A Reordering-free Multipath Traffic Engineering Architecture for Diffserv-MPLS Networks

Nail Akar, Ibrahim Hokelek, Muammer Atik, and Ezhan Karasan
Department of Electrical and Electronics Engineering
Bilkent University, Ankara, Turkey TR–06800
Email: {akar,hokelek,matik,ezhan}@ee.bilkent.edu.tr

Abstract— **In this paper, we propose a novel traffic engineering architecture for IP networks with Multi Protocol Label Switching (MPLS) backbones. In this architecture, two (primary and secondary) Label Switched Paths (LSPs) are established among every pair of IP routers located at the edge of an MPLS cloud. Traffic between a source-destination pair is then split between the primary and secondary LSPs using an ABR-like explicit-rate feedback gathered from the network. Taking into consideration the packet reordering effect of packet-based load balancing schemes, we propose a novel traffic splitting mechanism that operates on a per-flow basis. We show using a variety of scenarios that deploying flow-based multipath traffic engineering not only provides significantly and consistently better throughput than that of a single path but is also void of any packet reorderings.**

I. Introduction

Internet traffic engineering is defined as the set of mechanisms that controls how traffic flows through a network so as to optimize resource utilization and network performance [1]. Traditional IP networks use shortest path hop-by-hop routing using simple link metrics 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. Several researchers have thus proposed the use of nontrivial link metrics for a given traffic demand matrix to improve routing performance where the link metrics are computed using a centralized optimization algorithm [3], [4]. This approach is effective particularly in case when the traffic matrix does not change significantly in short time scales [5].

In the alternative 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 overlay approach therefore replaces the hop-by-hop routing paradigm using shortest paths in traditional IP networks with explicit routing via which logical connections use any one of the feasible paths through the network. The emergence of Multi Protocol Label Switching (MPLS) technology provides mechanisms in IP backbones for explicit routing to facilitate traffic engineering [1],[6]. In MPLS backbones, one can use a constraint-based routing scheme so that traffic may be controlled to flow optimally through certain routes [7],[8].

[1]This project is supported in part by The Scientific and Technical Research Council of Turkey (TUBITAK) under projects EEEAG-101E025 and EEEAG-101E048

In the multipath 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 multipath traffic engineering is to increase the performance of the network by splitting the traffic between a source-destination pair among the multiple logical connections dedicated to that pair. In [5], resource management packets are transmitted periodically to the egress node, which returns them back to the ingress node. Based on the information in the returning resource management packets, the ingress node computes the one-way statistics like delay and loss for all the paths, and uses a gradient projection algorithm for load balancing. In [9] and [10], loading information on network links is assumed to be flooded using an enhanced interior gateway protocol and a load adjustment algorithm is proposed for performance improvement.

In this paper, we propose a novel traffic engineering architecture for best-effort IP networks with MPLS backbones. In this architecture, two MPLS bidirectional Traffic Engineering TE-LSPs, one being the primary LSP (P-LSP) and the latter being the secondary LSP (S-LSP), are established between each IP router pair located at the edge of an MPLS cloud. These two LSPs are link disjoint and routes of these paths are found using a variant of Dijkstra's algorithm which ensures the P-LSPs having less or equal number of hops than the S-LSPs.

Once the two LSPs are established, the next step is the development of an algorithm that splits the traffic demand between the two LSPs in a way to improve the throughput. Motivated by the ABR (Available Bit Rate) service category used for flow control in ATM networks, we propose to incorporate an ABR-like mechanism for traffic engineering purposes in MPLS networks. In this architecture, resource management (RM) packets (akin to RM cells in ATM) are used for extracting the available bit rate information from the MPLS network. The MPLS data plane, on the other hand, is envisioned to support differentiated services (diffserv) through per-class queuing with the gold, silver, and bronze classes dedicated to RM packets, data packets of P-LSPs, and data packets of S-LSPs, respectively. A strict priority per-class queuing scheme is used for scheduling the packets, with the highest priority assigned to RM packets, then to packets belonging to P-LSPs, and the lowest priority assigned to S-LSPs. We propose that MPLS switches, for each of its interfaces, run a

0-7803-8199-8/03/$17.00 © 2003 IEEE

separate instance of an ABR control algorithm for the silver and bronze classes and provide an explicit rate information back to sender for every LSP using that interface. Each IP router at the edge of the MPLS cloud maintains two queues (silver and bronze queues) per destination and these queues are drained at the rates dictated by the ABR-like feedback mechanism. In the flow-based traffic engineering architecture, we propose a traffic splitting algorithm in which individual traffic flows are identified and assigned to one of the two LSPs based on the average occupancies of the corresponding silver and bronze queues. Once such an assignment for a new flow is made, all packets of the same flow are forwarded using the same LSP (primary or secondary). This mechanism ensures that packet reordering would not take place at the receiving end of the corresponding flow. The method we propose for traffic splitting is called RER (Random Early Reroute), which is motivated by the RED (Random Early Discard) algorithm [11] used for active queue management in the Internet.

The three main contributions of the current paper are given below:

- Traffic engineering proposed in this paper is not only effective in long but also in short time scales due to the promptness of the explicit-rate feedback mechanism. Therefore, when unpredictable hot spots arise in the network, it is possible using this methodology to move the traffic around the hot spots in a distributed, automated, and timely fashion.
- If the alternative paths use more resources (or hops), some improperly designed load balancing algorithms may even lead to degradation in overall performance relative to a scheme that uses a single path for every node pair, known as the knock-on effect [12]. Strict prioritization in the data plane which is proposed in the paper ensures that the amount of traffic using secondary LSPs does not have a deteriorating impact on the explicit rates of the primary LSPs. Therefore, such a prioritization in the data plane promises to eliminate the knock-on problem.
- Packet reordering is known to have an adverse effect on the application level performance for some services [13]. The flow-based nature of the proposed traffic engineering architecture strictly controls the probability of packet reordering and in particular the mechanism can be made effectively reordering-free.

The rest of the paper is organized as follows. In Section 2, we present our traffic engineering architecture. We describe the simulation framework to verify the effectiveness of this approach and we present our numerical results in Section 3. Conclusions and future work are provided in the final section.

II. Architecture

Our proposed traffic engineering architecture is comprised of the following three components:

- Network architecture,
- Feedback mechanism,
- Flow-based traffic splitting,

which are studied next.

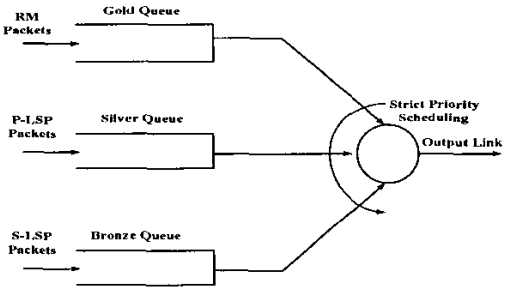

Fig. 1. Queuing architecture for MPLS switches

A. Network Architecture

As the network architecture, we propose an MPLS network that supports differentiated services (diffserv) with three Olympic services, namely the gold, silver, and bronze services. The gold service is dedicated for the Resource Management (RM) packets used for explicit rate feedback. The silver and bronze services are used by data packets in the way described below.

We establish two link disjoint LSPs between every source-destination pair of IP routers, i.e. the paths do not share a common link. For a particular source-destination pair, the primary LSP uses the minimum hop path found using Dijkstra's algorithm. When there is a tie in the algorithm, we break the tie randomly. The route for the secondary LSP is found by pruning the links used by the P-LSP and choosing one of the minimum hop paths in the remaining network graph. If the connectivity is lost after pruning links from the graph, the secondary LSP is not established. Other algorithms can also be used to find link-disjoint paths but a comparative analysis of these methods and their impact on overall throughput is left for future research. In our proposed TE architecture, data packets of P-LSPs and S-LSPs receive the silver and bronze services, respectively. We suggest to use the E-LSP (EXP-inferred-LSP) method for tagging the packets [14]. In this method, the three-bit experimental (EXP) field in the MPLS header is used to code the particular service a packet would receive.

A strict priority per-class queuing scheme is used for scheduling the packets, with the highest priority assigned to resource management packets, then to packets belonging to P-LSPs, and the lowest priority assigned to S-LSPs. The envisioned MPLS queuing architecture is given in Figure 1. To provide prompt feedback information, the highest service priority is given to the resource management packets. On the other hand, the incentive behind the isolation between the silver and bronze services by using strict priority scheduling in the data plane is to eliminate the so-called knock-on effect observed in load balancing algorithms [12],[15]. The knock-on effect refers to the phenomenon where using alternative paths by some sources force other sources whose minhop paths share links with these alternative paths to also use alternative paths. For a given source-destination pair, the primary LSP used by the silver service uses fewer hops than the secondary

LSP used by the bronze service because of the way we set up these LSPs. Therefore, strict priority scheduling is proposed for making sure that the performance of the silver service is not impacted by the load on the bronze queues.

B. MPLS Feedback Mechanism

The feedback information received from the network plays a crucial role in our TE approach. MPLS technology does not currently have a standards-based feedback mechanism, but we propose that a feedback mechanism very similar to the ABR service category in ATM networks, is to be used in MPLS networks as well. In this architecture, the ingress node of each LSP sends Resource Management (RM) packets (along with data packets) to the network, which are then returned back by the egress node to the ingress node. Similar to ABR, RM packets have Explicit Rate (ER), Congestion Indication (CI), and No Increase (NI) fields that can be used by the switches to provide feedback back to the sending sources. The MPLS switch runs a separate instance of an explicit rate algorithm to calculate the ER for the silver and bronze classes on all of its interfaces. In our experimental studies, we use a variable packet size extension of the ERICA ABR explicit rate algorithm [16] which is known to be max-min fair with proven transient performance.

For every LSP, RM packets are sent towards the network once in N_{RM} data packets. In order to be able to maintain the continuity of feedback, a new RM packet is always sent to the network if no data packets are generated in the last T_{RM} seconds. On the way to the destination, the RM packets are not modified. When the RM packet is on its way back from the destination to the source, three main operations are performed. Firstly, each switch sets the ER field to the minimum of the current ER value in the RM packet and the maximum rate the switch can support. Thus, every source should send at a rate no more than the ER calculated at its bottleneck point. Secondly, we assume a buffer size of B at the network buffers and the CI is set by the switch if the buffer occupancy of the switch is larger than B_{CI}. As a third operation, the NI is set if the buffer occupancy of the switch is between B_{NI} and B_{CI}. When the sending source receives the ER, CI and NI information, traffic will be sent using the standards-based ABR source behavior [17]. As in the ABR source behavior, the source node calculates Allowed Traffic Rate (ATR) using ER, CI, and NI fields of the RM packet and sends its traffic at a rate dictated by ATR. The algorithm given in Table I is used to calculate ATR. In this table, RDF and RIF correspond to Rate Decrease Factor and Rate Increase Factor, and MTR and PTR correspond to Minimum Traffic Rate and Peak Traffic Rate, respectively.

C. Flow-based Splitting

In this subsection, we describe how traffic is split among the primary and secondary LSPs. There are three stages in our proposed traffic splitting approach. The following operations will be performed in each stage for every new packet generated by a source Label Switch Router (LSR). In the first stage, we classify traffic flows and maintain a list which keeps track of each active flow. We note that traffic carried between the source and destination LSRs is an aggregation of multiple traffic flows generated by multiple users/applications. We assume that a hash function based on the quadruple <source IP addresses, source port, destination IP addresses, destination port> is applied to the incoming packet and the incoming packet is assigned to an existing flow in the list according to the outcome of the hash function. If the flow determined by the hashing function is not in the list, a new flow is inserted into the list; otherwise, the states of the active flows already in the list are updated. A flow is said to be active if a packet for that flow has arrived within the last T_{out} seconds. Otherwise, that flow is said to timeout and it is deleted from the list of active flows.

{
if CI is set
$ATR := ATR - ATR * RDF$
else if NI is not set
$ATR := ATR + RIF * PTR$
$ATR := min(ATR, PTR)$
}
$ATR := min(ATR, ER)$
$ATR := max(ATR, MTR)$

TABLE I
THE ABR SOURCE BEHAVIOR

In the second stage, a silver queue and a bronze queue are implemented on a per-destination basis. Both queues are drained using the ATR information calculated by using the standard ABR source behavior. In this stage, we decide which service queue each flow should join. When a packet arrives which is not associated with an existing flow, then a decision on how to forward the packets of this flow needs to be made at this stage. For this purpose, we compute the D_{P-LSP} and D_{S-LSP} delay estimates for the silver and bronze queues in the edge node, respectively. These delay estimates are calculated by means of dividing the corresponding queue occupancy by the drain rate ATR of that queue. The notation Δ denotes the average difference between the delay estimates which is updated at the epoch of nth packet arrival as follows:

$$\Delta_n = \gamma(D_{P-LSP} - D_{S-LSP}) + (1-\gamma)\Delta_{n-1}, \quad (1)$$

where γ is the averaging parameter to be set by the network operator. We also use the notation D_{max} to denote the maximum allowable delay for a packet through the MPLS backbone network. For flow management and routing, we propose the following policy that applies to the first packet of a new active flow:

$$\begin{array}{ll} \text{discard the packet,} & \text{if } D_{P-LSP}, D_{S-LSP} \geq D_{max} \\ \text{forward flow over P-LSP,} & \text{if } D_{P-LSP} < D_{max},\ D_{S-LSP} \geq D_{max} \\ \text{forward flow over S-LSP,} & \text{if } D_{P-LSP} \geq D_{max},\ D_{S-LSP} < D_{max} \\ \text{forward flow over S-LSP} & \\ \text{with probability } p(\Delta) \text{ or} & \\ \text{forward flow over P-LSP} & \\ \text{with probability } 1-p(\Delta), & \text{otherwise.} \end{array} \quad (2)$$

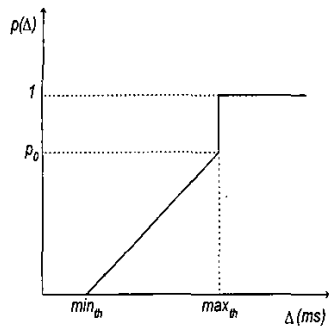

Fig. 2. The traffic splitting function $p(\Delta)$

Parameter	Value	Parameter	Value
N_{RM}	15	T_{RM}	200 ms
B_{CI}	1.2x10^6 bytes	B_{NI}	1x10^6 bytes
RDF	0.0625	RIF	0.125
MTR	0	PTR	Line Rate
L_f	10240 bytes	L_p	128 bytes
R_f	64 Kbps	p_0	1
min_{th}	30 ms	max_{th}	150 ms
D_{max}	180 ms	γ	0.3
B	2x10^6 bytes	T_{out}	300 ms

TABLE II

PROBLEM PARAMETERS USED THROUGHOUT THE SIMULATION STUDY (UNLESS OTHERWISE STATED)

where the probability $p(\cdot)$ is given in Figure 2. In this way, we adopt a variant of the Random Early Discard (RED) algorithm [11] used for active queue management. We call the policy we use for multipath traffic engineering a Random Early Rerouting (RER) policy. RED's goal is to control the average queue size whereas in multipath TE, it is the average delay difference between the two queues that is controlled by RER. Once an LSP is decided upon the arrival of the first packet of a new flow, all successive packets of the same flow will then be forwarded over the same LSP if delay constraints are satisfied. If the first packet of a new flow is discarded because of the delay constraint violation, then the policy (2) will have to apply to the second packet of the same flow to decide onto which LSP to forward the flow.

In the third stage, we employ per-class queuing at each physical port. The three stage traffic splitting mechanism is depicted in Figure 3 for an MPLS edge node with two destinations and two physical ports. In this example, we assume the primary and secondary LSPs for destination 1 use Port 1 and Port 2, and the primary and secondary LSPs for destination 2 use Port 2 and Port 1, respectively.

III. Simulation Study

In this section, we will present our simulation results to validate our proposed TE architecture. The platform we use is an event-driven packet-based MPLS simulator implemented using the Java programming language. The simulator allows us to specify the network topology and the traffic demand matrix. T_{ij} (in bps) denotes the long term average traffic demand between the nodes i and j and $T = \{T_{ij}\}$ denotes the traffic demand matrix. Motivated by recent research on flow-based Internet traffic modeling, in our simulation studies, the traffic demand between a source-destination pair is generated as a superposition of identical individual flows. The individual flow arrival process between the LSRs i and j is assumed to be Poisson with rate λ_{ij} (flows/s). The length for each flow is assumed to be deterministic and is denoted by L_f (in bytes). More general distributions for flow sizes are left for future research. Each flow consists of a sequence of packets of fixed length L_p (in bytes) where the rate of packet arrivals within a flow (i.e. flow rate) is exponentially distributed and its mean is denoted by R_f (bps). Since the flow sizes are fixed in this study, increasing the long term traffic demand between a source-destination pair leads to a larger average number of active flows at a given instant between the same pair.

The simulator reports Current Traffic Rate (CTR) for each LSP which is the traffic injection rate (in Mbps) from the source LSR towards the network. The number of total injected bytes throughout an averaging interval is first counted and this number is then divided by the averaging interval to find the CTR of this LSP at that particular time. Loss Rate (LR) for a given source-destination pair is defined as the ratio of the number of rejected/lost bytes of that pair to the number of total incoming bytes. We note that an incoming packet may either be rejected at the source node because of delay constraints or it can be dropped within the network because of congestion. The NLR (Network LR) is used for indicating the network-wide loss rate as a whole.

In our simulation studies, when transmitting a packet from source node, a sequence number is associated for that packet. If the sequence number of the currently arriving packet at the destination node is less than the sequence number of the previously arrived packet, then the current packet is counted as an "out of order packet". ROR (Reordering Rate) for a source-destination pair is then defined as the ratio of out of order packets to all packets belonging to this pair. NROR (Network ROR) denotes the network-wide reordering rate.

We will refer to the traffic engineering method described and proposed in the previous section as Flow-Based MultiPath Routing (FBMPR). When the policy 2 not only applies to the first packets of each new flow but to all packets without flow classification, we then use the term "Packet-Based MPR (PBMPR)" for the underlying method. We note that PBMPR does not take into consideration the packet reordering within a flow and therefore routes packets of the same flow independently over either the P-LSP or the S-LSP. The drawback of using PBMPR is that packet reordering within a TCP flow can falsely trigger congestion control mechanisms and cause unnecessary throughput degradation at the TCP level [13]. Single Path Routing (SPR) refers to the case when the S-LSP is absent in the system. SPR should be viewed as the MPLS counterpart of a flow controlled best-effort ATM network using the ABR service. In our simulation study, we compare and

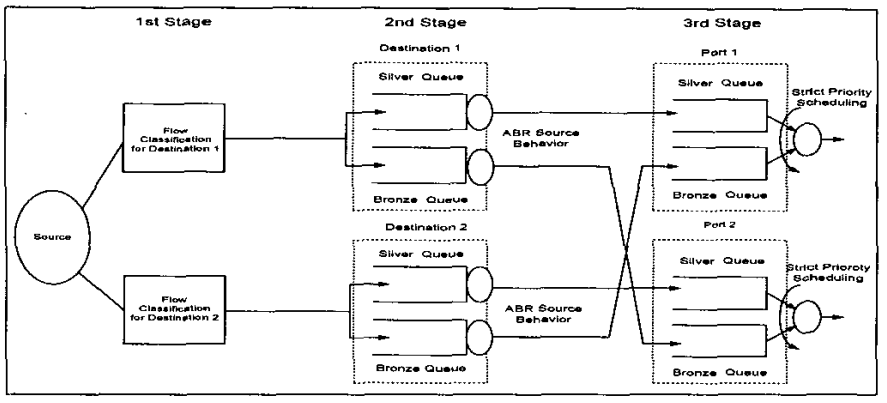

Fig. 3. MPLS edge node architecture and traffic splitting mechanism with two destinations and two physical ports

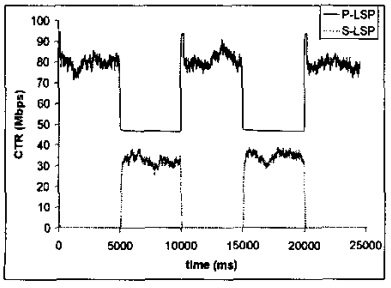

Fig. 4. Current traffic rate graph from node 1 to 0 when PBMPR is employed

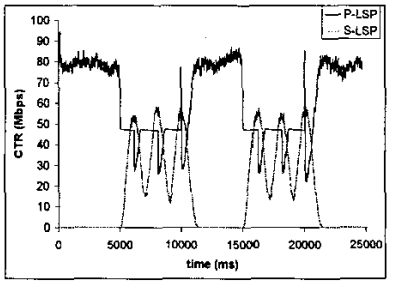

Fig. 5. Current traffic rate from node 1 to node 0 when FBMPR is employed

contrast the methods SPR, FBMPR, and PBMPR in terms of their network-wide loss rates (NLR) and reordering rates (NROR).

Unless otherwise stated, the algorithm parameters in Table II will be used throughout the simulation study.

A. Transient Performance

In this example, we will present how our TE method performs when network parameters change in short time scales. For this purpose, we use a simple three node ring topology where the nodes are numbered $0, 1, 2$. We assume a symmetric traffic demand of 80 Mbps between the nodes 0 and 1. The link from node 1 to node 0 is assumed to have a capacity that alternates between 100 Mbps and 50 Mbps. In particular, we assume a 100 Mbps capacity in the interval (0s,5s), 50 Mbps in the interval (5s,10s), and so on. For the traffic between node 1 and 0, we assume two LSPs, the P-LSP using the direct path, and the S-LSP using the indirect path via node 2. In Figure 4, the CTRs for the P-LSP and S-LSP from node 1 to node 0 are depicted when the method PBMPR is employed. Initially, the P-LSP is able to carry all traffic from node 1 to node 0. However, when the link capacity between node 1 and node 0 drops to 50 Mbps, this link is not able to carry the 80 Mbps traffic demand. In this case, the P-LSP carries about 47 Mbps traffic of the overall 80 Mbps traffic demand and the S-LSP carries about 33 Mbps. In Figure 5, we plot CTRfor the two LSPs when FBMPR is used. The CTRs for both PBMPR and FBMPR show similar average behavior when the network conditions vary, however, the FBMPR response is more oscillatory. For this oscillatory behavior, we note that all packets of the same flow are forwarded using the same LSP in FBMPR and the decision made for the first packet of a particular flow applies to all packets belonging to the same flow. Therefore there will be decision epochs when all the new flows are forwarded over the new LSP whereas the already active flows using the old LSP can still saturate the corresponding queue of the latter LSP. This phenomenon occasionally leads to underutilization in one queue and overutilization in the other and therefore oscillatory behavior. We view the oscillatory response as the price we pay for the elimination of packet reordering. We also monitored the long run network-wide loss rates for this scenario; we find NLR to be 0.478% for FBMPR and 0 when PBMPR is applied. However, we also note that NROR is zero

with FBMPR and 10.350% for PBMPR. In our simulations, we do not attempt to quantify the effect of the packet reordering rate on the application-level throughput but it has been noted in the literature that this level of NROR may cause severe degradation in the user-perceived performance [13].

B. Elimination of the Knock-On Effect

The goal of this simulation study is to demonstrate the knock-on effect and its implications on overall throughput. For this purpose, we use seven diffserv-MPLS nodes interconnected to each other using a ring topology. We assume nodes numbered as $0, 1, \ldots, 6$ in the clockwise direction. Each link is bidirectional and has a capacity of 155 Mbps in both directions. We assume source-destination pairs having 15 Mbps traffic demand if their shortest paths are in the clockwise direction, for example source node 0 and destination node 1. For a given pair, we assume a traffic demand of 35 Mbps if the shortest path for this pair is in the counterclockwise direction, e.g., source node 0 and destination node 4. We compare and contrast three mechanisms, the first one being SPR where a single path is used. The second mechanism is FBMPR where the P-LSPs and S-LSPs are differentiated using per-class queuing in the data plane. The final mechanism is denoted by FIFO-FBMPR where the P-LSPs and S-LSPs share a single FIFO (First-In-First-Out) queue in the data plane over which a single instance of an ABR algorithm runs to find the explicit rate of all LSPs sharing the queue.

From Table III, we find network-wide loss rate NLR to be 20.397% for SPR which only uses the shortest path for sending traffic. When we use FIFO-FBMPR which amounts to making no differentiation among the two silver and bronze services, NLR increases up to 39.192% despite the load balancing effort. This example demonstrates that when multiple paths are used, improperly designed load balancing algorithms can even deteriorate the overall performance due to the knock-on effect. The sharp difference between the two methods FIFO-FBMPR and FBMPR is that for this particular ring topology, there is a significant difference between the average number of hops used by the P-LSPs and the S-LSPs. Therefore, a max-min fair algorithm (e.g., ABR), while trying to be fair to all connections without a consideration of their hop lengths, favors the use of S-LSPs. However, S-LSPs use significantly more resources than P-LSPs in this ring topology and should be resorted to only if the P-LSPs cannot carry all the traffic. This is though achievable using the proposed architecture FBMPR which results in a network-wide loss rate NLR 13.452%. In this architecture, the P-LSPs have strict priority over the S-LSPs and the S-LSPs use only the remaining capacity from the use of the P-LSPs. Therefore, the CTRs of the P-LSPs are not impacted adversely by the number of S-LSPs.

C. Impact of Flow Model Parameters

In this simulation study, we study the impact of flow model parameters on the overall performance. The algorithm FBMPR is tested for different flow model parameters in a publicly available test network which is available at the URL:

Method	NLR (%)
SPR	20.397
FIFO-FBMPR	39.192
FBMPR	13.452

TABLE III
NLRS FOR THE 7 NODE RING TOPOLOGY.

Method	NLR (%)	NROR (%)
SPR	5.559	-
FBMPR	0.783	-
PBMPR	0.445	9.169

TABLE IV
NLR AND NROR FOR THE HYPOTHETIC DENSE US TOPOLOGY

`www.fictitious.org/omp` (called the hypothetic dense US topology). To reduce the simulation run-time requirements, we scaled down both the link capacities and the traffic demands for this test network which consists of 12 nodes and 19 links. Unlike the original test network, we use $c_1 = 45$ Mbps (as opposed to the original 155 Mbps) bidirectional links except for two links which have $c_2 = 2c_1 = 90$ Mbps capacity in both directions. Table IV gives the NLR values obtained through simulations for the hypothetic dense US topology for all the studied methods. We show that the network-wide loss rate can considerably be reduced by using multipath methods. We note that the NLR difference between PBMPR and FBMPR is small whereas NROR is 9.169% for PBMPR while there is no reordering observed with FBMPR. This demonstrates the efficiencies in making traffic splitting decisions on a per-flow basis as opposed to packet-based traffic engineering.

Next we study the effect of the flow arrival rates on the performance of FBMPR for the hypothetic dense US topology. For this purpose, we scale the link capacities c_1, c_2, and the traffic demands T_{ij} together by a multiplicative constant so as to vary λ_{ij}. We note that the flow arrival rates are related to the traffic demands by

$$\lambda_{ij} = T_{ij}/L_f \tag{3}$$

We compare and contrast the three methods SPR, PBMPR, and FBMPR when the link capacities are varied. The results are depicted in Figure 6. The x-axis of Figure 6 is the logarithm of the capacity parameter c_1 in Mbps which is the capacity of the 17 out of 19 links in the hypothetic dense US topology. We observe that the NLRs for PBMPR and SPR decrease with increasing link capacities which can be explained through the statistical multiplexing concept. The same effect can also be seen in the FBMPR case and there is further improvement in terms of NLR; the NLR for FBMPR approaches to that of the PBMPR case when the link capacities are increased and they are very close when the link capacity c_1 is larger than 45 Mbps. We therefore conclude that with the average flow sizes set to 10240 bytes as indicated in Table II, and when the link traffic demands are rich enough (i.e. $c_1 \geq 45$

Mbps), FBMPR works as good as its packet-based counterpart PBMPR while preserving the packet orders. This phenomenon can be explained by observing the relationship between flow-based traffic splitting and the flow arrival rates; the larger the flow arrival rates the more the number of decision epochs to split traffic. Increasing the control frequency then improves the performance of the controlled system. If, on the other hand, there are few flow arrivals in unit time, then fewer traffic splitting decisions would take place leading to over- or under-utilization of the associated silver and bronze queues maintained at the edge LSRs.

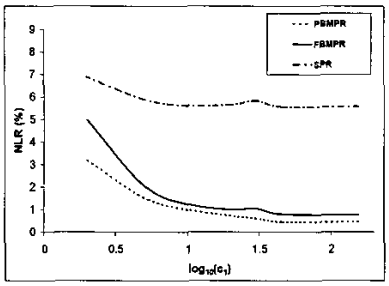

Fig. 6. NLR vs the logarithm of the link capacity parameter c_1

In our model the flow arrival rate also depends on the flow size L_f through Eqn. (3). We next study the impact of the flow sizes on the NLR while fixing the link capacities $c_1 = 45$ Mbps and $c_2 = 90$ Mbps. The results are given in Figure 7. We observe increasing network-wide loss rates with increasing flow sizes. It is clear that the smaller the flow sizes, the better the proposed TE method performs and the performance will approach to that of the PBMPR mechanism in terms of NLR.

IV. Conclusions

In this paper, we propose a multipath traffic engineering methodology in IP networks using MPLS backbones. A challenging requirement in such problems is the preservation of packet ordering of individual flows in IP networks. The proposed traffic engineering architecture in this paper handles this problem using flow-based multipath routing. In flow-based routing, each flow can be forwarded independently over the primary and the secondary LSPs so that packet reordering would not take place. The performance of the proposed TE architecture is shown to depend heavily on network speeds and the flow sizes. We conclude that the architecture is applicable to flow-rich national/regional backbone provider scenarios where the average number of flow arrivals in unit time is large enough to validate a flow-based traffic engineering approach. Future work will consist of using more realistic traffic models for the Internet and their implications on reordering-free multipath traffic engineering.

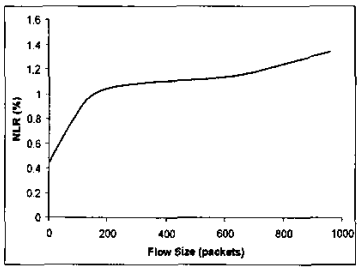

Fig. 7. NLR vs the flow size

References

[1] D. O. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao, "Overview and principles of Internet traffic engineering," IETF Informational RFC-3272, May 2002.

[2] Y. Wang, Z. Wang, and L. Zhang, "Internet traffic engineering without full mesh overlaying," in *Proceedings of INFOCOM*, Anchorage, USA, 2001.

[3] L. Berry, S. Kohler, D. Staehle, and P. Trangia, "Fast heuristics for optimal routing in IP networks," Universitat Wurzburg Institut fur Informatik Research Report Series, Tech. Rep. 262, July 2000.

[4] A. Elwalid, C. Jin, S. Low, and I. Widjaja, "MATE: MPLS adaptive traffic engineering," in *Proceedings of INFOCOM*, 2001, pp. 1300–1309.

[5] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol label switching architecture," RFC 2481, January 2001.

[6] S. Plotkin, "Competetive routing of virtual circuits in ATM networks," *IEEE Jour. Selected Areas in Comm.*, pp. 1128–1136, 1995.

[7] M. Kodialam and T. V. Lakshman, "Minimum interference routing with applications to MPLS traffic engineering," in *Proceedings of INFOCOM*, Tel-Aviv, Israel, March 2000.

[8] C. Villamizar, "OSPF Optimised Multipath (OSPF-OMP)," Internet Draft <draft-ietf-ospf-omp-02.txt>, 1998.

[9] ——, "MPLS Optimised Multipath (MPLS-OMP)," Internet Draft <draft-ietf-mpls-omp-01.txt>, 1999.

[10] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397–413, 1993.

[11] S. Nelakuditi, Z. L. Zhang, and R. P. Tsang, "Adaptive proportional routing: A localized QoS routing approach," in *Proceedings of INFOCOM*, Anchorage, USA, 2000.

[12] M. Laor and L. Gendel, "The effect of packet reordering in a backbone link on application throughput," *IEEE Network Magazine*, vol. 16, no. 5, pp. 28–36, 2002.

[13] F. L. Faucheur, L. Wu, B. Davie, S. Davari, P. Vaananen, R. Krishnan, and P. C. J. Heinanen, "MPLS support of differentiated services," RFC 3270, May 2002.

[14] F. P. Kelly, "Routing in circuit switched networks: Optimization, shadow prices and decentralization," *Advances in Applied Probability*, vol. 20, pp. 112–144, 1988.

[15] S. Kalyanaraman, R. Jain, S. Fahmy, R. Goyal, and B. Vandalore, "The ERICA switch algorithm for ABR traffic management in ATM networks," *IEEE/ACM Transactions on Networking*, vol. 8, no. 1, pp. 87–98, 2000.

[16] "ATM Forum Traffic Management Specification Version 4.1," Specification af-tm-0121.000, March 1999.