks (Figure 1) with the same populations of nodes as the power-law configurations. Certain performance measures, notably our *coefficient of variation* (see below) may be affected by the inherent irregularities present in any random network, even if it is large and obeys reasonable statistical laws. A regular topology eliminates those problems and helps us find out how much our performance measures are influenced by statistical aberrations in network configurations.

Every link in our network has the same bandwidth of 155 Mbps. The offered traffic consists of randomly generated sessions with exponentially distributed interarrival time. The mean value of this distribution is a simulation parameter that determines the global load in the network. The duration of a session is log-normally distributed with a mean of 180 s, consistent with observations of holding times for calls in the telephone networks [33]. The bandwidth requirement of a session is uniformly distributed between 1 and 5 Mbps. The source–destination pair is selected at random: every station in the network is equally likely to be selected for this role.

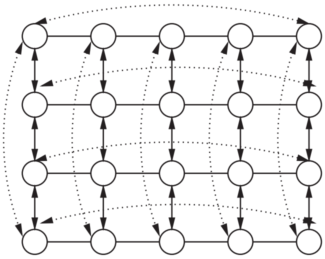


Figure 1. A 4 × 5 torus.

The total amount of simulated time for a single experiment was originally set at 24 h, and six independent experiments were used to obtain a single point of a performance curve. The high observed consistency of results allowed us to reduce the amount of simulated time by half, and the number of samples by the same factor.

In the first part of our simulation study, we are primarily interested in three performance measures as functions of the offered load and LS update period: the connection blocking rate, the *coefficient of variation* of the link utilization and the protocol overhead. The blocking rate is weighted by the connection bandwidth [9, 16] and defined as

$$\theta = \frac{\sum \text{bandwidth\_of\_blocked\_connections}}{\sum \text{bandwidth\_of\_requests}}$$

The *coefficient of variation* of the link utilization captures the variability of the load between different links and indicates how well the offered load is spread over the network resources. We define it as

$$C_v(\mathrm{LU}) = \frac{\mathrm{Std(LU)}}{u(\mathrm{LU})}$$

where LU stands for link utilization (i.e. the fraction of time the link is being used), Std is the standard deviation and u is the mean.

The overhead of our routing protocol can be expressed as the sum of two parts: the signaling component and the LS exchange component. With QoS routing, we need to make a reservation along the selected path before routing a request. The communication cost expended during this operation is called the signaling overhead and represented by the number of signaling messages. The LS exchange overhead is determined by the total number of LS update messages passed in the network. As demonstrated by our studies, the LS update cost is the dominating component of the total overhead.

In the second part of our study, in addition to calculating the bandwidth blocking rate and the link state exchange overhead, we also keep track of savings, i.e. the reduction of the LS exchange overhead with respect to flooding. This measure is defined as follows:

$$\text{Savings} = \frac{C_{\text{flood}} - C}{C_{\text{flood}}}$$

where C_{flood} is the LS overhead in the case of flooding, and C is the respective overhead for the scheme under investigation.

5. SIMULATION RESULTS

5.1. Multiple-path routing with inaccurate information

The primary goal of these experiments was to answer the following questions:

1. Is there a single path selection algorithm (among the ones listed in Section 3.2) that gives consistently the best performance?
2. How does the multiple-path approach fare in the context of inaccurate link state information? In particular, how does the performance of our multiple-path routing scheme depend on LSUP?
3. How large should the multiple-path selection be, i.e. how does k (the number of paths to choose from) affect the performance of our routing scheme?
4. What are the costs and benefits of the proposed multiple-path algorithms?
5. How does our routing scheme compare to other schemes?

Figure 2, obtained for a small and typical network, illustrates the blocking rate under all five path selection algorithms with the number of alternative paths $k = 3$. The trend shown in this figure has been clearly visible in all our experiments. First, it turns out that the hop-based selection algorithms (i.e. WKS and BKS) unquestionably outperform the bandwidth-based ones (BKW, RKW and SKW), with WKS winning over BKS. Although the superiority of WKS over BKS is not obvious in Figure 2, it becomes visible in a larger/denser network, e.g. see Figure 3.

The reason why the hop-based algorithms win in this competition is that by making the path length the primary optimization criterion they focus on minimizing the amount of resources needed to sustain a connection. Consequently, the network tends to use less of its total bandwidth per session and is thus able to accommodate more connections.

The two figures also demonstrate that past some narrow initial range of LSUP, the quality of our routing schemes (especially the hop-based ones) is not adversely affected by the inaccuracies in the link state information. In all cases, as the LS update period tends to infinity, the blocking rate stabilizes to a constant. This is not surprising, since then the calculated paths are not much

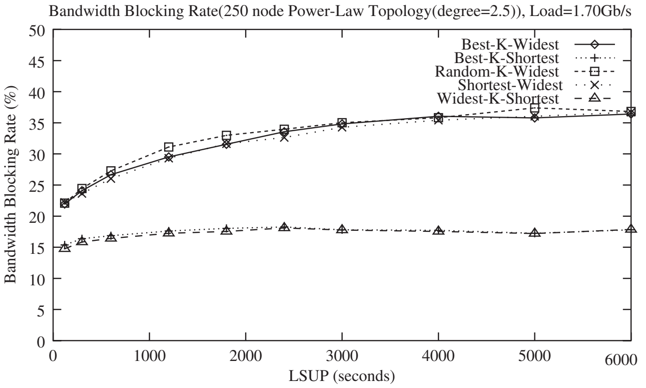


Figure 2. Small typical network, $k = 3$.

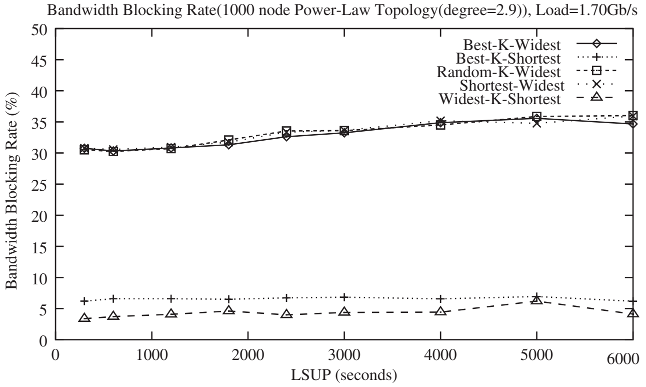


Figure 3. Large dense network, $k = 3$.

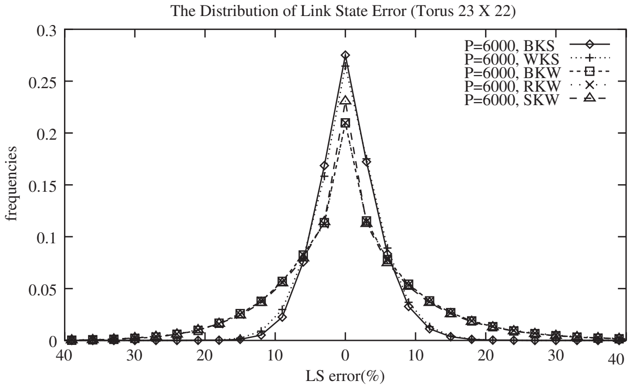


Figure 4. Link state error distribution.

better than ones selected at random. A more important observation is the relative insensitivity of the hop-based schemes to the LS update period.

Aside from blocking performance, we also compared the distribution of link state errors associated with the two classes of algorithms, i.e. how much the link bandwidth available to the routing node deviates from the actual correct value. Based on Figure 4, and using a quantile–quantile plot method [34], we found that the error is approximately normally distributed. Figure 4 also indicates that the hop-based algorithms generate a smaller variance of error. In particular, with BKS, 98.40% of all errors lie in a range as narrow as [−10%, 10%], while with BKW, only 66.41% of errors are in that range.

To answer question 2 and 3, we examined the impact of k on the performance of a path selection algorithm, and found it to depend on the algorithm and on the network density. Fortunately, the hop-based schemes perform quite well with a small number of alternative paths to select from. Figures 5 and 6 show the blocking rate of WKS in two medium-sized networks for different values of k. In the sparse network, the impact of k is quite negligible, which means that alternative path selection brings about little, if any, improvement. This is understandable, because low connectivity implies low number of alternative paths with comparable length (using

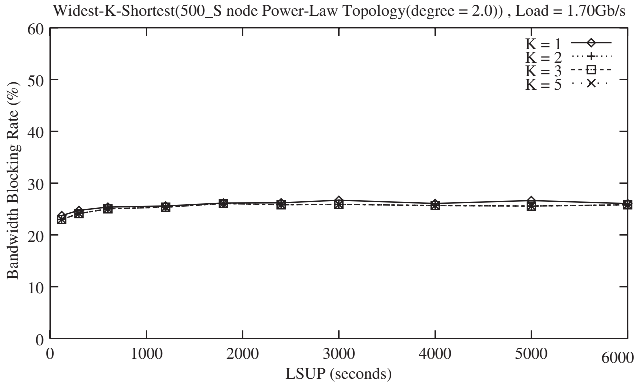


Figure 5. Medium sparse network, different values of k.

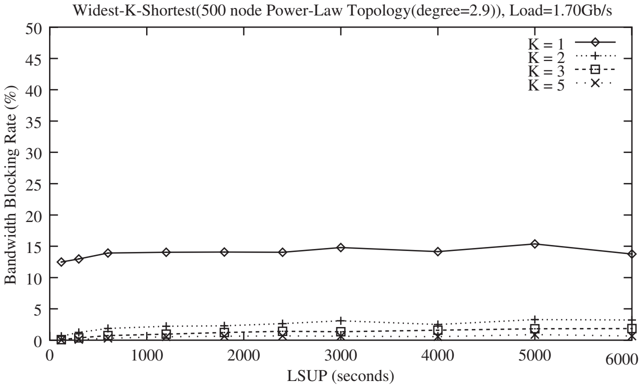


Figure 6. Medium dense network, different values of k.

a comparable amount of network resources). Consequently, even though multiple alternative paths may still exist, they tend to be of different length, so selecting an alternative to the shortest path in such a network is statistically a poorer choice than in a network offering multiple shortest paths. In Figure 6, we see a clear improvement caused by the statistical presence of sensible alternatives.

Regardless of the circumstances, the gap between the cases $k = 1$ and $k = 2$ is significantly bigger than between $k = 2$ and $k = \infty$. This indicates that while it is advantageous to have a choice, that choice need not be excessively big. This observation is good news. It means that the complexity of the routing algorithm can be well contained, because all it needs to do is to find just a few (e.g. 3) alternative paths, which effort is only moderately bigger than finding a single path.

Our primary intuition behind multiple paths was that with several alternative routes, we would be able to spread the offered load more evenly over the entire network. Thus, one would expect that better path selection algorithms should result in a lower observed value of the *coefficient of variation* of link utilization C_v.

Alas, as indicated by Figure 7, this kind of study must be done on a regular topology to make sense. With local irregularities that are bound to occur in a random network of any size, by trying to consistently follow the shortest paths, a routing scheme may quite legitimately create localized hot spots. Those departures from a uniform spread of the offered load will tend to increase with the decreasing average length of a connection path, because the irregularities in topology manifest themselves on a small scale, and they tend to disappear as we look at larger regions of the network. Consequently, in Figure 7, the observed value of C_v is higher for the path selection schemes that offer better performance, i.e. target shorter paths. On the other hand, in Figure 8, obtained for a perfectly regular torus network, the *coefficient of variation* shows a reversed trend, which remains in a perfect agreement with intuition.

So far we have only examined the performance improvement of the proposed multiple-path scheme; in the following we shall investigate whether it is a cost effective solution. Figure 9 shows that WKS with LSUP = 1200 and $K = 2$ or 3 outperforms the shortest path routing scheme with LSUP = 120. In other words, the multiple-path scheme with inaccurate information wins over the single path scheme with accurate information. This performance benefit is

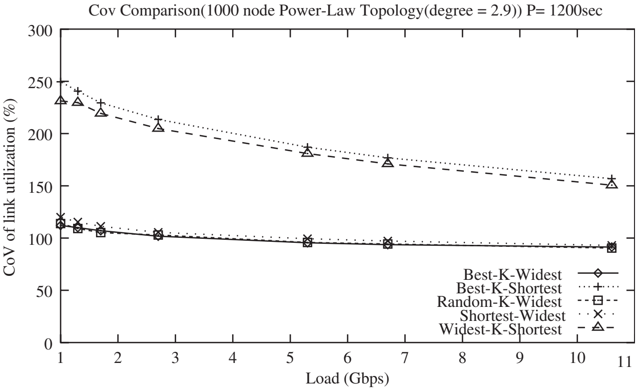


Figure 7. Large dense random network, $k = 3$.

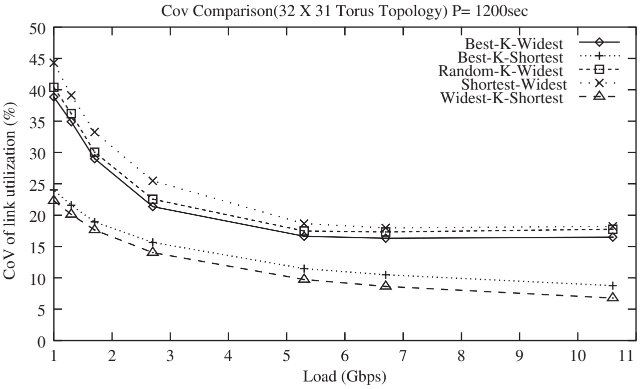


Figure 8. Large regular network, $k = 3$.

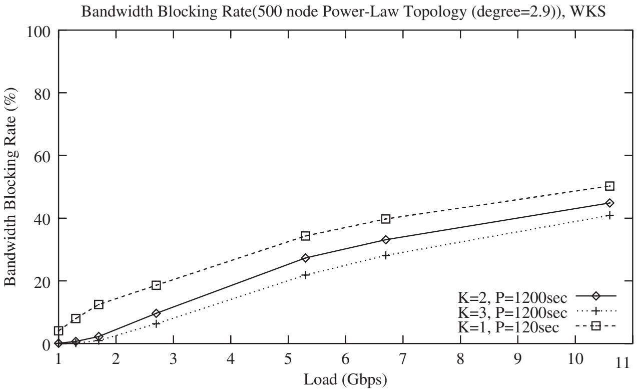


Figure 9. Blocking rate comparison of multiple-path and single-path routing.

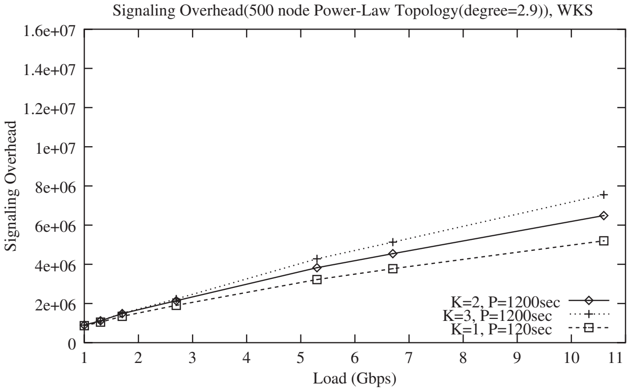


Figure 10. Signaling overhead comparison of multiple-path and single-path routing.

obtained at a cost of increased signaling overhead, as shown in Figure 10. Nevertheless, as revealed in Figure 11, the total cost of the multiple-path scheme with inaccurate information is far less than that of the single path scheme. By total cost, we mean the sum of the two overhead components, i.e. the signaling overhead and the LS exchange overhead. This lets us infer that the proposed multiple-path scheme is a scalable solution.

The confrontation of WKS with DA proposed in Reference [9] and WK introduced in Reference [15] illustrates why routing schemes that do not account for inaccuracies in the link state information may perform poorly, even if they appear superior when that information is accurate. This comparison, illustrated in Figures 13–17, has been done on three network configurations: the MCI network used in Reference [9], a regular torus, and a dense power-law network. We observe that if the LS information is accurate (LSUP = 60 s), DA and WK exhibit about the same (if not better) performance as WKS. However, with outdated LS information (LSUP = 1200 s), WKS wins over the other two approaches (Figures 12 and 15), even using k as low as 2.

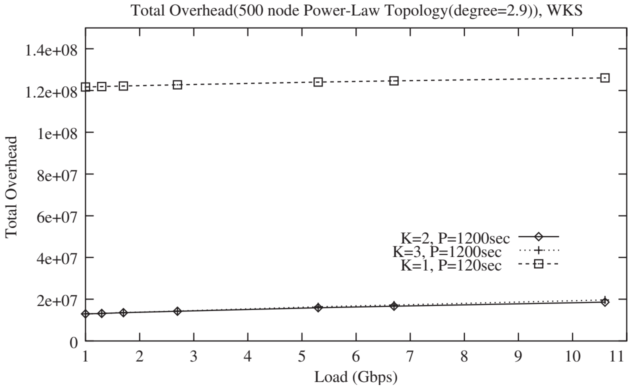


Figure 11. Total overhead comparison of multiple-path and single-path routing.

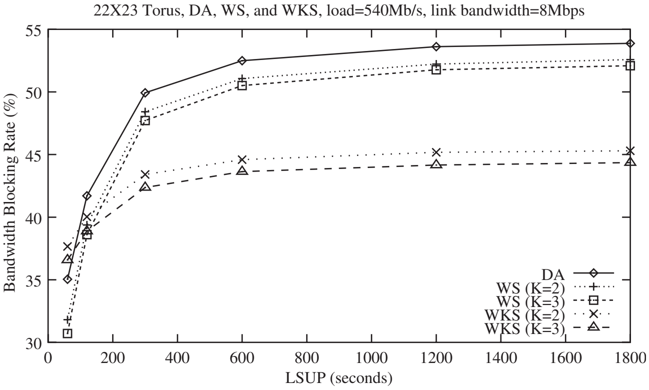


Figure 12. 22 × 23 Torus, blocking rate vs LSUP of DA, WK and WKS.

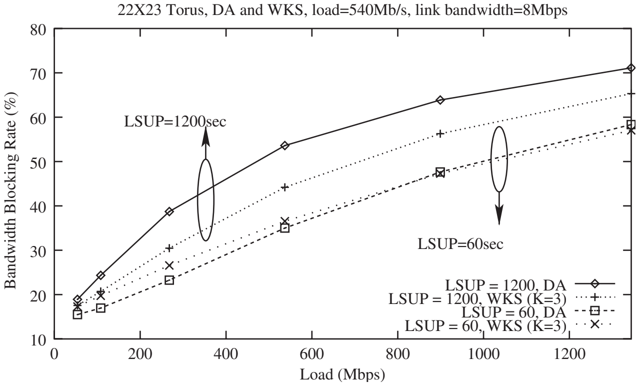


Figure 13. 22 × 23 Torus, blocking rate vs load of DA and WKS.

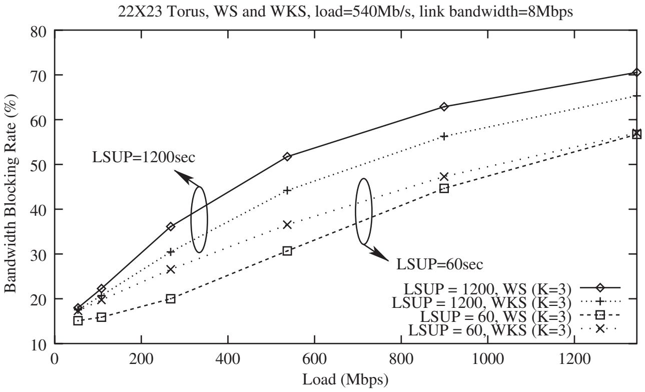


Figure 14. 22 × 23 Torus, blocking rate vs load of WS and WKS.

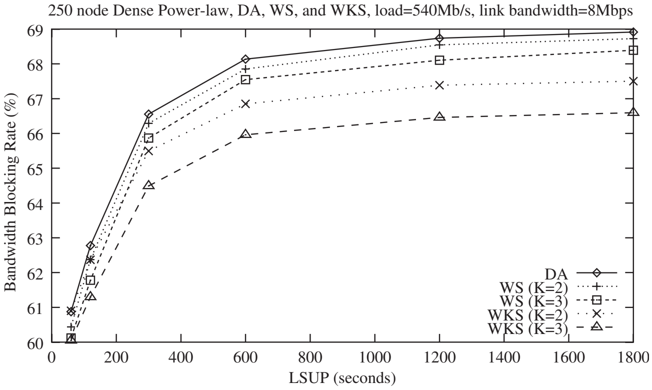


Figure 15. Two hundred and fifty node dense power law, blocking rate vs LSUP of DA, WK and WKS.

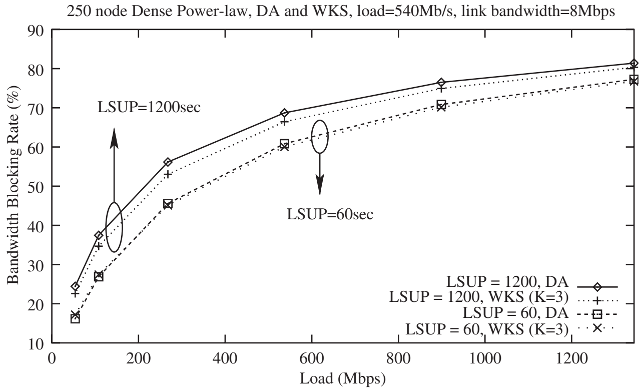


Figure 16. Two hundred and fifty node dense power law, blocking rate vs load of DA and WKS.

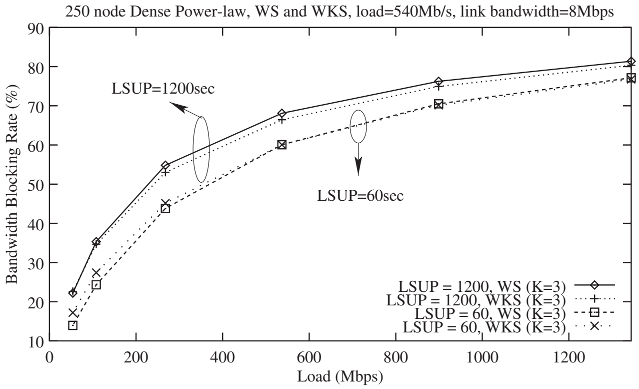


Figure 17. Two hundred and fifty node dense power law, blocking rate vs load of WS and WKS.

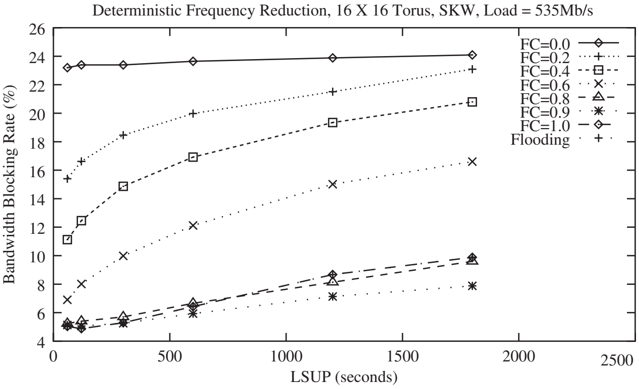


Figure 18. DFR blocking rate vs LSUP period in a 16 × 16 torus for different frequency coefficients (FC).

To explain this difference, let us introduce some notation. Assume that for a given source–destination pair: the paths in the routing table are denoted *path*(i) ($1 \leqslant i \leqslant k$ for WKS and $1 \leqslant i \leqslant 2$ for DA), *bw*(i) is the bottleneck bandwidth of the ith path, *hop*(i) is the hop count of the ith path, $h_{\min}$ is the hop count of the minimum hop path, H is the total number of different hop counts for all the k paths. Let n be the hop count of a minimum-hop path. According to Reference [9], a DA path is a widest-shortest path with no more than $n+1$ hops. DA actually is a multiple-path scheme with $k=2$, hop(2) – hop(1) = 1, and $bw(2) > bw(1)$. In the case of LSUP = 0, if there exist paths with $h_{\min}$ and $h_{\min}+1$ hops, DA has two alternative paths to choose from. But the situation is more complex for WKS. When the paths are calculated based on accurate link state information, there are in fact only H, not k, paths that can possibly succeed. This is because for all the paths with the same length, if the widest route fails, the others will also fail due to their smaller bandwidth. Only if $H > 1$ can WKS have at least two alternative paths. Thus, if the link state information is accurate, WKS may occasionally provide fewer feasible paths than DA. However, this is not the case when LSUP > 0. For WKS, even if

$H = 1$, all k paths can possibly be successful. Besides, for DA, some paths are eliminated by the restriction $bw(2) > bw(1)$, and those eliminated paths could be feasible due to the inaccurate information. Thus, if LSUP > 0, WKS always provides $k \geqslant 2$ routes to select from while DA provides at most two.

5.2. Controlling the dissemination of LS information

In Section 3.3, we introduced three LS update schemes whose purpose is to control the scope and frequency of LS dissemination. Here we investigate the overhead of those schemes and their performance in terms of the network blocking rate. In this study, we consider SKW, i.e. one of the bandwidth-based algorithms. Although the bandwidth-based schemes have been demonstrated to perform consistently worse than the hop-based solutions, their higher sensitivity to LS information inaccuracies make them a better setting in which to compare different dissemination schemes.

Figures 18 and 19 illustrate the behaviour of DFR applied to SKW on a regular torus network. We can see the dramatic cost reduction when the frequency coefficient (FC) is only slightly less than 1, which is caused by the cumulative decrease in the number of LS messages forwarded towards more distant nodes. On the other hand, the blocking rate increases insignificantly for the same range of FC, with FC $= 0.8$ appearing as a possible compromise value (with 80% cost savings). The results for PFR are similar, although the best value of FC is somewhat different.

Figure 20 depicts the blocking rate for DFR as a function of FC in the 16×16 torus network. With small LSUP values, the blocking rate decreases along with increasing FC. However, for large LSUP, if FC is increased above a certain point, the blocking rate surprisingly tends to increase slightly, until it settles at a value somewhat above the minimum (obtained for FC about 0.9). This counter-intuitive behaviour can be seen for the other two schemes, PFR and HDT, and for randomized power-law topologies, where it is even more pronounced (see Figure 21). These results suggest that there may exist an upper limit for LSUP. If this limit is exceeded, LS information becomes useless to the point of being harmful, no matter how much of it is disseminated over the network.

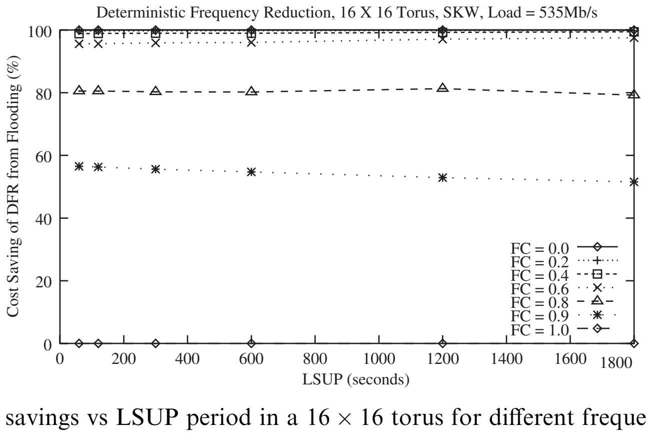


Figure 19. DFR cost savings vs LSUP period in a 16×16 torus for different frequency coefficients (FC).

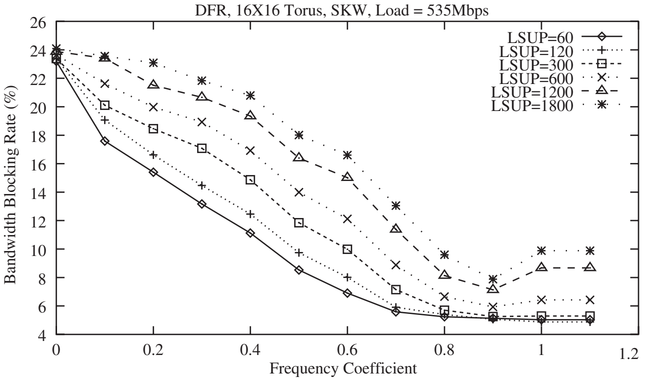


Figure 20. Blocking rate in a 16 × 16 torus topology vs frequency coefficient (for DFR) for different LSUP.

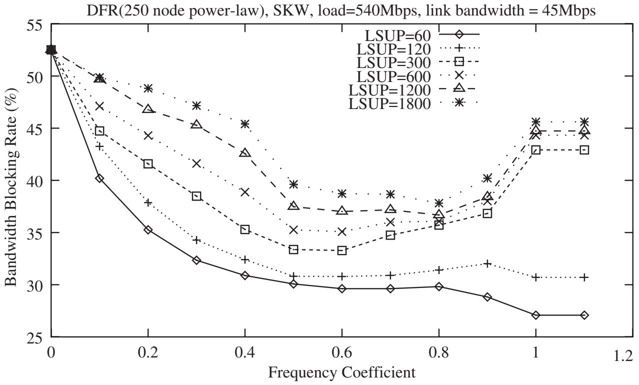


Figure 21. Blocking rate in a 250 node power-law topology vs frequency coefficient (for DFR) for different LSUP.

5.3. Qualitative link state dissemination schemes

In Section 3.3, we mentioned four qualitative link state dissemination schemes, GN, BN and GBN, confronted with PLS as the non-qualitative 'reference' solution. Here we discuss their cost and blocking performance. We conjecture that it makes sense to propagate 'good news' (i.e. the availability of new routing opportunities) faster than regular periodic updates. In this way, the inaccuracy of the status information obtained from last update that inhibited the use of a link can be quickly corrected.

To assess the validity of this claim, we investigate the blocking rate and LS exchange cost of GN, BN and GBN as functions of LSUP and network load. Figure 22 reveals that when LSUP is less than a certain point, the traditional PLS scheme performs the worst compared with the other three. This is understandable because in case of a small LSUP, the other three schemes result in more up-to-date information in addition to periodic updates. Among the qualitative schemes, GBN appears to perform the best, BN the worst, and GN places in-between, but the

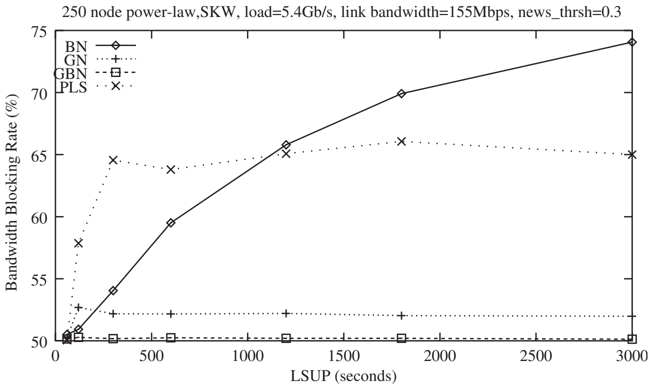


Figure 22. Blocking rate vs LSUP period on a power-law random topology of 250 nodes.

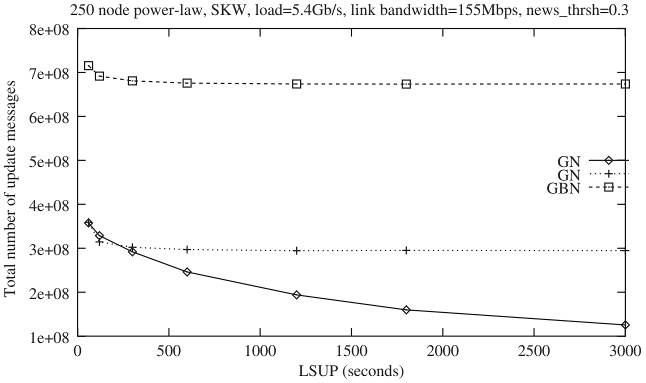


Figure 23. Cost vs LSUP period on a power-law random topology of 250 nodes.

upside of GN is that it achieves a comparable blocking ratio to that of GBN with significantly less message overhead, as shown in Figures 22 and 23.

The second observation is that BN is clearly the worst for larger values of LSUP. The explanation is fairly straightforward. The path selection algorithm chooses from among *k* paths. A 'bad news' message can only restrict the number of feasible paths to a number less than *k*. That is, we start with limited options that, along the way, are limited further, until the arrival of a next periodic LS message. Thus BN is even worse than PLS.

Finally, we investigate how the qualitative schemes perform under varying load in the network. In Figures 24–27, we change the load and observe the blocking rate and the number of LS messages sent. It is clear that regardless of the value of LSUP, GN and GBN perform the best. The impact of BN is not as detrimental, if LSUP is sufficiently short, but for a comparable cost in terms of LS messages, GN brings about better performance. For a significantly higher cost one can get the advantage of GBN. However, the overall improvement cannot be considered spectacular. Nonetheless, the smaller overhead of GN suggests that it is a reasonable

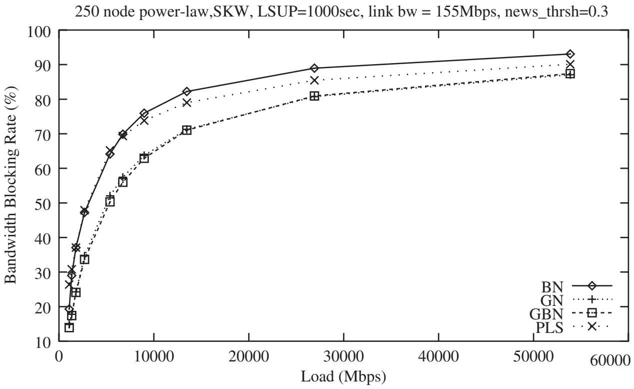


Figure 24. Blocking rate vs network load for LSUP of 1000 s. Power-law random topology of 250 nodes.

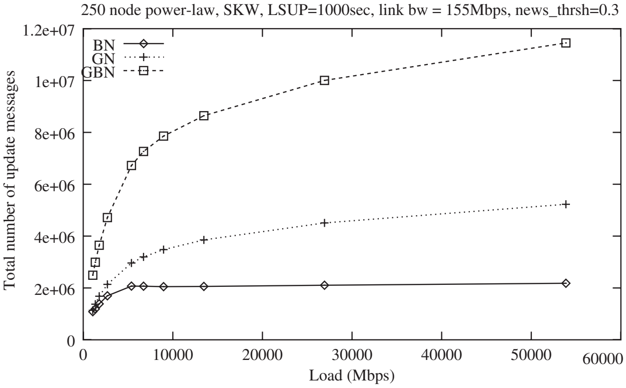


Figure 25. Message cost vs network load for LSUP of 1000 s. Power-law random topology of 250 nodes.

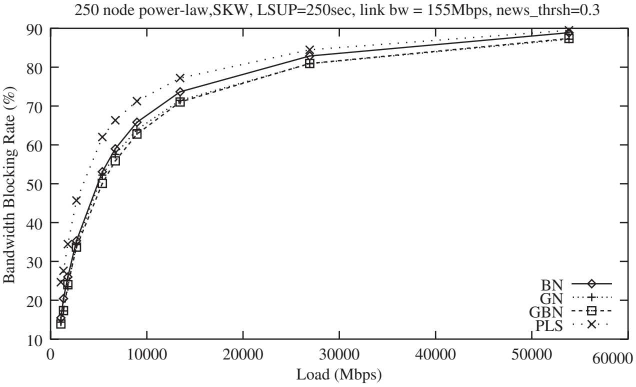


Figure 26. Blocking rate vs network load for LSUP of 250 s. Power-law random topology of 250 nodes.

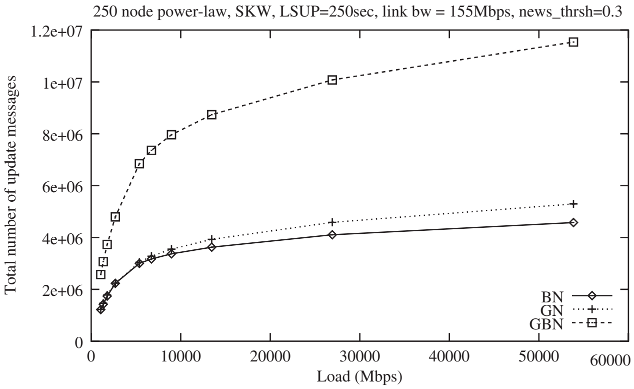


Figure 27. Message cost vs network load for LSUP of 250 s. Power-law random topology of 250 nodes.

compromise between the blocking rate improvement and the extra traffic incurred by the scheme. The experiments confirms our intuition that 'good news' should travel 'faster'. In other words, it is more important to learn when a loaded link becomes available than the other way around. The latter will be discovered anyway when needed, because attempting to setup a connection over the congested link will fail.

6. CONCLUSIONS

In this paper, we address the scalability problem of QoS routing along two dimensions: path construction/selection, and dissemination of the link state information. We have proposed a family of multiple-path routing schemes to deal with the inaccuracy of link state information, and a number of policies for reducing the number of link state messages propagated in the network.

Our experiments indicate that the proposed k-path approach to routing brings about a significant improvement over the single-path solution, especially if the link state information is outdated and inaccurate. This improvement is accomplished by trading off an insignificant increase in signaling overhead for improved blocking performance and reduced number of link state messages in the network.

Our performance studies show that hop-based algorithms consistently win over bandwidth-based ones. The best of our routing algorithms, widest-K-shortest, also outperforms the *Dynamic alternative* as well as widest-shortest when operating with the same value of $k = 2$.

We have demonstrated that a multiple choice of routing paths is one possible remedy for the problem of limited accuracy of the link state information, which is bound to haunt all networks of non-trivial size. Routing solutions that do not take this problem into account will poorly scale to large networks, in which the link state information cannot be updated and propagated too often. As it turns out, it is more important to have a choice at all than to be able to choose from a large selection. Consequently, in terms of computational complexity, our routing algorithms are comparable to those that prefer to stick to a single path. Overall, there is ample evidence that multiple-path schemes can be a scalable solution for QoS routing in environments with inaccurate link state information.

Our attempts at reducing the amount of link state information circulated in the network revealed a fundamental problem with long LSUP values. As it turns out, outdated link state information is not merely useless for guiding routing decisions, but it is actually harmful, in the sense that relying on it (and propagating it) worsens the blocking performance of the network. The results of our experiments with qualitative triggers, although interesting, do not promise spectacular improvements along this line.

Confronted by the dilemma of whether it is better to possess state information about the network, even if it is somewhat obsolete, or not to possess any altogether, we note that possessing obsolete state information may inhibit the establishment of connections that would be possible to establish if we were more 'agnostic' about the state of the network. However, one has to consider the control overhead of LS information on one hand and, on the other, several connection setup requests that fail due to the lack of knowledge about the state of the network.

We believe that most promising is the direction that associates link state information with an age field. Recent information can be accepted at the face value, while older information can be considered to be moving to an antithetical direction of the currently assumed link state. For example, if the residual bandwidth is known to be low recently, then we can assume that it will be higher later. Finally, after a point, random choice may be the best approach.

REFERENCES

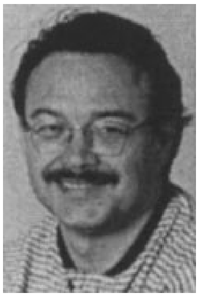
1. Chen S, Nahrstedt K. An overview of QoS routing for the next generation high-speed networks: problems and solutions. *IEEE Network* 1998; **12**:64–79.
2. Davie B, Rekhter Y. *MPLS Technology and Applications*. Morgan Kaufmann: Los Altos, CA, 2000.
3. Jamoussi B. Constraint-Based LSP Setup using LDP. Internet Draft draft-ietf-mpls-cr-ldp-05.txt, February 2001.
4. Awduche DO *et al*. RSVP-TE: Extensions to RSVP for LSP Tunnels. Internet Draft draft-ietf-mpls-rsvp-lsp-tunnel-08.txt, February 2001.
5. Iwata A, Fujita N, Ash GR. Crankback Routing Extensions for MPLS Signaling. Internet Draft draft-iwata-mpls-crankback-01.txt, July 2001.
6. Jaffe J. Algorithms for finding paths with multiple constraints. *Networks* 1984; **14**:95–116.
7. Wang Z, Crowcroft J. QoS routing for supporting resource reservation. *IEEE Journal of Selected Areas of Communications* 1996; **14**(7):1228–1234.
8. Zhang H. Service disciplines for guaranteed performance service in packet-switching networks. In *Proceedings of the IEEE*, vol. 83, October 1995.
9. Ma Q, Steenkiste P. Quality-of-service routing for traffic with performance guarantees. In *Proceedings of IFIP Fifth International Workshop on Quality of Service*, Columbia University, New York, May 1997; 115–126.
10. Shaikh A, Rexford J, Shin KG. Dynamics of quality-of-service routing with inaccurate link-state information. *Technical Report* CSE-TR-350-97, Department of Electrical Engineering and Computer Science, University of Michigan, November, 1997.
11. Guerin R, Orda A. QoS-based routing in networks with inaccurate information: theory and algorithms. *IEEE/ACM Transaction on Networking* 1999; **7**(3):350–364.
12. Apostolopoulos G, Guerin R, Tripathi SK. Quality of service routing: a performance perspective. In *SIGCOMM'98*. September 1998, ACM.
13. Apostolopoulos G, Guerin R, Kamat S. Implementation and performance measurements of QoS routing extensions to OSPF. In *Proceedings of INFOCOM'99*. IEEE: New York, March 1999.
14. Apostolopoulos G, Guerin R, Kamat S, Orda A, Tripathi SK. Intra-domain QoS routing in IP networks: a feasibility and cost/benefit analysis. *IEEE Network*, September/October 1999.
15. Apostolopoulos G, Guerin R, Kamat S, Przygienda T, Orda A, Williams D. QoS routing mechanisms and OSPF extensions. Internet Request For Comments RFC RFC2676, Internet Engineering Task Force, December 1998.
16. Ma Q. Quality-of service routing in integrated services networks. *Ph.D. Thesis*, Carnegie Mellon University, January 1998.
17. Magoni D, Pansiot J-J. Comparative study of internet-like topology generators. *Technical Report*, LSIIT laboratory, Universite Louis Pasteur, May 2001.
18. Yuan X, Zheng W. A comparative study of quality of service routing schemes that tolerate imprecise state information. *Technical Report* TR-010704, Department of Computer Science, Florida State University, 2001.
19. Apostolopoulos G, Guerin R, Kamat S, Tripathi SK. Improving QoS routing performance under inaccurate link state information. In *16th International Teletraffic Congress (ITC-16)*, June 1999.

20. Zhang Z-L, Nelakuditi S, Tsang RP. Adaptive proportional routing: a localized QoS routing approach. In *Proceedings of INFOCOM'00*, Tel Aviv, Israel. IEEE: New York, April 2000; 1566–1575.
21. Hsu C, Hui JY. Load-balanced *K*-shortest path routing for circuit-switched networks. In *Proceedings of IEEE NY/NJ Regional Control Conference*. IEEE: New York, August 1994.
22. Cidon I, Rom R, Shavitt Y. Multi-path routing combined with resource reservation. In *IEEE INFOCOM'97*. IEEE: New York, April 1997; 92–100.
23. Rao NSV, Batsell SG. QOS routing via multiple paths using bandwidth reservation. *Technical Report* 13547, INRIA, October 1998.
24. Guerin R, Orda A, Williams D. QoS routing mechanisms and OSPF extensions. In *Proceedings of GLOBECOM'97*, Phoenix, Arizona, November 1997.
25. Hao F, Zegura EW. Scalability techniques in QoS routing. *Technical Report* GIT-CC-99-04, College of Computing, Georgia Institute of Technology, 1999.
26. The ATM Forum. Private network–network interface specification version 1.0. *Technical Report* af-pnni-0055.000, ATM Forum, March 1996.
27. Hao F, Zegura EW. On scalable QoS routing: performance evaluation of topology aggregation. In *IEEE Proceedings of the INFOCOM'00*, New York, March 2000.
28. Lee WC. Topology aggregation for hierarchical routing in ATM networks. *ACM Computer Communication Review* 1995; **25**(2):82–92.
29. Breslau L, Estrin D, Zappala D, Zhang L. Limited distribution updates to reduce overhead in adaptive internetwork routing, February 1993.
30. Jia Y, Nikolaidis I, Gburzynski P. Multiple path routing in networks with inaccurate link state information. In *IEEE International Conference on Communications (ICC'2001)*, June 2001.
31. Martins EQV, Pascoal MMB, Santos JLE. The K shortest paths problem. *Technical Report*, CISUC, June 1998.
32. Guangyu Pei, Mario Gerla, Tsu-Wei Chen. Fisheye state routing in mobile ad hoc networks. In *ICDCS Workshop on Wireless Networks and Mobile Computing*, Taipei, Taiwan, April 2000; D71–D78.
33. Bolotin VA. Modeling calling holding time distributions for CCS network design and performance analysis. *IEEE JSAC* 1994; **12**(3):433–438.
34. Jain R. *The Art of Computer Systems Performance Analysis*. Wiley, Inc.: New York, 1991.

AUTHORS' BIOGRAPHIES

Yanxia Jia received her PhD degree in Computer Science from the University of Alberta, Canada in 2003. She received her MSc and BSc in Computer Science from North China Institute of Computing Technology (1997) and the Harbin Engineering University (1994), China. Currently she is an assistant professor in the Department of Mathematics and Computer Science, Ashland University, USA. Dr. Jia's research interests are protocol design and performance evaluation in computer networks.

Ioanis Nikolaidis is an Associate Professor with the Computing Science Department at the University of Alberta. He received his BSc from the University of Patras, Greece, in 1989 and his MSc and PhD from the Computer Science Department of Georgia Tech in 1991 and 1994 respectively. During 1995–96 he worked for the European Computer Industry Research Center in Munich, Germany. In January 1997, he joined the University of Alberta. His research interests include the design and performance evaluation of computer network protocols.



Pawel Gburzynski received his MSc and PhD in Computer Science from the University of Warsaw, Poland in 1976 and 1982, respectively. Before coming to Canada in 1984, he had been a research associate, systems programmer, and consultant in the Department of Mathematics, Informatics, and Mechanics at the University of Warsaw. Since 1985, he has been with the Department of Computing Science, University of Alberta, where he is a Professor. Dr. Gburzynski's research interests are in communication networks, operating systems, simulation, and performance evaluation.