Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®

Computer Networks 44 (2004) 79–102

COMPUTER NETWORKS

www.elsevier.com/locate/comnet

On selection of candidate paths for proportional routing ☆

Srihari Nelakuditi [a,*], Zhi-Li Zhang [b], David H.C. Du [b]

[a] *Department of Computer Science & Engineering, University of South Carolina, Columbia, SC 29208, USA*
[b] *Department of Computer Science & Engineering, University of Minnesota, Minneapolis, MN 55455, USA*

Received 2 October 2002; received in revised form 9 May 2003; accepted 25 June 2003

Responsible Editor: S. Fdida

Abstract

QoS routing involves selection of paths for flows based on the knowledge at network nodes about the availability of resources along paths, and the QoS requirements of flows. Several QoS routing schemes have been proposed that differ in the way they gather information about the network state and select paths using this information. Most of these schemes can be categorized as *best path* routing where a source node selects the "best" path for each incoming flow based on its current view of the global network state. It has been shown that best path routing schemes require frequent exchange of network state, imposing both communication overhead on the network and processing overheads on the core routers. On the other hand, *proportional* routing schemes proportion incoming flows among a set of *candidate* paths. Two key questions that arise under proportional routing are how to select candidate paths and how to proportion flows among candidate paths. We propose a scheme that selects a few widest disjoint paths as candidates and equalizes the blocking probabilities of the candidate paths. We show that our proportional routing approach yields higher throughput with lower overhead than best path routing approach. Furthermore, we present a method for aggregating the state of an area and extend the proportional routing approach to provide hierarchical routing across multiple areas in a large network.

Keywords: QoS routing; Proportional routing; Widest disjoint paths

1. Introduction

Routing in the current Internet focuses primarily on connectivity and typically supports only the "best-effort" datagram service. The routing protocols deployed, such as OSPF [17], use the *shortest-path* routing paradigm, where routing is optimized based on *static* metrics such as hop count or administrative weight. While the service offered by these protocols is suitable for traditional data applications such as ftp and telnet, it is not adequate for many emerging applications such as IP telephony, video on-demand and teleconferencing, which require stringent delay and bandwidth guarantees. The "shortest paths" chosen for the best-effort service may not have sufficient resources to provide the requisite service for these

☆ An earlier abridged version of this paper appeared in the Proceedings of IWQoS'01, June 2001 [18].

* Corresponding author.
E-mail addresses: srihari@cse.sc.edu (S. Nelakuditi), zhzhang@cs.umn.edu (Z.-L. Zhang), du@cs.umn.edu (D.H.C. Du).

1389-1286/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/S1389-1286(03)00346-3

applications. Moreover, with explosive growth of Internet traffic, the shortest-path routing paradigm of current Internet also leads to unbalanced traffic distribution—links on frequently used shortest paths become increasingly congested, while links not on shortest paths are underutilized [29].

QoS based routing has been proposed [6] as a way to address these issues. Under QoS routing, a flow requests for a specific quality of service and it is admitted only if the requested QoS can be guaranteed. Paths for flows are dynamically selected based upon the knowledge of resource availability (referred to as *QoS state*) at network nodes and the QoS requirements of flows. Upon arrival of a flow, the source router first selects, [1] based on its view of the network state, a path that is likely to satisfy the requirements of the flow. It then sends a setup request to reserve the requested bandwidth along the path. This request is accepted and the flow is admitted if sufficient bandwidth is available at all links along the path. Otherwise the request is rejected and in that case the flow is blocked, i.e., no attempt is made to reroute the flow. The goal of a QoS routing scheme is then to minimize the overall flow blocking probability. A survey of various QoS routing schemes can be found in [4].

In QoS routing, some knowledge regarding the (global) network QoS state is crucial in performing judicious path selection. This knowledge, for example, can be obtained through (periodic) information exchange among routers in a network. Under *best-path* routing approach, each router constructs a global view of the network QoS state by piecing together the QoS state information obtained from other routers, and selects the "best path" for a flow based on this global view of the network state. Examples of the best-path routing approach are various QoS routing schemes [1,15] based on QoS extensions to the OSPF routing protocol. Best-path routing schemes work well when each source node has a reasonably *accurate* view of the network QoS state. However, as the network resource availability changes with each flow arrival and departure, maintaining an accurate view of the network QoS state is impractical, due to prohibitive communication and processing overheads entailed by frequent QoS state information exchange. In the presence of inevitable *inaccurate* information regarding the global network QoS state, best-path routing schemes suffer degraded performance.

As a viable alternative to the best-path routing approach, we proposed [20] a novel *localized proportional routing* approach to QoS routing. Under this proportional routing approach, instead of (periodically) exchanging information with other routers to obtain a global view of the network QoS state, a source router attempts to *infer* the network QoS state from *locally collected flow statistics* such as flow arrival/departure rates and flow blocking probabilities, and performs adaptive proportioning of flows among a set of *candidate* paths based on this local information. As a result, the localized proportional routing approach avoids the drawbacks of the conventional best-path routing approach.

Under pure localized approach, the candidate path set remains static while their proportions are adjusted dynamically. A network node under localized approach can judge the quality of paths only by routing some traffic along them. So, it is not possible to update the candidate path set based on local information alone. On the other hand, due to changing network conditions, a few good candidate paths cannot be selected statically whereas dynamic selection of candidate paths requires global QoS state updates. However, these updates would not cause significant burden on the network as long as their frequency is not more than what is needed to convey connectivity information in traditional routing protocols like OSPF. The QoS state of each link could then be piggybacked along with the conventional link state updates. Hence it is important to devise proportional routing schemes that work well even with infrequent global updates. We propose such a scheme *widest disjoint paths* (wdp) that uses *infrequently* exchanged *global* information for selecting a few good candidate paths based on their long term available bandwidths. Flows are proportioned among the candidate paths using *local* information to cushion the short term variations in their

[1] Here we assume source routing with bandwidth guarantees.

available bandwidths. This *hybrid* proportional routing approach adapts at different time scales to the changing network conditions.

The schemes discussed so far assume that each router in the network is aware of the topology and the state of the whole network. This is referred to as *flat* routing and under flat routing, each router participates in link state updates and maintains detailed information about the entire network. This introduces significant burden on every router and as the size of the network grows, the overhead at each router increases tremendously. To provide a scalable solution, *hierarchical routing* is suggested [3,17] as an alternative to flat routing. Under hierarchical routing, a network is divided into multiple areas. [2] The routing within the area is flat with each router having detailed information about routers and links in that area. But the routers have only sketchy *aggregate* information about other areas. To route traffic destined for other areas, a source router may select a partial higher level path, based on the aggregate information, that gets expanded, based on the detailed information, at the ingress border router of each area along the path. Such a hierarchical routing reduces the overhead at each router by limiting the scope of link state updates and maintaining only summary information about other areas.

The hierarchical routing approach while reduces the burden on a router, introduces inaccuracy in the information available for routing. The performance of hierarchical routing hinges critically on how topology and state of a network are aggregated and how paths across areas are selected based on aggregate information. It is somewhat straightforward to summarize routing information when best-path routing is employed. On the other hand, it is not obvious how to aggregate the state when multiple paths are used to route traffic between a pair of routers. We propose an aggregation method that summarizes the state of multiple paths between two routers using a single metric and extend the proportional routing approach to provide hierarchical routing across multiple areas in a large network. We evaluate the performance and show that the proposed hierarchical proportional routing scheme performs as well as flat proportional routing. Furthermore, we demonstrate that with only aggregate information it outperforms even flat best-path routing schemes having detailed information about the network.

The rest of the paper is organized as follows. In Section 2, we introduce the proportional routing framework and motivate hybrid approach to proportional routing that selects a few good candidate paths using global information and proportions traffic among these paths using local information. Section 3 describes our scheme for selecting widest disjoint paths as candidates. Section 4 presents the proposed aggregation metric and the hierarchical proportional routing scheme. Section 5 discusses the related work. Section 6 evaluates the performance of the proposed schemes. Section 7 concludes the paper.

2. Proportional routing framework

In this section, we first lay out the basic assumptions regarding the proportional routing framework we consider in this paper. We then present a global optimal proportional routing procedure (*opr*), where we assume that the traffic loads among all source–destination pairs are known. The *opr* procedure gives the least blocking probability that can be achieved by a proportional routing scheme. However, it is quite complex and time consuming. We use the performance of *opr* as a reference to evaluate the proposed scheme. We then describe a localized adaptive proportioning approach that uses only locally collected path state metrics and assigns proportions to paths based on their quality. We then propose a hybrid approach to proportional routing that uses global information to select a few good candidate paths and employs localized adaptive proportioning to proportion traffic among these paths. [3]

[2] Also referred to as peer groups in PNNI [3].

[3] Note that proportioning of traffic among multiple paths is performed at the flow level. Once a flow is admitted, all its packets follow the same path.

2.1. Problem setup

In all the QoS routing schemes considered in this paper we assume that source routing (also referred to as explicit routing) is used. More specifically, we assume that the network topology information is available to all source nodes (e.g., via the OSPF protocol), and one or multiple explicit-routed paths or label switched paths are set up *a priori* between each source and destination pair using, e.g., MPLS [24]. Flows arriving at a source to a destination are routed along one of the explicit-routed paths (hereafter referred to as the *candidate* paths between the source–destination pair). For simplicity, we assume that all flows have the same bandwidth requirement—one unit of bandwidth. When a flow is routed to a path where one or more of the constituent links have no bandwidth left, this flow will be blocked. The performance metric in our study will be the overall blocking probability experienced by flows. We assume that flows from a source to a destination arrive randomly with a Poisson distribution, and their holding time is exponentially distributed. Hence the offered traffic load between a source–destination pair can be measured as the product of the average flow arrival rate and holding time. Given the offered traffic load from a source to a destination, the task of proportional routing is to determine how to distribute the load (i.e., route the flows) among the candidate paths between a source and a destination so as to minimize the overall blocking probability experienced by the flows.

2.2. Global optimal proportioning

The global optimal proportioning has been studied extensively in the literature (see [25] and references therein). Here it is assumed that each source node knows the complete topology information of the network (including the maximum capacity of each link) as well as the offered traffic load between every source–destination pair. With the global knowledge of the network topology and offered traffic loads, the *optimal* proportions, for distributing flows among the paths between each source–destination pair, can be computed as described below.

$\mathcal{N}$:	$\{1, 2, ..N\}$, vector of N nodes in the given network.
$\mathcal{L}$:	$\{1, 2, ..L\}$, vector of L links in the given network.
$\hat{C}$:	$\{\hat{c}_1, \hat{c}_2, ..\hat{c}_L\}$, vector of capacities of the links, $\hat{c}_i > 0$.
σ :	a pair (s, d), where $s, d \in \mathcal{N}$.
r :	a path, i.e a set of links $\in \mathcal{L}$, from source to destination.
$\hat{R}_\sigma$:	a set of feasible paths between a source-destination pair σ.
R_σ :	a set of candidate paths between a source-destination pair σ.
ν_l :	load on the link l.
ν_σ :	load on the source-destination pair σ.
b_l :	blocking probability of link l.
b_r :	blocking probability of path r.
α_r :	proportion of load assigned to path r.

Fig. 1. Notation.

Consider an arbitrary network topology with N nodes and L links (please refer to Fig. 1 for the notation). For $l = 1, \ldots, L$, the maximum capacity of link l is $\hat{c}_l > 0$, which is assumed to be fixed and known. The links are unidirectional, i.e., carry traffic in one direction only. Let $\sigma = (s, d)$ denote a source–destination pair in the network. Let λ_σ denote the average arrival rate of flows arriving at source node s destined for node d. The average holding time of the flows is $1/\mu_\sigma$. Recall that each flow is assumed to request one unit of bandwidth, and that the flow arrivals are Poisson, and flow holding times are exponentially distributed. Thus the offered load between the source–destination pair σ is $v_\sigma = \lambda_\sigma/\mu_\sigma$.

Let $\widehat{R}_\sigma$ denote the set of *feasible* paths for routing flows between the pair σ. The global optimal proportioning problem can be formulated [10–12] as the problem of finding the optimal proportions $\{\alpha_r^*, r \in \widehat{R}_\sigma\}$ where $\sum_{r \in \widehat{R}_\sigma} \alpha_r^* = 1$, such that the overall flow blocking probability in the network is minimized. Or equivalently, finding the optimal proportions $\{\alpha_r^*, r \in \widehat{R}_\sigma\}$ such that the total carried traffic in the network, $W = \sum_\sigma \sum_{r \in \widehat{R}_\sigma} \alpha_r v_\sigma (1 - b_r)$ is maximized. Here b_r is the blocking probability on path r when a load of $v_r = \alpha_r v_\sigma$ is routed through r. Then the set of *candidate* paths R_σ are a subset of feasible paths $\widehat{R}_\sigma$ with proportion larger than a negligible value ϵ, i.e., $R_\sigma = \{r\colon r \in \widehat{R}_\sigma, \alpha_r^* > \epsilon\}$. This global optimal proportional routing problem is a constrained nonlinear optimization problem and can be solved using sequential quadratic programming [5,23].

2.3. Localized adaptive proportioning

The optimal proportioning procedure described above requires global information about the offered load between each source–destination pair. It is also quite complex and thus time consuming. We have shown [19,20] that it is possible to obtain near-optimal proportions using simple localized strategies such as equalization of blocking probabilities (*ebp*) and equalization of blocking rates (*ebr*). Let $\{r_1, r_2, \ldots, r_k\}$ be the set of k candidate paths between a source–destination pair. The objective of the *ebp* strategy is to find a set of proportions $\{\alpha_{r_1}, \alpha_{r_2}, \ldots, \alpha_{r_k}\}$ such that flow blocking probabilities on all the paths are equalized, i.e., $b_{r_1} = b_{r_2} = \cdots = b_{r_k}$, where b_{r_i} is the flow blocking probability on path r_i. On the other hand, the objective of the *ebr* strategy is to equalize the flow blocking rates, i.e., $\alpha_{r_1} b_{r_1} = \alpha_{r_2} b_{r_2} = \cdots = \alpha_{r_k} b_{r_k}$. By employing these strategies a source node can adaptively route flows among multiple paths to a destination, in proportions that are commensurate with the *perceived* qualities of these paths. The perceived quality of a path between a source and a destination is inferred based on locally maintained flow statistics: the offered load on the path and the resulting blocking probability of the flows routed along the path. This information can be easily collected at a source by keeping track of the number of flows routed along a path and the number of flows blocked along that path.

In this work, we use a simpler approximation to *ebp* that computes new proportions as follows. The proportions are changed gradually using two parameters *maximum proportional change*, δ and the corresponding *expected proportional change in blocking probability*, ϕ to approximate Erlang's loss formula. These parameters capture the relative change in blocking probability corresponding to a change in the load, i.e., if the load is changed by a fraction δ, the blocking probability would change by a fraction ϕ. Based on these parameters, the new load $v_{r_i}^{(1)}$ onto a path r_i is computed based on the current offered load $v_{r_i}^{(0)}$ and blocking probability $b_{r_i}^{(0)}$. Let $\bar{b}^{(0)} = \sum_{i=1}^{k} \alpha_{r_i}^{(0)} b_{r_i}^{(0)}$ be the current mean blocking probability. The load onto a path r_i is decreased as follows if its current blocking probability $b_{r_i}^{(0)}$ is higher than the mean $\bar{b}^{(0)}$:

$$v_{r_i}^{(1)} = \frac{v_{r_i}^{(0)}}{1 + \min\left(\delta, \frac{b_{r_i}^{(0)} - \bar{b}^{(0)}}{b_{r_i}^{(0)}} \frac{\delta}{\phi}\right)}.$$

Similarly the load is increased as follows if $b_{r_i}^{(0)}$ is lower than $\bar{b}^{(0)}$:

$$v_{r_i}^{(1)} = v_{r_i}^{(0)}\left(1 + \min\left(\delta, \frac{\bar{b}^{(0)} - b_{r_i}^{(0)}}{\bar{b}^{(0)}} \frac{\delta}{\phi}\right)\right).$$

The corresponding proportions would then be $\alpha_i^{(1)} = v_{r_i}^{(1)} / \sum_{j=1}^{k} v_{r_j}^{(1)}$. Essentially the load onto a path r_i is increased or decreased based on whether its current blocking probability $b_{r_i}^{(0)}$ is lower or higher respectively than the mean $\bar{b}^{(0)}$. The magnitude of change is determined based on the relative distance of $b_{r_i}^{(0)}$ from $\bar{b}^{(0)}$. Furthermore it is ensured that the change is gradual by limiting the magnitude of change to a fraction δ. The *mean time between proportion computations* is controlled by a configurable parameter θ. This period θ should be large enough to allow for a reasonable measurement of the quality of the candidate paths. The blocking performance of the candidate paths are observed for a period θ and at the end of the period the proportions are recomputed.

2.4. Hybrid approach to proportional routing

The global proportioning procedure described above computes optimal proportions α_r^* for each path r given a feasible path set $\widehat{R}_\sigma$ for each source–destination pair σ. Taking into account the overhead associated with setting up and maintaining the paths, it is desirable to minimize the number of candidate paths while minimizing the overall blocking probability. However achieving both the minimization objectives may not be practical. Note that the blocking probability minimization alone, for a fixed set of candidate paths, is quite complex and time consuming. Minimizing the number of candidate paths involves experimenting with different combinations of paths and the complexity grows exponentially as the size of the network increases. Hence it is not feasible to find an optimal solution that minimizes both the objectives. Considering that achieving the absolute minimal blocking is not very critical, it is

worthwhile investigating heuristic schemes that *tradeoff slight increase in blocking for significant decrease in the number of candidate paths*.

The localized approach to proportional routing is simple and has several important advantages. However it has a limitation that routing is done based solely on the information collected locally. A network node under localized proportional routing approach can judge the quality of paths/links only by routing some traffic along them. It would have no knowledge about the state of the rest of the network. While the proportions for paths are adjusted to reflect the changing qualities of paths, the candidate path set itself remains static. To ensure that the localized scheme adapts to varying network conditions, many feasible paths have to be made candidates. It is not possible to preselect a few good candidate paths statically. Hence it is desirable to *supplement localized proportional routing* with a mechanism that *dynamically selects a few good candidate paths*.

We propose such a hybrid approach to proportional routing where locally collected path state metrics are supplemented with globally exchanged link state metrics. The traffic is proportioned among the candidate paths using local information. These proportions are computed after every θ interval. A set of few good candidate paths R_σ are maintained for each pair σ and this set is updated based on the global information. The mean time between candidate path updations is controlled by a configurable parameter ξ, where $\xi \gg \theta$. In the next section, we describe a scheme that selects widest disjoint paths as candidates.

3. Candidate path selection

Two key questions that arise in candidate path selection are how many paths are needed and how to find these paths. Clearly, the number and the quality of the candidate paths selected dictate the performance of a proportional routing scheme. There are several reasons why it is desirable to minimize the number of paths used for routing. First, there is a significant overhead associated with establishing, maintaining and tearing down of paths. Second, the complexity of the scheme that distributes traffic among multiple paths increases considerably as the number of paths increases. Third, there could be a limit on the number of explicitly routed paths such as label switched paths in MPLS [24] that can be setup between a pair of nodes. Therefore it is desirable to use *as few paths as possible* while at the same time *minimize the congestion* in the network.

In this section, we present the procedure for selecting a few good candidate paths. To help determine whether a path is good and whether to include it in the candidate path set, we define *width* of a path and introduce the notion of *width* of a *set of paths*. The candidate path set R_σ for a pair σ is changed only if it increases the width of the set R_σ or decreases the size of the set R_σ without reducing its width. The widths of paths are computed based on link state updates that carry *average residual bandwidth* information about each link.

A basic question that needs to be addressed by any path selection procedure is what is a "good" path. In general, a path can be categorized as good if its inclusion in the candidate path set decreases the overall blocking probability considerably. It is possible to judge the utility of a path by measuring the performance with and without using the path. However, it is not practical to conduct such inclusion–exclusion experiment for each feasible path. Moreover, each source has to independently perform such trials without being directly aware of the actions of other sources which are only indirectly reflected in the state of the links. Hence each source has to try out paths that are likely to decrease blocking and make such decisions with some local objective that leads the system towards a global optimum.

When identifying a set of candidate paths, another issue that requires attention is the sharing of links between paths. A set of paths that are good *individually* may not perform as well as expected *collectively*. This is due to the sharing of *bottleneck* links. When two candidate paths share a bottleneck link, it may be possible to remove one of the paths and shift all its load to the other path without increasing the blocking probability. Thus by ensuring that candidate paths do not share bottleneck links, we can reduce the number of candidate paths without increasing the blocking

probability. A simple guideline to enforce this could be that the candidate paths be mutually disjoint, i.e., they do not share *any* links. This is overly restrictive, since even with shared links, some paths can cause reduction in blocking if those links are not congested. What matters is not the sharing itself but *the sharing of bottleneck links*. While the sharing of links among the paths is *static* information independent of traffic, identifying bottleneck links is *dynamic* since the congestion in the network depends on the offered traffic and routing patterns. Therefore it is essential that candidate paths be *mutually disjoint w.r.t. bottleneck links*.

To judge the quality of a path, we define *width* of a path as the residual bandwidth on its bottleneck link. Let $\hat{c}_l$ be the maximum capacity of link l and v_l be the average load on it. The difference $c_l = \hat{c}_l - v_l$ is the average residual bandwidth on link l. Then the *width* w_r of a path r is given by $w_r = \min_{l \in r} c_l$. The larger its width is, the better the path is, and the higher its potential is to decrease blocking. Similarly we define *distance* [15] of a path r as $\sum_{l \in r} (1/c_l)$. The shorter the distance is, the better the path is. The widths and distances of paths can be computed given the residual bandwidth information about each link in the network. This information can be obtained through periodic link state updates. To discount short term fluctuations, the *average residual bandwidth* information is exchanged. Let τ be the update interval and u_l^t be the utilization of link l during the period $(t - \tau, t)$. Then the average residual bandwidth at time t, $c_l^t = (1 - u_l^t)\hat{c}_l$. Hereafter without the superscript, c_l refers to the most recently updated value of the average residual bandwidth of link l.

To aid in path selection, we also introduce the notion of *width* for a *set of paths* R, which is the maximum flow carriable by paths in the set R that is computed as follows. We first pick the path r^* from R with the largest width w_{r^*}. If there are multiple such paths, we choose the one with the shortest distance d_{r^*}. We then decrease the residual bandwidth on all its links by an amount w_{r^*}. This effectively makes the residual bandwidth on its bottleneck link to be 0. We remove the path r^* from the set R and then select a path with the next largest width based on the just updated residual bandwidths. Note that this change in residual bandwidths of links is local and only for the purpose computing the width of R. This process is repeated till the set R becomes empty. The sum of all the widths of paths computed thus is defined as the *width of* R. Note that when two paths share a bottleneck link, the width of two paths together is same as the width of a single path. The width of a path set computed thus essentially accounts for the sharing of links between paths.

The procedure to compute the width of a path set R is shown in Fig. 2. In each iteration, a subset of paths R^* with the largest width w^* are identified (lines 4–5). From these widest paths, a path r^* with the shortest distance d^* is selected (lines 6–7). The width w^* of path r^* is added to the total width W (line 8). The residual capacities of all the links along the path r^* is reduced by an amount w^* (lines 9–10). This in turn affects the widths of other paths in R. Note that this change in residual capacities of links is local and only for the purpose computing width of R. The path r^* is removed from the set (line 11) and this process is repeated till the set R becomes empty (line 3). The resulting W is considered to be the width of R. The narrowest path, i.e., the last path removed from the set R is referred to as NARROWEST(R).

Based on this notion of width of a path set, we propose a path selection procedure that *adds* a new candidate path only if its inclusion *increases the width*. It *deletes* an existing candidate path if its exclusion *does not decrease* the total width. When the number of candidate paths reaches the specified limit, it replaces a candidate path with another

```
1.  PROCEDURE WIDTH(R)
2.    W = 0
3.    While R ≠ ∅
4.      w* = max_{r∈R} w_r
5.      R* = {r : r ∈ R, w_r = w*}
6.      d* = min_{r∈R*} d_r
7.      r* = {r : r ∈ R*, d_r = d*}
8.      W = W + w*
9.      For each l in r*
10.       c_l = c_l − w*
11.     R = R \ r*
12.   Return W
13. END PROCEDURE
```

Fig. 2. The procedure to compute width for a path set R.

path if this change increases the width. In other words, each modification to the candidate path set either *increases the width* or *decreases the number* of candidate paths. We refer to this scheme as widest disjoint paths (*wdp*). Essentially *wdp* selects *widest paths* that are *mutually disjoint w.r.t. bottleneck links*.

The selection procedure is shown in Fig. 3. First, the load contributed by each existing candidate path is deducted from the corresponding links (lines 2–4). After this *local adjustment*, the residual bandwidth c_l on each link l reflects the load offered on l by all source destination pairs other than σ. Given these adjusted residual bandwidths, the candidate path set R_σ is modified as follows.

The benefit of inclusion of a feasible path r is determined based on the number of existing candidate paths (lines 6–8). If this number is below the specified limit η, the resulting width W_r is the width of $R_\sigma \cup r$. Otherwise, it is the width of $R_\sigma \cup r \setminus$ NARROWEST($R_\sigma \cup r$), i.e., the width after excluding the narrowest path among $R_\sigma \cup r$. Let W^+ be the largest width that can be obtained by adding a feasible path (line 9). This width W^+ is compared with width of the current set of candidate paths. A feasible path is made a candidate if its inclusion in set R_σ increases the width by a fraction ψ (line 10). Here $\psi > 0$ is a configurable parameter to ensure that each addition improves the width by a significant amount. It is possible that many feasible paths may cause the width to be increased to W^+. Among such paths, the path r^+ with the shortest distance is chosen for inclusion (lines 11–13). Let r^- be the narrowest path in the set $R_\sigma \cup r$ (line 14). The path r^- is replaced with r^+ if either the number of paths already reached the limit or the path r^- does not contribute to the width (lines 15–16). Otherwise the path r^+ is simply added to the set of candidate paths (lines 17–18). When no new path is added, an existing candidate path is deleted from the set if it does not change the width (lines 20–22). In all other cases, the candidate path set remains unaffected. It is obvious that this procedure always either increases the width or decreases the number of candidate paths.

```
1.  PROCEDURE SELECT(σ)
2.    For each path r in R_σ
3.      For each link l in r
4.        c_l = c_l + (1 − b_r)ν_r
5.    If |R_σ| < η
6.      W_r = WIDTH(R_σ ∪ r), ∀r ∈ R̂_σ \ R_σ
7.    Else
8.      W_r = WIDTH(R_σ ∪ r \ NARROWEST(R_σ ∪ r)), ∀r ∈ R̂_σ \ R_σ
9.    W^+ = max_{r∈R̂_σ\R_σ} W_r
10.   If (W^+ > (1 + ψ) WIDTH(R_σ))
11.     R^+ = {r : r ∈ R̂_σ \ R_σ, W_r = W^+}
12.     d^+ = min_{r∈R^+} d_r
13.     r^+ = {r : r ∈ R^+, d_r = d^+}
14.     r^− = NARROWEST(R_σ ∪ r^+)
15.     If (|R_σ| = η or WIDTH(R_σ ∪ r^+ \ r^−) = W^+)
16.       R_σ = R_σ ∪ r^+ \ r^−
17.     Else
18.       R_σ = R_σ ∪ r^+
19.   Else
20.     r^− = NARROWEST(R_σ)
21.     If WIDTH(R_σ \ r^−) = WIDTH(R_σ)
22.       R_σ = R_σ \ r^−
23.   END PROCEDURE
```

Fig. 3. The candidate path set selection procedure for pair σ.

It should be noted that though *wdp* uses link state updates it does not suffer from the *synchronization* problem unlike global best path routing schemes. There are several reasons contributing to the stability of *wdp*: (1) The information exchanged about a link is its *average* not *instantaneous* residual bandwidth and hence less variable. (2) The traffic is proportioned among few "good" paths instead of loading the "best" path based on inaccurate information. (3) Each source–destination pair uses only a few candidate paths and makes only gradual changes to the candidate path set. (4) The new candidate paths are selected for a pair only after deducting the load contributed by the current candidate paths from their links. Due to such local adjustment, even with global link state updates, mass synchronization is avoided since the view of the network for each node is different. (5) When network is in a stable state of convergence, the information carried in link state updates would not become outdated and consequently each node would have reasonably accurate view of the network. Essentially the nature of information exchanged and the manner in which it is utilized work in a mutually beneficial fashion and lead the system towards a stable optimal state.

4. Hierarchical proportional routing

The above discussion on candidate path selection and proportional routing assumes that the network is flat. To reduce the overhead and to increase the scalability of routing, a large network is generally divided into multiple areas. Routers inside an area have detailed information about their area but only aggregated information about other areas. The border routers perform *topology and state aggregation* of their areas, exchange this aggregated information among themselves and also propagate a summary of this information to interior routers within their areas. Due to incomplete information about the whole network, an interior source router can only select a *skeletal* path to a destination which gets *expanded* by the border routers along the way to the destination. This is referred to as *hierarchical routing*. The key issue in extending the multipath routing solution to hierarchical routing is how to perform aggregation when multiple paths are used to route between a pair of border routers.

4.1. Topology and state aggregation

Topology aggregation is concerned with capturing the structure of an area which is relatively static. There are several proposals for topology aggregation such as [13,14]. We take a simple approach where a border router makes other routers in its area appear as directly connected to it by a *logical link*. This approach is similar to the one employed by OSPF [17]. Then state aggregation is about summarizing the state of the network by assigning the attributes to these logical links. When best-path routing is used to route traffic within an area, it is straightforward to perform state aggregation: the attributes of a logical link are that of the best-path. For example, when traffic within an area is routed along shortest paths, then the *distance* of the logical link between a pair of routers would simply be the distance of the shortest path. On the other hand, it is not obvious how to summarize the state when multiple paths are used to route traffic between a pair of routers.

The multipath routing scheme *wdp* described in the previous section lends itself very well to aggregation. Note that *wdp* selects candidate paths such that the total *width* of these candidates is as large as possible. This width essentially captures the traffic-carrying capacity of all the paths between a pair of routers. Hence, we propose to summarize the state of multiple paths between a pair of routers by a single metric, *the width of its candidate paths*. This metric not only provides more accurate but also more stable description of the state of the network than the best-path metric which could change quite frequently. Given a set of candidate paths R_σ of a pair σ, its width, W_σ can be computed using the procedure shown in Fig. 2, i.e., $W_\sigma = \mathrm{WIDTH}(R_\sigma)$.

We propose a hierarchical routing scheme based on this aggregate metric. This scheme employs *wdp* for intra-area routing and exchanges aggregate width information among border routers. This aggregate information is propagated to interior routers by border routers and consequently used in the selection of skeletal higher level paths to destinations by source routers.

4.2. Skeletal path selection

Suppose that a border router injects the aggregate information it receives from other border routers into its area. For routing to a destination in a different area, an interior source router could then select an higher level skeletal path consisting of border routers. Each border router along the skeletal path would then use intra-area routing to find a path to the next border router along the path till the destination is reached.

The *wdp* scheme can be extended naturally to provide hierarchical proportional routing. At the higher level each area is represented by a set of logical links. The average available bandwidth of a logical link corresponding to a pair σ is set to the width of σ, W_σ. Now we can form a higher level network consisting of physical backbone links and the logical aggregate links representing each area. For example, consider the topology shown in Fig. 4(a). The corresponding higher level view of border routers is shown in Fig. 4(b). Under hierarchical routing, an interior source router would have a detailed view similar to Fig. 4(a) about its

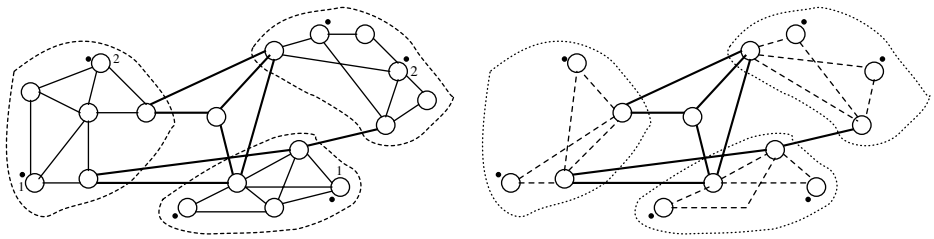

Fig. 4. (a) Flat and (b) hierarchical views of the hisp topology.

own area and an aggregate view similar to Fig. 4(b) about the rest of the network.

Given the higher level view of the network, a source router can apply *wdp* scheme shown in Fig. 3 *as is* on this higher level network to identify a set of widest disjoint paths as candidates and perform *ebp* based proportioning among these higher level candidate paths. However, there is one difference in that two paths that are considered disjoint at the higher level may not be really disjoint at the lower level. We can be conservative and treat two higher level paths as disjoint only if they do not pass through the same area. In our study, enforcement of such a constraint did not make much difference since such paths were anyway not chosen due to the sharing of inter-area links. We refer to this hierarchical version of *wdp* where *wdp* is used for both intra-area and inter-area routing as *hwdp*.

5. Related work

Several multipath routing schemes have been proposed for balancing the load across the network. The Equal Cost Multipath (ECMP) [17], Optimized Multipath (OMP) [28,29], and MPLS Adaptive Traffic Engineering (MATE) [7] schemes perform packet level forwarding decisions. ECMP splits the traffic *equally* among multiple equal cost paths. However, these paths are determined statically and may not reflect the congestion state of the network. Furthermore, it is desirable to apportion the traffic according to the quality of each path. OMP is similar in spirit to our work. OMP also uses updates to gather link loading information, selects a set of best paths and distributes traffic among them. However, our scheme makes routing decisions at the flow level and consequently the objectives and procedures are different. MATE is different from our work in that it assumes that several explicit LSPs are setup and uses active probing to measure the quality of these paths. This paper addresses the problem of selection of good candidate paths while proportioning among them using locally collected flow blocking information without any explicit probes.

Another approach to path selection is to precompute maximally disjoint paths [27] and attempt them in some order. This is static and overly conservative. What matters is not the sharing itself but *the sharing of bottleneck links*, which change with network conditions. In our scheme we dynamically select paths such that they are disjoint *w.r.t.* bottleneck links.

QoS routing schemes have been proposed [4,8,15,30] where flow level routing decisions are made based upon the knowledge of the resource availability at network nodes and the QoS requirements of flows. This knowledge is obtained through global link state information exchange among routers in a network. These schemes, which we refer to as global best path routing schemes, construct a global view of the network QoS state by piecing together the information about each link, and perform path selection based solely on this global view. Examples of global best path routing schemes are *widest shortest path* (wsp) [8], *shortest widest path* (swp) [30], and *shortest distance path* (sdp) [15]. While *wdp* also uses link state updates, the nature of information exchanged and the manner in which it is utilized is quite different from global best path routing schemes. In

Section 6, we demonstrate that *wdp* provides higher throughput with lower overhead than these schemes.

The best path routing based hierarchical routing is studied in [9]. Consider a hierarchical version of *wsp* which we refer to as *hwsp*. Under *hwsp*, the state of routing between a pair of routers is summarized by the widest shortest path. Hence, the bandwidth and the hop count of a logical link between a pair of routers is given by bottleneck bandwidth and the hop count of the corresponding widest shortest path. The *hwsp* is a hierarchical source routing scheme where a source router selects a widest shortest higher level logical path based on the bandwidths and hop counts of the logical links. This skeletal path is then expanded by the border routers using *wsp* to route within the area. In the next section, we show that our hierarchical proportional routing scheme *hwdp* outperforms *hwsp*.

6. Performance analysis

In this section, we evaluate the performance of the proposed hybrid proportional routing scheme *wdp*. [4] We start with the description of the simulation environment. First, we compare the performance *wdp* with the optimal scheme *opr* and show that *wdp* converges to near-optimal proportions. Furthermore, we demonstrate that the performance of *wdp* is relatively insensitive to the values chosen for the configurable parameters. We then contrast the performance of *wdp* with global QoS routing scheme *wsp* in terms of the overall blocking probability and routing overhead. Finally, we compare the performance of hierarchical versions of these schemes.

6.1. Simulation environment

The Fig. 5 shows the *isp* topology used in our study. This topology of an ISP backbone network is also used in [1,15]. For simplicity, all the links

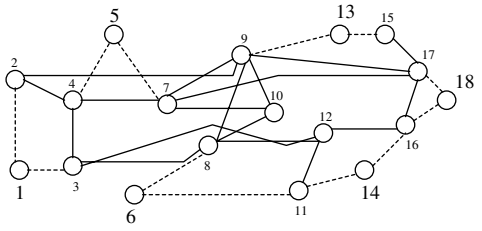

Fig. 5. The topology used for performance evaluation.

are assumed to be bidirectional and of equal capacity in each direction. There are two types of links: *solid* and *dotted*. All solid links have same capacity with C_1 units of bandwidth and similarly all the dotted links have C_2 units. The dotted links are the access links and for the purpose of our study their capacity is assumed to be higher than solid links. Otherwise, access links become the bottleneck limiting the impact of multipath routing and hence not an interesting case for our study. Flows arriving into the network are assumed to require one unit of bandwidth. Hence a link with capacity C can accommodate at most C flows simultaneously.

The flow dynamics of the network is modeled as follows (similar to the model used in [26]). The nodes labeled with bigger font are considered to be source (ingress) or destination (egress) nodes. Flows arrive at a source node according to a Poisson process with rate λ. The destination node of a flow is chosen randomly from the set of all nodes except the source node. The holding time of a flow is exponentially distributed with mean $1/\mu$. Following [26], the offered network load on *isp* is given by $\rho = \lambda N \bar{h}/\mu(L_1C_1 + L_2C_2)$, where N is the number of source nodes, L_1 and L_2 are the number of solid and dotted links respectively, and $\bar{h}$ is the mean number of hops per flow, averaged across all source–destination pairs. The parameters used in our simulations are $C_1 = 20$, $C_2 = 30$, $1/\mu = 1$ min (here after written as just *m*). The topology specific parameters are $N = 6$, $L_1 = 36$, $L_2 = 24$, $\bar{h} = 3.27$. The average arrival rate at a source node λ is set depending upon the desired load ρ.

The *wdp* scheme has several configurable parameters as listed in Fig. 6. These parameters are set as follows by default. Any change from these

[4] Ideally, this hybrid scheme should be referred to as *wdp + ebp*. But we simply refer to it as *wdp*.

η :	maximum number of paths allowed between a pair
ψ :	width increase threshold for changing the path set
τ :	mean time between link state updates
θ :	mean time between computation of proportions
ξ :	mean time between computation of candidate paths

Fig. 6. Configurable parameters in *wdp*.

settings is explicitly mentioned wherever necessary. The default values for these parameters are $\psi = 0.2$, $\tau = 30$ m, $\theta = 60$ m, $\xi = 180$ m. For each pair σ, all the paths between them whose length is at most one hop more than the minimum number of hops is included in the feasible path set $\widehat{R}_\sigma$. The maximum number of candidate paths allowed between a source–destination pair is varied from 1 to 4 but in most simulations it is set to 3. The amount of offered load on the network ρ is set to 0.55. Each run simulates arrival of 1,000,000 flows and the results corresponding to the latter half of the simulation are reported here. These simulations are performed using ns2 [22]. The scripts corresponding to the results presented in this paper can be found in [21].

6.2. Performance of wdp

In this section, we compare the performance of *wdp* and *opr* to show that *wdp* converges to near-optimal proportions using only a few paths for routing traffic. We also demonstrate that *wdp* is relatively insensitive to the settings for the configurable parameters.

6.2.1. Convergence

Fig. 7 illustrates the convergence process of *wdp*. The results are shown for different values of $\eta = 1, \ldots, 4$. Fig. 7(a) compares the performance of *wdp*, *opr* and *ebp*. The performance is measured in terms of the overall flow blocking probability, which is defined as the ratio of the total number of blocked flows to the total number of flow arrivals. The overall blocking probability is plotted as a function of time. In the case of *opr*, the algorithm is run offline to find the optimal proportions given the set of feasible paths and the offered load between each pair of nodes. The resulting proportions are then used in simulation for statically proportioning the traffic among the set of feasible paths. The *ebp* scheme refers to the localized scheme used in isolation for adaptively proportioning across all the feasible paths. As noted earlier all paths of length either *minhop* or *minhop* + 1 are chosen as the set of feasible paths in our study.

There are several conclusions that can be drawn from Fig. 7(a). First, the *wdp* scheme converges for all values of η. Given that the time between changes to candidate path sets, ξ, is 180 m, it reaches steady state within (on average) 5 path recomputations per pair. Second, there is a marked reduction in the blocking probability when the number of paths allowed, η, is changed from 1 to 2. It is evident that there is quite a significant gain in using multipath routing instead of single path routing. When the limit η is increased from 2 to 3

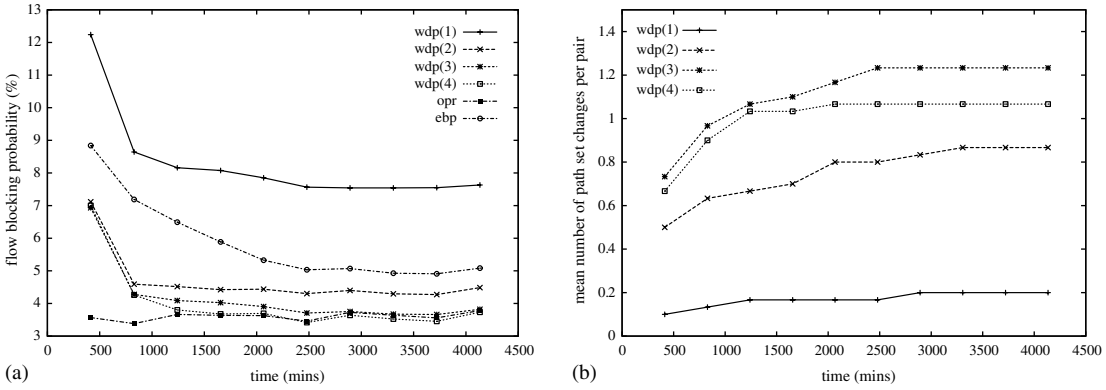

Fig. 7. Convergence process of *wdp*. (a) Blocking probability and (b) number of changes to candidate path.

the improvement in blocking is somewhat less but significant. Note that in our topology there are at most two paths between a pair that do not share any links. But there could be more than two paths that are mutually disjoint w.r.t. bottleneck links. The performance difference between η values of 2 and 3 is an indication that we only need to ensure that candidate paths do not share congested links. However using more than 3 paths per pair helps very little in decreasing the blocking probability. Third, the *ebp* scheme also converges, albeit slowly. Though it performs much better than *wdp* with single path, it is worse than *wdp* with $\eta = 2$. But when *ebp* is used in conjunction with path selection under *wdp* it converges quickly to lower blocking probability using only a few paths. Finally, using at most 3 paths per pair, the *wdp* scheme approaches the performance of optimal proportional routing scheme. [5]

Fig. 7(b) establishes the convergence of *wdp*. It shows the average number of changes to the candidate path set as a function of time. Here the change refers to either addition, deletion or replacement operation on the candidate path set R_σ of any pair σ. Note that the cumulative number of changes are plotted as a function of time and hence a plateau implies that there is no change to any of the path sets. It can be seen that the path sets change incrementally initially and after a while they stabilize. Thereafter each pair sticks to the set of chosen paths. It should be noted that starting with at most 3 minhop paths as candidates and making as few as 1.2 changes to the set of candidate paths, the *wdp* scheme achieves almost optimal performance.

We now compare the average number of paths used by a source–destination pair for routing. The mean number of paths used by each pair for routing are shown in Fig. 8. Note that in *wdp* scheme η only specifies the maximum allowed number of paths per pair. The actual number of paths selected for routing depends on their widths. The average number of paths used by *wdp* for η of

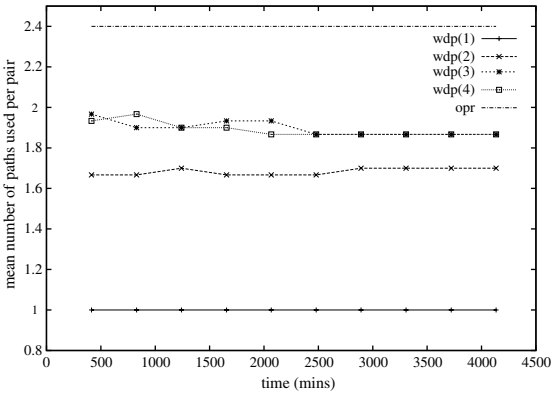

Fig. 8. Number of paths used for routing.

2 and 3 are 1.7 and 1.9 respectively. The number of paths used stays same even for higher values of η. The *ebp* scheme uses all the given feasible paths for routing. It can measure the quality of a path only by routing some traffic along that path. The average number of feasible paths chosen are 5.6. In case of *opr* we count only those paths that are assigned a proportion of at least 0.10 by the optimal offline algorithm. The average number of such paths under *opr* scheme are 2.4. These results support our claim that *wdp* performs almost like *opr* while using fewer paths.

6.2.2. Sensitivity

The *wdp* scheme requires periodic updates to obtain global link state information and to perform path selection. To study the impact of update interval on the performance of *wdp*, we conducted several simulations with different update intervals ranging from $\tau = 1$ to 60 m. The Fig. 9(a) shows the flow blocking probability as a function of update interval. At smaller update intervals there is some variation in the blocking probability, but much less variation at larger update intervals. It is also clear that increasing the update interval does not cause any significant change in the blocking probability. To study the effect of update interval on the stability of *wdp*, we plotted the average number of path set changes as a function of update interval in Fig. 9(b). It shows that the candidate path set of a pair changes often when the updates are frequent. When the update interval is small, the average residual bandwidths of links resemble

[5] It may be surprising to see *wdp*(4) performing better, though very slightly, than *opr*. This is due to the statistical fluctuations in traffic and the static proportioning in case of *opr*.

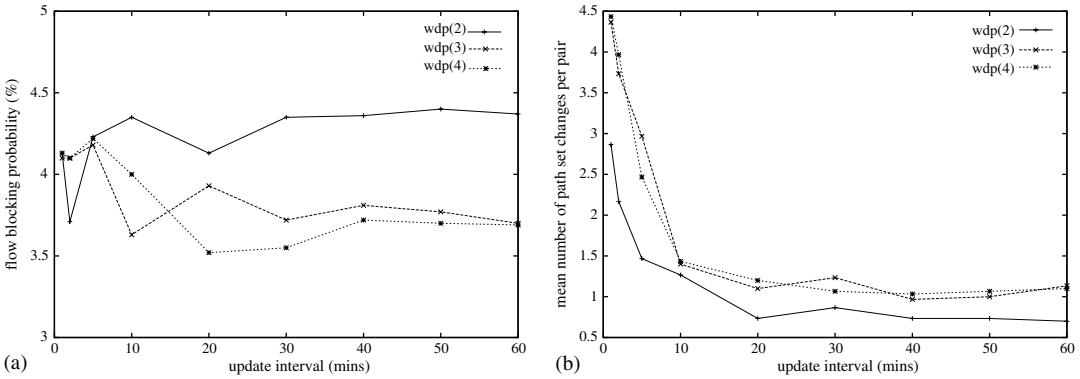

Fig. 9. Sensitivity of *wdp* to update interval τ. (a) Blocking probability and (b) number of path set changes.

their instantaneous values, thus highly varying. Due to such variations, paths may appear wider or narrower than they actually are, resulting in unnecessary changes to candidate paths. However, this does not have a significant impact on the blocking performance due to adaptive proportional routing among the selected paths. For the purpose of reducing overhead and increasing stability, we suggest that the update interval τ be reasonably large, while ensuring that it is much smaller than the path recomputation interval ξ.

We also studied the sensitivity of *wdp* to the width increase threshold parameter for changing the candidate path set, ψ. The simulations were run for five different values of ψ from 0.10 to 0.30. Fig. 10(a) shows the flow blocking probability as a function of ψ. It can be seen that the blocking performance of *wdp* is relatively insensitive to the value of ψ. Once again in Fig. 10(b) the number of path set changes is shown as a function of ψ. As expected, there are fewer changes as the ψ value increases. However this does not effect the blocking significantly since much wider paths are included in the set anyway. Only those paths that may contribute to the width by a small amount are excluded by choosing higher ψ value. To ensure that good paths are not excluded and not so good paths are not included we suggest setting ψ to a value around 0.20.

6.3. Comparison of wsp and wdp

We now compare the performance of hybrid QoS routing scheme *wdp* with a global QoS routing scheme *wsp*. The *wsp* is a well-studied [1,9,15] scheme that selects the widest shortest path for

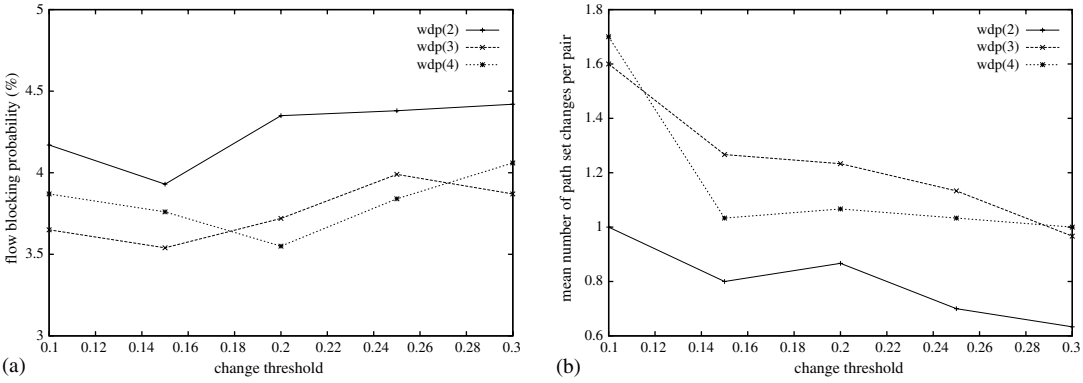

Fig. 10. Sensitivity of *wdp* to change threshold ψ. (a) Blocking probability and (b) number of path set changes.

each flow based on the global network view obtained through link state updates. The information carried in these updates is the residual bandwidth at the instant of the update. Note that *wdp* also employs link state updates but the information exchanged is average residual bandwidth over a period not its instantaneous value. We use *wsp* as a representative of global QoS routing schemes as it was shown to perform the best among similar schemes such as shortest widest path and shortest distance path. In the following, we first compare the performance of *wdp* with *wsp* in terms of flow blocking probability and then the routing overhead.

6.3.1. Blocking probability

Fig. 11(a) shows the blocking probability as a function of update interval τ used in *wsp*. The τ for *wdp* is fixed at 30 m. The offered load on the network ρ was set to 0.55. It is clear that the performance of *wsp* degrades drastically as the update interval increases. The *wdp* scheme, using at most two paths per pair and infrequent updates with $\tau = 30$ m, blocks fewer flows than *wsp*, that uses many more paths and frequent updates with $\tau = 0.5$ m. The performance of *wdp* even with a single path is comparable to *wsp* with $\tau = 1.5$ m. Fig. 11(b) displays the flow blocking probability as a function of offered network load ρ which is varied from 0.50 to 0.60. Once again, the τ for *wdp* is set to 30 m and the performance of *wsp* is plotted for three different settings of τ: 0.5, 1.0 and 2.0 m. It can be seen that across all loads the performance of *wdp* with $\eta = 2$ is better than *wsp* with $\tau = 0.5$. Similarly with just one path, *wdp* performs better than *wsp* with $\tau = 2.0$ and approaches the performance of $\tau = 1.0$ as the load increases. It is also worth noting that *wdp* with two paths rejects significantly fewer flows than with just one path, justifying the need for multipath routing.

The simulation environment used for the performance evaluation of *wdp* so far has six source nodes. To understand the ability of *wdp* in dealing with many sources, we have performed a set of simulations with more nodes generating traffic. Fig. 12(a) shows the results of a scenario when nine border nodes are acting as sources and destinations. The performance of *wdp* is shown with $\eta = 3$. The performance of *wsp* is plotted for two different settings of τ: 0.25, 0.50. Similarly, Fig. 12(b) shows the blocking probability of these schemes when all 18 nodes are source nodes. In both cases, the performance of *wdp* is better than *wsp* with update interval of 15 s. To further validate the relative performance of *wdp*, we have conducted simulations using bigger topologies. We have generated a 50 node topology and a 100 node topology using BRITE [16]. We then chose a set of node pairs that are at least certain hops (4 for 50-node topology and 5 for 100-node topology) away as the source–destination pairs. The number of such node pairs are 101 for 50-node topology and 135 for 100-node topology. The total load applied on the network is varied from 400 to 600. Fig. 13

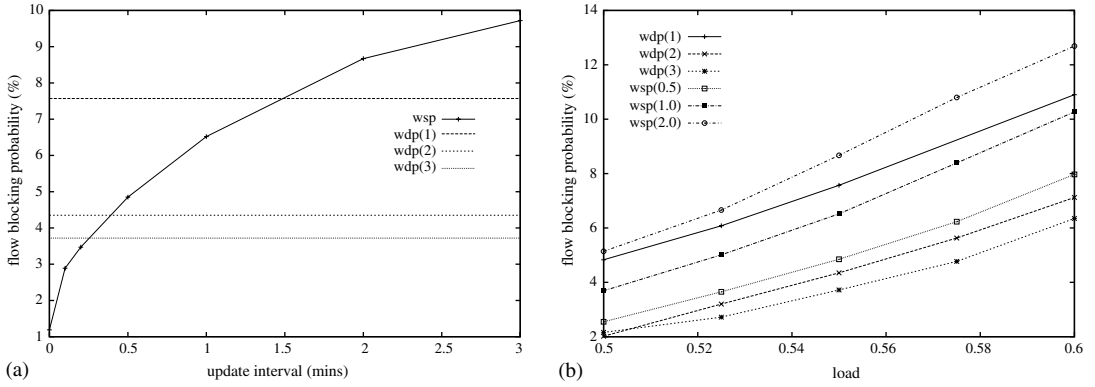

Fig. 11. Performance comparison of *wdp* and *wsp*. (a) Varying update interval and (b) varying load.

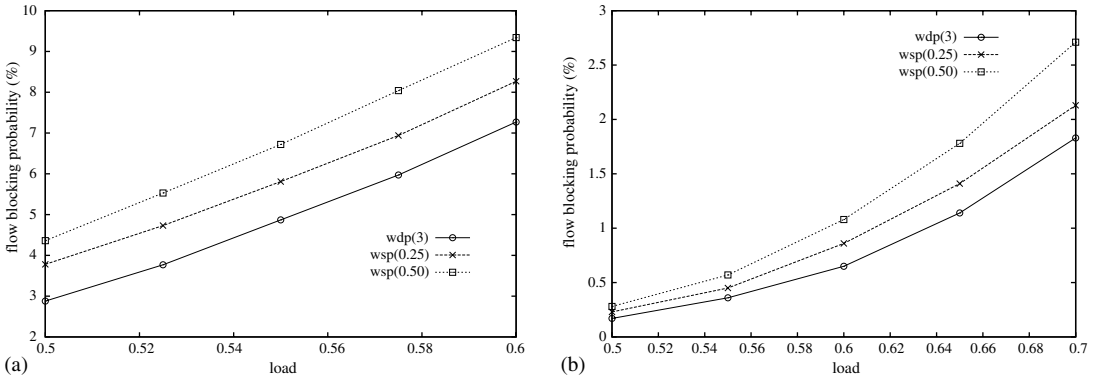

Fig. 12. Comparison using more traffic sources. (a) Border nodes as sources and (b) fall nodes as sources.

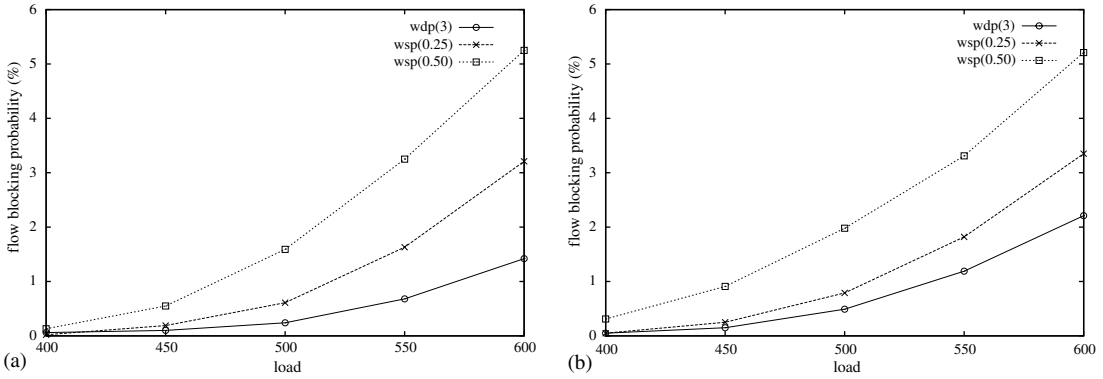

Fig. 13. Comparison using bigger network topologies: (a) 50 nodes and (b) 100 nodes.

compares the blocking performance of *wdp* and *wsp* under these settings. Once again, the blocking probability of *wdp* is smaller than *wsp* with update interval of 15 s. These results further confirm that irrespective of the traffic and topology settings, with very infrequent updates *wdp* outperforms *wsp* with frequent updates.

There are several factors contributing to the superior performance of *wdp*. First, it is the nature of information used to capture the link state. Under *wdp*, the information exchanged about a link is its *average* not *instantaneous* residual bandwidth and hence less variable. Second, before picking the widest disjoint paths, the residual bandwidth on all the links along the current candidate path are adjusted to account for the load offered on that path by this pair. Such a *local adjustment* to the global information makes the network state appear differently to each source. It is as if each source receives a customized update about the state of each link. The sources that are currently routing through a link perceive higher residual bandwidth on that link than other sources. This causes a source to continue using the same path to a destination unless it finds a much wider path. This in turn reduces the variation in link state and consequently the updated information does not get outdated too soon. In contrast, *wsp* exchanges highly varying instantaneous residual bandwidth information and all the sources have the same view of the network. This results in mass synchronization as every source prefers *good* links and avoids *bad* links. This in turn increases the variance in instantaneous residual bandwidth

values and causes route oscillation. [6] The *wdp* scheme, on the other hand, by selecting paths using both local and global information and by employing *ebp* based adaptive proportioning delivers stable and robust performance.

6.4. Heterogeneous traffic

The discussion so far is focused on the case where the traffic is homogeneous, i.e., all flows request for one unit of bandwidth and their holding times are derived from the same exponential distribution with a fixed mean value. Here we study the applicability of *wdp* in routing heterogeneous traffic where flows could request for varying bandwidths with their holding times derived from different distributions. We demonstrate that *wdp* is insensitive to the duration of individual flows and hence we do not need to differentiate flows based on their holding times. We also show that when the link capacities are considerably larger than the average bandwidth request of flows, it may not be necessary to treat them differently and hence *wdp* can be used *as is* to route heterogeneous traffic.

Consider the case of traffic with k types of flows, each flow of type i having a mean holding time $1/\mu_i$ and requesting bandwidth B_i. Let ρ_i be the offered load on the network due to flows of type i, where the total offered load, $\rho = \sum_{i=1}^{k} \rho_i$. The fraction of total traffic that is of type i, $\phi_i = \rho_i/\rho$. The arrival rate of type i flows at a source node, λ_i is given by $\lambda_i = \rho_i \mu_i LC/N\bar{h}B_i$, which is an extension of the formula presented in Section 6.1. To account for the heterogeneity of traffic, bandwidth blocking ratio [15] is used as the performance metric for comparing different routing schemes. The bandwidth blocking ratio is defined as the ratio of the bandwidth usage corresponding to blocked flows and the total bandwidth usage of all the offered traffic. Suppose b_i is the observed blocking probability for flows of type i, then the bandwidth blocking ratio is given by

$$\frac{\sum_{i=1}^{k} (b_i \lambda_i B_i/\mu_i)}{\sum_{i=1}^{k} (\lambda_i B_i/\mu_i)}.$$

In the following, we compare the performance of *wdp* and *wsp*, measured in terms of bandwidth blocking ratio, under different traffic conditions, varying the fractions ϕ_i to control the traffic mix.

Mixed holding times. We now examine the case of traffic with two types of flows that request for the same amount of bandwidth, i.e., $B_1 = B_2 = 1$, but with different holding times. We consider three scenarios. In the first scenario, both types of flows have their holding times derived from exponential distribution but their means are different: 60 and 120 s. In the second scenario, both types have the same mean holding time of 60 s but their distributions are different: exponential and pareto. In the third scenario, holding times of both types of flows follow pareto distribution but their means are different: 60 and 120 s. In all these scenarios, a load of 0.55 is offered between the nodes labeled with bigger font in *isp* topology given in Fig. 5. Fig. 14 shows the performance of *wdp* and *wsp* under these different scenarios.

Consider the first scenario where type 1 flows are *short* ($1/\mu_1 = 60$ s) and type 2 flows are *long* ($1/\mu_1 = 120$ s), but both are exponentially distributed. Fig. 14(a) shows the bandwidth blocking ratio plotted as a function of the fraction ϕ_1 corresponding to short flows. It is quite evident that the performance of *wsp* degrades as the proportion of *short* flows increases while that of *wdp* stays almost constant. The behavior of *wsp* is as expected since the shorter flows cause more fluctuation in the network QoS state and the information at a source node becomes more inaccurate as the QoS state update interval gets larger relative to flow dynamics. On the contrary, *wdp* is insensitive to the duration of flows.

In the second scenario, a fraction of flows have their holding times derived from a pareto distribution while the rest have their holding times derived from an exponential distribution. The mean holding time of both the types is the same, 60 s. The pareto distribution is long tailed with its tail

[6] Some remedial solutions were proposed in [1,2] to deal with the inaccuracy at a source node. However, the fundamental problem remains and the observations made in this paper still apply.

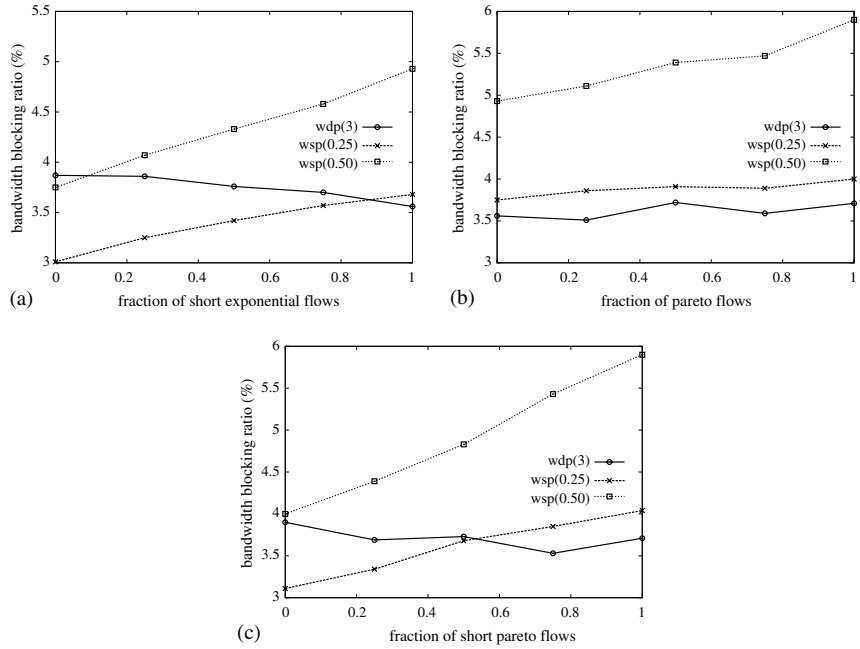

Fig. 14. Comparison under traffic with mixed holding times: (a) long and short exponential, (b) exponential and pareto, (c) long and short pareto.

controlled by a *shape* parameter. We have experimented with different shape values in the range 2.1–2.5 and found that results are similar. The results reported here correspond to a shape value of 2.2. In Fig. 14(b), bandwidth blocking ratio is plotted as a function of the fraction of pareto type flows. As the fraction of pareto flows increases, the blocking under *wsp*(30) increases while it stays almost same under *wsp*(15). The number of short (much less than mean holding time) flows are more under the pareto distribution than the exponential distribution because of the long tail of pareto. Consequently, update interval has to be small to capture the fluctuations due to such short flows. That is why the performance of *wsp*(30) degrades while *wsp*(15) is not affected. The relative performance of these schemes in the third scenario is similar to the first scenario with short and long flows. An important thing to note is that in all the scenarios the performance of *wdp* is insensitive to the holding times of flows.

Varying bandwidth requests. Now, consider the case of traffic with two types of flows, each requesting for *different amount of bandwidth* but having the same mean holding time. The bandwidth requests of flows are derived uniformly from a range: 0.5–1.5 for *small* flows and 1.5–2.5 for *large* flows, i.e., the mean bandwidth of small flows is 1 while it is 2 for large flows. The holding times of all the flows are drawn from an exponential distribution with mean 60 s. The performance is measured varying the mix of small and large flows. Fig. 15 shows the bandwidth blocking ratio as a function of the fraction of small flows. First thing to note is that *wdp* performs poorly when the majority of flows are large. However, as the number of small flows increases, it approaches the performance of *wsp*(15). The reason is that routing under *wdp* is independent of the amount of bandwidth requested while *wsp* is conscious of the bandwidth requested. However, when the link capacity is much larger than a flow's bandwidth request, *wdp* performs fine even though it is unconscious of the requested amount. To illustrate this, we doubled the capacity of all links and measured the performance of both the schemes

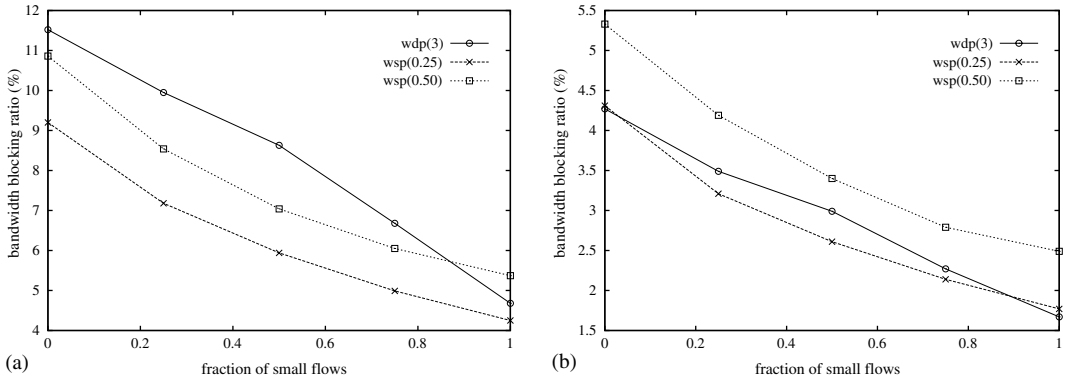

Fig. 15. Comparison under variable bandwidth traffic. (a) $C_1 = 20$, $C_2 = 30$ and (b) $C_1 = 40$, $C_2 = 60$.

under similar load conditions as the previous case. Fig. 15 shows that *wdp* performs as well as *wsp*(15). In general, we found that when bandwidth requests are significantly smaller than the link capacity, it is not necessary for *wdp* to differentiate between different bandwidth requests.

6.4.1. Routing overhead

Now we compare the amount of overhead incurred by *wdp* and *wsp*. This overhead can be categorized into per-flow routing overhead and operational overhead. We discuss these two separately in the following.

The *wsp* scheme selects a path by first pruning the links with insufficient available bandwidth and then performing a variant of Dijkstra's algorithm on the resulting graph to find the shortest path with maximum bottleneck bandwidth. This takes at least $O(E \log N)$ time where N is the number of nodes and E is the total number of links in the network. Assuming precomputation of a set of paths R_σ to each destination, to avoid searching the whole graph for path selection, it still need to traverse all the links of these precomputed paths to identify the widest shortest path. This amounts to an overhead of $O(L_\sigma)$, where L_σ is the total number of links in the set R_σ. On the other hand, in *wdp* one of the candidate paths is chosen in a weighted round robin fashion whose complexity is $O(\eta)$ which is much less than $O(L_\sigma)$ for *wsp*.

Now consider the operational overhead. Both schemes require link state updates to carry residual bandwidth information. However the frequency of updates needed for proper functioning of *wdp* is no more than what is used to carry connectivity information in traditional routing protocols such as OSPF. Therefore, the average residual bandwidth information required by *wdp* can be piggybacked along with the conventional link state updates. Hence, *wdp* does not cause any additional burden on the network. On the other hand, the *wsp* scheme requires frequent updates consuming both network bandwidth and processing power. Furthermore *wsp* uses too many paths. The *wdp* scheme uses only a few preset paths, thus avoiding per-flow path setup. Only admission control decision need to be made by routers along the path. The other overheads incurred only by *wdp* are periodic proportion computation and candidate path computation. The proportion computation procedure is extremely simple and costs no more than $O(\eta)$. Assuming all paths are feasible, the candidate path computation amounts to finding η widest paths and hence its worst case time complexity is $O(\eta N^2)$. However, this cost is incurred only once every ξ period. Considering both the blocking performance and the routing cost, we conclude that *wdp* yields much higher throughput with much lower overhead than *wsp*.

6.5. Performance of hwdp

In this section, we evaluate the performance of the hierarchical routing schemes. We first compare

the performance of *hwdp* scheme with flat *wdp* scheme and then with *wsp* and *hwsp* schemes.

6.5.1. Simulation environment

The simulation setup to evaluate hierarchical routing schemes is somewhat different from that used in the evaluation of flat routing schemes. In the following, we provide the additional information.

Figs. 4(a) and 16 show the topologies used in our study. The *hisp* topology is a slightly altered hierarchical version of the *isp* topology. The *rand* topology is similar to the one used in [9]. In both the topologies the *thin* links connect routers within an area and the *thick* links are the backbone links. All thin links are assumed have same capacity of 20 units and all thick links have 30 units. The nodes marked with a '·' are considered to be source or destination routers.

The default values for the configurable parameters of the schemes being simulated are as follows. The update interval τ in *wsp* and *hwsp* is set to 0.5 or 30 m and in the rest of our proportional routing schemes it is set to 30 m. The observation interval between recomputations of proportions θ in the flat *wdp* scheme is set to 40 m and candidate path selection is done every $\xi = 120$ m. Same settings are used for inter-area routing in case of *hwdp* and the corresponding values for intra-area routing are 20 and 60 m respectively. These values are chosen such that inter-area paths are changed more gradually than intra-area paths for the purpose of stability. The number of candidate paths allowed between a pair of nodes η is set to 3 for flat routing and intra-area routing.

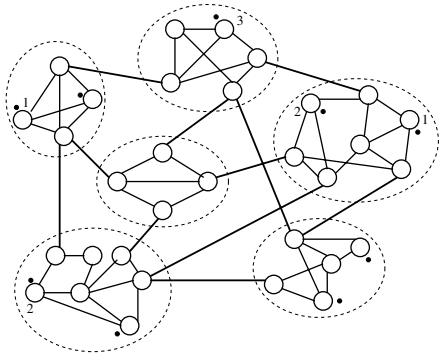

Fig. 16. The rand topology.

6.5.2. Convergence and adaptivity

Fig. 17 illustrates the convergence and adaptivity of proportional routing schemes. The results corresponding to *hisp* topology are shown in Fig. 17(a) and that of *rand* topology are shown in Fig. 17(b). The overall flow blocking probability is plotted as a function of time. We consider two traffic scenarios. In scenario I, all source–destination pairs are offered an equal amount of load and in scenario II, certain "hot pairs" exchange more traffic than other pairs. We have chosen two hot pairs marked by 1 and 2 in case of *hisp*. Similarly in case of *rand* three pairs are chosen as hot. In case of *hisp*, we offer a load of 200 in scenario I which is equally split between all source–destination pairs and in scenario II a load of 150 that is equally split among all pairs and a load 50 that is equally split among hot pairs only. The corresponding values for *rand* case are 300, 200 and 100 respectively.

The performance of *hwdp* is shown for two cases with the number of higher level candidate paths set to 1 and 2. We start the simulations with scenario I and switch to scenario II after sometime. There are several observations that can be made from these results. Starting with an arbitrary set of candidates and proportions, proportional routing schemes gradually converge to stable state. This gets disturbed when the traffic scenario changes and the blocking probability shoots up as the candidates and their proportions chosen for one traffic pattern are not perfect for a different traffic pattern. Consequently the proposed proportional routing schemes gradually adapt to the new traffic conditions and converge to stable state again.

With a single candidate path hierarchical *hwdp* scheme performs worse than flat *wdp* scheme. However, when the number of candidates is increased to two, there is almost no difference in blocking performance between these two schemes. These results indicate that there is significant gain in using multiple candidates for inter-area routing also. The gain is more pronounced in case of *hisp* than in *rand* and also in scenario II with hot pairs

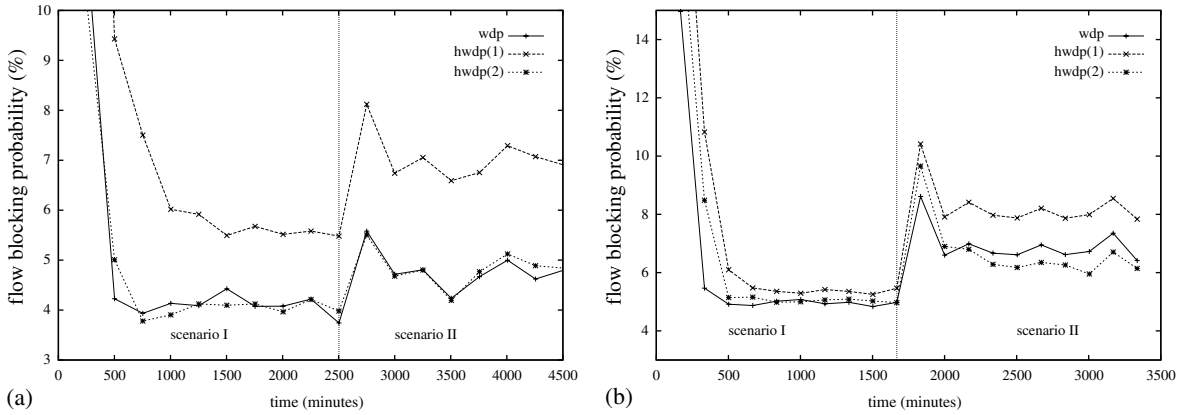

Fig. 17. Convergence and adaptivity of multipath routing schemes: (a) isp and (b) rand.

and thus non-uniform load than in scenario I with uniform load.

6.5.3. Blocking performance

Fig. 18(a) and (b) show the performance of these schemes under different load conditions for hisp and rand topologies respectively. The blocking probability is plotted as a function of offered load which is varied from 180 to 220 in case of hisp and 260 to 340 in case of rand. Note that the update interval τ in *wsp* and *hwsp* is set to 30 s while that in all our proportional routing schemes is set to 30 m. We also show the performance of flat *wsp* scheme with update interval of 0 s as a reference. This corresponds to the case of instantaneous updates, i.e., an update generated for every change in a link's available bandwidth, or to the case of a server that keeps track of the current state of the network and performs path selection and admission control in a centralized manner.

First, let us look at the impact of update interval and state aggregation on the performance of *wsp* scheme. There is a drastic increase in the blocking probability when the update interval is changed from an unrealistic setting of 0 s to a more realistic value of 30 s. This shows the impact of inaccuracy on best-path routing schemes and that they work well only with very frequent

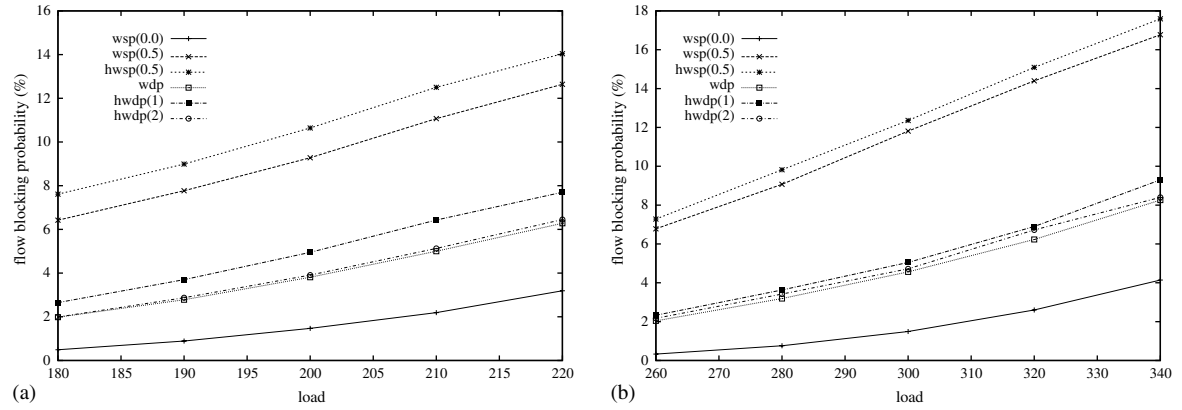

Fig. 18. Performance comparison with best-path routing schemes: (a) isp and (b) rand.

updates. Essentially best-path routing is suitable for centralized routing but not appropriate for distributed routing. The hierarchical version of *wsp*, *hwsp* fares worse than *wsp*. This is because the impact of inaccuracy introduced by the best-path based aggregation gets further amplified by the selection of the best higher level logical path based on inaccurate information.

Now, let us compare the proposed hierarchical proportional routing scheme *hwdp* with best-path routing schemes *wsp* and *hwsp*. It is expected that with always up-to-date information (update interval of 0 s), the flat *wsp* scheme would perform better than proportional routing schemes. However, it is surprising that with only aggregate information and update interval of 30 m, *hwdp* performs much better than even the flat *wsp* scheme with update interval of 30 s. It is worth noting that this is the case even when the number of candidates is limited to one. Furthermore, the gap in blocking performance between *hwdp* and *wsp* widens with the larger rand topology. These results demonstrate that proportional routing schemes with their minimal update overhead are more suitable than best path routing schemes for routing in large networks.

7. Conclusions

The performance of multipath routing hinges critically on the number and the quality of the selected paths. We addressed these issues in the context of the proportional routing paradigm, where the traffic is proportioned among a few good paths instead of routing it all along the best path. We proposed a hybrid approach that uses both global and local information for selecting a few good paths and for proportioning the traffic among the selected paths. We presented a *wdp* scheme that performs *ebp* based proportioning over widest disjoint paths. A set of widest paths that are disjoint *w.r.t.* bottleneck links are chosen based on globally exchanged link state metrics. The *ebp* strategy is used for adaptively proportioning traffic among these paths based on locally collected path state metrics. We compared the performance of our *wdp* scheme with that of optimal proportional routing scheme *opr* and shown that the proposed scheme achieves almost optimal performance using much fewer paths. We also demonstrated that the proposed scheme yields higher throughput with lower overhead compared to other link state update based schemes such as *wsp*.

We have also discussed how proportional routing can be extended to provide hierarchical routing across large networks that are divided into multiple areas. We proposed an aggregation method that summarizes the state of multiple paths between two routers in an area using a single metric. We presented a hierarchical multipath routing scheme *hierarchical widest disjoint paths* (hwdp) based on this aggregate metric. We evaluated its performance and shown that the proposed hierarchical proportional routing scheme performs as well as flat proportional routing scheme *wdp*. Furthermore, we demonstrated that this scheme with only aggregate information outperforms even the flat best-path routing scheme *wsp* having detailed information about the network. Based on our results, we conclude that *hwdp* scheme, with its low overhead and high throughput, is a suitable choice for hierarchical routing across large networks.

Acknowledgements

This work is partly supported by National Science Foundation Grants CAREER Award ANI-9734428, ANI-0073819, and ITR ANI-0085824. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] G. Apostolopoulos, R. Guerin, S. Kamat, S. Tripathi, Quality of service based routing: a performance perspective, ACM SIGCOMM, September 1998.

[2] G. Apostolopoulos, R. Guerin, S. Kamat, S. Tripathi, Improving QoS routing performance under inaccurate link state information, ITC'16, June 1999.

[3] ATM Forum, Private network–network interface, Specification Version 1 (PNNI 1.0), March 1996.
[4] S. Chen, K. Nahrstedt, An overview of quality-of-service routing for the next generation high-speed networks: Problems and solutions, IEEE Network Magazine, Special Issue on Transmission and Distribution of Digital Video 12 (6) (1998) 64–79.
[5] T. Coleman, Y. Li, An interior trust region approach for nonlinear minimization subject to bounds, SIAM Journal on Optimization 6 (1996) 418–445.
[6] E. Crawley, R. Nair, B. Rajagopalan, H. Sandick, A framework for QoS-based routing in the Internet, RFC 2386, August 1998.
[7] A. Elwalid, C. Jin, S. Low, I. Widjaja, MATE: MPLS adaptive traffic engineering, IEEE INFOCOM, 2001.
[8] R. Guerin, S. Kamat, A. Orda, T. Przygienda, D. Williams, QoS routing mechanisms and OSPF extensions, RFC 2676, August 1999.
[9] F. Hao, E. Zegura, On scalable QoS routing: Performance evaluation of topology aggregation, IEEE INFOCOM, 2000.
[10] F.P. Kelly, Routing in circuit-switched networks: optimization, shadow prices and decentralization, Advances in Applied Probability 20 (1988) 112–144.
[11] F.P. Kelly, Routing and capacity allocation in networks with trunk reservation, Mathematics of Operations Research 15 (1990) 771–793.
[12] F.P. Kelly, Fixed point models of loss networks, Journal of Australian Mathematical Society, Series B 31 (1989) 204–218.
[13] T. Korkmaz, M. Krunz, Source-oriented topology aggregation with multiple QoS parameters in hierarchical ATM networks, IWQOS, 1999.
[14] W.C. Lee, Topology aggregation for hierarchical routing in ATM networks, Computer Communications Review 25 (2) (1995) 82–92.
[15] Q. Ma, P. Steenkiste, On path selection for traffic with bandwidth guarantees, IEEE ICNP, October 1997.
[16] A. Medina, A. Lakhina, I. Matta, J. Byers, BRITE: An approach to universal topology generation, Proceedings of MASCOTS 2001, Cincinnati, August 2001.
[17] J. Moy, OSPF Version 2, Request For Comments 2328, Internet Engineering Task Force, April 1998.
[18] S. Nelakuditi, Z.-L. Zhang, On selection of paths for multipath routing, IWQoS'01, Karlsruhe, Germany, June 2001.
[19] S. Nelakuditi, S. Varadarajan, Z.L. Zhang, On localized control in quality-of-service routing, IEEE Transactions on Automatic Control, Special Issue on Systems and Control Methods for Communication Networks 47 (6) (2002) 1026–1038.
[20] S. Nelakuditi, Z.-L. Zhang, R.P. Tsang, D.H.C. Du, Adaptive proportional routing: a localized QoS routing approach, IEEE Transactions on Networking 10 (6) (2002) 790–804.
[21] http://www.cse.sc.edu/srihari/soft/qosr.tar.gz.
[22] Network Simulator NS2, http://www.isi.edu/nsnam/ns/index.html.
[23] M. Powell, A fast algorithm for nonlinear constrained optimization calculations, in: G.A. Watson (Ed.), Numerical Analysis, Lecture Notes in Mathematics, vol. 630, Springer, Berlin, 1978.
[24] E. Rosen, A. Viswanathan, R. Callon, Multi-protocol label switching architecture, RFC 3031, January 2001.
[25] K.W. Ross, Multiservice Loss Models for Broadband Telecommunication Networks, Springer, 1995.
[26] A. Shaikh, J. Rexford, K. Shin, Evaluating the overheads of source-directed quality-of-service routing, ICNP, October 1998.
[27] N. Taft-Plotkin, B. Bellur, R. Ogier, Quality-of-service routing using maximally disjoint paths, IWQOS, June 1999.
[28] C. Villamizar, MPLS optimized multipath (MPLS-OMP), Internet Draft draft-ietf-ospf-omp-02.txt, February 1999.
[29] C. Villamizar, OSPF optimized multipath (OSPF-OMP), Internet Draft draft- ietf-mpls-omp-01.txt, February 1999.
[30] Z. Wang, J. Crowcroft, Quality-of-service routing for supporting multimedia applications, IEEE Journal on Selected Areas in Communications 14 (7) (1996) 1228–1234.

Srihari Nelakuditi received his B.E. in Computer Science from Andhra University, Visakhapatnam, India, M.Tech. in Computer Science from Indian Institute of Technology, Chennai, India, and Ph.D. in Computer Science from University of Minnesota, Minneapolis. He is currently an Assistant Professor at University of South Carolina, Columbia. He was a research fellow and an adjunct faculty member in the Department of Computer Science at University of Minnesota, Minneapolis. He also worked as a software design engineer at Texas Instruments, Bangalore, India. His research interests include quality-of-service based routing, inter-domain routing and video streaming. He is a member of IEEE and ACM.

Zhi-Li Zhang received the B.S. degree in Computer Science from Nanjing University, China, in 1986 and his M.S. and Ph.D. degrees in Computer Science from the University of Massachusetts in 1992 and 1997. In 1997 he joined the Computer Science and Engineering faculty at the University of Minnesota, where he is currently an Associate Professor. From 1987 to 1990, he conducted research in Computer Science Department at Aarhus University, Denmark, under a fellowship from the Chinese National Committee for Education. His research interests are in computer networks and multimedia systems. He received the National Science Foundation CAREER Award in 1997. He has also awarded the prestigious McKnight Land-Grant Professorship at the University of Minnesota. He is a co-recipient of an ACM SIGMETRICS best paper award. He currently serves on the Editorial Boards of IEEE/ACM Transactions on Networking and Computer Networks. He is a member of IEEE, ACM and INFORMS Telecommunication Section.

David H.C. Du received the B.S. degree in Mathematics from National Tsing-Hua University, Taiwan, ROC in 1974, and the M.S. and Ph.D. degrees in Computer Science from the University of Washington, Seattle, in 1980 and 1981, respectively. He is currently a Professor at the Computer Science and Engineering Department, University of Minnesota, Minneapolis. His research interests include multimedia computing, storage systems, high-speed networking, high-performance computing over clusters of workstations, database design and CAD for VLSI circuits. He has authored and co-authored more than 150 technical papers, including 75 referred journal publications in his research areas. He is an IEEE Fellow and was an Editor of IEEE Transactions on Computers from 1993 to 1997. He is currently serving on the Editorial Boards of Journal of Cluster Computing, Journal of Parallel and Distributed Computing Practices and Journal of Information Sciences and Engineering. He served as a guest editor for a number of journals including IEEE Computer, IEEE and Communications of ACM. He has also served as Conference Chair and Program Committee Chair to several conferences in multimedia and database areas.