AIMD-based online MPLS traffic engineering for TCP flows via distributed multi-path routing

Onur ALPARSLAN*, Nail AKAR*, Ezhan KARASAN*

Abstract

With this paper, we propose a distributed online traffic engineering architecture for MPLS networks. In this architecture, a primary and secondary MPLS LSP are established from an ingress LSR to every other egress LSR. We propose to split the TCP traffic between the primary and secondary paths using a distributed mechanism based on ECN marking and AIMD-based rate control. Inspired by the random early detection mechanism for active queue management, we propose a random early reroute scheme to adaptively control the delay difference between the primary and secondary LSPs. Considering the adverse effect of packet reordering on TCP performance for packet-based load balancing schemes, we propose that the TCP splitting mechanism operates on a per-flow basis. Using flow-based models developed for Internet traffic and simulations, we show that flow-based distributed multi-path traffic engineering outperforms on a consistent basis the case of a single path in terms of per-flow goodputs. Due to the elimination of out-of-order packet arrivals, flow-based splitting also enhances TCP performance with respect to packet-based splitting especially for long TCP flows that are hit hard by packet reordering. We also compare and contrast two queuing architectures for differential treatment of data packets routed over primary and secondary LSPs in the MPLS data plane, namely first-in-first-out and strict priority queuing. We show through simulations that strict priority queuing is more effective and relatively more robust with respect to the changes in the traffic demand matrix than first-in-first-out queuing in the context of distributed multi-path routing.

Key words: Teletraffic, Internet, TCP/IP, Network routing, Resource sharing, Queue, Distributed system.

INGÉNIERIE DU TRAFIC MPLS POUR LES FLUX TCP VIA UN ROUTAGE MULTIPLE DISTRIBUÉ, EFFECTUÉ EN LIGNE SELON UN ALGORITHME AIMD

Résumé

Cet article présente une architecture distribuée d'ingénierie du trafic des réseaux MPLS (Multi-Protocol Label Switching ou commutation multiprotocole par étiquettes). Dans cette

* Electrical and Electronics Engineering Department, Bilkent University, Bilkent, Ankara 06800, Turkey, e-mail: akar@ee.bilkent.edu.tr.
This work is supported in part by the Science and Research Council of Turkey (Tübitak) under projects EEEAG-101E048 and EEEAG – 101E025.

architecture, on établit un chemin commuté par étiquette (LSP : Label Switched Path) primaire et un secondaire d'un routeur-commutateur (LSR : Label Switch Router) d'entrée à un routeur-commutateur de sortie. On propose de partager le trafic TCP entre les chemins primaire et secondaire en utilisant un mécanisme basé sur un marquage ECN (Explicit Congestion Notification : annonce explicite de congestion) et une commande basée sur l'algorithme AIMD (Additive Increase Multiplicative Decrease : croissance linéaire, diminution exponentielle). En s'inspirant des mécanismes de rejet aléatoire précoce (random early discard) de la gestion de files d'attente, on propose un mécanisme de reroutage précoce pour commander de manière adaptative la différence de délai entre les chemins primaire et secondaire. En considérant l'effet du réordonnancement des paquets sur la performance de TCP, on propose que le mécanisme de partage de trafic s'appuie sur les flux de communication. En utilisant des modèles à base de flux développés pour le trafic de l'internet et des simulations, on montre que l'ingénierie du trafic multichemin distribuée à partir des flux permet de meilleurs résultats qu'avec un seul chemin. Et puisqu'on élimine l'arrivée de paquets dans le désordre, cette séparation des trafics à partir des flux est encore meilleure dans le cas des gros flux TCP. On compare également deux architectures de files d'attente pour le traitement des paquets de données routés sur les chemins primaire et secondaire : la politique premier-entré premier-servi et la politique de respect de priorité. On montre à l'aide de simulations qu'une politique de strict respect de priorité est plus efficace et relativement plus robuste dans le contexte de routage multichemin distribué.

Mots clés : Télétrafic, Internet, TCP/IP, Routage réseau, Partage ressource, File attente, Système réparti.

Contents

I. INTRODUCTION

Internet Traffic Engineering (TE) is defined as the set of mechanisms required for enhancing the resource utilization of an operational network. In particular, traffic engineering controls how traffic flows through a network so as to optimize resource utilization and network performance [1]. TE mechanisms can be applied to hop-by-hop, explicit, or multi-path routed networks.

Traditional hop-by-hop routed IP networks using OSPF or IS-IS routing protocols resort to shortest path routing with simple link weights such as hop-count or delay. 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. A number of research studies have recently been carried out on traffic engineering in hop-by-hop routed networks. For a given traffic demand matrix, these studies seek an optimal set of link weights to improve the routing performance and these link metrics are computed using a centralized optimization algorithm [2, 3, 4]. Once

computed, the link metrics can be configured into the core IP routers either manually or automatically. We note that an accurate estimate of the traffic demand matrix should be available for this approach for acceptable performance, which is generally hard to obtain [5]. This approach is effective particularly when the traffic matrix does not change significantly in short time scales [6]. A robust extension of this approach that can cope with failures is also available in [7].

In the overlay approach, service providers establish logical connections between the edge nodes of a backbone, and then overlay these logical connections onto the physical topology. The hop-by-hop routing paradigm using shortest paths is overridden in this approach since logical connections can take any feasible path through the network. The emergence of Multi-Protocol Label Switching (MPLS) technology provides the necessary protocols and mechanisms in IP backbones for explicit routing to facilitate traffic engineering [1, 8]. In MPLS backbones, a constraint-based routing scheme can be used so that the traffic may be controlled to flow optimally through certain routes [9, 10]. In the general overlay approach, typically an initial logical connection layout is obtained for traffic engineering using a long-term traffic matrix and constraint-based routing. The bandwidth allocated to each virtual connection in this layout is directly related to the actual long-term traffic between the end points of the virtual connection. However, the actual traffic may occasionally deviate from the long-term value and this deviation generally cannot be predicted in advance. When there is a significant traffic increase in the actual traffic for a certain logical connection, additional bandwidth allocation will be needed, and the network will be signaled for an increase in the allocated bandwidth. If extra bandwidth is available, this request will be accepted and an additional allocation will be made. Otherwise, a new constraint-based route will be sought and if found, the original logical connection will be torn down and a new logical connection will be established. When there is a significant traffic reduction, the network will be signaled for the deallocation. A crucial performance impacting issue in such TE mechanisms is the determination of the amount of traffic change required to initiate such a signaling process.

Another TE approach that we focus on in the current paper is "multi-path routing", the goal of which is to improve the utilization of resources of an underlying physical network by providing multiple paths between source-destination (s-d) pairs. In the multi-path overlay approach, multiple logical connections with disjoint paths are established between the two end points of a network. These paths can be determined by using the long-term traffic demand. The goal of multi-path overlay traffic engineering is to increase the resource utilization of the network by intelligently splitting the traffic between an s-d pair among multiple alternative logical connections. The work in [11] proposes a dynamic multi-path routing algorithm in connection-oriented networks where the shortest path is used under light traffic conditions and multiple paths are used as the shortest path becomes congested. The adaptive multi-path approach, proposed in [12, 13, 14, 15] considers a general cognitive packet network carrying smart, dumb, and acknowledgment packets. In this approach, smart packets explore and learn optimal routes using reinforcement learning in an adaptive manner and dumb packets that carry actual payload follow these learned routes. Multi-path routing is studied in the context of wireless networks as well; a distributed multi-path routing scheme that selects a network path with sufficient resources in a dynamic multi-hop mobile setting is described in [16].

Recently, there have been a number of multi-path traffic engineering proposals specifically for MPLS networks that are amenable to distributed online implementation. In [6], probe packets are transmitted periodically to the egress Label Switch Router (LSR) which then returns them back to the ingress LSR. Based on the information in the returning probe packets, the ingress LSR computes the one-way congestion statistics that can be delay or loss, and uses a gradient projection algorithm for load balancing. In the model [6], all paths between an s-d pair are treated equally which may be problematic in scenarios for which some paths may have significantly longer hop lengths than their corresponding min-hop paths. Additive Increase/Multiplicative Decrease (AIMD) feedback algorithms are used generally for flow and congestion control in computer and communication networks [17, 18]. The multi-path AIMD-based approach of [19] uses binary feedback information regarding the congestion state of the LSPs and a traffic splitting heuristic using AIMD is proposed which ensures that source LSRs never send traffic to secondary paths of longer length before they make full use of their primary paths.

A challenging issue in multi-path routing is the potential de-sequencing (or reordering) of packets through the network since packets will occasionally take different routes with different delays. A queuing analysis of packet de-sequencing (or reordering) and required re-sequencing at the receiver site was carried out in [20] to show the impact of de-sequencing on total delays. The majority of the traffic in the current Internet is TCP-based and the impact of de-sequencing of TCP packets on application layer performance is crucial. However, TCP receiver behavior is quite complex and varies from one implementation to another and is not amenable to a stochastic analysis as described in [20]. Experimental studies demonstrate that packet de-sequencing within a TCP flow can significantly deteriorate TCP performance [21, 22]. When traffic is split in a static manner (i.e., splitting ratios are fixed over time), hashing-based splitting schemes are shown to be effective in terms of both scalability and de-sequencing performance [21]. In dynamic traffic splitting, the splitting ratios change adaptively over time with the changing congestion status of the corresponding alternative paths. Flow-based multi-path routing is a dynamic splitting scheme that operates on a per-flow basis with the aim of avoiding packet de-sequencing within a flow; see for example [23] and [24] for related work. We refer the reader to [25] and [26] for flow-based multipath routing with emphasis on routing of elastic flows like TCP. Flow-based routing in the QOS routing context in MPLS networks is described in [27] and [28], but these studies require a flow aware core network and therefore have scalability problems with increasing number of instantaneous flows [27].

Recently, a new flow-based multi-path traffic engineering approach for best-effort networks is proposed in [29]; this reordering-free architecture is tested for UDP traffic, and its performance improvements relative to the case of non-flow based multi-path routing and that of single path are reported. The approach imposes flow awareness only at the edge devices and therefore does not have the scalability problems of other flow-based QOS routing schemes. In the current paper, we extend the work in [29] towards a reordering-free flow-based traffic engineering architecture for TCP traffic in MPLS backbones. In this architecture, two link disjoint MPLS LSPs, one being the primary Label Switched Path (P-LSP) and the latter being the secondary LSP (S-LSP), are established between LSR pairs between which there is direct TCP traffic. The proposed architecture is amenable to extension to arbitrary number of paths but this generalization is left outside the scope of the current paper to improve its readability.

After establishing these two LSPs, we develop an algorithm that splits the traffic between the two LSPs in order to improve the overall throughput. Motivated by the ABR (Available Bit Rate) service category used for flow control in ATM networks, we proposed an explicit rate feedback mechanism for traffic engineering purposes in MPLS networks in [29]. Since explicit rate feedback is not an MPLS standard, we propose in this paper, a binary feedback mechanism that can be implemented with the help of standards-based ECN (Explicit Congestion Notification)-capable LSRs with little additional complexity. In the proposed mechanism, edge LSRs maintain one drop-tail queue for the P-LSP and another drop-tail queue for the S-LSR. These queues are drained using an AIMD algorithm triggered by the binary feedback mechanism.

In the proposed flow-based multi-path traffic engineering architecture, a traffic splitting algorithm is proposed in which individual traffic flows are identified and probabilistically assigned to one of the two LSPs based on the average difference between the delays in the corresponding queues. The algorithm we propose for traffic splitting is called RER (Random Early Reroute), which is inspired by the RED (Random Early Detection) algorithm used for active queue management in the Internet. The flow-based mechanism ensures that packet reordering would not take place at the receiving end of the corresponding flow.

One other important issue in dynamic load-balancing is the design of the MPLS data plane in terms of its queuing architecture. It is well-known that using longer alternative paths by some sources force other sources whose min-hop paths share links with these alternative paths to also use alternative paths [30]. If the alternative paths use more resources (or hops) on the average, some improperly designed load balancing algorithms may perform poorly even relative to a scheme that uses a single path for every s-d pair. This fact is called the knock-on effect in literature, and precautions should be taken to minimize this effect [30]. Inspired by the well-known principles of dynamic routing in circuit switched networks, a trunk reservation based approach is presented to deal with this effect in [26]. We take a different approach in this paper for the same problem using per-class queueing. In [29], we proposed a queuing architecture in the MPLS data plane that favors packets of P-LSPs over those of S-LSPs in Strict Priority (SP) sense to cope with the knock-on effect. In this paper, we compare and contrast the SP and the widely deployed First-In-First-Out (FIFO) queuing architectures in the TCP load-balancing context and show through simulations that SP queuing is more effective and relatively more robust with respect to the changes in the traffic demand matrix than FIFO queuing.

The remainder of the paper is organized as follows. In Section II, we present our traffic engineering architecture. Our simulation results are presented in Section III and conclusions and future work are provided in the final section.

II. ARCHITECTURE

In this study, we envision an MPLS network consisting of edge and core LSRs depicted in Figure 1 TCP/IP traffic is sourced from and destined to the edge LSRs while the core LSRs carry only transit traffic. Flow classification, traffic splitting, per-destination queuing, rate control,

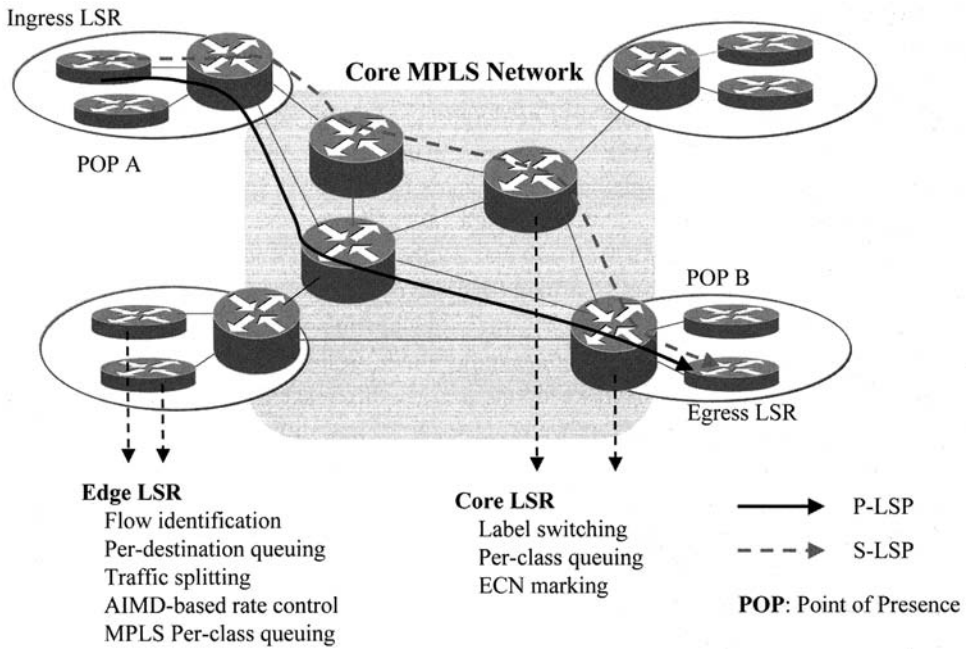

FIG. 1 – MPLS network with edge and core LSRs and the minimum hop P-LSP and the S-LSP from the ingress LSR at POP A to the egress LSR at POB B.

Réseau MPLS (commutation multiprotocole par étiquettes) avec routeurs de cœur et routeurs frontière et les chemins primaire à nombre minimal de bonds et secondaire entre un routeur entrant au point de présence A et un routeur sortant au point de présence B.

and the conventional edge MPLS functionalities are carried out by the edge LSRs in the proposed architecture. The core LSRs are not flow aware and their sole responsibility is label switching, per-class queuing, and ECN marking. Consequently, the proposed architecture has the potential to scale to large networks since flow awareness and flow processing capabilities are required only on the edge LSRs, but not on the core LSRs where scalability requirements are elaborate.

Our proposed distributed on-line traffic engineering architecture is comprised of the following three main components: (i) LSP establishment and queuing model used in core LSRs, (ii) feedback mechanism and rate control, (iii) traffic splitting algorithm. We now study each of these components and their interactions.

We propose in this study that the core LSRs employ output queuing and they support differentiated services (diffserv) with the gold, silver, and bronze services (i.e., Olympic services). These services can be implemented with per-class queuing using three drop-tail queues, namely gold, silver, and bronze queues, at every outgoing physical interface. In terms of scheduling, we propose that the gold queue has strict priority over the silver queue that has strict priority over the bronze queue. In our proposed architecture, the gold service is dedicated not only to Resource Management (RM) packets (their role will be described in

detail later) used for gathering binary congestion status from the network but for TCP ACK (i.e., acknowledgment) packets as well. The motivation behind this choice is to provide prompt feedback to TCP end users and also traffic splitters deployed at edge LSRs. How the remaining silver and bronze services will be used for TCP data packets are based on the way LSPs are established which is described below.

We assume in this study that edge LSRs are single-homed, i.e., they have a link to a single core LSR. Two LSPs, which are link disjoint in the core MPLS network, are then set-up from an ingress LSR to every other edge LSR for which there is direct TCP/IP traffic. If edge LSRs are multi-homed then the "link disjointness" requirement could be imposed on the entire MPLS cloud including the edge LSRs; we do not study multi-homing in the current paper. The P-LSP uses the minimum hop path found using Dijkstra's algorithm. When there is a tie in the algorithm, this tie is broken randomly. The route for the S-LSP is found by pruning the links in the core MPLS network used by the P-LSP and randomly choosing one of the minimum hop paths in the remaining network graph. For an example, we refer the reader to Figure 1 in which the P-LSP and S-LSP traverse two and three hops, respectively, in the core network. If the connectivity is lost after pruning the links from the graph, the secondary LSP is not established. We note that if an accurate estimate of the traffic demand matrix is known a-priori, more sophisticated algorithms might be used to select the routes LSPs take. However, we do not assume a-priori knowledge on traffic demands in this study.

There are two queuing models based on the work in [29] that we study in this paper. In FIFO queuing, data packets of P-LSPs and S-LSPs join the same silver queue and we do not make use of the bronze queue at all. In other words, in FIFO queuing there is no preferential treatment to packets using fewer resources (i.e., traversing fewer hops). However, this policy might carry the risk of promoting the knock-on effect, i.e., using longer secondary paths by some sources may force other sources whose primary paths share links with these secondary paths to also use secondary paths. We remark that secondary paths use more resources (i.e., longer hop lengths) and they should be resorted to only when the primary paths can no longer accommodate additional TCP traffic. Based on the work described in [29], we propose SP queuing in which TCP data packets belonging to P-LSPs use the silver service and those belonging to S-LSPs use the bronze service. We note that the above-mentioned queuing models are implementable using the standards-based E-LSP (EXP-inferred-PSC LSP) method by which the three-bit experimental (EXP) field in the MPLS header is used by edge LSPs to code the PSC (Per Hop Behavior Scheduling Class) [31]. We propose that two of the EXP bits are devoted to marking the packet as a

(A) RM packet for a P-LSP

(B) RM packet for an S-LSP,

(C) TCP data packet for a P-LSP,

(D) TCP data packet for an S-LSP.

We also propose to mark TCP ACK packets in the same way as RM packets (marked as either (A) or (B)) since they should receive the same gold service. However, this means that ACK packets will also go through additional RM packet processing. The reason behind this pragmatic choice is that we would like to reserve the final bit of the three EXP bits for binary congestion indication and not use it for separately marking TCP ACK packets.

The second basic component of the overall architecture is the feedback mechanism and rate control to be used for traffic engineering. MPLS technology does not currently have a standards-based feedback mechanism, but we proposed in [29] that a protocol very similar to that of the ABR service category in ATM networks to be used in MPLS networks as well. Explicit rate feedback was shown to be useful for traffic engineering purposes in [29] due to its promptness and well-proven transient properties. However, we relax this requirement in the current paper for ease of potential implementation. In the current architecture, we require that the core LSRs are ECN-capable [32], and upon congestion, they mark the packet as Congestion Experienced (CE), where the marking can be done with the remaining EXP bit; see [33] for an initial proposal on ECN support in MPLS which has not advanced. In the architecture discussed in this paper, the ingress LSR of each LSP periodically sends RM packets along with data packets on the same LSP towards the egress LSR. If the RM packet belongs to a P-LSP (i.e., marked as (A)), the LSRs check the percentage queue occupancy of the silver queue of that interface and sets the CE bit (if not already set) if this value exceeds the configuration parameter μ. Similarly, in the case of an S-LSP RM packet (i.e., marked as (B)), the transit LSR compares the percentage queue occupancy of the bronze (silver) queue with μ if SP (FIFO) queuing is employed and sets the CE bit accordingly.

Table I. – The AIMD algorithm.

L'algorithme AIMD.

```
if RM packet market as CE
        ATR: = ATR – RDF × ATR
else
        ATR: = ATR + RIF × PTR
ATR: = min (ATR, PTR)
ATR: = max (ATR, MTR)
```

This RM packet is then returned back by the egress LSR to the ingress node indicating congestion status of the forward LSP it belongs to. For every LSP, RM packets are sent to the network once in every T_{RM} seconds. When the ingress LSR receives the congestion information about the LSP, it will invoke an AIMD algorithm to compute the ATR (Allowed Transmission Rate) of the corresponding LSP. The AIMD algorithm is given in Table I [17]. In this algorithm, *RDF* and *RIF* denote the Rate Decrease Factor and Rate Increase Factor, and *MTR* and *PTR* correspond to Minimum Transmission Rate and Peak Transmission Rate, respectively.

We now describe the splitting process for the TCP traffic at the edge LSRs. The edge LSRs identify TCP traffic flows and maintain a list which keeps track of each active flow. We note that the traffic carried between the source and destination LSRs is an aggregation of multiple traffic flows generated by multiple users/applications. Two drop-tail queues per egress LSR, namely the P-LSP and S-LSP queues, are maintained at the edge LSRs. Both queues are drained using the ATR information calculated by the AIMD algorithm given in Table I. At this stage, we

decide on which service queue each TCP flow would join. When a packet arrives, which is not associated with an existing flow, a decision on how to forward the packets of this new flow needs to be made. For this purpose, we compute D_{P-LSP} and D_{S-LSP}, the delay estimates for the P-LSP and S-LSP queues in the edge LSR, respectively. These delay estimates are calculated by means of dividing the corresponding queue occupancy by the drain rate, ATR, of that queue. The notation d_n denotes the exponential weighted moving averaged difference between the delay estimates at the epoch of the nth packet arrival and is updated as follows:

$$d_n = \beta (D_{P-LSP} - D_{S-LSP}) + (1 - \beta) d_{n-1} \quad (1)$$

where β is the averaging parameter to be set by the network operator. We refer the reader to earlier work described in [13, 14, 15] for the use of exponentially averaged delay or other QoS estimates for making routing decisions.

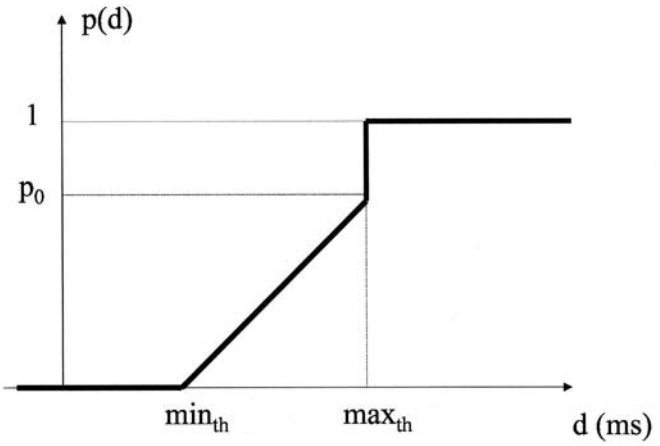

FIG. 2 – The traffic splitting function $p(d)$.

La fonction de partage de trafic p(d).

In our traffic engineering architecture, every new active flow is identified with the arrival of the first packet of the new flow, say nth packet arrival. This new flow is assigned to the secondary LSP with probability $p(d_n)$ which is given in Figure 2 as a function of the estimated delay difference at the arrival of the nth packet. Equivalently, the new flow is said to be assigned to the primary LSP with probability $(1 - p(d_n))$. This LSP assignment curve is similar to the Random Early Detection (RED) curve used for active queue management [34]. We call this policy for multi-path traffic engineering as the Random Early Reroute (RER) policy. RED has the goal of controlling the average queue occupancy whereas in multipath TE, the average (smoothed) delay difference between the two queues is controlled by the RER. The RER uses a proportional control rather than a simple threshold policy in order to control the potential

fluctuations in the controlled system. The RER favors the min-hop path and resorts probabilistically to the secondary path when the P-LSP queue starts to build up. Once an LSP is selected upon the arrival of the first packet of a new flow, all successive packets of the same flow will be forwarded over the same LSP. Finally, the edge LSRs are assumed to have the same per-class queuing functionality at their outgoing physical interfaces as core LSRs.

The architecture is summarized in an example network with three edge LSRs (0-2) and three core LSRs depicted in Figure 3. In this figure, the internals of only the edge LSR 0 are given. P-LSP(n) queue refers to the queue maintained for TCP data packets destined for the egress LSR n and using the primary path. These packets then join the silver queue of the per-class queuing stage for later transmission towards the core LSR. S-LSP(n) queue can be similarly defined. If SP is used, then TCP data packets using the secondary paths will enter the bronze queue in the second stage. In Figure 3, the SP queuing case is depicted. If FIFO queuing were employed, then the TCP data packets of secondary paths would also join the silver queue as those of primary paths. All queues in the per-destination queuing stage are drained by the ATR of the corresponding queue which is calculated by the AIMD-based algorithm. RM packets and TCP ACK packets bypass the first stage, and they directly join the gold queue.

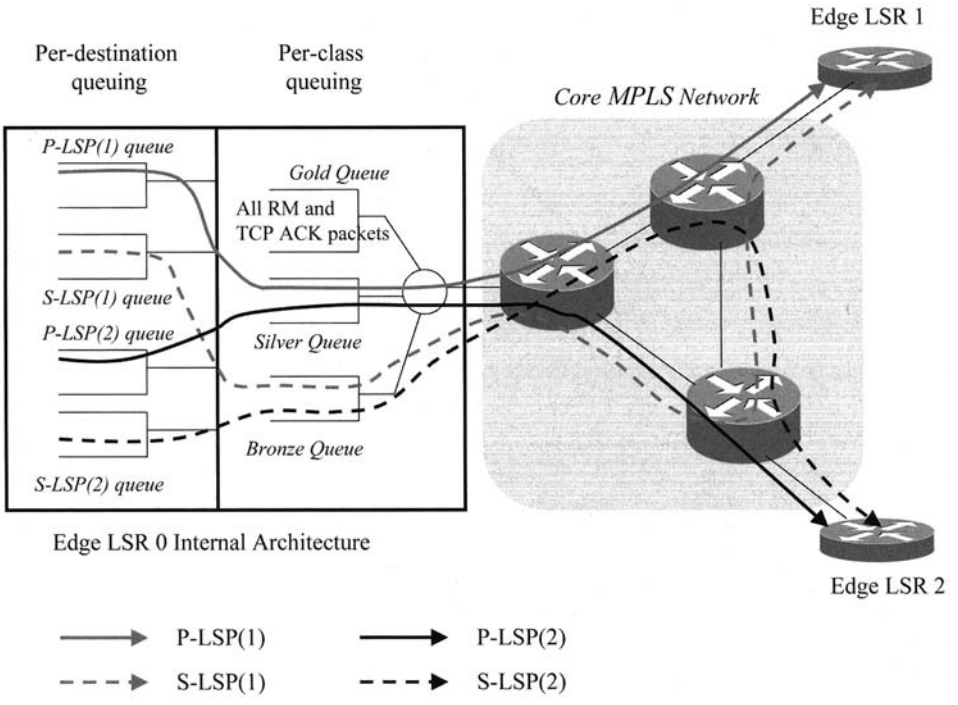

FIG. 3 – MPLS network with 3 edge LSRs and 3 core LSRs with strict priority queuing at the physical interfaces. All the outgoing routes for the edge LSR 0 for the TCP data packets are depicted.

Réseau MPLS (Multi-Protocol Label Switching : commutation multiprotocoles par étiquette) avec 3 routeurs (LSR : Label Switch Router) de coeur et 3 routeurs de frontière dont les interfaces physiques sont gérées avec priorité stricte. Toutes les routes sortantes des paquets de données TCP pour la frontière LSR 0 sont décrites.

III. SIMULATION RESULTS

The performance of the AIMD-based multi-path TE algorithm for TCP traffic is evaluated for the three-node network shown in Figure 3. We assume that the core LSRs are connected by links each with a capacity of 50 Mbit/s, and each link has a propagation delay of 10 ms. We also assume that edge LSRs are connected to the core LSRs with 1 Gbit/s links and therefore the core-to-core links are potential bottleneck links.

In our simulations, flow arrivals occur according to a Poisson process, and flow sizes have a bounded Pareto distribution [35]. The bounded Pareto distribution is used instead of the normal Pareto (similar to [36]) because normal Pareto distribution has infinite variance, and thus very long simulations are required for convergence. Meanwhile, bounded Pareto 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 bounded Pareto distribution is denoted by $BP(k, p, \alpha)$, 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 is decreased, the tail gets larger, and the ratio of long flows increases. The probability density function for the $BP(k, p, \alpha)$ is given by

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

The average flow size, m, for the $BP(k, p, \alpha)$ distribution is given by [35]

$$m = \frac{\alpha}{(1 - \alpha)(p^{\alpha} - k^{\alpha})} (pk^{\alpha} - kp^{\alpha}).$$

The following parameters are used for the bounded Pareto distribution in this study: $k = 4000$ Bytes, $p = 50 \times 10^6$ Bytes, and $\alpha = 1.06$ or 1.20. The mean flow sizes are given by $m = 30{,}544$ Bytes for $\alpha = 1.06$, and $m = 20{,}362$ Bytes for $\alpha = 1.20$.

The average outgoing traffic from each edge LSR is fixed to 70 Mbit/s in our simulations. The offered traffic from edge LSR i and to edge LSR j is denoted by $T_{i,j}$. For simplicity, we assume that $T_{i,((i+1) \bmod 3)} = \gamma T_{i,((i-1) \bmod 3)}$ for all $0 \leq i \leq 2$. The traffic spread parameter, γ, is introduced in order to characterize the effect of traffic distribution on multi-path TE. While $\gamma = 1$ corresponds to fully symmetric traffic, $\gamma = 0$ corresponds to totally asymmetric traffic.

In order to evaluate the performance of the flow-based multi-path TE algorithm, we use single-path routing and packet-based TE algorithms for reference. In packet-based TE, packets are routed over the P-LSP or S-LSP using the RER mechanism given in Figure 2, but irrespective of the flow they belong to. The packet-based multi-path TE causes out-of-order packet delivery at the destination, and this is known to adversely affect the TCP performance [23, 24]. We study this effect in our numerical experiments. Single-path routing uses the minimum-hop path with the AIMD-ECN capability turned on. We use the term "shortest-path routing" to refer to this scheme.

Two sets of buffer threshold parameters for the RER curve are used in this study:

- Shortest Delay (SD): $min_{th} = max_{th} = 0$ ms and $p_0 = 1$.
- RER: $min_{th} = 1$ ms, $max_{th} = 15$ ms and $p_0 = 1$.

SD forwards each flow (for flow-based TE) or packet (for packet-based TE) simply to the path with the shorter estimated queuing delay at the ingress edge LSR, and thus it does not favor the P-LSP. SD is used in conjunction with the FIFO queuing discipline where there is no preferential treatment between the P-LSP and the S-LSP at the core LSRs. For UDP traffic, we had used RER parameters $min_{th} = 30$ ms, $max_{th} = 150$ ms and $p_o = 1$ in [29]. However, for the TCP traffic, the queue buildup is less severe due to the rate adaptivity of users so we had to reduce the thresholds accordingly. We experimented extensively with different RER parameters but we observed that in the neighborhood of the chosen RER parameter set, the performance of the RER is quite robust. The results of these exploratory numerical studies are not included in this paper in order to make the presentation more concise. The delay averaging parameter is selected as $\beta = 0.3$.

The TCP data packets are assumed to be 1040 Bytes long including MPLS headers. We assume that the RM packets are 50 Bytes long. All the buffers at the edge and core LSRs, including per-destination (primary and secondary) and per-class queues (gold, silver and bronze), have a size of 104,000 Bytes each. The TCP receive buffer is of length 19,840 Bytes.

The following parameters are used for the AIMD algorithm:

- $T_{RM} = 0.1$ s
- $RDF = 0.0625$
- $RIF = 0.125$
- $PTR = 50$ Mbit/s
- $MTR = 0$
- $\mu = 50\%$

We leave the study of the impact of AIMD parameters on routing performance for future research.

This TCP TE architecture is implemented over ns-2 (Network Simulator) version 2.27 [37] and the TCP-Reno is used in our simulations. The simulation runtime is selected as 300 s. In all simulation results, events concerning flow arrivals only in the period [95 s, 295 s] are reported. The following five algorithms are compared in terms of their performance:

- Flow-based multi-path with RER and Strict Priority
- Flow-based multi-path with Shortest Delay and FIFO
- Packet-based multi-path with RER and Strict Priority
- Packet-based multi-path with Shortest Delay and FIFO
- Shortest-path (i.e., Single Path using the min-hop path)

The goodput of a TCP flow i (in Kbytes/s) 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 Bytes successfully delivered to the application layer by the TCP receiver within the simulation duration. The parameter T_i is the sojourn time of the flow i within the simulation runtime. We note that if flow i has terminated within the simulation time, then Δ_i will be equal to the flow size in Bytes. The average goodputs for TCP flows as a function of the flow size are given in Figures 4 and 5 for $\alpha = 1.06$ and $\alpha = 1.20$, respectively. We first observe that plots for both

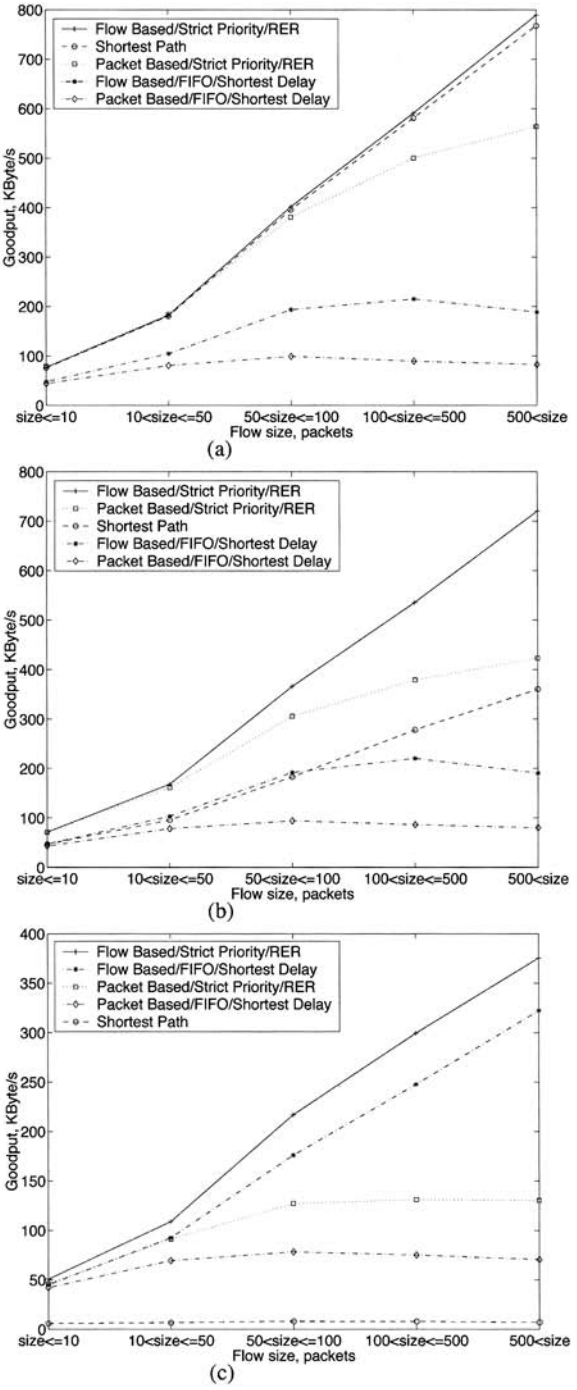

FIG. 4 – Goodput as a function of flow size for $\alpha = 1.06$ and (a) $\gamma = 1.0$, (b) $\gamma = 0.4$, and (c) $\gamma = 0$.

Débit utile moyen en fonction de la taille du flux pour $\alpha = 1{,}06$ et (a) $\gamma = 1{,}0$, (b) $\gamma = 0{,}4$, et (c) $\gamma = 0$.

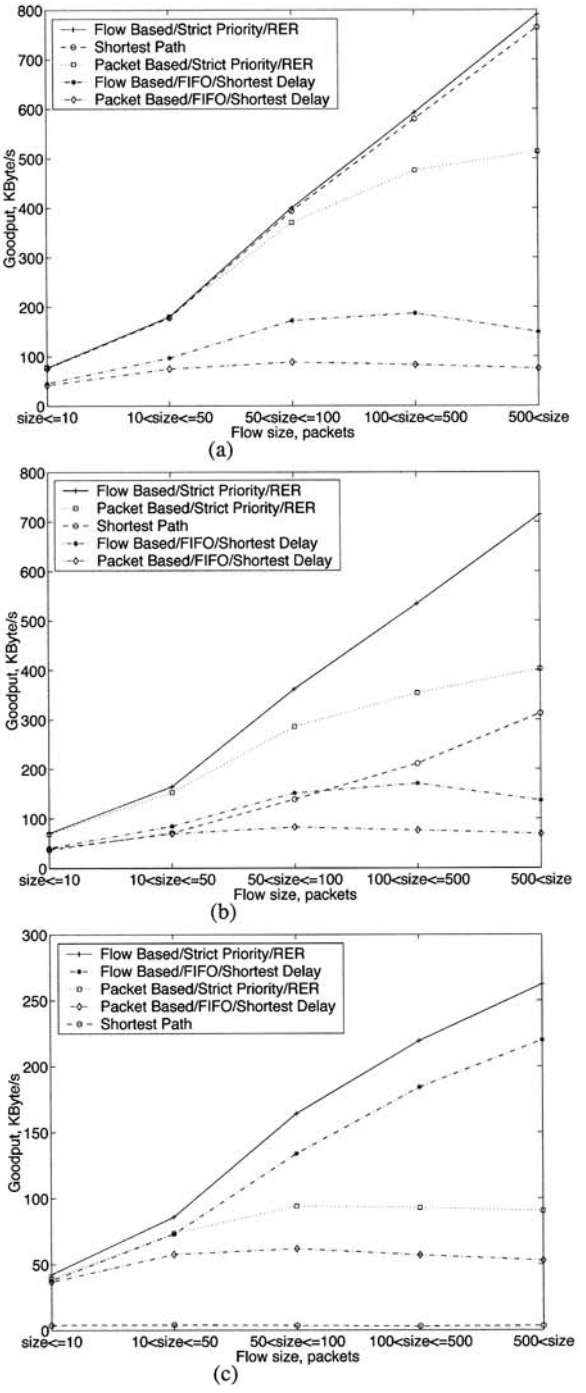

FIG. 5 – Goodput as a function of flow size for $\alpha = 1.20$ and (a) $\gamma = 1.0$, (b) $\gamma = 0.4$, and (c) $\gamma = 0.0$.

Débit utile moyen en fonction de la taille du flux $\alpha = 1{,}20$ et (a) $\gamma = 1{,}0$, (b) $\gamma = 0{,}4$, et (c) $\gamma = 0{,}0$.

values of α exhibit similar behavior. The average goodput for each flow size range is computed by taking the arithmetic mean of the goodputs of the flows that have a size within the given range. The flow-based multi-path TE algorithms always attain the highest average goodput for different values of α, γ and flow size. The flow-based TE with RER and Strict Priority performs better than the flow-based TE with Shortest Delay and FIFO in all cases.

Both packet-based TE algorithms, i.e., Strict Priority/RER and FIFO/Shortest Delay, generally perform worse than their flow-based counterparts. This is a result of the packet reordering occurring with packet-based algorithms. Large flows that are active for a longer period are more adversely affected with this, whereas packet reordering affects shorter flows less since the smoothed delay difference d_n does not change significantly during the short period while a small flow is active. The packet-based TE algorithm with Shortest Delay and FIFO performs much worse than the packet-based algorithm with RER and Strict Priority, since the former alternates packets between P-LSP and S-LSP as d_n fluctuates around zero, causing larger number of packet re-orderings. The performance of the shortest path algorithm deteriorates compared to flow-based and packet-based TE algorithms as the traffic becomes more asymmetric and P-LSP becomes more congested, i.e., as γ decreases.

The average goodputs generally increase with the flow size since larger flows have the advantage of achieving larger TCP congestion windows. The average goodputs for the five routing algorithms as a function of the traffic distribution parameter γ are shown in Figure 6.

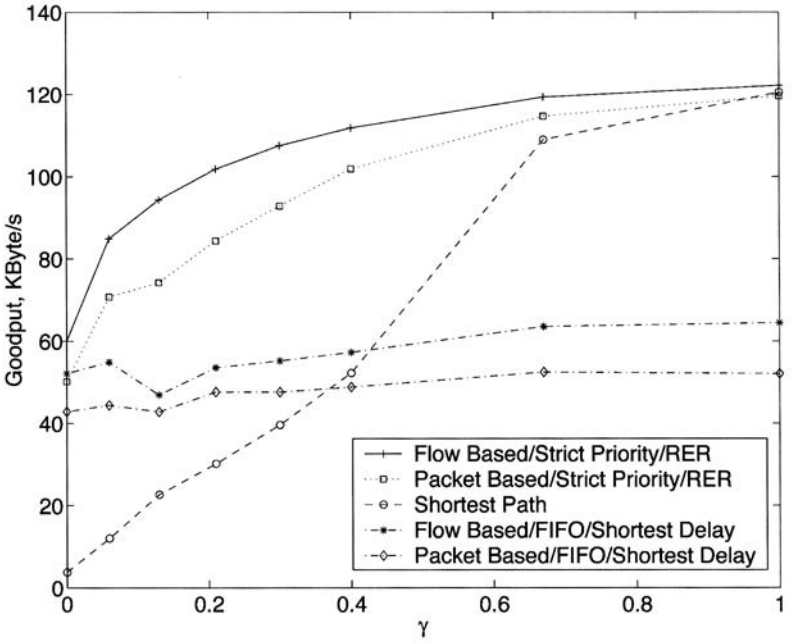

FIG. 6 – Average per-flow goodput as a function of γ for $\alpha = 1.20$.

Débit utile moyen par flux en fonction de gamma pour γ de $\alpha = 1{,}20$.

In this figure, the average goodput is computed as the arithmetic mean of all flow goodputs. We observe that the flow-based TE algorithm with RER and Strict Priority achieves the highest average goodput among all TE algorithms considered in this study. The average goodputs for the flow and packet-based TE algorithms with RER and Strict Priority decrease as γ decreases since P-LSP becomes more congested. On the other hand, performances of the flow and packet-based TE algorithms with Shortest Delay and FIFO do not change significantly as γ changes between 0.0 and 1.0. This is a consequence of the equal treatment between P-LSP and S-LSP with these algorithms. TE algorithms with RER and Strict Priority and the shortest path routing algorithm have similar performances for large γ since routing with TE algorithms boils down to the shortest path routing when γ is large as P-LSP becomes very lightly loaded.

In order to have a more accurate representation of the goodputs achieved by individual packets, we compute the normalized goodput which is defined as

$$G_{norm-avg} = \frac{\sum_i n_i G_i}{\sum_i n_i}$$

where G_i is the average throughput attained by flow i, and n_i is the number of packets for flow i. The normalized goodputs for the five routing algorithms as a function of the traffic distribution parameter γ are shown in Figure 7. Since $G_{norm-avg}$ demonstrates the effect of

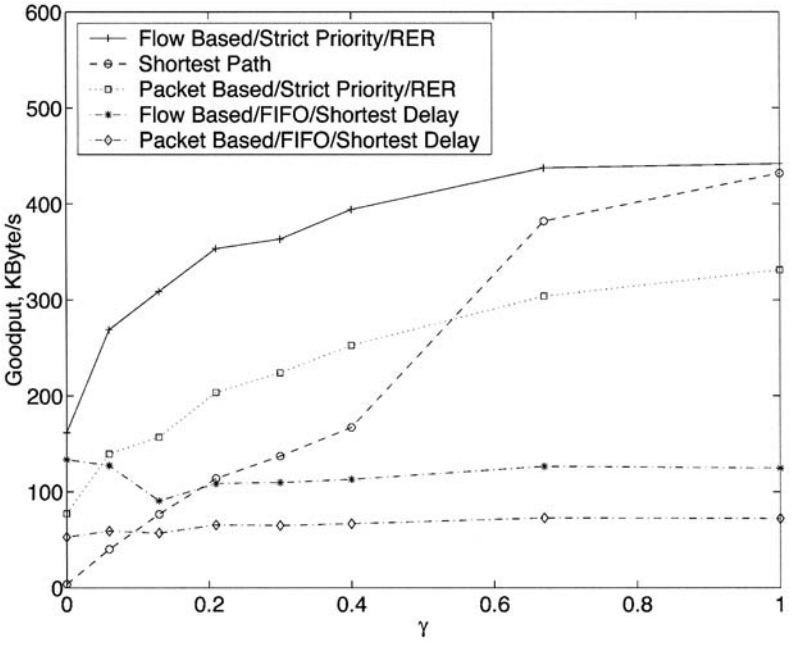

FIG. 7 – Normalized average goodput as a function of γ for $\alpha = 1.20$.

Débit utile moyen normé en fonction de γ pour $\alpha = 1.20$.

larger flows better, the difference between average goodputs achieved by flow and packet-based TE algorithms can be seen more clearly.

The relative change of the normalized goodputs with the four TE algorithms with respect to the shortest path routing are given in Table II. This relative change, Δ^{TE}, is computed for a generic TE method as

$$\Delta^{TE} = \frac{G^{TE}_{norm-avg} - G^{ShortestPath}_{norm-avg}}{G^{ShortestPath}_{norm-avg}}$$

where $G^{ShortestPath}_{norm-avg}$ is the normalized goodput with the shortest path routing, and $G^{TE}_{norm-avg}$ denotes the normalized goodput with one of the four TE algorithms used for the calculation of the corresponding Δ^{TE}. The flow-based TE algorithm with RER and Strict Priority achieves the highest normalized goodput compared with the other TE algorithms. Although the flow-based TE algorithm with Shortest Delay and FIFO has higher goodput relative to the shortest path algorithm for small values of γ, its performance degrades to worse than the shortest path routing for large values of γ, i.e., with more symmetric traffic distribution and less congested P-LSP. The packet-based TE algorithms also perform worse than the shortest path routing for large values of γ.

IV. CONCLUSIONS

In this paper, we propose a flow-based distributed scheme for traffic engineering TCP flows in MPLS networks. In this architecture, TCP traffic is split at the flow level between the primary and secondary paths using a random early rerouting principle that controls the

Table II. – Relative increase/decrease of normalized goodput, Δ^{TE}, for the four TE algorithms with respect to shortest path routing.

Augmentation/diminution du débit moyen utile normé, Δ^{TE}, pour quatre algorithmes d'ingénierie du trafic qui s'appuient sur le routage du plus court chemin.

γ	Flow-based		Packet-based	
	SP/RER	FIFO/SD	SP/RER	FIFO/SD
0.00	42.55	34.99	19.83	13.21
0.06	5.71	2.18	2.48	0.47
0.13	3.03	0.18	1.05	– 0.26
0.21	2.10	-0.05	0.79	– 0.43
0.30	1.65	-0.20	0.63	– 0.53
0.40	1.36	-0.32	0.51	– 0.60
0.67	0.15	-0.67	-0.20	– 0.81
1.00	0.02	-0.71	-0.23	– 0.83

queuing delay difference between the two alternative paths. We propose ECN marking and AIMD-based rate control so that the traffic splitting decisions can be made at the flow-aware edges of the MPLS network. Moreover, we propose a strict priority queuing mechanism not for QoS delivery but for improving routing performance via the mitigation of the knock-on effect that is known to exist for general load balancing schemes. Using Poisson flow arrivals and bounded Pareto-distributed flow sizes, we show that the flow-based distributed multi-path traffic engineering based on random early reroute and strict priority queuing, consistently outperforms the case of a single path in terms of the TCP goodput whereas the gain in using this scheme increases when the traffic becomes less uniform. Due to the elimination of the out-of-order packet delivery, flow-based splitting also enhances the TCP performance with respect to packet-based splitting especially for long TCP flows which are hit hard under packet reordering.

Manuscrit reçu le 20 décembre 2003
Accepté le 13 avril 2004

REFERENCES

[1] AWDUCHE (D. O.), CHIU (A.), ELWALID (A.), WIDJAJA (I.), XIAO (I.), "Overview and principles of Internet traffic engineering", IETF Informational RFC-3272, May 2002.

[2] BERRY (L.), KOHLER (S.), STAEHLE (D.), TRANGIA (P.), "Fast heuristics for optimal routing in IP networks", Universitat Wurzburg Institut fur Informatik Research Report Series, Tech. Rep. 262, July 2000.

[3] FORTZ (B.), THORUP (M.), "Internet traffic engineering by optimizing OSPF weights", in *Proceedings of INFOCOM*, Tel-Aviv, Israel, pp. 519-528, 2000.

[4] WANG (Y.), WANG (Z.), ZHANG (L.), "Internet traffic engineering without full mesh overlaying", in *Proceedings of INFOCOM*, Anchorage, USA, 2001.

[5] MEDINA (A.), TAFT (N.), SALAMATIAN (K.), BHATTACHARYYA (S.), DIOT (C.), "Traffic matrix estimation: Existing techniques and new directions", in *Proceedings of the ACM SIGCOMM*, August 2001.

[6] ELWALID (A.), JIN (C.), LOW (S.), WIDJAJA (I.), "MATE: MPLS Adaptive Traffic Engineering", in *Proceedings of INFOCOM*, pp. 1300-1309, 2001.

[7] YUAN (D.), "A bi-criteria approach for robust OSPF routing",' in *Proceedings of IEEE Workshop on IP Operations and Management*, Kansas City, Missouri, USA pp. 91-98, 2003.

[8] ROSEN (E.), VISWANATHAN (A.), CALLON (R.), "Multiprotocol label switching architecture", RFC 2481, January 2001.

[9] KODIALAM (M.), LAKSHMAN (T. V.), "Minimum interference muting with applications to MPLS traffic engineering", in *Proceedings of INFOCOM*, Tel-Aviv, Israel, March 2000.

[10] PLOTKIN (S.), "Competitive routing of virtual circuits in ATM networks", *IEEE Jour. Selected Areas in Comm.*, **13**, pp. 1128-1136, 1995.

[11] BAHK (S.), ZARKI (M. E.), "Dynamic multi-path routing and how it compares with other dynamic routing algorithms for high speed wide area networks", in *ACM SIGCOMM*, pp. 53-64, 1992.

[12] GELENBE (E.), LENT (R.), XU (Z.), "Towards networks with cognitive packets", in *MASCOTS, Proceedings of the 8th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, San Fransisco, CA, pp. 3-12, 2000.

[13] GELENBE (E.), LENT (R.), XU (Z.),"Measurement and performance of cognitive packet networks", *Computer Networks*, **37**, pp. 691-701, 2001.

[14] GELENBE (E.), LENT (R.), XU (Z.), "Design and performance of cognitive packet networks", *Performance Evaluation*, vol. 46, pp. 155-176, 2001.

[15] GELENBE (E.), LENT (R.), MONTUORI (A.), XU (Z.), "Cognitive packet networks: QoS and performance", in *MASCOTS, Proceedings of the 10th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, Fort Worth, TX, 2002, pp. 3-12.

[16] CHEN (S.), NAHRSTEDT (K.), "Distributed quality of service routing in ad-hoc networks", *IEEE Jour. Selected Areas in Comm.*, **17**, pp. 1488-1504, 1999.

[17] CHIU (D. M.), JAIN (R.), "Analysis of the increase/decrease algorithms for congestion avoidance in computer networks", *Computer Networks and ISDN Systems*, **17**, n° 1, pp. 1-14, June 1989.

[18] JACOBSON (V.), "Congestion avoidance and control", *ACM Computer Communication Review; Proceedings of the Sigcomm '88 Symposium in Stanford, CA, August, 1988*, vol. 18, 4, pp. 314-329, 1988.

[19] WANG (J.), PATEK (S.), WANG (H.), LIEBEHERR (J.), "Traffic engineering with AIMD in MPLS networks", in *7th IFIP/IEEE International Workshop on Protocols for High-Speed Networks,* 2002, pp. 192-210.

[20] BACCELLI (F.), GELENBE (E.), PLATEAU (B.), "An end to end approach to the resequencing problem", *Journal of the ACM,* vol. 31, n° 3, pp. 474-485, 1984.

[21] CAO (Z.), WANG (Z.), ZEGURA (E. W.), "Performance of hashing-based schemes for Internet load balancing", in *INFOCOM* (1), pp. 332-341, 2000.

[22] LAOR (M.), GENDEL (L.), "The effect of packet reordering in a backbone link on application throughput", *IEEE Network Magazine*, vol. 16, n° 5, pp. 28-36, 2002.

[23] SHAIKH (A.), REXFORD (J.), SHIN (K. G.), "Load-sensitive routing of long-lived IP flows", in *ACM SIGCOMM*, pp. 215-226, 1999.

[24] LEE (Y.), CHOI (Y.), "An adaptive flow-level load control scheme for multipath forwarding", in *Networking – ICN 2001,* 2001.

[25] OUESLATI-BOULAHIA (S.), OUBAGHA (E.), "An approach to elastic flow routing", in *International Teletraffic Congress,* Edinburgh, UK, June 1999.

[26] OUESLATI-BOULAHIA (S.), ROBERTS (J. W.), "Impact of trunk reservation on elastic flow routing", in *Networking 2000,* March 2000.

[27] LIN (Y.-D.), HSU (N.-B.), HWANG (R.-H.), "QoS routing granularity in MPLS networks", *IEEE Comm. Mag.*, **46**, n° 2, pp. 58-65, 2002.

[28] NELAKUDITI (S.), ZHANG (Z.-L.), "A localized adaptive proportioning approach to QoS routing", *IEEE Comm. Mag.*, **46**, n° 2, pp. 66-71, 2002.

[29] AKAR (N.), HOKELEK (I.), ATIK (M.), KARASAN (E), "A reordering-free multipath traffic engineering architecture for Diffserv/MPLS networks", in *Proceedings of IEEE Workshop on IP Operations and Management,* Kansas City, Missouri, USA, pp. 107-113, 2003.

[30] NELAKUDITI (S.), ZHANG (Z. L.), TSANG (R. P.), "Adaptive proportional routing: A localized QoS routing approach" in *Proceedings of INFOCOM*, Anchorage, USA, 2000.

[31] FAUCHEUR (F. L.), (WU L.), B. DAVIE, S. DAVARI, P. VAANANEN, R. KRISHNAN, AND P. C. J. HEINANEN, "MPLS support of differentiated services", RFC 3270, May 2002.

[32] RAMAKRISHNAN (K.), FLOYD (S.), BLACK (D.), "Addition of explicit congestion notification (ECN) to IP", RFC 3168, Proposed standard, 2001.

[33] RAMAKRISHNAN (K. K.), FLOYD (S.), DAVIE (B.), "A proposal to incorporate ECN in MPLS", internet-draft draft-mpls-ecn-00.txt, June 1999.

[34] FLOYD (S.), JACOBSON (V.), "Random early detection gateways for congestion avoidance", *IEEE/ACM Transactions on Networking,* **1**, n° 4, pp. 397-413, 1993.

[35] RAI (I. A.), URVOY-KELLER (G.), BIERSACK (E. W.), "Analysis of LAS scheduling for job size distributions with high variance", in *Proceedings of ACM Sigmetrics*, pp. 218-228, 2003.

[36] GUO (L.), MATTA (I.), "The war between mice and elephants", in ICNP'2001: *The 9th IEEE international Conference on Network Protocols,* Riverside, CA, 2001.

[37] MCCANNE (S.), FLOYD (S.), "ns Network Simulator", Web page: http://www.isi.edu/nsnam/ns/, July 2002.