EUROPEAN TRANSACTIONS ON TELECOMMUNICATIONS
Eur. Trans. Telecomms. 2011; **22**:185–199
Published online 4 March 2011 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/ett.1468

COMMUNICATION NETWORK

TCP flow aware adaptive path switching in diffserv enabled MPLS networks

Onur Alparslan[1], Nail Akar[2] and Ezhan Karasan[2]*

[1] Department of Information Networking, Graduate School of Information Science and Technology, Osaka University, 1-5 Yamadaoka, Suita, Osaka 565-0871, Japan

[2] Electrical and Electronics Engineering Department, Bilkent University, 06800 Bilkent, Ankara, Turkey

ABSTRACT

We propose an adaptive flow-level multi-path routing-based traffic engineering solution for an IP backbone network carrying TCP/IP traffic. Incoming TCP flows are switched between two explicitly routed paths, namely the primary and secondary paths (PP and SP), for resilience and potential goodput improvement at the TCP layer. In the proposed architecture, PPs receive a preferential treatment over SPs using differentiated services mechanisms. The reason for this choice is not for service differentiation but for coping with the detrimental knock-on effect stemming from the use of longer SP that is well known for conventional network load balancing algorithms. Moreover, both paths are congestion-controlled using Explicit Congestion Notification marking at the core and Additive Increase Multiplicative Decrease rate adjustment at the ingress nodes. The delay difference between PP and SP is estimated using two per-egress rate-controlling buffers maintained at the ingress nodes for each path, and this delay difference is used to determine the path over which a new TCP flow will be routed. We perform extensive simulations using ns-2 in order to demonstrate the viability of the proposed distributed adaptive multi-path routing method in terms of per-flow TCP goodput. The proposed solution consistently outperforms the single-path routing policy and provides substantial per-flow goodput gains under poor PP conditions. Moreover, highest goodput improvements under the proposed scheme are achieved by flows that receive the lowest goodputs with single-path routing, while the performance of the flows with high goodputs with single-path routing does not deteriorate with the proposed path switching technique. Copyright © 2011 John Wiley & Sons, Ltd.

KEYWORDS

multi-path routing; path switching; traffic engineering; MPLS; differentiated services

***Correspondence**

Ezhan Karasan, Electrical and Electronics Engineering Department, Bilkent University, 06800 Bilkent, Ankara, Turkey.
E-mail: ezhan@ee.bilkent.edu.tr

Received 16 March 2009; Revised 2 September 2010; Accepted 28 November 2010

1. INTRODUCTION

Traffic engineering (TE) is defined as the set of mechanisms that control how traffic flows through a network so as to optimise resource utilisation and network performance [1]. TE mechanisms can be applied to hop-by-hop, explicit or multi-path routing networks. Traditional hop-by-hop routed IP networks using intra-domain routing protocols, such as Open Shortest Path First (OSPF) or Intermediate System to Intermediate System (IS-IS), resort to shortest path routing with simple link weights such as hop-count, delay or the inverse of the link bandwidth. Although the simplicity of this approach allows IP routing to scale to very large networks, it does not make the best use of network resources. If the traffic demand matrix is known a-priori, the link weights can be determined by solving an optimisation problem for TE purposes [2]. Although the problem of obtaining optimal weights is known to be NP-hard, a number of heuristics can be used to find near-optimal solutions [2,3]. Despite the relative ease of implementation of link weight-based methods, it is not only hard to reach optimality by using shortest path routing but also hard to estimate traffic demands that may comprise unpredictable spikes [4]. On the other hand, explicitly routed networks allow traffic to follow any desired route as opposed to one that is computed by hop-by-hop destination-based routing protocols. Multi-Protocol Label Switching (MPLS) is a technology which provides the necessary protocols and mechanisms for IP backbones to enable explicit routing to facilitate TE. There are a variety

of on-line and off-line TE proposals for explicitly routed MPLS networks; see Ref. [5] for an extensive treatment of MPLS TE methods and applications. Off-line methods solve mathematical programs for optimal routes using long term average traffic demands as inputs. However, the drawback of using offline methods is that they cannot react to real-time traffic changes or traffic spikes. On the other hand, online methods should be designed to detect real-time changes in the offered traffic and react to them, which is the scope of the current paper.

Multi-path routing is a general umbrella term referring to routing mechanisms that take advantage of path diversity between source–destination (s–d) pairs. Dispersity routing, for example subdivides a certain message and disperses it through a multiplicity of paths using redundancy so as to cope with paths with long delays or high loss rates [6]. Path switching, on the other hand, refers to the ability of an end system to dynamically switch among multiple paths (one path at a time) for a given message or flow, to a destination depending on the current status of these paths made available by path probing mechanisms [7]. In MATE [8] proposed for MPLS networks, the ingress label switch router (LSR) transmits probe packets periodically to the egress LSR which then returns the probe packets back to the ingress LSR, carrying information on the label switched path (LSP) characteristics. Based on the gathered information, the MATE algorithm tries to equalise the congestion measure, for example delays, for each LSP, by appropriately splitting traffic across multiple LSPs. Using a similar technique [9], the authors focus on elastic traffic and make adaptive path switching decisions at the ingress device based on round trip time (RTT) measurements. Some of the existing path switching proposals rely on path diversity provided by multihoming or overlay scenarios. One such scenario from the existing literature is a VoIP service provider multihomed to several ISPs which dynamically determines the path to forward voice packets [10]. The work of [11] uses similar principles but for an overlay networking scenario in conjunction with redundant dispersity routing to improve application-level performance under time-varying congestion on Internet paths and failures. Although most path switching proposals rely on path probing for which no network involvement is required, a number of methods exist that are based on network assistance. For example, TeXCP proposed in Ref. [12] splits the multi-path routing problem into two components; namely the load balancer and the rate controller. The load balancer component takes the state of the network as the input and makes path switching decisions to minimise utilisation. On the other hand, each path is congestion controlled by network involvement with the control loop operating at a faster time scale compared to the load balancer.

Another line of recent research focuses on finding stability conditions for rate control in conjunction with dynamic multi-path routing. For example, in Ref. [13], sufficient conditions for the local stability of end-to-end algorithms for joint routing and rate control are given for a network with arbitrary interconnection of sources and resources, and heterogeneous propagation delays. The Ref. [14] provides theoretical justification of certain decentralised algorithms for joint optimisation of congestion control and multi-path routing. However, experimental studies of moderately sized TCP/IP networks are not presented in these two papers. In the current paper, as opposed to a theoretical justification, we take the approach of experimenting with a heuristical dynamic multi-path routing algorithm in moderately sized TCP/IP networks and for a wide range of scenarios. In particular, we address the problem of path switching in an explicitly routed IP network provider scenario (e.g. MPLS network) in which the ingress nodes of the network decide on which explicitly routed path to forward the incoming IP packets. Flow-based TE for UDP traffic using a similar approach as in this paper was studied earlier [15]. In this paper, we concentrate on elastic TCP traffic based on the observation that TCP is the dominant transport protocol used in the Internet today. However, we also note that this situation might quickly change in the near future due to the current explosion of video streaming traffic. The case of networks with TCP/UDP traffic mix is left for future research. The case of two paths, one being the min-hop primary path (PP) and the second one being the alternative secondary path (SP), is studied here, but the architecture is amenable for extension to two or more SP. Experimental studies reveal that packet de-sequencing within a TCP flow can significantly deteriorate end-system TCP performance [16] and therefore we focus on flow-level path switching mechanisms as in Refs. [9,17] that identify the TCP flows at the ingress of the network and dynamically determine the most convenient path on a per flow-basis with the aim of avoiding packet de-sequencing within a flow. Although flow-level granularity for traffic splitting suffices for shorter flows (flows of 'mice' type), it may not give as satisfactory results for longer flows (flows of 'elephants' type). For this reason, the concept of flowlet-level traffic splitting as opposed to packet- or flow-level splitting is introduced in Ref. [18], where a flowlet is defined as a burst of packets appropriately chosen to avoid reordering. Our work is similar in spirit to Ref. [12] which uses feedback based rate control for each path and packet-level non-elastic UDP traffic splitting whereas we study elastic TCP flow-level splitting in the current paper. Improved splitting granularity, such as flowlet-level switching, is left outside the scope of the paper.

It is well known that using alternative longer paths by some traffic sources force other sources whose min-hop paths share links with these alternative paths to also use alternative paths [19]. This fact is called the knock-on effect in the literature and is studied in depth for alternately routed circuit switched networks [20]. Precautions should be taken to mitigate the knock-on effect, such as the well-known 'trunk reservation' concept in circuit switched networks [20]. One of the key ingredients of our proposed architecture for IP-MPLS packet networks is the use of modified deficit round robin (MDRR) per-class queuing

and scheduling that favours packets routed along PPs over those routed along SPs to cope with the knock-on effect. In this way, we are able to use min-hop paths as long as they are not congested and start moving some of the traffic to alternative paths when congestion arises along PPs without jeopardising the already existing flows on min-hop paths. In this paper, we also compare MDRR scheduling with the widely deployed First In First Out (FIFO) queuing in terms of their capabilities to deal with the knock-on effect in the context of TCP flow-level TE. Prioritised routing or trunk reservation concepts have also appeared recently in Refs. [17,21] in the context of Internet and optical burst switching networks, respectively.

From a positioning standpoint, we note two distinctive features of our model compared with the multihoming or overlay path switching scenarios that are already studied in the literature:

- In the current network provider scenario, the path switching entity belongs to the network provider and therefore the network can actively participate in making path switching decisions.
- While choosing the paths to route flows, the network addresses the question of 'what is good for the network' rather than 'what is good for the application' since all the path switching entities in the network would collectively operate for optimising the resource utilisation and network performance as a whole. In this sense, we use path switching for TE purposes within a network provider domain.

This paper finds its roots in our preliminary studies [22,23], but differs from those in the following:

- We propose MDRR scheduling at the core nodes as opposed to strict priority queuing (SPQ) to mitigate the knock-on effect, which leads to significant performance improvement.
- Using ns-2, we simulate a 12-node network at the TCP level for validation purposes and obtain entirely new results that quantify the gain in TCP-level goodput using the proposed approach in comparison with the shortest path only routing which is most common in the current Internet. We note that TCP flow-level simulation experiments in the context of multi-path routing are rare due to the difficulty of maintaining states for a very large number of TCP flows, which we were able to achieve in this study.
- An important shortcoming of the solutions proposed in Refs. [22,23] is that although the average per-flow goodput increases with respect to the shortest path routing, there are a number of flows routed over SPs whose performance degrades due to starvation of service. The proposed MDRR scheduling technique addresses this starvation problem. We show that substantial performance improvements are obtained in terms of per-flow TCP goodput using MDRR scheduling, especially for those flows that perform poorly under shortest path routing. Meanwhile, the performance of flows that are already performing well with the shortest path routing is not adversely affected.

The remainder of the paper is organised as follows. In Section 2, we present our adaptive flow-level multi-path routing-based TE architecture. Experimental results using ns-2 simulations are presented in Section 3. Conclusions and future work are given in the final section.

2. PATH SWITCHING ARCHITECTURE

In the proposed path switching architecture, we envision an IP backbone network which consists of edge and core nodes (i.e. routers) which has mechanisms for establishing explicitly routed paths such as MPLS. In this network, edge (ingress or egress) nodes are gateways that originate/terminate explicitly routed paths and core nodes carry transit traffic only. Edge nodes are in charge of per-egress and per-class based queuing, flow identification, rate control and path switching as will be described in the sequel. Core nodes support per-class queuing and Explicit Congestion Notification (ECN) marking. In this architecture, flow awareness requirement is restricted to edge nodes only, making the overall architecture scalable.

The proposed architecture is based on the following building blocks: (i) path establishment, (ii) queuing in network nodes, (iii) feedback mechanism and rate control and (iv) traffic splitting, that are described next.

2.1. Path establishment

We assume in this study that edge nodes are single-homed, that is they have a link to a single core node. We set up one PP and one SP from each ingress node to every other egress node. We impose that the two paths are link-disjoint within the scope of the core network. The paths are computed using a two-step approach. First, the PP is established as the min-hop path. If there are multiple min-hop paths, the one with the minimum propagation delay is chosen as the PP. In order to find the route for the SP, we prune the links used by the PP and compute the min-hop path in the remaining network graph. A tie in this step is broken similarly. If the connectivity is lost after the first step, we do not establish an SP. We prefer to use this simple path computation scheme which does not perform any load balancing since we do not assume a-priori knowledge of the traffic demand matrix. We also note that the min-hop path may not always be the min-delay path but we are mostly concerned about the utilisation of network links from the network operator standpoint and we therefore use the min-hop paths as the PP to minimise network resource use.

2.2. Queuing in network links

As far as queuing is concerned, all the links in the network employ a differentiated services (diffserv) mechanism,

namely per-class queuing, with three drop-tail queues, namely gold, silver and bronze queues. For scheduling, we propose to use the MDRR algorithm inspired by Ref. [24] which is also described in Ref. [25]. In our implementation, the gold queue has strict priority over the silver and bronze queues. The gold queue is used for resource management (RM) and TCP ACK packets (the role of RM packets will be explained in the sequel). We propose that ACK packets are identified by the ingress node and the encapsulation header for ACK packets are marked accordingly. In this paper, we assume that TCP ACK packets are not piggybacked on TCP data packets on the reverse path. If so, one could use the gold queue only for RM packets, the case of which is not studied in the current paper. Silver and bronze queues are used for TCP data packets only.

It is well known that using alternative longer paths by some sources may force other sources whose min-hop paths share links with these alternative paths to also use alternative paths [19]. This cascading effect is called the *knock-on effect* in the literature and is studied in depth for alternate routing algorithms in circuit switched networks. Preventive measures are taken to mitigate the knock-on effect, for example the *trunk reservation* mechanism used in circuit switched networks. We address this problem in IP networks by using per-class queuing mechanisms. In this paper, we compare and contrast two such methods. The first method is FIFO queuing in which all TCP data packets join the silver queue irrespective of the type of path (primary or secondary) they use. However, this queuing policy triggers the knock-on effect due to lack of preferential treatment to packets using fewer resources (i.e. traversing fewer hops). In order to mitigate the knock-on effect, longer SP should be resorted to only if PP can no longer accommodate additional traffic and traffic flowing over longer SP should not adversely affect the traffic flowing on the PP. We therefore propose per-class queuing for all the links in the network such that TCP data packets routed over PPs use the silver queue and those using SPs join the bronze queue.

In our MDRR implementation, the gold queue has the highest priority and all packets residing at the gold queue are served at each visit of the gold queue of the MDRR scheduler. Once all packets waiting at the gold queue are dequeued, then the MDRR scheduler first visits the silver queue and decides on whether to dequeue any number of packets. Once this decision is made and corresponding packets are transmitted from the silver queue, the gold queue is visited back to dequeue all waiting packets if any. The MDRR scheduler then visits the bronze queue and decides on whether to transmit any packets from this queue or not. The gold queue is visited back to dequeue all waiting packets. This pattern then repeats itself. In this setting, the gold queue is said to have non-preemptive strict priority over the silver and bronze queues. In MDRR, the decision to dequeue a number of packets from the silver and bronze queues is made as follows. In our MDRR implementation, the silver queue is associated with a quantum ϕ_s and a deficit counter denoted by δ_s. When the MDRR scheduler visits the silver queue, the deficit counter δ_s is first increased by ϕ_s. Then, a number of packets at the head of the silver queue whose cumulative packet lengths do not exceed δ_s are transmitted from the silver queue. Then, δ_s is reduced by the sum of the sizes of all packets that are transmitted at this visit. It is clear that multiple packets can be served from the silver queue during the same visit as long as the total length of the served packets do not exceed δ_s. When the silver queue is empty at a certain visit, its deficit counter δ_s is set to 0. The MDRR operation for the bronze queue is similar except that the bronze queue is associated with a different quantum ϕ_b and a deficit counter denoted by δ_b. In our MDRR implementation, a weight is assigned to each of the silver and bronze queues reflecting the relative bandwidth allocated to that queue. The quantum of a queue is then chosen to be proportional with the assigned weight of the queue. By setting the weight of the silver queue sufficiently larger than that of the bronze queue, IP packets using the silver queue, that is PP packets, are given preferential treatment, thus mitigating the knock-on effect. In this paper, the silver and bronze queues are assigned weights of 90% and 10%, respectively. In order to ensure at least a packet to be transmitted from the bronze queue at each visit of a non-empty bronze queue, the quantum ϕ_b of the bronze queue is dynamically set to the length of the packet size of the head-of-the-line packet at the bronze queue. Otherwise, ϕ_b does not change. In the successive visit of the silver queue, ϕ_s is set to 9 ϕ_b to be compliant with the assigned weights.

2.3. Feedback mechanism and rate control

Another building block of the proposed architecture is the feedback mechanism and rate control. For intelligent path switching, the current status of the two paths needs to be available at the path switching entity. This information can be provided using path probing mechanisms without network assistance by measuring delay, loss and other attributes of the paths. On the contrary, we assume network assistance in this paper and in our proposed architecture, ingress nodes periodically send RM packets to egress nodes, one over the PP (P-RM) and the other over the SP (S-RM). These RM packets are sent towards the network every T_{RM} seconds with the direction bit set to indicate the direction of flow. If MDRR is used and when a P-RM (S-RM) packet arrives at the core node on its forward path, the node compares the percentage queue occupancy of its silver (bronze) queue on the outgoing interface with a predetermined configuration parameter μ and it sets the congestion experienced (CE) bit (if not already set) of the P-RM (S-RM) packet accordingly. If FIFO queuing is used then it is the silver queue occupancy that needs to be checked for both P-RM and S-RM packets. When the RM packet arrives back at the ingress node, the CE bit indicates the congestion status of the path it was sent over. According to the information, the ingress node updates the allowed transmission rate (ATR) of the corresponding

Eur. Trans. Telecomms. 2011; **22**:185–199 © 2011 John Wiley & Sons, Ltd.
DOI: 10.1002/ett

rate-controlled path according to the Additive Increase Multiplicative Decrease (AIMD) algorithm, which is described in Ref. [23]. The AIMD algorithm [23] allows ATR to change in the interval [MTR,PTR] where MTR and PTR denote the minimum transmission rate and peak transmission rate, respectively. The other AIMD algorithm parameters are rate decrease factor (RDF) and rate increase factor (RIF).

The proposed architecture is depicted in Figure 1 for a simple 3-node core network in which solid lines correspond to PPs whereas the dotted lines denote SPs originating at ingress node 0. For the sake of simplicity, the internal architecture of only ingress node 0 is given which maintains two queues for each egress node for rate control purposes at the per-egress queuing stage, one for the PP and the other for the SP. IP packets leaving the per-egress queuing stage join their respective queues in the per-class queuing stage for the edge-core link. Also note that the per-class queuing stage is not limited to edge-core links but to all links in this network including core–core links.

2.4. Path switching

The final ingredient to the proposed approach is the way path switching is performed for an incoming TCP flow. The edge nodes first identify new flows. The delay estimates for the PP and SP queues (denoted by D_{PP} and D_{SP}, respectively) at the edge nodes are then calculated by dividing the occupancy of the corresponding per-egress queue by the current drain rate (i.e. ATR). Upon arrival of the first packet of the nth flow (i.e. a TCP SYN segment), a running estimate of the delay difference (denoted by Δ_n) is calculated as:

$$\Delta_n = \beta(D_{PP} - D_{SP}) + (1-\beta)\Delta_{n-1} \quad (1)$$

where β is a smoothing parameter. If $\Delta_n \leq \min_{th}$ ($\Delta_n \geq \max_{th}$), then the flow is switched towards the PP (SP). When $\max_{th} > \Delta_n > \min_{th}$, the new flow is switched towards the SP with probability p_0 and towards the PP with probability $1 - p_0$, where p_0 is given by:

$$p_0 = \frac{\Delta_n - \min_{th}}{\max_{th} - \min_{th}} \quad (2)$$

where $\min_{th}$, $\max_{th}$ and p_0 are the algorithm parameters. This path switching mechanism is called Random Early Reroute (RER) which is inspired by the Random Early Detection (RED) algorithm used for active queue management in the Internet [26]. RER favours shorter PP over longer SP in order to mitigate the negative results of the knock-on effect and reduce the delay experienced by TCP flows. Furthermore, since RER makes path switching decisions on a per-flow basis, all packets of a TCP flow follow the same path.

3. SIMULATION STUDY

We obtain the simulation results using the network-assisted path switching architecture by simulating (with ns-2) IP/MPLS networks with enhanced ECN capability and

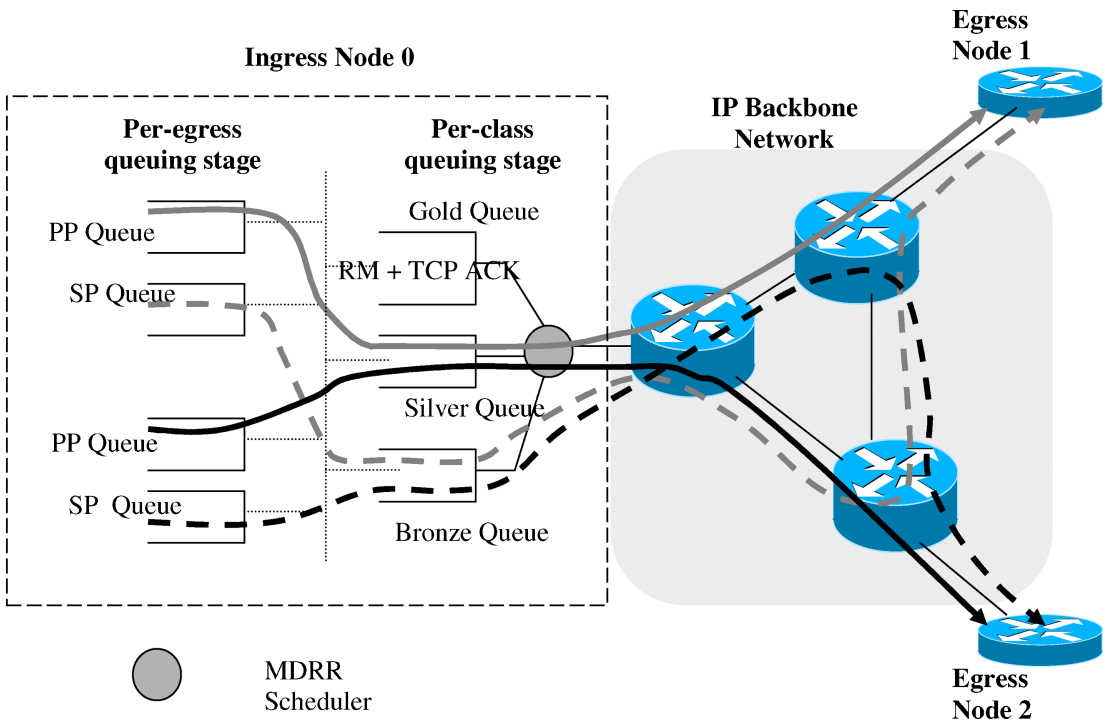

Figure 1. The proposed architecture for path switching.

modified MPLS header that accommodates a field to indicate the packet type which is necessary for proper operation.

3.1. Topology and traffic matrix

We study two topologies in this paper, a small 3-node triangle core network depicted in Figure 1 and a medium sized 12-node network available in the public domain [27], which is depicted in Figure 2. For the 3-node network, core nodes are connected by links each with a capacity of 50 Mbps, and each link has a propagation delay of 10 ms. There is an edge node connected via a 1 Gbps link to each core node therefore the core links are the potential bottleneck links in the 3-node network. The average traffic sourced out of each edge node is fixed at 70 Mbps. The traffic demand $T_{i,i+1}$ from node i to node $i+1 (\mathrm{mod}\, 3), i = 0, 1, 2,$ is uniformly distributed in the interval (0,70) Mbps and that of $T_{i,i-1}$ is set to 70 Mbps$-T_{i,i+1}$. The distribution of traffic sourced out of each node is independent of others. Fifty random traffic demand matrices are generated as inputs to the simulation study based on the above statistical characterisation. The goodput results are obtained from the collection of 50 independent runs each with its own randomly drawn traffic matrix. On the other hand, the 12-node network we study is known as the hypothetic US topology which is available in Ref. [27] together with the link capacities and the traffic demand matrix. This topology has 19 links and has been used for several optimal multi-path studies. In the original network, all network links have a capacity of 155 Mbps except for the two bidirectional links de $\leftrightarrow$ ch and ch $\leftrightarrow$ cl which have a capacity of 310 Mbps in both directions. In order to keep the memory requirements of the simulator to a reasonable level, all the link capacities and traffic demands are scaled by a factor of 1/3 resulting in a scaled down version of the original network but operating at the same load.

3.2. Traffic model

Given a traffic demand in Mbps between a s–d pair, we need to have a TCP traffic model that generates traffic according to the traffic demand. Motivated by recent studies on Internet traffic modeling, in our simulations we assume that flow arrivals occur according to a Poisson process with rate λ, and flow sizes obey a Bounded Pareto (BP) distribution. The BP distribution is preferred in this study rather than the ordinary Pareto distribution since the latter distribution has infinite variance, and thus excessively long simulations would be required for convergence. Moreover, the BP distribution still exhibits the large variance and heavy tail properties of the flow size distribution of Internet traffic and allows us to set a bound on the largest flow size. The BP distribution is denoted by BP(k, p, α), where k and p denote the minimum and maximum flow sizes, respectively, and the shape parameter α is the exponent of the power law.

As α gets smaller the tail gets heavier, and the likelihood of having large flows increases. The probability density function for the BP(k, p, α) is given by:

$$f(x) = \frac{\alpha k^{\alpha}}{1-(k/p)^{\alpha}} x^{-\alpha-1}, k \leq x \leq p, 0 \leq \alpha \leq 2 \quad (3)$$

In this study, we use $k = 4$ Kbytes, $p = 50$ Mbytes and $\alpha = 1.20$, corresponding to an average flow size of 20.4 Kbytes.

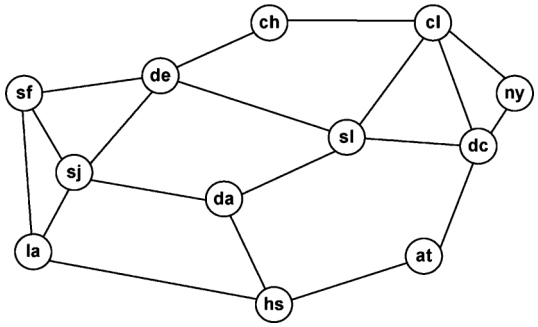

Figure 2. Hypothetic US topology as given in Ref. [27].

3.3. Routing policies

In order to validate the effectiveness of the path switching idea introduced in this paper, we compare and contrast several routing policies using simulations. As a reference policy, we use rate-controlled single-path routing in which the traffic follows the minimum-hop route with the ECN and AIMD capabilities turned on but with no path switching. In short, we call this method 'single-path routing'. It would be desirable to have a path switching mechanism whose performance surpasses that of single-path routing in all possible scenarios. Viability of such a mechanism is the scope of the current simulation study.

When path switching is used, two further policies are used to handle the implications of the knock-on effect: (i) RER used for path switching at the edge node and (ii) queuing and scheduling used in the network links.

In this paper, we study two sets of RER parameters:

- $\min_{\mathrm{th}} = \max_{\mathrm{th}} = 0$ ms and $p_0 = 1$, called the shortest delay (SD) policy since the path with the shorter smoothed average buffering delay is chosen when SD policy is employed. The SD policy switches each new TCP flow simply to the path with the shorter estimated buffering delay at the ingress edge node, and therefore the SD policy does not favour the PP in path switching.
- $\min_{\mathrm{th}} = 1$ ms, $\max_{\mathrm{th}} = 15$ ms and $p_0 = 1$, called the RER policy; this choice of RER parameters fits well with the philosophy of RER discussed in Section 2 that favours the min-hop path over the longer SP.

For RER, we fix the delay averaging parameter as $\beta = 0.3$. For the queuing and scheduling policies in

the network links, we consider the following three alternatives:

- *FIFO queuing* in which all TCP data packets (PP and SP) join a single queue, say the silver queue, and RM and TCP ACK packets use the gold queue.
- *SPQ* in which we have the gold, silver, and bronze queues used by RM/TCP ACK packets, PP TCP data packets and SP TCP data packets, respectively, drained according to an SPQ policy with the gold (bronze) queue having the highest (lowest) priority.
- *MDRR scheduling* as explained in detail in Section 2.2 with 90% weight for the silver queue and 10% weight for the bronze queue.

In this paper, we consider the following three combined policies comprising different queuing/scheduling and RER mechanisms, namely FIFO/SD, SPQ/RER and MDRR/RER.

3.4. IP/MPLS and simulation parameters

Data packets are assumed to be 1040 bytes long including the MPLS header. We assume that the RM packets are 50 bytes long. All the buffers at the edge and core nodes including per-egress and per-class queues have a size of 104,000 bytes each. The TCP receive buffer is of length 20,000 bytes. This proposed path switching architecture is implemented over ns-2 (Network Simulator) version 2.27 [28] and the TCP-Reno is used in the simulations. The simulation runtime is selected as 300 s. In all the simulation results, events concerning flow arrivals occurring only in the period (90 , 250 s) are reported in order to avoid transient effects.

3.5. AIMD algorithm parameters

The AIMD-based rate control algorithm that we use in the current article is already described in Ref. [23]. We now describe how we set the AIMD algorithm parameters in our simulation-based study. The frequency of probe packets is chosen such that $T_{\mathrm{RM}} = 0.1$ s for the 3-node network, and $T_{\mathrm{RM}} = 0.02$ s for the 12-node network. The RDF parameter of the AIMD-based rate control algorithm is set to as 0.0625 in all simulations. We use different set of values for the RIF parameter of the AIMD algorithm as described below. For the 3-node network, RIF = 0.125 for PPs and SPs in FIFO/SD and PPs in single-path routing and MDRR/RER. The MDRR weight ratio of the silver and bronze queues in MDRR/RER is 9:1, so in case of congestion due to PP flows, the maximum drain rate of bronze queues is limited to only 10% of the link bandwidth. Gradual traffic increase by the AIMD algorithm of SPs must be in smaller steps in order to prevent buffer overflows in bronze queues in case their drain rate is limited to 10% of the link speed. Therefore, the SPs in MDRR/RER are proposed to have RIF = 0.0138. For the 12-node network, RIF = 0.0625 for PPs and SPs in FIFO/SD and PPs in single-path routing and MDRR/RER. SPs in MDRR/RER have RIF = 0.00694 due to the MDRR weight ratio as explained above.

One of the problems of ECN-based congestion detection algorithms is the so-called beat-down problem [29]. When a path between a s–d pair goes through several congested switches, CE bit of this s–d pair's RM packets is marked more often than other s–d pairs having paths going through fewer number of congested switches, causing unfairness among s–d pairs. It is possible to improve fairness for PPs in single-path routing and for PPs and SPs in MDRR/RER by properly setting the MTR parameter of the AIMD algorithm. MTR can be calculated separately for PPs and SPs in MDRR/RER. Each ingress node calculates PP MTR (SP MTR) value for each link along PP (SP) as $\mathrm{MTR} = C \times \mathrm{BW} \times W/N$, where BW is the bandwidth of the link, W is the weight of the bronze (silver) queue, N is the number of PPs (SPs) using the link and $C < 1$ is a constant. For MDRR/RER, $W = 0.9$ for the silver queue and $W = 0.1$ for the bronze queue. For single-path routing, $W = 1$ for the silver queue. Each s–d pair compares the PP MTR (SP MTR) values of links along the PP (SP) and chooses the minimum value as its MTR for PP (SP). For PPs and SPs when FIFO/SD is used, MTR = 1 bps is chosen in order to eliminate division by 0. The PTR parameter of the AIMD algorithm is chosen as the speed of the slowest link on the corresponding path. The constant $C = 0.9$ is chosen in the simulations. The percentage queue occupancy threshold μ for setting the CE bit is set to $\mu = 50\%$ for the 3-node network and $\mu = 20\%$ for the 12-node network.

3.6. Performance metrics

The goodput of a TCP flow i, G_i, is defined as the service rate received by flow i during its lifetime or equivalently it is the ratio Δ_i/T_i, where Δ_i is the number of bits successfully delivered to the application layer by the TCP receiver and T_i is the sojourn time of the flow i within the simulation runtime. We note that if flow i terminates within the simulation runtime then Δ_i is equal to the flow size S_i. Note that some flows may not be entirely carried or may not even be admitted (i.e. SYN packets dropped) during the simulation duration due to overloading of certain links in the network. We therefore introduce a performance measure, called the net average goodput over all TCP flows, denoted by $G_{\mathrm{net}} = \sum_i \Delta_i G_i / \sum_i S_i$, where the service rate for uncarried packets is set to 0. G_{net} can be computed for an individual s–d pair or for the network as a whole.

3.7. Simulation results

We first study the effect of the RER parameters $\mathrm{min}_{\mathrm{th}}$ and $\mathrm{max}_{\mathrm{th}}$ on TCP goodput when MDRR/RER is employed. The network-wide average goodput is shown in Figure 3 as

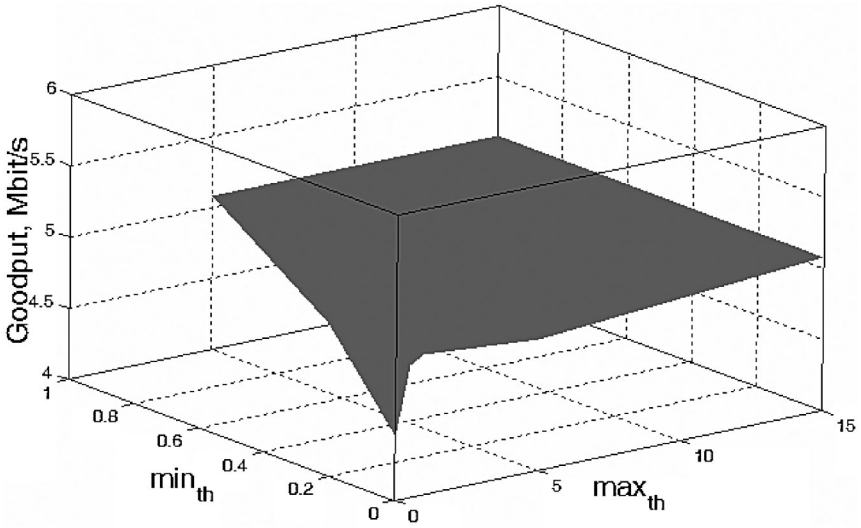

Figure 3. The network-wide goodput as a function of $\min_{th}$ and $\max_{th}$.

a function of RER parameters $\min_{th}$ and $\max_{th}$. Based on these results, in our simulation studies, we propose to use $\min_{th} = 1$ ms and $\max_{th} = 15$ ms. It can be observed that the goodput performance possesses a smooth behaviour around the selected values of the parameters. However, when these parameters are chosen close to 0 (mimicking the SD policy), the goodput drops considerably as an outcome of the knock-on effect and this particular regime needs to be avoided by the operator. More work is needed to tune the RER parameters but our preliminary results show that the performance of the system is quite robust with respect to our choice of the specific RER parameters.

Next, we compare SPQ/RER and MDRR/RER. The goodputs received by the PPs and the SPs for MDRR/RER and SPQ/RER for the 3-node and 12-node topologies are depicted in Figure 4. Note that the obtained goodputs are over the 50 independent runs and for all pairs simulated for the 3-node network whereas we have a single run for the 12-node network. It is observed that in SPQ, some flows receive lower goodput when forwarded over the SP compared to PP. If SP is passing through a congested link, SPQ causes ATR of the SP to get close to 0. In this case, RER algorithm may give incorrect routing decisions and route more flows to SPs resulting in flow starvation. On the other hand, MDRR/RER assigns a non-zero weight to the bronze queue eliminating the possibility of starvation of SPs. As seen in the figures, with MDRR/RER, the goodput ratio of the PP and the SP becomes more stable especially

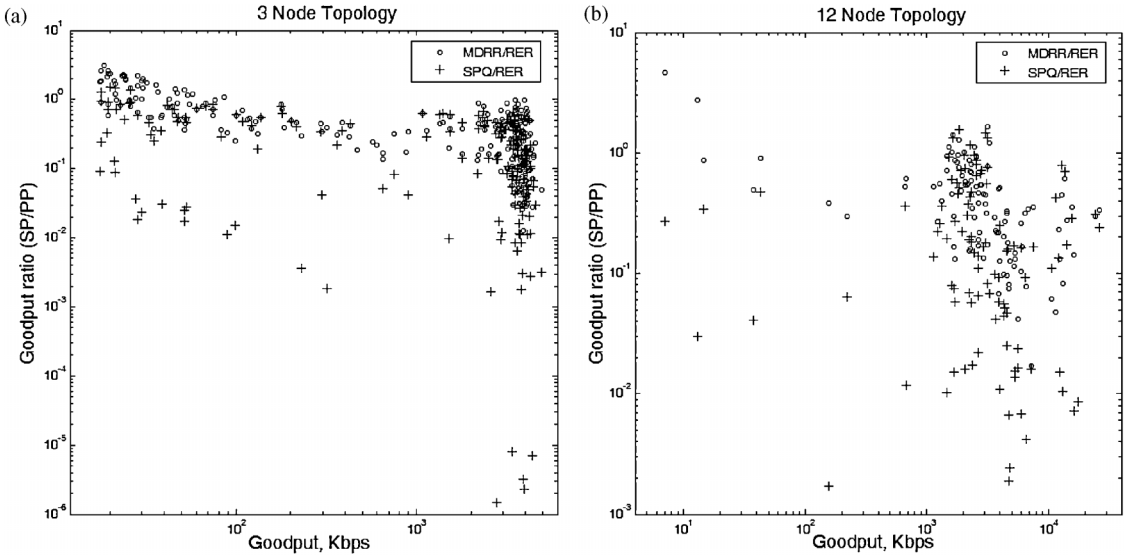

Figure 4. The ratio of goodputs received by PP and SP for MDRR/RER and SPQ/RER.

for slow s–d pairs. The path goodput ratio becomes smaller for high goodput s–d pairs, however this is less important since the number of flows forwarded to SP is small for high goodput s–d pairs due to RER. For the rest of the paper, we use MDRR/RER in order to mitigate the flow starvation problem instead of SPQ/RER which was used in our earlier work [22].

The impact of the architectural constructs RER and MDRR proposed in this study is studied in terms of their ability to mitigate the knock-on effect. MDRR/RER policy, where the MDRR method is used in network links for queuing and scheduling and RER is used at the edges for intelligent path switching, is compared with single-path routing and FIFO/SD schemes. We present the cumulative histogram diagram in Figure 5 showing the fraction of s–d pairs receiving an average goodput G_{net} less than a certain value in Kbps with the x-axis using logarithmic scale. The cumulative histograms are given for both the 3-node (using 50 randomly generated traffic matrices) and 12-node networks (where the histogram is obtained using the goodput distribution over node pairs using the given traffic demand matrix). First consider the 3-node case. With respect to single-path routing, FIFO/SD treats favourably the low goodput node pairs but while doing this high goodput pairs are seriously hampered, resulting in reduction of overall average goodput. We observe that when the goodputs increase, the performances of the primary and SP of FIFO/SD simultaneously deteriorate which can be explained by the knock-on effect stemming from FIFO queuing and SD-type path switching. We show that it is feasible to favour low goodput pairs without having to jeopardise high goodput pairs by using the MDRR/RER method which takes the best of the two worlds, namely that of FIFO/SD and single-path routing. We note that the MDRR method consistently outperforms the single-path routing where the performance improvement is magnified for the cases where the shortest path is congested and TCP flows using congested shortest paths in single-path routing receive low goodputs. The fact that these results are obtained for a set of 50 randomly generated traffic matrices is indicative of the robustness and resilience of the proposed path switching architecture. Similar conclusions can be drawn for the 12-node case for which we ran a single instance of a traffic matrix that was obtained out of a publicly available test case and a handful of node pairs took advantage of path switching for this particular scenario.

We present in Figure 6 the gain in per node-pair goodputs via the use of MDRR/RER and FIFO/SD with respect to the reference single-path routing. The x and y coordinates of each marked point in this figure correspond to the goodput obtained with single-path routing for a certain node pair and the goodput gain in using MDRR/RER or FIFO/SD for the same pair, respectively. Consider first the 3-node case and MDRR/RER. All the marked points are located in a *right triangle* whose bottom side coincides with the unity (or no) gain line, which shows that MDRR/RER consistently outperforms single-path routing. There are a number of marked points along this horizontal side implying that path switching does not introduce any gain for those pairs but does not worsen the results either. On the other hand, goodput gains up to two orders of magnitude are observed for some node pairs. Most importantly, the maximum gain is achieved for node pairs for which the single-path routing produced the lowest per-pair goodput (along the vertical side of the right triangle). There are also a number of marked points along the hypotenuse of the right triangle, which shows that gains using MDRR/RER drop as goodputs achieved by single-path routing increase. Other marked points lie inside the right triangle revealing gains with varying intensity. On the contrary, the FIFO/SD approach yields marked points inside a *parallelogram* rather than a triangle implying goodput losses with respect to single-path routing for many node pairs, especially for those with high goodputs. This parallelogram type behaviour of FIFO/SD is a consequence of the knock-on effect. Similar conclusions can be drawn for the 12-node network.

In order to demonstrate the effect of the total amount of the traffic demand, the traffic matrix is scaled down for the 12-node network according to a traffic scaling parameter $\gamma \leq 1$. We observe from Figure 7 that the network-wide G_{net} of MDRR/RER and single-path routing closely follow each other as γ changes, but the performance of FIFO/SD is far lower. We do not observe a crucial difference in network-wide goodput between MDRR/RER and single-path routing since MDRR/RER improves mainly the performance of some low goodput s–d pairs that are limited in number. When $\gamma = 1$, single-path routing even gives a slightly better network-wide G_{net} than MDRR/RER. Revisiting Figure 6b, we observe that the goodputs of low goodput s–d pairs are improved up to 15 times while the goodputs of some of high goodput node pairs may reduce by 0–15% due to the choice of MDRR weights that slightly favour longer paths. The fact that high goodput node pairs dominate in the network-wide average goodput calculations explains the slight outperformance of single-path routing over MDRR/RER.

The ratios of total data transmitted over PP and SP for each flow using MDRR/RER and FIFO/SD are shown in Figure 8 as a function of the goodput received by the flow. These figures show that MDRR/RER controls the ratio of amount of data sent to PP and SP according to the performance of paths. It uses SP only if PP is congested and SP is better. A substantial portion of the data is transmitted through PP if the goodput achieved through PP is high. On the other hand, SD/FIFO routes flows to the SP regardless of the PP goodput resulting in unnecessary forwarding of flows to the SP. Since SPs are typically longer and use more resources, aggressive use of SPs induces the knock-on effect.

The average goodputs achieved by MDRR/RER, SPQ/RER, FIFO/SD and the single-path schemes as a function of the flow size are depicted in Figure 9 for the 3-node topology. In these plots, TCP flow sizes are grouped into five bins as shown in the horizontal axis. In order

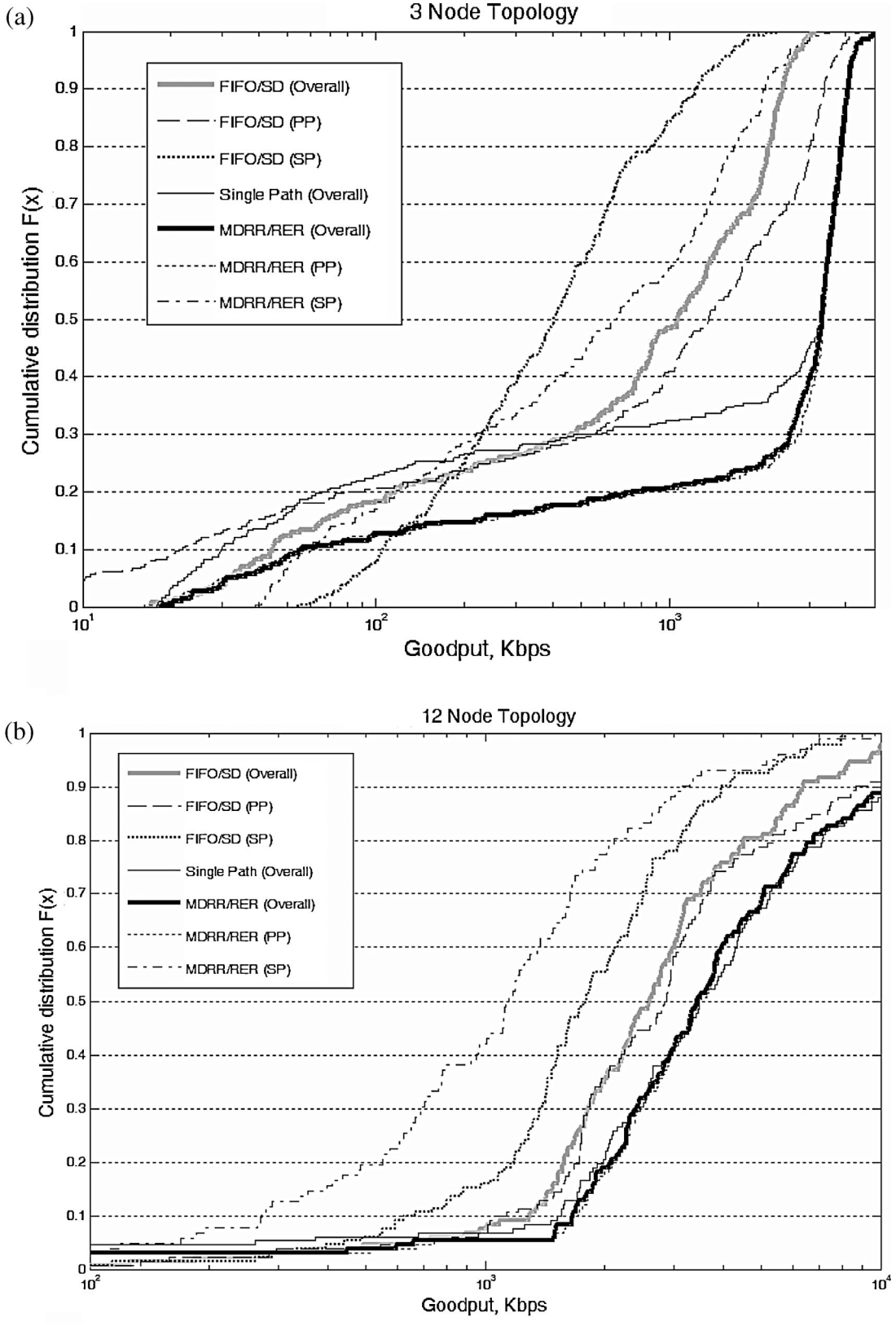

Figure 5. Cumulative histogram of the goodputs seen by source–destination pairs, primary path (PP) only, secondary path (SP) only and overall for (a) the 3-node topology (b) the 12-node topology.

Eur. Trans. Telecomms. 2011; **22**:185–199 © 2011 John Wiley & Sons, Ltd.
DOI: 10.1002/ett

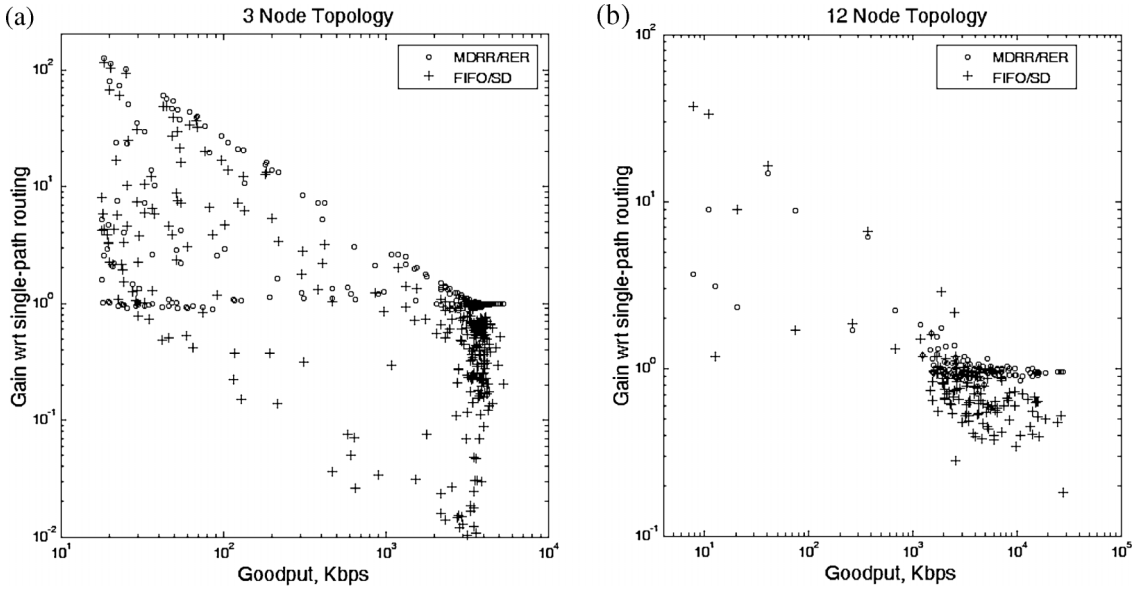

Figure 6. The gain in per-node pair goodputs via the use of MDRR/RER and FIFO/SD algorithms with respect to single-path routing.

to study the effects of the traffic distribution, three different cases are considered: (a) $T_{i,i+1} = T_{i,i-1} = 35$ Mbps, (b) $T_{i,i+1} = 50$ Mbps, $T_{i,i-1} = 20$ Mbps and (c) $T_{i,i+1} = 70$ Mbps, $T_{i,i-1} = 0$, where $i = 0, 1, 2$ and the addition and subtraction operations are modulo 3. When the traffic sourced at a node is evenly split between the two possible destinations, that is case (a), MDRR/RER, SPQ/RER and single-path schemes achieve nearly the same goodputs, whereas the goodput obtained by FIFO/SD is significantly lower due to the knock-on effect as shown in Figure 9a. As the traffic becomes more unevenly split between the two possible destinations (cases (b) and (c)), the performance of the shortest path scheme deteriorates since the PP becomes congested, as can be observed from Figure 9b and c. On the other hand, the interaction between traffic routed over PP and SP decreases when the traffic is more unevenly distributed, and the performance of FIFO/SD improves. In the extreme case when $T_{i,i+1} = 70$ Mbps, $T_{i,i-1} = 0$, that is case (c), the packets routed over PP and SP do not share any link bandwidth and hence there is no knock-on effect. In this case, FIFO/SD achieves nearly the same goodput as MDRR/RER and SPQ/RER. We also observe from

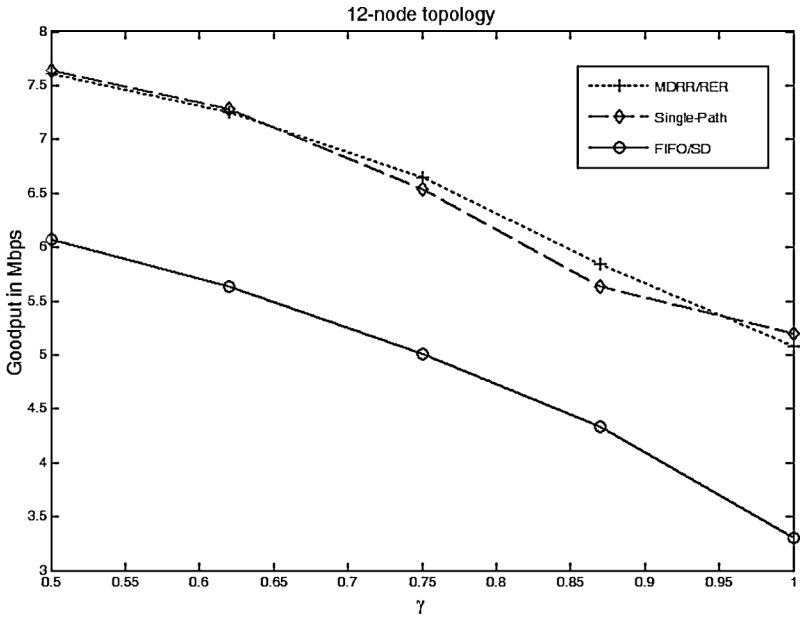

Figure 7. The network-wide goodput as a function of the scaling parameter γ for the 12-node topology.

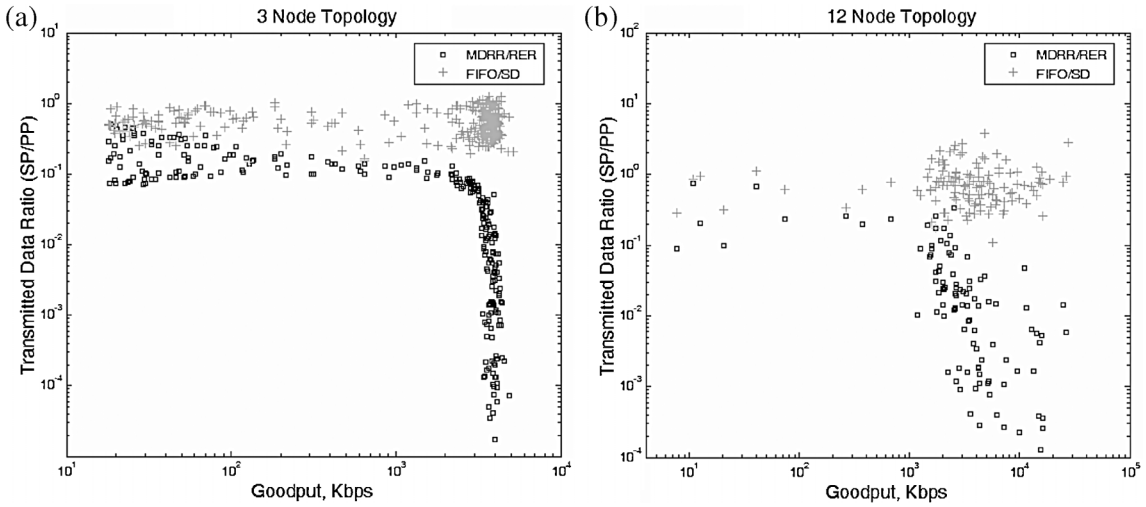

Figure 8. The ratio of total data transmitted over the primary path PP and the secondary path SP.

Figure 9 that the average goodputs for MDRR/RER and SPQ/RER increase as the flow size increases. This is mainly due to the fact that larger flows have the advantage of achieving larger TCP congestion windows whereas shorter flows cannot reach such large congestion windows due to the slow-start mechanism.

Finally, the adaptability of the proposed MDRR/RER scheme with respect to the changing network conditions is studied. To this end, we started running the simulation using the 3-node topology as described in Section 3.1, then at $t = 100$ s we reduced the bandwidth of the link between nodes 0 and 1 from 50 to 30 Mbps, and finally at $t = 200$ s, we increased the bandwidth of link (0,1) from 30 Mbps back to 50 Mbps. In this study, the traffic between node pairs is given by $T_{i,i+1} = 50$ Mbps, $T_{i,i-1} = 20$ Mbps, where $i = 0, 1, 2$, during the whole simulation period.

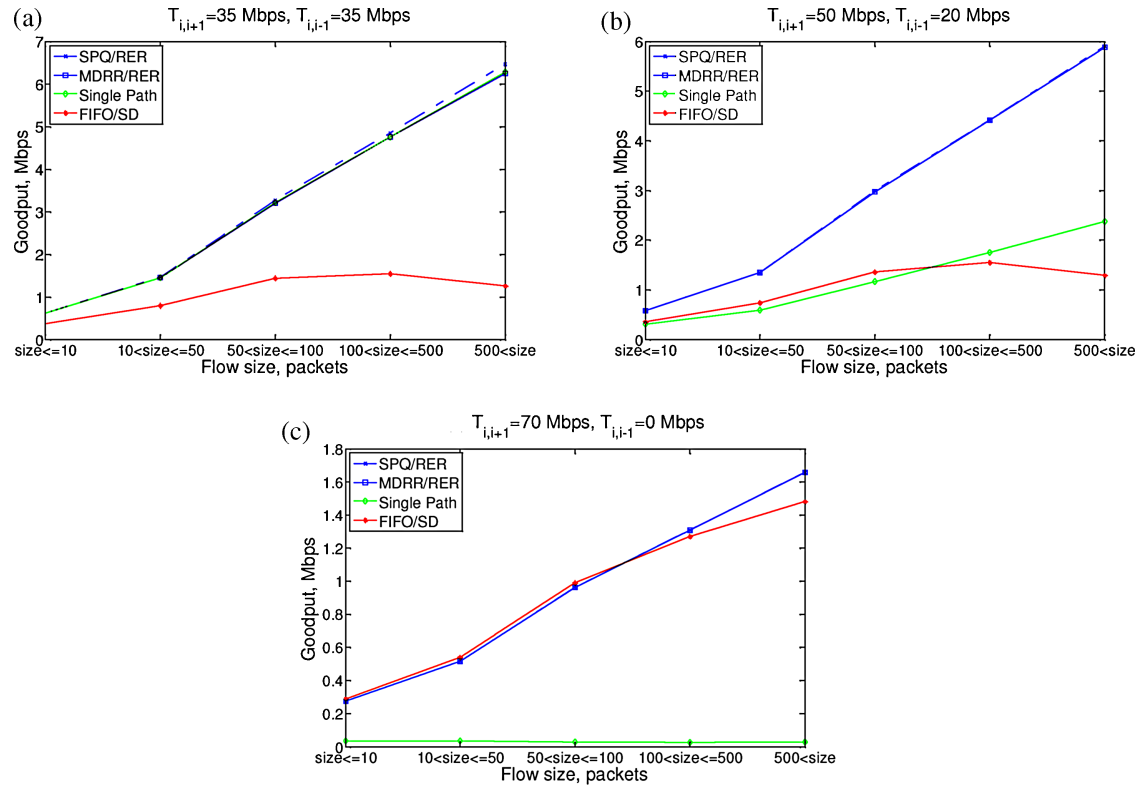

Figure 9. The network-wide goodput as a function of the flow sizes for the 3-node topology.

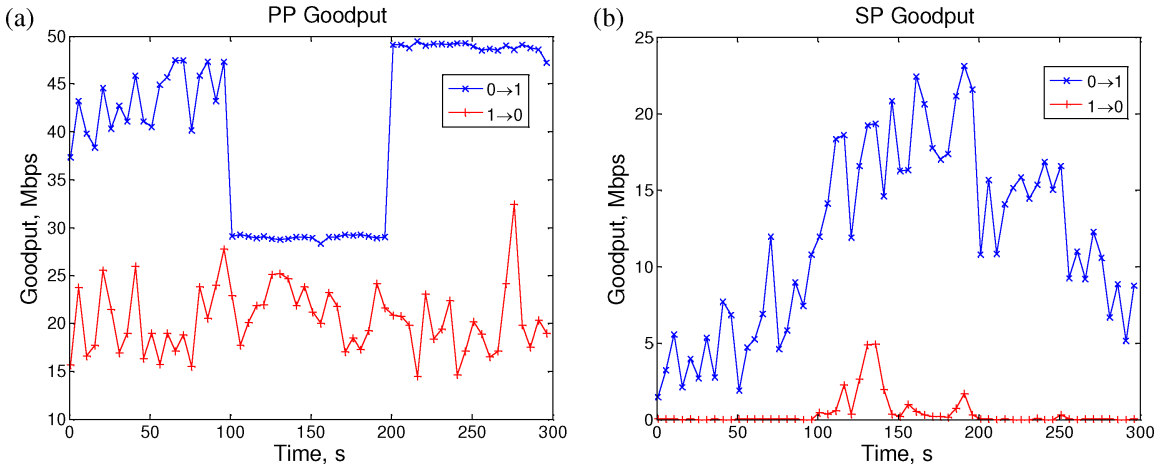

Figure 10. Time evolution of total goodputs over PP and SP as the bandwidth of (0,1) link changes.

The evolution of the total goodputs achieved by all TCP flows between node pairs $0 \rightarrow 1$ and $1 \rightarrow 0$ over primary and SP are plotted in Figure 10. Since $T_{0,1} = 50$ Mbps, the total PP goodput is saturated at 30 Mbps in the period (100 , 200 s). Consequently, the total $0 \rightarrow 1$ goodput over the SP increases in this period, which is plotted in Figure 10b, as more $0 \rightarrow 1$ flows are routed over the SP. On the other hand, the total $1 \rightarrow 0$ goodput is only marginally affected from this bandwidth change (a slight increase in the total $1 \rightarrow 0$ goodput over SP due to increased congestion over the PP) since $T_{1,0} = 20$ Mbps, which is still less than the reduced link bandwidth in the period (100 , 200 s). We also observe from Figure 10 that the sum of the PP and SP goodputs for $0 \rightarrow 1$ flows is higher in the period (200 , 300 s) compared to the period (0 , 100 s). This is due to the fact that some TCP flows are backlogged during the congested period (10 , 200 s), and they are still active in the period (200 , 300 s).

4. CONCLUSIONS

A flow-based distributed path switching mechanism is proposed for TE of TCP flows in networks with explicit routing capability such as MPLS. In this architecture, TCP flows are assigned to the primary and SP using a random early rerouting mechanism which uses the queuing delay difference between the two alternative paths in order to make routing decisions. An AIMD-based rate control using ECN marking is employed such that the traffic splitting decisions are made at the flow-aware edges of the MPLS network. A MDRR queuing and scheduling mechanism is proposed not for QoS delivery but for favouring shortest paths and thus improving routing performance. Extensive simulations are performed using a TCP traffic model with Poisson flow arrivals and BP distributed flow sizes from which it is shown that it is possible to substantially improve the performance of low-speed TCP flows with network-assisted path switching without having to jeopardise the performance of already high-speed TCP flows. Consistent outperformance against single-path routing and substantial per-flow goodput gains under poor PP conditions are the main outcomes of this study. We believe that prioritised routing, flow-aware networking, and switching decisions that favour min-hop paths are the minimal requirements for success for an adaptive multi-path routing technique to be useful. Simulation of larger topologies with a more diversified set of traffic demand matrices, use of more than two paths, and flowlet switching especially for long flows, are topics considered for future work. Another future work is to use probing based bandwidth estimation techniques (rather than network assistance that uses RMs and ECN capability in the core) to reduce implementation complexity.

ACKNOWLEDGEMENTS

This work is supported in part by the Scientific and Technological Research Council of Turkey (TÜBİTAK) under projects EEEAG-104E047, EEEAG-105E065 and EEEAG-106E046.

REFERENCES

1. Awduche DO, Chiu A, Elwalid A, Widjaja I, Xiao X. *Overview and principles of Internet traffic engineering*. IETF Informational RFC-3272, May 2002.
2. Fortz B, Thorup M. *Internet traffic engineering by optimizing OSPF weights*. In *Proceedings of INFOCOM*, 519–528, Tel-Aviv, Israel, March 2000.
3. Salles RM, Rolla VG. Efficient routing heuristics for Internet traffic engineering. *Computer Communications* 2007; **30**(9): 1942–1952.
4. Wang H, Xie H, Qiu L, Yang YR, Zhang Y, Greenberg A. *COPE: traffic engineering in dynamic networks*. In *Proceedings of the ACM SIGCOMM*, 99–110, Pisa, Italy, September 2006.

5. Osborne E, Simha A. Traffic Engineering with MPLS. Cisco Press: Indianapolis, USA, 2002.
6. Maxemchuk NF. Dispersity routing in high-speed networks. *Computer Networks and ISDN Systems* 1993; **25**(6): 645–661.
7. Tao S, Xu K, Xu Y, et al. *Exploring the performance benefits of end-to-end path switching*. In *Proceedings of IEEE ICNP*, 304–315, Berlin, Germany, October 2004.
8. Kodialam M, Lakshman TV. *Minimum interference routing with applications to MPLS traffic engineering*. In *Proceedings of INFOCOM*, Vol. 2, 884–893, Tel-Aviv, Israel, March 2000.
9. Zhao Z, Shu Y, Zhang L, Yang O. Flow-level multipath load balancing in MPLS network. *IEICE Transactions on Communications* 2005; **E88-B**(5): 2015–2022.
10. Tao S, Xu K, Estepa A, et al. *Improving VoIP quality through path switching*. In *Proceedings of INFOCOM*, 2268–2278, Miami, March, 2005.
11. Andersen DG, Snoeren AC, Balakrishnan H. *Best-path vs. multi-path overlay routing*. In *Proceedings of ACM/SIGCOMM Internet Measurement Conference*, 91–100, Miami, FL, October 2003.
12. Kandula S, Katabi D, Davie B, Charny A. *Walking the tightrope: responsive yet stable traffic engineering*. In *Proceedings of ACM SIGCOMM*, 253–264, Philadelphia, PA, August 2005.
13. Kelly F, Voice T. Stability of end-to-end algorithms for joint routing and rate control. *ACM Computer Communication Review* 2005; **35**(2): 5–12.
14. Paganini F. *Congestion control with adaptive multipath routing based on optimization*. In *Proceedings of the Conference on Information Sciences and Systems*, 333–338, Princeton, NJ, March 2006.
15. Akar N, Hokelek I, Karasan E. Available bit rate traffic engineering in MPLS networks with flow-based multipath routing. *IEICE Transactions on Communications* 2004; **E87-B**(10): 2913–2921.
16. Laor M, Gendel L. The effect of packet reordering in a backbone link on application throughput. *IEEE Network Magazine* 2002; **16**(5): 28–36.
17. Oueslati S, Roberts J. *Comparing flow-aware and flow-oblivious adaptive routing*. In *Proceedings of 40th Annual Conference on Information Sciences and Systems*, 655–660, March 2006.
18. Kandula S, Katabi D, Sinha S, Berger A. Dynamic load balancing without packet reordering. *ACM Computer Communication Review* 2007; **37**(2): 51–62.
19. Nelakuditi S, Zhang Z-L, Tsang RP, Du DHC. Adaptive proportional routing: a localized QoS routing approach. *IEEE/ACM Transactions on Networking* 2002; **10**(6): 790–804.
20. Kelly FP. Routing and capacity allocation in networks with trunk reservation. *Mathematics of Operations Research* 1990; **15**: 771–793.
21. Zalesky A, Vu HL, Rosberg Z, Wong EWM, Zukerman M. Stabilizing deflection routing in optical burst switched networks. *IEEE Journal on Selected Areas in Communications* 2007; **25**(6): 3–19.
22. Alparslan O, Akar N, Karasan E. *Combined use of prioritized AIMD and flow-based traffic splitting for robust TCP load balancing*. In *Proceedings of Fifth International Workshop on Quality of Future Internet Services (QofIS'04)*, 124–133, Barcelona, Spain, July 2004.
23. Alparslan O, Akar N, Karasan E. AIMD-based online MPLS traffic engineering for TCP flows via distributed multi-path routing. *Annals of Telecommunications* 2004; **59**(11-12): 1353–1371.
24. ''Understanding and Configuring MDRR/WRED on the Cisco 12000 Series Internet Router'', Cisco technical note, Doc. ID 18841, http://www.cisco.com/warp/public/63/mdrr_wred_overview.html.
25. Lenzini L, Mingozzi E, Stea G. *Bandwidth and latency analysis of modified deficit round robin scheduling algorithms*. In *Proceedings of the 1st International Conference on Performance Evaluation Methodologies and Tools (Valuetools 2006)*, Pisa, Italy, October 2006.
26. Floyd S, Jacobson V. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking* 1993; **1**(4): 397–413.
27. ''Optimized Multipath,'' available at www.faster-light.net/omp/ 1999, topology: http://www.faster-light.net/omp/usa-omp.html traffic demand matrix: http://www.faster-light.net/omp/usa-traffic.html.
28. The Network Simulator – ns-2, developed by L. Berkeley Network Laboratory and University of California Berkeley, http://www.isi.edu/nsnam/ns.
29. Ohsaki H, Murata M, Suzuki H, Ikeda C, Miyahara H. Rate-based congestion control for ATM networks. *ACM SIGCOMM Computer Communication Review* 1995; **25**(2): 60–72.

Authors' Biographies:

Onur Alparslan received B.S. and M.S. degrees from Bilkent University, Ankara, Turkey, in 2002 and 2005, in electrical and electronics engineering. He received Ph.D. degree from Osaka University, Japan, in information science and technology, in 2008. He was the recipient of Monbukagakusho Graduate Scholarship from the Ministry of Education, Science, Sports and Culture of Japan during 2005–2008. He is currently a post-doctoral researcher at Osaka University.

Nail Akar received his B.S. degree from Middle East Technical University, Turkey, in 1987 and M.S. and Ph.D. degrees from Bilkent University, Turkey, in 1989 and 1994, respectively, all in electrical and electronics engineering. From 1994 to 1996, he was a visiting scholar and a visiting assistant professor in the Computer Science Telecommunications program at the University of Missouri-Kansas City. In 1996, he joined

the Technology Planning and Integration group at the Long Distance Division, Sprint, where he held a senior member of technical staff position from 1999 to 2000. Since 2000, he has been a faculty member at Bilkent University, currently as an associate professor. His current research interests include performance analysis of computer and communication networks, queuing systems, traffic engineering, network control and resource allocation and optical networking. Dr. Akar actively participates in European Commission FP7 NoE projects BONE and WiMAGIC.

Ezhan Karasan received B.S. degree from Middle East Technical University, Ankara, Turkey, M.S. degree from Bilkent University, Ankara, Turkey and Ph.D. degree from Rutgers University, Piscataway, New Jersey, USA, all in electrical engineering, in 1987, 1990 and 1995, respectively. During 1995–1996, he was a postdoctorate researcher at Bell Labs, Holmdel, New Jersey, USA. From 1996 to 1998, he was a Senior Technical Staff Member in the Lightwave Networks Research Department at AT&T Labs-Research, Red Bank, New Jersey, USA. He has been with the Department of Electrical and Electronics Engineering at Bilkent University since 1998, where he is currently an associate professor. Dr. Karasan is a member of the Editorial Board of Optical Switching and Networking journal. He is the recipient of 2004 Young Scientist Award from Turkish Scientific and Technical Research Council (TUBITAK), 2005 Young Scientist Award from Mustafa Parlar Foundation and Career Grant from TUBITAK in 2004. Dr. Karasan received a fellowship from NATO Science Scholarship Program for overseas studies in 1991–94. Dr. Karasan is participating in the FP7-IST NoE BONE project. His current research interests are in the application of optimisation and performance analysis tools for the design, engineering and analysis