Computer Communications 26 (2003) 388–403

computer communications

www.elsevier.com/locate/comcom

Invited Paper

A new traffic engineering manager for DiffServ/MPLS networks: design and implementation on an IP QoS Testbed

I.F. Akyildiz[a,*], T. Anjali[a], L. Chen[a], J.C. de Oliveira[a], C. Scoglio[a], A. Sciuto[b], J.A. Smith[b], G. Uhl[b]

[a] *Broadband and Wireless Networking Lab, School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA*
[b] *NASA Goddard Space Flight Center, Greenbelt, MD 20771, USA*

Received 12 May 2002; accepted 14 May 2002

Abstract

In a multi-service network, different applications have varying QoS requirements. The IETF has proposed the DiffServ architecture as a scalable solution to provide Quality of Service (QoS) in IP Networks. In order to provide quantitative guarantees and optimization of transmission resources, DiffServ mechanisms should be complemented with efficient traffic engineering (TE) mechanisms, which operate on an aggregate basis across all classes of service. The MultiProtocol Label Switching (MPLS) technology is a suitable method to provide TE, independent of the underlying layer2 technology. Currently, the combined use of Differentiated Services (DiffServ) and MPLS is a promising technique to provide Quality of Service (QoS), while efficiently exploiting network resources. In this paper, TEAM, an automated manager for DiffServ/MPLS networks is introduced and its design. The design and implementation details are discussed.

Keywords: DiffServ; MultiProtocol Label Switching; Traffic Engineering; Automated Network Management; Testbed

1. Introduction

One of the most actively studied open issues in several areas of communication networks is the problem of bandwidth reservation and management. Load balancing is another important issue. It is desirable to avoid portions of the network becoming over-utilized and congested, while alternate feasible paths remain underutilized. These issues are addressed by Traffic Engineering (TE). The MultiProtocol Label Switching (MPLS) technology is a suitable method to provide TE, independent of the underlying layer2 technology [1,2]. MPLS per se cannot provide service differentiation, which brings up the need to complement it with another technology capable of providing such feature: DiffServ. DiffServ is becoming prominent in providing scalable network designs supporting multiple classes of services. When optimization of resources is sought, DiffServ mechanisms need to be complemented by existing MPLS traffic engineering mechanisms, which then becomes DiffServ-aware Traffic Engineering (DS-TE) [3], currently under discussion in Internet Engineering Task Force (IETF). In this case, DiffServ and MPLS both provide their respective benefits. It is obvious that such future networks cannot be managed manually when all new protocols are implemented. Therefore, automated managers need to be developed to simplify network management and to engineer traffic efficiently [4].

With the objective to studying and researching the issues mentioned above, we assembled an IP QoS testbed in our laboratory (http://www.ece.gatech.edu/research/labs/bwn). The testbed is a high-speed top-of-the-line mix of highly capable routers and switches for testing DiffServ and MPLS functionalities. During our experiences with the testbed, we realized the need for an improved set of algorithms for network management and also an integrated architecture for an automated network manager. This led to the design and implementation of Traffic Engineering Automated Manager (TEAM).

Individual problems addressed by TEAM may already have been considered, but to the best of our knowledge, an integrated solution does not exist. We are developing TEAM as a centralized authority for managing a

* Corresponding author. Tel.: +1-404-894-5141; fax: +1-404-894-7883.
E-mail addresses: ian@ece.gatech.edu (I.F. Akyildiz), tricha@ece.gatech.edu (T. Anjali), leochen@ece.gatech.edu (L. Chen), jau@ece.gatech.edu (J.C. de Oliveira), caterina@ece.gatech.edu (C. Scoglio), asciuto@rattler-e.gsfc.nasa.gov (T. Sciuto), jsmith@rattler-e.gsfc.nasa.gov (J.A. Smith), uhl@rattler-e.gsfc.nasa.gov (G. Uhl).

0140-3664/03/$ - see front matter © 2002 Elsevier Science B.V. All rights reserved.
PII: S0140-3664(02)00157-3

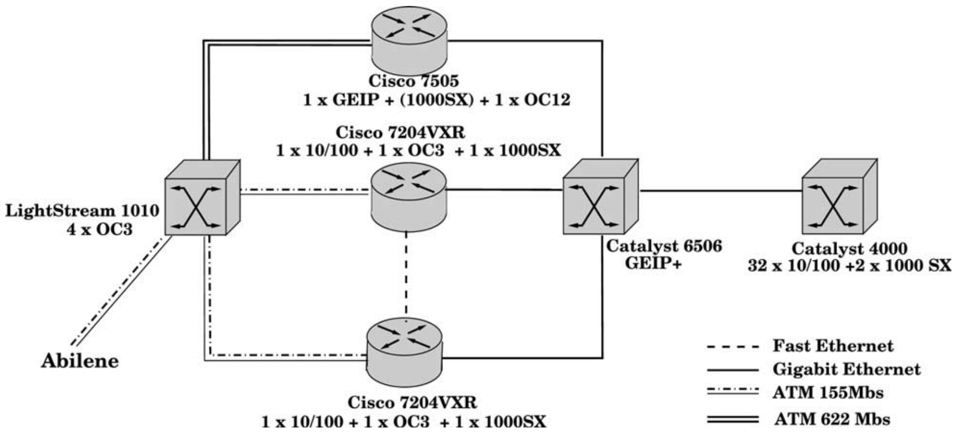


Fig. 1. BWN-Lab IP QoS Testbed.

DiffServ/MPLS domain. Our proposal is a comprehensive study that describes practical solutions for MPLS network management. TEAM is responsible for dynamic bandwidth and route management. Based on the network state, TEAM takes the appropriate decisions and reconfigures the network accordingly. TEAM is designed to provide a novel and unique architecture capable of managing large scale MPLS/DiffServ networks.

The structure of the rest of the paper is as follows. In Section 2, we enlist the components of our IP QoS testbed. The following section, Section 3, includes a design description of TEAM along with comparison with other MPLS network management tools. In Section 4 we present our proposed algorithms for bandwidth management, namely Label Switched Path (LSP) setup and dimensioning, LSP preemption and LSP capacity allocation. Section 5 discusses the route management aspects of TEAM, followed by the description of the measurement tool employed by TEAM in Section 6. In Section 7, we present the implementation details for TEAM. Finally, we conclude the paper in Section 8.

2. Physical testbed

We have a full-fledged Next Generation Internet routers physical testbed in our Broadband and Wireless Networking Laboratory (BWN-Lab) at Georgia Institute of Technology, equipped with DiffServ capable routers and switches manufactured by Cisco. We have a Cisco 7500 router with a Gigabit Ethernet card and a layer 3 switch Catalyst 6500 with an enhanced Gigabit Ethernet card and also other routers and switches. These routers and switches are widely deployed in the backbones of current high-speed networks. All our routers support MPLS and a variety of QoS technologies such as RSVP and DiffServ. Currently all devices have SNMP enabled and different measurement tools like MRTG and Netflow are being evaluated. During the analysis of MRTG, a new improved version of the tool was developed by our group: MRTG++ (described in Section 6). It allows managers to monitor traffic with up to 10 s interval, rather than the original 5 min sampling of MRTG, providing fine-grained detail about the state of the network. Our testbed is connected via an OC3 link to Abilene, the advanced backbone network of Internet2 society, that supports development and deployment of new applications. We perform end-to-end QoS performance experiments with NASA Goddard in Maryland and NASA Ames in California. The objective of the experiments is to study the advantages and disadvantages of using DiffServ in a heterogeneous traffic environment. The traffic under study is generated from voice, video and data sources. Fig. 1 shows a schematic of our testbed assembly. Next, we present a brief description of the experiments we performed.

2.1. DiffServ experiments

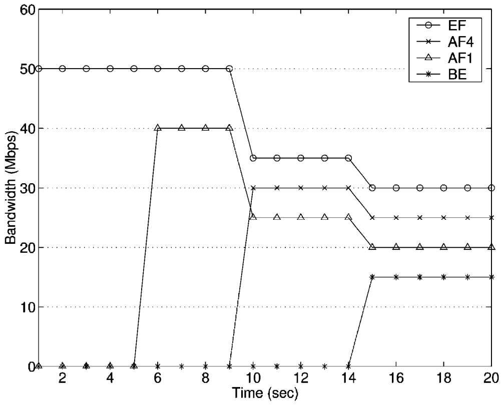
We validated the use of ALTQ and CAR as the scheduling and policing mechanisms, respectively, for providing DiffServ [5]. Furthermore, we investigated the use of Class-Based Weighted Fair Queueing (CBWFQ) as another scheduling mechanism, as explained next. Four UDP flows are separated into four different DiffServ PHBs and CBWFQ applied at the intermediate hops according to Fig. 2(a). In Fig. 2(b), we show the result of the experiments and the efficiency of the operation of CBWFQ. In the case of no congestion, the flows get the desired bandwidth. In case of congestion, the flow throughputs are reduced to fairly share the link according to the minimum guarantees provided by CBWFQ. With these experiments, we concluded that DiffServ is a suitable technology for providing QoS on an aggregated basis. DiffServ can be enhanced with the TE capabilities of MPLS to provide end-to-end QoS.

2.2. MPLS experiments

We conducted some experiments to show the efficiency of TE provided by MPLS technology. Various TCP and UDP flows were sent from router 1 to router 2. In the absence of TE, all the flows chose the shortest path between

Class (UDP)	Time of Start	Requested bw (Mbps)	Guaranteed bw (Mbps)
EF	0	50	30
AF4	9	40	25
AF1	5	40	20
BE	14	40	None

(a)



(b)

Fig. 2. DiffServ experiments.

the two routers, in effect starving the TCP flows for bandwidth. If LSPs are created between the two routers, but TCP and UDP flows are still forced to share bandwidth, again TCP flows starve. If the flows are separated into different LSPs by themselves, they do not interfere with each other. For the example case shown in Fig. 3, the UDP flows of 40 Mbps are sharing Tunnel 1 whereas the TCP flows are occupying the Tunnels 2 and 3 individually.

Running experiments on our physical testbed, we discovered shortcomings in some of the current DiffServ-MPLS functionalities. In particular, the LSP setup is manual with human intervention and no optimal decision policy exists. The LSP preemption policy is based purely on priority, which leads to bandwidth wastage. The LSP capacity is manually set to the sevice level agreement values plus a small cushion for bandwidth guarantee. This leads to bandwidth wastage. Also LSPs can be routed explicitly, thus creating the need for a policy to optimize the routing. Based on these revelations, we investigated and proposed solutions to each of these issues [6–11]. Furthermore, there exists a need to obtain a balance between the objectives of efficient resource utilization and efficient QoS provisioning. A manager entity for the whole domain is best-suited to provide such a balance. Therefore, we realized the potential for an integrated automated MPLS network manager and proposed TEAM.

3. Team Traffic Engineering automated Manager

The design and management of an MPLS network is a fundamental key to the success of the QoS provisioning. Many problems need to be solved such as LSP dimensioning, set-up/tear-down procedures, routing, adaptation to actual carried traffic, preemption, initial definition of the network topology, etc. To illustrate the inter-relations of the listed problems for MPLS network management, let us consider the scenario where network planning methods have provided an initial topology of the MPLS networks which needs to be adapted to the changing traffic demands. Possible events could be arrival of a request for LSP setup based on the SLS agreements or arrival of a bandwidth request in the MPLS network. The first event can be handled by the combined use of three of our proposed methods in the order: LSP routing, LSP preemption, and LSP capacity allocation. The LSP routing aims to find the route on the physical network over which the LSP will be routed. LSP preemption decides if any existing LSPs can be preempted on the route to make way for the new LSP if there is not enough available bandwidth. LSP capacity allocation method tries to fine-tune the LSP capacity allocation to avoid unused reserved bandwidth. The second event of arrival of a bandwidth request triggers the LSP setup and dimensioning which may in turn trigger the LSP creation steps of routing, preemption and capacity allocation.

The above-mentioned problems can be handled at two different time scales, namely short-term and long-term. *Short-term Network Management* (minutes, hours) is based on the current state of the network. Dynamic methods for re-dimensioning and routing are designed to provide efficient resource utilization and load balance for MPLS networks. These methods perform real-time adaptation to the actual network state (Bandwidth and Route Management). *Long-term Network Management* (months, years) is used to provide an initial design and dimension of the network topology based on the predicted utilization of the network. TEAM performs both short-term and long-term management. In the following sections, we present the various approaches developed by us for short-term management.

Several TE servers have been already proposed in literature. The RATES server [12] is a software system developed at Bell Laboratories for MPLS traffic engineering and is built using centralized paradigm. RATES communicate only with the source of the route and spawns off signaling from the source to the destination for route setup. RATES views this communication as a policy decision and therefore uses Common Open Policy Service (COPS) [13] protocol. RATES uses a relational database as its information store.

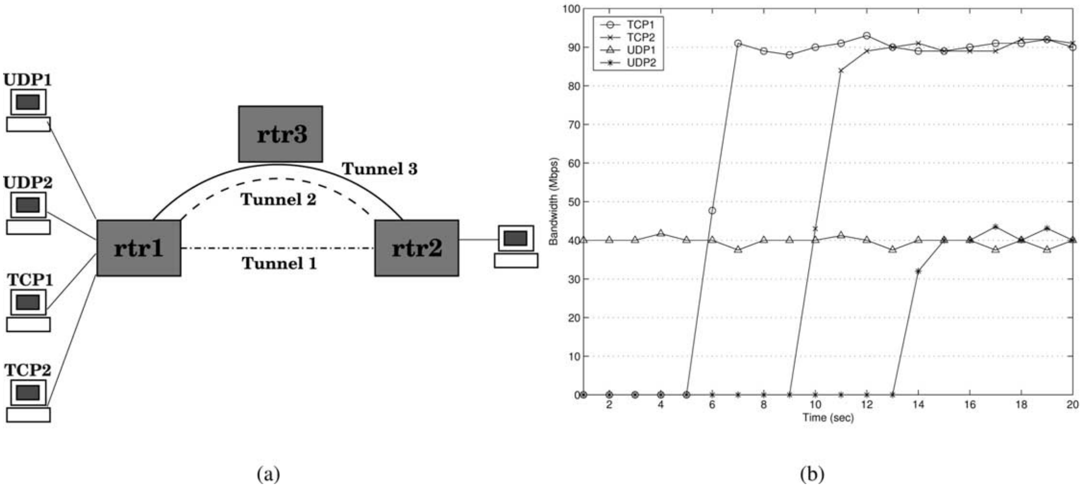


Fig. 3. MPLS experiment setup and results.

RATES implements Minimum Interference Routing Algorithm (MIRA) [14], based on the concept of minimum interference, to route LSPs. It consists of the following major modules: explicit route computation, COPS server, network topology and state discovery, dispatcher, GUI, an open Application Programming Interface, data repository, and a message bus connecting these modules. Summarizing, RATES is a well-designed TE tool, but TE is only performed by the routing of bandwidth guaranteed LSPs.

Another state dependent TE mechanism to distribute network load adaptively is suggested in Ref. [15]. MATE assumes that several explicit LSPs have been established between an ingress and egress node in an MPLS domain using a standard protocol like RSVP-TE. The goal of the ingress node is to distribute the traffic across the LSPs. It is important to note that MATE is intended for traffic that does not require bandwidth reservation, with best-effort traffic being the most dominant type. Since the efficacy of any state-dependent TE scheme depends crucially on the traffic measurement process, MATE requires only the ingress and the egress nodes to participate in the measurement process. Based on the authors' experience, available bandwidth was considered difficult to be measured, so packet delay and loss have been selected for measurement purposes. The network scenario for which MATE is suitable is when only a few ingress-egress pair are considered. In fact for a network with N nodes, having x LSPs between each pair of nodes, the total number of LSP is in the order of xN^2 which can be a large number. Furthermore it is not designed for bandwidth guaranteed services.

TEQUILA [16] is a European collaborative research project looking at an integrated architecture and associated techniques for providing end-to-end QoS in a DiffServ-based Internet. In TEQUILA, a framework for Service Level Specification has been produced, an integrated management and control architecture has been designed and currently MPLS and IP-based techniques are under investigation for TE. The TEQUILA architecture includes control, data and management planes. The management plane aspects are related to the concept of Bandwidth Broker (BB) and each Autonomous System should deploy its own BB. The BB includes components for monitoring, TE, SLS management and policy management. The TE subsystem is further decomposed into modules of traffic forecast, network dimensioning, dynamic route management, and dynamic resource management. The MPLS network dimensioning is based on the hose model which is associated with one ingress and more than one egress node. The dynamic route management module considers: (a) setting up the forwarding parameters at the ingress node so that the incoming traffic is routed to LSPs according to the bandwidth determined by network dimensioning, (b) modifying the routing according to feedback received from network monitoring and (c) issuing alarm to network dimensioning in case available capacity cannot be found to accommodate new connection requests. The dynamic resource module aims at ensuring that link capacity is fairly distributed among the traffic classes sharing a link, by appropriately setting buffer and scheduling parameters. TEQUILA architecture is very interesting and shows a similar approach for MPLS networks design and management compared to TEAM. However, the algorithms and techniques to be implemented in TEQUILA are not defined in detail at the moment, and their quantitative evaluation has not been carried out.

The use of MPLS for TE, quality of service and virtual private networks has been decided at GlobalCenter [17], one of the 10 largest ISPs in the USA. Approximately 200 routers participate in the MPLS system. Since a full meshed network would result in an MPLS system of about 40,000 LSPs, it is decided to deploy a hierarchical MPLS system of two layers of LSPs. To deploy an MPLS system for TE, the following procedure is proposed based on the network operator experience: (a) Statistics collection for traffic utilizing LSPs, (b) LSP deployment with bandwidth constraints, (c) Periodic update of LSP bandwidth and (d) Off-line constraint based routing. To provide QoS, MPLS is used in combination with the DiffServ architecture. Since it

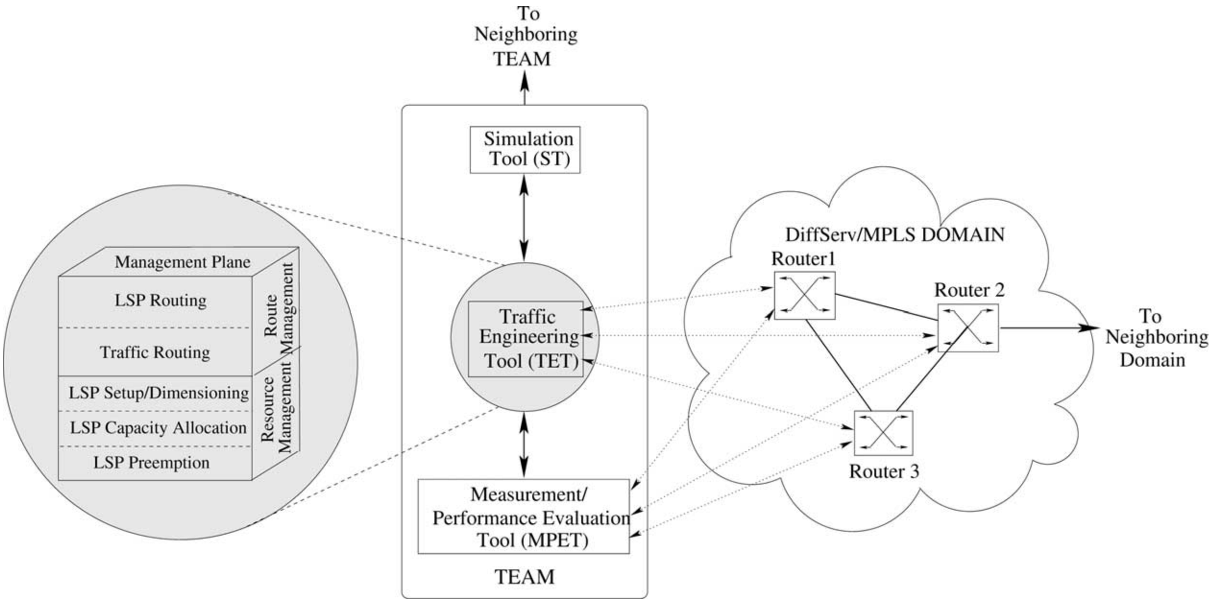


Fig. 4. TEAM: traffic engineering automated manager.

is desirable to use different LSPs for different classes, the physical network is divided into multiple virtual networks, one per class. These networks can have different topology and resources. The end effect is that premium traffic can use more resources. Many tools are needed for designing and managing these virtual networks. The use of MPLS for TE and QoS decided by an important ISP is the confirmation that MPLS is a very promising technique even from a business point of view. The solution provided by TEAM is in line with the QoS architecture defined by GlobalCenter.

The architecture of our TEAM is shown in Fig. 4. As shown, TEAM will have a central server, the Traffic Engineering Tool (TET), that will be supported by two tools: Simulation Tool (ST) and Measurement/Performance Evaluation Tool (MPET). The TET and the MPET will interact with the routers and switches in the domain. The MPET will provide a measure of the various parameters of the network and routers like the available bandwidth, overall delay, jitter, queue lengths, number of packets dropped in the routers, etc. This information will be input to the TET. Based on this measured state of the network, the TET decides the course of action, such as to vary the capacity allocated to a given LSP or to preempt a low priority LSP to accommodate a new one, or to establish the path for a traffic requiring a specified QoS. The TET also will automatically implement the action, configuring accordingly the routers and switches in the domain. Whenever required, the TET can consolidate the decision using the ST. The ST will simulate a network with the current state of the managed network and apply the decision of the TET to verify the achieved performance. The TET management tasks include Bandwidth Management (LSP setup/dimensioning, LSP preemption, LSP capacity allocation) and Route Management (LSP routing), as shown in Fig. 4. In the following sections, we provide details of our already proposed and implemented algorithms.

The prototype of TEAM is being implemented on our Testbed. The TEAM protocol stack will reside on an end-host. MPET and TET will interact with the routers and switches of the domain controlled by the TEAM. The interfaces to the routers will be based on Simple Network Management Protocol (SNMP), Common Open Policy Service Protocol for Provisioning (COPS-PR), and Command-Line Interfaces (CLI). SNMP is an open-source protocol to retrieve the Management Information Bases (MIBs) and COPS-PR for Policy Information Bases (PIBs) from the routers. They can also make router configuration changes. To ensure security, we will use SNMPv3. MPET is currently based on MRTG++, our enhanced version of MRTG, and retrieves traffic rate information on every interface with up to 10 s interval.

4. Bandwidth management

Bandwidth management deals with managing the resources of an MPLS network in an efficient manner to meet QoS requirements. It comprises of LSP setup and dimensioning (Section 4.1), preemption (Section 4.2), and capacity allocation (Section 4.3). In the event of an LSP setup request, the LSP preemption and capacity allocation functions are invoked. In the case of bandwidth reservation request, LSP setup and dimensioning procedures are triggered which may in turn initiate the LSP creation steps of routing, preemption and capacity allocation.

4.1. LSP setup and dimensioning

An important aspect in designing an MPLS network is to determine an initial topology and to adapt it to the traffic load. A topology change in an MPLS network occurs when a new LSP is created between two nodes. The LSP creation involves determining the route of the LSP and the according resource allocation to the path. A fully connected MPLS network can be used to minimize the signaling. The objective of our algorithm is to determine when an LSP should be created and how often it should be re-dimensioned.

4.1.1. Related work

Two different approaches, *traffic-driven* and *topology-driven* [1], can be used for MPLS network design. In the *traffic-driven* approach, the LSP is established on demand according to a request for a flow, traffic trunk or bandwidth reservation. The LSP is released when the request becomes inactive. In the *topology-driven* approach, the LSP is established in advance according to the routing protocol information, e.g. when a routing entry is generated by the routing protocol. The LSP is maintained as long as the corresponding routing entry exists, and released when the routing entry is deleted. The advantage of the traffic-driven approach is that only required LSPs are set-up; while in the topology-driven approach, the LSPs are established in advance even if no data flow occurs.

A simple LSP set-up policy based on the traffic-driven approach has been proposed in Ref. [18], in which an LSP is established whenever the number of bytes forwarded within 1 min exceeds a threshold. This policy reduces the number of LSPs in the network; however, it has very high signaling costs and needs high control efforts for variable and bursty traffic as in the case of a fully connected network.

A threshold-based policy for LSP set-up is suggested in Ref. [6]. It provides an on-line design for MPLS network depending on the current traffic load. The proposed policy is a traffic-driven approach and balances the signaling and switching costs. By increasing the number of LSPs in a network, the signaling costs increase while the switching costs decrease. In the policy, LSPs are set-up or torn-down depending on the actual traffic demand. Furthermore, since a given traffic load may change depending on time, the policy also performs filtering in order to avoid oscillations, which may occur in case of variable traffic. The policy is 'greedy' in the sense that it tries to minimize instantaneous costs, rather than trying to find the optimal solution.

An approach to setup inter-domain LSPs is given in Ref. [19]. In order to connect LSPs in different domains, the use of specialized bandwidth broker agent called Bandwidth Management Point (BMP) is proposed. The architecture and the signaling protocols to establish inter-domain LSPs and distribute inter-domain labels are presented. The approach assumes that the decision for LSP setup has already been made based on the nominal capacity allocation from the SLS agreements. It does not perform decisions about the need for the LSP and whether it is cost-effective.

The network design scheme in Ref. [20], utilizes an online recursive estimation of effective bandwidths for bandwidth allocation and buffer dimensioning under QoS requirements on queuing delay and packet loss probability. The scheme can be extended for MPLS networks but the authors have performed traffic demand estimation for each source destination pair based on inverse Erlang-B formula followed by optimal allocation algorithm for the estimation. This may not be true for the Internet traffic in general.

Class-level and path-level aggregation based network designs for statistical end-to-end delay guarantees are presented and compared in Ref. [21]. The authors have shown that the class-level aggregation results in a better statistical multiplexing gain because path-level aggregation for a fully meshed path network is viable only for small networks. In our approach, we will show that even path-level aggregation is beneficial for large networks if they are not fully meshed.

4.1.2. Optimal and sub-optimal solution

Our proposed optimal and sub-optimal decision policies [7] for LSP set-up are based on continuous-time Markov Decision Process (MDP) [22] theory. The cost functions for the MDP theory have been defined in Ref. [7]. Following the theory of MDPs, we define the expected infinite-horizon discounted total cost, $v^{\pi}(S_0)$, with discounting rate α, given that the process occupies state S_0 at the first decision instant and the decision policy is π by:

$$v_{\alpha}^{\pi}(S_0) = E_{S_0}^{\pi}\left\{\sum_{m=0}^{\infty} \mathrm{e}^{-\alpha t_m}\left[W_{\mathrm{sign}}(S_m, a) + \int_{t_m}^{t_{m+1}} \mathrm{e}^{-\alpha(t-t_m)}[w_b(S_m, a) + w_{sw}(S_m, a)]\mathrm{d}t\right]\right\}. \quad (1)$$

where $t_0, t_1, \ldots$ represent the times of successive instants when events occur and $W_{\mathrm{sign}}(S_m, a)$ represents the fixed part of the cost incurred whereas $[w_b(S_m, a) + w_{sw}(S_m, a)]$ represents the continuous part of the cost between times t_m and t_{m+1}.

The optimization objective is to find a policy π^* such that:

$$v_{\alpha}^{\pi^*}(s) = \inf_{\pi \in \Pi} v_{\alpha}^{\pi}(s).$$

The optimal decision policy can be found by solving the optimality equations for each initial state S. We assume that the bandwidth requests arrive according to a Poisson process with rate λ and the request durations are exponentially distributed with rate μ. With our assumptions of a discounted infinite-horizon CTMDP, the optimality equations can be written as:

$$v(S) = \min_{a \in A}\left\{r(S, a) + \frac{\lambda + \mu}{\lambda + \mu + \alpha}\sum_{j \in \bar{S}} q(j|S, a)v(j)\right\} \quad (2)$$

where $r(S, a)$ is the expected discounted cost between two decision instants and $q(j|S, a)$ is the probability that the system occupies state j at the subsequent decision instant, given that the system is in state S at the earlier decision instant and action a is chosen.

Some assumptions have been made in the approach. The first assumption concerns the capacity of the underlying physical links. It is assumed that there is always enough capacity, which can be reserved by the overlaid LSPs. The second assumption is about the routing algorithm in use for the LSPs and for the bandwidth requests. It is assumed that the LSPs are always routed along the min-hop path between the ingress and egress nodes. Furthermore, the bandwidth requests are routed either on the direct LSP or on the multiple-LSP path overlaying the min-hop path. These assumptions imply that the analysis holds for any node pair independent of traffic for other node pairs.

The optimal policy is derived by the solution of the optimality Eq. (2) for each initial state. The optimal policy π^* has a control-theoretic structure and it is stationary implying same decision rule at each decision instant. The threshold structure of the optimal policy facilitates the solution of the optimality Eq. (2) but still it is difficult to pre-calculate and store the solution because of the large number of possible system states. So, we propose a sub-optimal policy, called the *Least One-Step Cost Policy*, that is easy and fast to calculate.

The proposed Least One-Step Cost policy is an approximation to the solution of the optimality Eq. (2). It minimizes the cost incurred between two decision instants. Instead of going through all the iterations of the value iteration algorithm, we perform only the first iteration. The one-step optimal policy $\pi^\#$ is also control-theoretic in structure and stationary implying same decision rule at each decision instant.

4.2. LSP preemption

In IETF RFC 2702, [1], issues and requirements for TE in an MPLS network are highlighted. In order to address both traffic oriented and resource oriented performance objectives, the authors point out the need for *priority* and *preemption* parameters as TE attributes of *traffic trunks*. A *traffic trunk* is an aggregate of traffic flows belonging to the same class. Traffic trunks are routable objects, distinct from the LSP which they traverse, and are unidirectional. A request for resources for a new traffic trunk implies the setup of a new LSP.

The *preemption attribute* determines whether an LSP with a certain *priority attribute* can preempt another LSP with a lower *priority attribute* from a given path, when there is a competition for available resources. The preempted LSP may then be rerouted. Preemption can be used to assure that high priority LSPs can be always routed through relatively favorable paths in a differentiated services environment. In the same context, preemption can be used to implement various prioritized access policies as well as restoration policies following fault events [1].

4.2.1. Related work

Although not a mandatory attribute in the traditional IP world, preemption becomes indeed a more attractive strategy in a differentiated services scenario [23,24]. Moreover, in the emerging optical network architectures, preemption policies can be used to reduce restoration time for high priority traffic trunks under fault conditions. Nevertheless, in the DiffServ-aware Traffic Engineering (DS-TE) approach, whose issues and requirements are discussed in Ref. [3], the preemption policy is again considered an important piece on the bandwidth reservation and management puzzle, but no preemption strategy is defined.

Running preemption experiments using commercial routers, we could conclude that the preempted LSPs were always the ones with the lowest priority, even when the bandwidth allocated was much larger than the one required for the new LSP. This policy would result in high bandwidth wastage for cases in which rerouting is not allowed. An LSP with a large bandwidth share would be preempted to give room to a higher priority LSP that requires a much lower bandwidth.

A new LSP setup request has two important parameters: bandwidth and preemption level. In order to minimize wastage, the set of LSPs to be preempted can be selected by optimizing an objective function that represents these two parameters, and the number of LSPs to be preempted. More specifically, the objective function could be any or a combination of the following [23–25]:

1. Preempt the connections that have the least priority (preemption level). The QoS of high priority traffics would be better satisfied.
2. Preempt the least number of LSPs. The number of LSPs that need to be rerouted would be lower.
3. Preempt the least amount of bandwidth that still satisfies the request. Resource utilization would be better.

In Ref. [25], the authors propose connection preemption policies that optimize the discussed criteria in a given order of importance: number of connections, bandwidth, and priority; and bandwidth, priority, and number of connections. The novelty in our approach, [8], is to propose an objective function that can be adjusted by the service provider in order to stress the desired criteria. No particular criteria order is enforced. Moreover, our preemption policy is complemented by an adaptive rate scheme. The resulting policy reduces the number of preempted LSPs by adjusting the rate of selected low-priority LSPs that can afford to have their rate reduced in order to accommodate a higher-priority request. This approach minimizes service disruption and rerouting decision and signaling.

4.2.2. Proposed new preemption policy: optimal solution and heuristic

Consider a request for a new LSP setup with bandwidth b and a certain preemption level. When preemption is needed, due to insufficient available resources, the preemptable LSPs will be chosen among the ones with lower preemption level in order to fit r (the difference between the requested bandwidth b and the available bandwidth on a given link). Without loss of generality, we assume that bandwidth is available in bandwidth modules, which implies that variables such as r and b are integers.

Using an integer optimization formulation to implement our preemption policy, [8], we minimize the following objective function:

$$\begin{aligned} F = {} & \alpha(\text{priority of preempted LSPs}) \\ & + \beta(\text{number of preempted LSPs}) \\ & + \gamma(\text{total preempted capacity}) \qquad (3) \end{aligned}$$

where the coefficients α, β, and γ are suitable weights that can be configured by the network operator in order to stress the desired importance of each component in F. As a constraint, we must ensure that the preempted LSPs release enough bandwidth to satisfy the new request.

Our proposed formulation, [8], allows the balance of the three important criteria, and does not imply any order of importance. The network operator is free to adjust the coefficients according to the best interest of each particular network. However, improvements regarding computational issues can be done.

The choice of LSPs to be preempted is known to be an NP-complete problem [26]. For networks of small and medium size, or for a small number of LSPs, the online use of an optimization method is a fast and accurate way to find the solution. However, for large networks and large number of LSPs, a simple heuristic that could approximate the optimal result would be preferable. In order to simplify the online choice of LSPs to be preempted, we propose an heuristic, [8], in which a 'cost function' is calculated for each LSP, and the ones with smaller cost that add enough bandwidth to accommodate r are chosen to be preempted.

4.2.3. New preemption policy with adaptive rate scheme: optimal adaptive solution and heuristic

In Section 4.2.2, when a set of LSPs was chosen to be preempted, those LSPs were torn down and could be rerouted, which implied extra signaling and routing decisions. In order to avoid or minimize rerouting, we propose to reduce the number of preempted LSPs by selecting a few low-priority LSPs that would have their rate reduced by a certain maximum percentage in order to accommodate the new request. In the future, whenever there exists available bandwidth in the network, the lowered-rate LSPs would fairly increase their rate to the original reserved bandwidth.

Some applications such as non-real-time video or data transfer can afford to have their transmission rate reduced, and would be the most likely to be assigned to such Class-Types and preemption levels. By reducing the rate in a fair fashion, the LSPs would not be torn down, there would not be service disruption, extra setup and torn down signaling, or rerouting decisions. For the DiffServ technology, traffic aggregates assigned to the Assured Forward Per-Hop Behavior (AF PHB) would be the natural candidates for rate reduction. Whereas Expedited Forward Per-Hop Behavior (EF PHB) supports services with 'hard' bandwidth and jitter guarantees, the AF PHB allows for more flexible and dynamic sharing of network resources, supporting the 'soft' bandwidth and loss guarantees appropriated for bursty traffic [27].

Similarly to the preemption policy, we formulate the adaptive preemption policy as an integer optimization problem, [8]. We assume that bandwidth is available in bandwidth modules, and choose a set of LSPs, which can afford to have their rate reduced.

We propose the following new objective function $\mathscr{F}$:

$$\begin{aligned} \mathscr{F} = {} & \alpha(\text{priority of preemted bandwidth modules}) \\ & + \beta(\text{number ofpreempted LSPs}) \\ & + \gamma(\text{total preempted capacity}) \\ & + \text{bandwidth module cost per LSP} \qquad (4) \end{aligned}$$

In this equation, the bandwidth module cost per LSP is proportional to the number of modules reserved by the LSP. Coefficients α, β, and γ are used for the same purpose as previously: in order to stress the importance of each component in $\mathscr{F}$.

As for constraints, we must make sure that the bandwidth requirement is met, that all the bandwidth modules from an LSP are made available when that LSP is preempted, that the respective modules for the LSPs that will reduce their rate are also preempted, and that the preempted rate will not be more than $\Delta\%$ of the reserved bandwidth for that LSP.

We propose to use an heuristic for the adaptive policy in order to simplify and expedite the online choice of LSPs that will have their rate reduced or will be completely preempted. The cost function (calculated individually for each LSP) for this heuristic is composed by terms representing the cost of preempting an LSP, the choice of minimum number of LSPs for preemption or rate-reduction, the amount of bandwidth to be preempted, and an additional cost by preempted bandwidth module, similar to the one in $\mathscr{F}$. This additional cost is calculated as the inverse of the amount of bandwidth reserved by the considered LSP. In this way, an LSP with more bandwidth modules will be more likely to have its rate reduced than one with just a few number of modules.

4.3. LSP capacity allocation

In the LSP setup and dimensioning area of bandwidth management, the LSPs are dimensioned based on the nominal requirements derived from the SLS agreements. In reality, these nominal specifications may not be enough or too elaborate or too conservative. So, there is a need for a scheme for capacity allocation for LSPs, which is based on some measurements of the actual traffic carried on the LSP.

Conventional approaches to resource allocation on a link rely on predetermined traffic characteristics which may be difficult to predict. Network traffic can be divided into elastic (e.g. TCP) and non-elastic (e.g. UDP) traffic [28]. These two types differ in their requirements from the network. Packet level characteristics of elastic traffic are controlled by the transport protocol and its interactions with the network whereas the non-elastic flows have inherent rate characteristics that must be preserved in the network to avoid losses. This implies that the source characteristics may not be known ahead of time, specified parameters may not characterize the source adequately or a large number of parameters may be required to define traffic characteristics.

Current capacity allocation methods can be either *off-line* or *on-line*. Off-line, or static, methods determine the allocation amount before the transmission begins. These approaches [29] are simple and predictable but lead to resource wastage. On-line, or dynamic, methods [30–32] periodically renegotiate resource allocation based on predicted/measured traffic behavior. These methods undergo a large number of re-negotiations to satisfy the QoS.

The capacity allocation scheme for LSPs can be extended to be used for Bandwidth Brokers [33] in a DiffServ domain. The scheme can be utilized to determine the capacity allocation for an inter-domain link.

4.3.1. Related work

There are various approaches that can be employed to obtain the capacity allocation for a link. The simplest ones include prediction based on Gaussian assumption (from the central limit theorem) and the Autobandwidth Allocator for MPLS from Cisco [34]. The Gaussian assumption may not hold for Internet traffic in general. The autobandwidth allocator decides the allocation with a phase lag, i.e. the allocation during a time interval is based upon the maximum traffic during the previous interval. This can lead to inefficient allocation decisions. Another proposed scheme for resource provisioning is to have a bandwidth 'cushion', wherein extra bandwidth is reserved over the current usage. As proposed in Ref. [35], if the traffic volume on a link exceeds a certain percentage of the agreement level, it leads to a multiplicative increase in the agreement. A similar strategy is proposed in case the traffic load falls below a considerable fraction of the reservation. This scheme satisfies the scalability requirement but leads to an inefficient resource usage. This drawback can become increasingly significant once the bandwidth requirements of the users become considerable.

The goal of the allocation scheme should be not to derive a near-perfect prediction, but to obtain an upper bound on the resource requirement which is not too conservative. This is because the resources on the links will be provisioned based on the predicted values and if the prediction is near perfect, it can lead to blocking of new requests or degradation of service. With this aim in mind, the Minimum Mean Square Error Linear Predictor in Ref. [30] cannot be employed because it tries to predict the actual value of the measured sequence.

There are currently several implementations of Bandwidth Brokers underway. An architecture of a Bandwidth Broker for scalable support of guaranteed services is given in Ref. [36]. It considers mainly the admission control issues for a Bandwidth Broker. A list of current bandwidth broker implementations and their status is given on the Internet2 QBone Bandwidth Broker Advisory Council webpage [37].

4.3.2. Bandwidth estimation and forecast

We propose an on-line scheme called Estimation and Prediction Algorithm for Bandwidth Brokers (EPABB), [9], to estimate in an optimal manner, the amount of traffic utilizing an LSP based on a measurement of the instantaneous traffic load. This estimate is then used to forecast the traffic bandwidth requests so that resources can be provisioned on the LSP to satisfy the QoS of the requests. The estimation is performed by the use of Kalman Filter [38] theory while the prediction procedure is based on deriving the transient probabilities of the possible system states. This scheme would lead to reduction in the cushion value without introducing per-flow modifications in the resource reservation. Kalman Filters have been previously applied to flow control in high-speed networks. In Ref. [39], Kalman Filter was given for state estimation in a packet-pair flow control mechanism. In Ref. [40], Kalman Filter was used to predict traffic in a collection of VC sources in one VP of an ATM network. Our work distinguishes itself from previous work as the Kalman Filter is used as an optimal estimation algorithm, instead of filtering or smoothing and it is an input to the capacity prediction step.

Consider an LSP$(1,2)$ between the Label Switched Routers (LSRs). We estimate the utilization of the LSP based on a periodic measurement of the aggregate traffic on LSP$(1,2)$. We assume that the traffic measurements are performed at discrete time-points mT, $m=1,2,\dots,M$ for a given value of T. The value of T is a measure of the granularity of the estimation process. Larger values imply less frequent estimation, which can result in larger estimation errors. At the time instant m (corresponding to mT), the aggregate traffic on LSP$(1,2)$ in the direction AS1 to AS2 is denoted by $y(m)$. We also assume that for the duration $(0,MT]$, the number of established sessions that use LSP$(1,2)$ is N. For each session, flows are defined as the active periods. So, each session has a sequence of flows separated by periods of inactivity. We denote by $x(m)$ the number of flows at the instant m and by $x(mT+t), t \in (0,T]$ the number of flows in the time interval $(mT,(m+1)T]$, without notational conflict. Clearly, $x(m) \leq N$ and is not known/measurable. We also assume that each flow has a constant rate of b bits per second. So, nominally

$$y(m) = bx(m). \tag{5}$$

The underlying model for the flows is assumed to be Poisson with exponentially distributed inter-arrival times (parameter λ) and durations (parameter μ). The analysis has been carried out using this assumption and then our simulation results show that, inspite of this assumption, the capacity prediction is very close to the actual traffic.

Our scheme [9] for resource allocation is split into two steps. In the first step, a rough measure of the aggregate traffic $y(m)$ is taken and it is used to evaluate the number of flows through the Kalman Filter estimation process. In the second step, we reserve the resources $R(m)$ on the link LSP(1, 2) for the time $t \in (mT,(m+1)T]$ based on the forecast of the evolution of $x(m)$.

We denote by $p_k(t)$, for $t \in (mT,(m+1)T]$ the probability that the number of active flows at time t is k, given that the estimated number of active flows at the previous measurement instant was j. By using queuing theory [41], we can write the differential equations for the probabilities, solution of which will provide the matrices in the classical Kalman filter formulation. The only measurable variable in our system is $\bar{y}(m)$. It is a measure, corrupted by noise, of the aggregate traffic on the link. Using $\bar{y}(m)$ we evaluate $\hat{x}(m)$, an estimate of $x(m)$, using the Kalman filter setup as:

$$\hat{x}(m) = A\hat{x}(m-1) + B + K(m)[\bar{y}(m) - CA\hat{x}(m-1) - CB] \tag{6}$$

where $K(m)$ is Kalman Filter gain. This gives an estimate of the traffic on the LSP. This estimate will be used to forecast the traffic for the purpose of resource reservation.

The optimal estimate $\hat{x}(m)$ of the number of active flows can now be used to forecast $R(m+1)$, the resource requirement on the LSP(1, 2).

5. Route management

Route management deals with deciding the routes for LSPs over a physical network (Section 5.1) and for bandwidth requests over an MPLS network. It is triggered by the arrival of either an LSP setup request or a bandwidth reservation requests in MPLS networks.

5.1. LSP routing

In MPLS networks, bandwidth guaranteed LSPs can be used to provide QoS guarantees to customers in an IP Virtual Private Network (VPN) fashion. The possibility of explicit routing of LSPs enables the choice of non-shortest paths to perform load balance and, therefore, TE in MPLS networks [42]. A dynamic routing algorithm for LSP setup is a must in this scenario, since off-line algorithms are not suitable due to the necessary a priori knowledge of future LSP setup requests.

5.1.1. Related work

Currently adopted routing schemes, such as OSPF [43] and IS–IS [44], have the number of hops as the only metric used for routing calculations, which may not be enough for QoS routing purposes. In order to introduce QoS requirements in the routing process, the Widest-Shortest Path (WSP) [45] and the Shortest-Widest Path (SWP) [46] algorithms were proposed. In WSP, the shortest paths are computed and the one with larger available bandwidth is chosen to route the traffic. In SWP, paths with the larger available bandwidth are computed and, in case more than one exists, the shortest one is selected. Modifications to these routing algorithms have been proposed in order to reduce complexity, such as the K shortest path, K widest-shortest path, etc. which consider only K path options in their decisions [47]. In Ref. [48], the authors discuss two cost components of QoS routing: complexity and increased routing protocol overhead. Improvements are suggested in order to diminish these cost components, such as path pre-computation and non-pruning triggering policies. Some of these suggestions were used in our routing algorithm. Finally, another QoS routing algorithm called MIRA [14] tries to minimize the interference between different routes in a network for a specific set of ingress–egress nodes. This process involves the computation of maximum flow values for all ingress–egress pairs, computation of the so-called critical links (links that are likely to suffer from interference), and the computation of weights to be used for a weighted-shortest path algorithm that chooses the final route. Before running the weighted-shortest path algorithm, links that have residual bandwidth smaller than the demand are pruned. The shortcomings of MIRA include its computation burden, uses longer paths than shortest-path routing schemes, it is not able to estimate the interference effects on clusters of nodes, and its is not very likely to be implemented by vendors because of its complexity.

5.1.2. SPeCRA: Stochastic Performance Comparison Routing Algorithm

For our algorithm, [10], we consider the problem of setting up bandwidth guaranteed LSPs in an MPLS network, where LSP setup requests arrive individually, and future requests are not a priori known. We propose Stochastic Performance Comparison Routing Algorithm (SPeCRA) to solve the LSP routing problem in an MPLS network characterized by heavy uncertainties of the offered traffic.

Assume a LSP routing scheme (e.g. shortest-path) is denoted by θ. A set of possible LSP routing schemes (e.g. shortest-path, K-shortest-path, shortest-widest-path, etc.)

will be denoted by Θ. We will consider the percentage of rejected LSP setup requests in an interval to be an estimate of the probability of LSP setup rejection. This percentage depends on the LSP routing scheme adopted and also on the particular realization w of the stochastic process characterizing the traffic. In conclusion, the percentage of rejected LSP setup requests from time 0 to t can be denoted by the function $f(0, t, \theta, w)$.

We denote by $p_B(\theta, t)$ the value assumed by the objective function if the system would remain stationary after the time t with the LSP routing scheme $\theta(t)$. Consider now a short time interval $[t, t+T]$, which is contained in a steady interval (namely, there exists an l such that $\tau_l \le t < (t+T) < \tau_{l+1}$), during which no change is made in the LSP routing scheme. An estimate of $p_B(\theta, t)$ is given by the fraction of LSP setup requests rejected in such an interval, $f(t, t+T, \theta, w)$.

If we change the LSP routing scheme θ into θ', we can verify that, in general, $f(t, t+T, \theta', w) \ne f(t, t+T, \theta, w)$. Under very conservative assumptions, it is possible to prove [49] that the estimate of the order between θ and θ' in terms of LSP setup rejection probability, is more robust than the estimate of the cardinal values of the two LSP setup rejection probabilities. In fact, if there are N independent estimates of $p_B(\theta, t)$ and $p_B(\theta', t)$ taken on N different and non-overlapping intervals, the convergence rate of the estimated order to the real order is an exponential function of N and is much larger than the convergence rate of the cardinal estimates, whose variance approaches 0 with $1/N$. Such an interesting feature is used in SPeCRA, where we assume that the piecewise stationary characterizations of the traffic are denoted by SS_i, with $0 \le i \le I$.

Summarizing, we have a discrete and finite set $\{SS_0, SS_1, \dots, SS_I\}$ of stationary stochastic processes, from which we can compose a non-stationary traffic by selecting any combination of SS_is (A non-stationary traffic composed by several SS_is). For each element SS_i, there exists a routing scheme θ_i which is optimal, within a set of possible routing schemes Θ (routing scheme θ_i leads to the minimum rejection rate $p_B(\theta_i, t)$). We do not know a priori which is the current SS_i, neither where the switching times among the SS_is are located. SPeCRA should be able to determine the optimal θ_i without knowing which SS_i is the current traffic offered to the network. The details of SPeCRA are described next.

In order to explain SPeCRA's implementation, we define an increasing sequence of time instants $t_k = kT_c$, $k = 1, 2, \dots$, and denote the interval $[t_k, t_{k+1}]$ as the kth control interval. The algorithm always behaves as a homogeneous Markov chain and the optimal routing scheme is a state of the chain, which is visited at the steady state with a certain probability. Aiming to reduce the chance that we could leave the state due to estimate error, we introduce a noise filter that reduces the effect of such estimate errors. Changes are less likely to happen if the adopted routing scheme (state) is 'good'. This is achieved by introducing a state variable Q, which reduces the changes from 'good' to 'bad' routing schemes, while does not reduce the changes from 'bad' to 'good'. The algorithm is detailed below.

- *Data*:
  - The set of possible routing schemes Θ;
  - The probability function $R(\theta, \theta')$, which represents the probability of choosing θ' as candidate routing scheme when the current routing scheme is θ;
  - An initial routing scheme θ_0;
  - A time duration T_c.
- *Initialization*: Set $x_0 = \theta_0$, $Q_k = 0$ and $k = 0$.
- *Iteration* k:
  1. Let $x_k = \theta$ be the current routing scheme and choose a set $S_k = [z_1, z_2, \dots, z_s]$ of s candidate routing schemes, where the selection of z_i is made according to $R(\theta, z_i)$;
  2. Record all the LSP setup requests arrived and ended during the interval $[t_k, t_{k+1}]$: Compute $f(t_k, t_k + T_c, z_i, w)$, $i = 1, 2, \dots, s$, the estimates of the LSP setup rejection probabilities $p_B(z_i, t)$ for each routing scheme z_i. Select $\theta' = \arg\min_{i=1,2,\dots,s} f(t_k, t_k + T_c, z_i, w)$;
  3. Choose a new routing scheme according to the estimates computed in the previous step: let $f(t_k, t_k + T_c, \theta, w)$ and $f(t_k, t_k + T_c, \theta', w)$ be the estimates of the LSP setup rejection probabilities $p_B(\theta, t)$ and $p_B(\theta', t)$ for the two schemes θ and θ', respectively, in the kth control interval.

     If $f(t_k, t_k + T_c, \theta, w) - Q_k > f(t_k, t_k + T_c, \theta', w)$

     $$x_{k+1} = \theta';$$

     $$Q_{k+1} = 0;$$

     else

     $$x_{k+1} = x_k;$$

     $$Q_{k+1} = [Q_k + f(t_k, t_k + T_c, \theta', w) - f(t_k, t_k + T_c, \theta, w)]/2;$$

  4. Set $k = k + 1$ and go to step 1.

In order to analyze SPeCRA's performance, we ran simulations (see [10]) of the same network topology and compared the traffic described in Ref. [48] and compare the LSP rejection ratio obtained by SPeCRA and MIRA. We observe that SPeCRA outperforms MIRA in the static and also dynamic cases. The set of routing schemes chosen for SPeCRA's simulations is comprised of simple shortest-hop or shortest-cost algorithms, which are interesting solutions for vendors that would rather not implement a complicated algorithm. SPeCRA is easy to be implemented and not as computationally heavy as other routing algorithms.

6. Measurement/performance evaluation tool

There are various measurable quantities of interest that can be insightful about the state of the network. Available bandwidth (together with other metrics like latency, loss, etc.) can predict the performance of the network. Based on the bandwidth available, the network operator can obtain information about the congestion in the network, decide the admission control, perform routing, etc. For MPLS networks, the available bandwidth information can be used to decide about the LSP setup [7], LSP preemption [8], routing (Widest Shortest Path [45], Shortest Widest Path [46]), LSP preemption [8], etc. Each of these processes needs available bandwidth information at a suitable time-scale. It is desirable to obtain the available bandwidth information by measurements from the actual LSPs because they give more realistic information about the available bandwidth. The available bandwidth information can also be obtained by subtracting the nominal reservation for the tunnels from the link capacity, which gives a lower bound. The available bandwidth on a link is indicative of the amount of load that can be routed on the link. Obtaining an accurate measurement of the available bandwidth can be crucial to effective deployment of QoS services in a network. Available bandwidth can be measured using both active and passive approaches.

6.1. Related work

Various tools and products are available that can be used to measure available bandwidth of a link in the network. In Ref. [50], the authors have described a few bottleneck bandwidth algorithms. They can be split into two families: those based on Pathchar [51] algorithm and those based on Packet Pair [39] algorithm. The Pathchar algorithm is an active approach which leads to the associated disadvantage of consumption of significant amount of network bandwidth, etc. The packet pair algorithm measures the bottleneck bandwidth of a route. Active implementations have bandwidth consumption whereas passive implementations may not give correct measurement. In Ref. [52], the authors have proposed another tool to measure bottleneck link bandwidth based on packet pair technique. Some other tools based on the same technique for measuring bottleneck bandwidth of a route have been proposed in Refs. [53,54]. None of them measures the available bandwidth or utilization of a desired link of a network. In Ref. [55], the authors have proposed a tool to measure the available bandwidth of a route which is the minimum available bandwidth along all links of the path. It is an active approach based on transmission of self-loading periodic measurement streams. Another active approach to measure the throughput of a path is Iperf [56] from NLANR that sends streams of TCP/UDP flows. Cisco has introduced the NetFlow [57] technology that provides IP flow information for a network. NetFlow provides detailed data collection with minimal impact on the performance on the routing device and no external probing device. But in a DiffServ environment, the core of a network is interested in aggregate rather than per-flow statistics, due to the scalability issues.

All the tools, except NetFlow, give path measurements based on an active approach. A network operator, on the other hand, would be interested in finding the available bandwidth on a certain link of the network. He has access to the routers/switches of the network and can measure available bandwidth from the routers without injecting pseudo-traffic. Thus, he does not need the end-to-end tools that utilize the active approach of measurement. One approach is to use SNMP [58] as a passive technique to monitor a specific device. Multi Router Traffic Grapher (MRTG) [59] is a tool based on SNMP to monitor the network links. It has a highly portable SNMP implementation and can run on most operating systems. Thus, the network operator requires a tool for measuring the available bandwidth on a certain link of the network in a passive manner whenever he desires. Since the operator has access to the router, he can use MRTG [59]. But MRTG has the limitation that it gives only 5 min averages of link utilization. For applications like routing, this large interval averaging may not be enough. MRTG can be enhanced to decrease the averaging interval down to 1 min. This may still be large for some applications. Thus, we have modified MRTG to MRTG++, to obtain averages over 10 s durations. This gives us the flexibility to obtain very fine measurements of link utilization. Even though the operator can have these measurements, he may not desire each measurement and also this will increase the load on the routers. So, we proposed an adaptive linear regression algorithm to predict the utilization of a link. The algorithm is adaptive because a varying number of past samples can be used in the regression depending on the traffic profile. Using the algorithm, we predict the utilization and the reliability interval for the prediction.

6.2. MRTG++

MRTG [59] is a tool to monitor the traffic load on network links. It generates HTML pages containing PNG images, which provide a visual representation of this traffic. MRTG is consists of a Perl script which uses SNMP to read the traffic counters of routers and a C program which logs the traffic data. The log is then used to create graphs representing the traffic on the monitored connection. The pictures are then embedded into webpages, which can be viewed using any modern Web-browser.

The original MRTG must be run every 5 min to query devices. Whenever it is run, it updates the database with new information and regenerates all graphs and all reports.

This behavior creates a scalability problem since it must recreate all graphs every 5 min. In order to handle this limitation, MRTG can be used with RRDTool, an acronym for Round Robin Database. It is a system to store time-series data in a compact way and create graphs to present it. RRDs are databases with a fixed amount of slots to store data. When the last element is stored, the system reuses old locations replacing the first ones as in a circle. This way, RRDs does not expand over time [60]. RRDtool replaces MRTG's capabilities for logging and displaying graphs. The idea is to still use MRTG to query devices but to store the information in a RRD. Web pages are created on demand only when needed. Integrating RRDtool with MRTG has another advantage. The original MRTG is unable to provide more details than in a 5 min interval. Even running MRTG more often, the database and the graph engine will summarize the information into 5 min intervals. RRDtool can be used to provide more room to store MRTG information. A patch must be applied to the mrtg perl script to make it store up to 1 min traffic information. 14all, routers.cgi and RRGrapher are RRDtool user's interfaces available. They provide a web interface to command and show graphs generated by RRDtool. Since these tools send commands to RRDtool to create the graphs, they also need to be patched in order to show more detail than 5 min. Although the patch provides us up to 1 min detail, our group was interested in getting information more frequently than 1 min. MRTG++ is the modified script to store information of 10 s interval. The modifications were relatively simple. First of all, the new RRD must be created with more storage space and smaller time steps between the slots. Next is to modify 14all to send the correct queries to RRDtool when creating graphs. Finally, MRTG must be run every 10 s to retrieve the counters from the interfaces. Iperf was used to generate traffic to validate the results.

6.3. Available bandwidth estimation scheme

Our approach, [11], is based on the use of MRTG where the manager TEAM will enquire each router in the domain through SNMP and obtain the information about the available bandwidth on each of its interfaces. The most accurate approach will be to collect information from all possible sources at the highest possible frequency allowed by the MIB update interval constraints. However, this approach can be very expensive in terms of signaling and data storage. Furthermore, it can be redundant to have so much information. Thus, our scheme tries to minimize the query frequency while still estimating the available bandwidth efficiently.

We can set the measurement interval of MRTG and measure the average link utilization statistic for that interval. We define for a link between two nodes i and j:

- $L(t)$: Traffic load at time t in bits per s,
- $A(t)$: Available capacity at time t in bits per s,
- τ : Length of the averaging interval of MRTG,
- $L_\tau[k], k \in \mathbf{N}$: Average load in $[(k-1)\tau, k\tau]$,
- p: the number of past measurements in prediction,
- h: the number of future samples reliably predicted,
- $A_h[k]$: the estimate at $k\tau$ valid in $[(k+1)\tau, (k+h)\tau]$.

Our problem can be formulated as linear prediction:

$$L_\tau[k+a] = \sum_{n=0}^{p-1} L_\tau[k-n]w_a[n] \qquad \text{for } a \in [1,h] \tag{7}$$

where on the right side are the past samples and the prediction coefficients $w_a[n]$ and on the left side, the predicted values. The problem can be solved using covariance method [61]. We propose to dynamically change the values of p and h based on the traffic dynamics.

The covariance equations are given in a matrix form as $\mathbf{R}_L\mathbf{w}_a = \mathbf{r}_a$, $a = 1, \ldots, h$, i.e.

$$\begin{bmatrix} r_L(0,0) & \cdots & r_L(0,p-1) \\ \vdots & \ddots & \vdots \\ r_L(p-1,0) & \cdots & r_L(p-1,p-1) \end{bmatrix} \begin{bmatrix} w_a(0) \\ \vdots \\ w_a(p-1) \end{bmatrix} = \begin{bmatrix} r_L(0,-a) \\ \vdots \\ r_L(p-1,-a) \end{bmatrix}$$

where the covariance of the sequence is estimated as

$$r_L(n,m) = \sum_{i=k-N+p}^{k} L_\tau[i-n]L_\tau[i-m].$$

Once the w_a's are found by solving the covariance equations, the estimates for $L_\tau[k+a]$ can be obtained. Using these estimates, $A_h[k]$ can be estimated and also the new values for p and h can be chosen depending on the error in estimation. Thus, the objective of the algorithm is to minimize the computational effort while providing a reliable estimate of available bandwidth of a link. It provides a balance of the processing load and accuracy. The algorithm is based on the dynamics of the traffic, i.e. it adapts itself.

7. TEAM implementation

7.1. SNMPv3

SNMP is the protocol used for communication between the routers and our manager. Despite the recent vulnerabilities, SNMP is widely accepted as the de facto standard for network management and patches are being released to fix it.

A major deficiency of the current version of SNMP is information security. Community strings are transported as clear text and if compromised, an attacker could have access

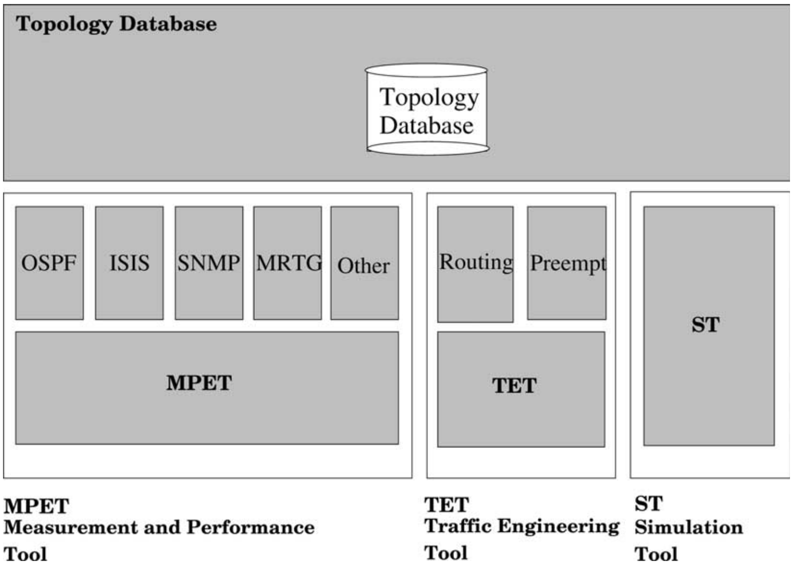


Fig. 5. Modules architecture.

to all management information and configuration rights. Preventive measurements such as packet filtering and configuring agents to only accept packets whose source IP address matches the manager can be used, but its effectiveness is questionable since the packet can easily be modified. SNMPv3 addresses this problem proposing a new framework for the protocol. It enhances security incorporating authentication, encryption and integrity checks on messages sent between the agent and the manager. Data integrity is accomplished by adding an MD5 or SHA digest to the message. It ensures that the message was not tampered and that is coming from where it claims to be. SNMPv3 also includes a timestamp on each message, so that replay attacks cannot be used. Finally, it provides support for data encryption between the agent and the manager. The recommendation is to use DES, although the specification is open to other alternatives.

SNMPv3 also has the concept of custom tree views for each user registered in the system. This way, we can restrict access of a user to only a set of the MIB tree, limiting the damage, in case the strings are compromised.

All commands between the manager and the routers are carried over SNMPv3. Unfortunately, Cisco IOS only exports read-only TE variables and tunnel setup cannot be done using SNMP. We identified two options to work out this limitation. The first one it to use telnet to connect to the router and send the CLI commands to create the MPLS tunnel. The second approach is to use SNMP and instruct the router to retrieve the configuration from a TFTP server to create the new tunnel.

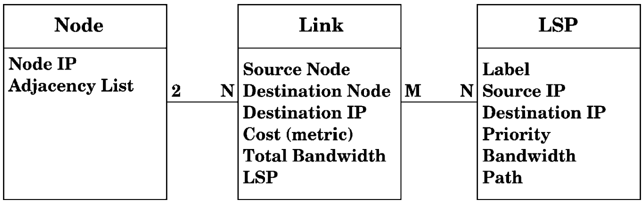


Fig. 6. Database structure.

In our implementation, we adopted the second approach since telnet is not secure and passwords are sent in clear text. Adopting SNMPv3 as the management protocol, we make sure passwords and commands are secured. Although the new configuration comes from the TFTP server in clear text, it does not contain any sensitive information. These commands are only relevant to the new tunnel setup and the router merges it to the current configuration.

7.2. TEAM architecture

The general architecture for TEAM is shown in Fig. 5. Since all modules (TET, MPET and ST) need access to the network topology, there is a centralized database and specific functions to read and manipulate its content. MPET is the module responsible to learn the topology of the network and evaluate its current conditions. It is structured in different modules and the main program executes each one on a separate thread. This enables us to interpret different information to build the topology map. For example, link state routing protocols like OSPF and ISIS require that every router in the area maintain a topology database of the area they belong. One of the modules would be responsible for listening to messages exchanged and updating the topology database. SNMP can also be used to retrieve routing information from the routers to update the database. A separate set of thread will be responsible for retrieving current bandwidth utilization, like MRTG.

Once the topology database is in place, the Traffic Engineering Tool (TET) is responsible for routing and resources management and the Simulation Tool (ST) is responsible for checking and validating TET decisions.

They will also access the topology database through the same set of common functions.

TEAM keeps a database with the network topology and connected LSPs in the domain. The database is updated by MPET, which adds and remove links as they become available or not. It also retrieves bandwidth utilization statistics similarly to MRTG and updates link records. In addition to the topology database, TEAM also maintains some information about the established LSPs under its domain, e.g. the LSP path. The database model is shown in Fig. 6.

8. Conclusions

We propose the design of a set of new algorithms to provide QoS and better resource utilization in an MPLS network and an integrated architecture, TEAM, as an MPLS domain manager. The new algorithms concern resource management and route management. All algorithms will be developed and evaluated through simulation and experiments individually on our physical testbed. Now, we are focusing on their inter-working and the development of TEAM as a whole. We have a full-fledged physical testbed, with Next Generation Internet routers, equipped with DiffServ-capable routers and switches manufactured by Cisco. The integration of the above mentioned algorithms will result in a unique and powerful method for Internet management.

Acknowledgement

This work was supported by NASA Goddard. The work of J. C. de Oliveira was also supported in part by CAPES (The Brazilian Ministry of Education Agency).

References

[1] D.O. Awduche, J. Malcom, J. Agogbua, M. O'Dell, J. McManus, Requirements for Traffic Engineering over MPLS, IETF RFC 2702, September 1999, http://www.ietf.org/rfc/rfc2702.

[2] D.O. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, G. Swallow, RSVP-TE: Extensions to RSVP for LSP Tunnels, IETF RFC 3209, December 2001, http://www.ietf.org/rfc/rfc3209.

[3] D. Awduche, A. Chiu, A, Elwalid, I. Widjaja, X. Xiao, Overview and Principles of Internet Traffic Engineering, IETF RFC 3272, May 2002, http://www.ietf.org/rfc/rfc3272.

[4] R. Callon, Predictions for the core of the network, IEEE Internet Computing 4 (1) (2000).

[5] J. de Oliveria, T. Anjali, B. King, C. Scoglio, Building an IP Differentiated Services Testbed, Proceedings of IEEE International Conference on Telecommunications, Bucharest, Romania, June, 2001.

[6] C. Scoglio, T. Anjali, J. de Oliveira, I.F. Akyildiz, G. Uhi, A New Threshold-Based Policy for Label Switched Path Setup in MPLS Networks, Proceedings of 17th International Teletraffic Congress'01, Salvador, Brazil, September, 2001.

[7] T. Anjali, C. Scoglio, J. de Oliveira, I.F. Akyildiz, G. Uhi, Optimal policy for LSP setup in MPLS networks, Computer Networks 39 (2002) 2.

[8] J. de Oliveira, C. Scoglio, I.F. Akyildiz, G. Uhi, A New Preemption Policy for DiffServ-Aware Traffic Engineering to Minimize Rerouting, Proceedings of IEEE INFOCOM'02, New York, USA, 2002.

[9] T. Anjali, C. Scoglio, I.F. Akyildiz, G. Uhi, A New Scheme for Traffic Estimation and Resource Allocation for Bandwidth Brokers, BWN Lab Technical Report BWN-02-J-32.

[10] J. de Oliveira, F. Martinelli, C. Scoglio, SPeCRA: a Stochastic Performance Comparison Routing Algorithm for LSP Setup in MPLS Networks, Proceedings of IEEE GLOBECOM'02, Taipei, Taiwan, November 2002.

[11] T. Anjali, C. Scoglio, L. Chen, I.F. Akyildiz, G. Uhl, ABEst: an Available Bandwidth Estimator within an Autonomous System, Proceedings of IEEE GLOBECOM'02, Taipei, Taiwan, November 2002.

[12] P. Aukia, M. Kodialam, P.V.N. Koppol, T.V. Lakshman, H. Sarin, B. Suter, RATES: a server for MPLS traffic engineering, IEEE Network March/April (2000) 14.

[13] S. Herzog, J. Boyle, R. Cohen, D. Durham, R. Rajan, A. Sastry, COPS usage for RSVP, IETF RFC 2749, January 2000, http://www.ietf.org/rfc/rfc2749.

[14] K. Kar, M. Kodialam, T.V. Lakshman, Minimum interference routing of bandwidth guaranteed tunnels with MPLS traffic engineering application, IEEE Journal on Selected Areas in Communications 18 (12) (2000) 2566–2579.

[15] A. Elwalid, J. Ching, S. Low, I. Widjaja, MATE: MPLS Adaptive Traffic Engineering, Proceedings of IEEE INFOCOM'01, Anchorage, USA, April, 2001.

[16] P. Trimintzios, L. Georgiadis, G. Pavlou, D. Griffin, C.F. Cavalcanti, P. Georgatsos, C. Jacquenet, Engineering the Multi-Service Internet: MPLS and IP-Based Techniques, Proceedings of IEEE International Conference on Telecommunications, Bucharest, Romania, June, 2001.

[17] X. Xiao, A. Hannan, B. Bailey, L.M. Ni, Traffic engineering with MPLS in the Internet, IEEE Network Magazine March (2000).

[18] S. Uhlig, O. Bonaventure, On the Cost of Using MPLS for Interdomain Traffic, Proceedings of Quality of Future Internet Services'00, Berlin, Germany, September, 2000, pp. 141–152.

[19] I.T. Okumus, J. Hwang, H.A. Mantar, S.J. Chapin, Inter-Domain LSP Setup Using Bandwidth Management Points, Proceedings of IEEE GLOBECOM'01, San Antonio, USA, November, 2001.

[20] Q. Hao, S. Tartarelli, M. Devetsikiotis, Self-Sizing and Optimization of High-Speed Multiservice Networks, Proceedings of IEEE GLOBECOM'00, San Francisco, USA, November, 2000.

[21] J. Liebehetrr, S.D. Patek, E. Yilmaz, Trade-offs in Designing Networks with End-to-end Statistical QoS Guarantees, Eighth International Workshop on Quality of Service, 2000.

[22] M.L. Puterman, Markov Decision Processes: Discrete Stochaftic Dynamic Programming, Wiley, New York, 1994.

[23] S. Poretsky, Connection Precedence and Preemption in Military Asynchronous Transfer Mode (ATM) Networks, Proceedings of MILCOM'98, 1998, pp. 86–90.

[24] S. Poretsky, T. Gannon, An Algorithm for Connection Precedence and Preemption in Asynchronous Transfer Mode (ATM) Networks, Proceedings of IEEE ICC'98, 1998, pp. 299–303.

[25] M. Peyravian, A.D. Kshemkalyani, Decentralized network connection preemption algorithms, Computer Networks and ISDN Systems 30 (11) (1998) 1029–1043.

[26] J.A. Garay, I.S. Gopal, Call Preemption in Communication Networks, Proceedings of IEEE INFOCOM'92, 1992, pp. 1043–1050.

[27] G. Armitage, Quality of Service in IP Networks, Macmillan Technical Publishing, 2000, Technology Series.

[28] J.W. Roberts, Traffic theory the Internet, IEEE Communications Magazine 39 (1) (2001) 94–99.

[29] W.C. Feng, F. Jahaniam, S. Sechrest, An optimal bandwidth allocation strategy for the delivery of compressed prerecorded video, ACM/Springer Verlag Multimedia Systems Journal 5 (5) (1997).

[30] A. Adas, Supporting Real-time VBR Video Using Dynamic Reservation Based on Linear Prediction, Proceedings of IEEE INFOCOM'96, San Francisco, USA, 1996, pp. 1476–1483.

[31] H. Zhang, E.W. Knightly, RED-VBR: a renegotiation based approach to support delay sensitive VBR video, Multimedia Systems 5 (1997) 164–176.

[32] D. Reininger, G. Ramamurthy, D. Raychaudhuri, VBR MPEG video coding with dynamic bandwidth renegotiation, IEEE International Conference on Communications, Seattle, USA, June, 1995, pp. 1773–1777.

[33] K. Nichols, V. Jacobson, L. Zhang, A Two-bit Differentiated Services Architecture for the Internet, IETF RFC 2638, July 1999, http://www.ietf.org/rfc/rfc2638.

[34] White Paper, Cisco MPLS AutoBandwidth Allocator for MPLS Traffic Engineering, June 2001, http://www.cisco.com/warp/public/cc/pd/iosw/prodlit/mpatb_wp.htm.

[35] A. Terzis, L. Wang, J. Ogawa, L. Zhang, Two-Tier Resource Management Model for the Internet, Proceedings of IEEE GLOBECOM'99, December, 1999, pp. 1779–1791.

[36] Z. Zhang, Z. Duan, L. Gao, Y.T. Hou, Decoupling QoS Control from Core Routers: a Novel Bandwidth Broker Architecture for Scalable Support of Guaranteed Services, Proceedings of ACM SZGCOM'00, Sweden, August, 2000.

[37] QBone Bandwidth Broker Advisory Council, http://www.internet2.edu/qos/qbone/QBBAC.shtml.

[38] P.S. Maybeck, Stochastic Models, Estimation, and Control, Academic Press, New York, 1979.

[39] S. Keshav, A Control-Theoretic Approach to Flow Control, Proceedings of ACM SZGCOM'91, Zurich, Switzerland, 1991.

[40] A. Kolarov, A. Atai, J. Hui, Application of Kalman Filter in High-Speed Networks, Proceedings of IEEE GLOBECOM'94, San Francisco, USA, 1994, pp. 624–628.

[41] L. Kleinrock, Queueing Systems, Wiley, New York, 1975.

[42] E. Rosen, A. Viswanathan, R. Callon, Multiprotocol Label Switching Architecture, IETF RFC 3031, January 2001, http://www.ietf.org/rfc/rfc3031.

[43] J. Moy, OSPF Version 2, IETF RFC 2328, April 1998, http://www.ietf.org/rfc/rfc2328.

[44] R. Callon, Use of IS-IS for Routing in TCP/IP and Dual Environments, IETF RFC 1195, December 1990, http://www.ietf.org/rfc/rfc1195.

[45] R. Guerin, A. Orda, D. Williams, QoS Routing Mechanisms and OSPF Extensions, Proceedings of Second Global Internet Miniconference (Joint with GLOBECOM'97), Phoenix, USA, November, 1997.

[46] Z. Wang, J. Crowcroft, Quality-of-service routing for supporting multimedia applications, IEEE Journal on Selected Areas in Communications 14 (7) (1996) 1234–1288.

[47] D. Eppstein, Finding the *k* shortest paths, SIAM Journal of Computing 28 (2) (1998) 652–673.

[48] G. Apostolopoulos, R. Guerin, S. Kamat, S.K. Tripathi, Quality of Service Based Routing: a Performance Perspective, Proceedings of SIGCOMM'98, 1998.

[49] L. Dai, C.H. Chen, Rates convergence of ordinal comparison for dependent discrete event dynamic systems, Journal of Optimization Theory and Applications 94 (1) (1997) 29–54.

[50] K. Lai, M. Baker, Measuring Bandwidth, Proceedings of IEEE INFOCOM'99, New York, USA, March, 1999.

[51] V. Jacobson, Pathchar, 1997, ftp://ftp.ee.lbl.gov/pathchar/.

[52] K. Lai, M. Baker, Nettimer: a Tool for Measuring Bottleneck Link Bandwidth, Proceedings of Third USENIX Symposium on Internet Technologies and Systems, San Francisco, USA, March, 2001.

[53] R.L. Carter, M.E. Crovella, Measuring Bottleneck Link Speeds in Packet-Switched Networks, Boston University Technical Report BU-CS-96-006, 1996.

[54] V. Paxson, End-to-end Internet Packet Dynamics, Proceedings of ACM SIGCOMM'97, Cannes, France, September, 1997.

[55] M. Jain, C. Dovrolis, Pathload: a Measurement Tool for End-to-End Available Bandwidth, A Workshop on Passive and Active Measurements, Fort Collins, USA, March, 2002.

[56] Iperf Tool, http://dast.nlanr.net/Projects/Iperf.

[57] Cisco IOS Netflow Website, http://www.cisco.com/warp/public/732/Tech/netflow/.

[58] J. Case, K. McCloghrie, M. Rose, S. Waldbusser, Introduction to version 2 of the Internet-standard Network Management Framework, IETF RFC 1441, 1993, http://www.ietf.org/rfc/rfcl441.

[59] MRTG Website, http://people.ee.ethz.ch/oetiker/webtools/mrtg/.

[60] RRDTool Website, http://people.ee.ethz.ch/oetiker/webtools/rrdtool/.

[61] M.H. Hayes, Statistical Digital Signal Processing and Modeling, Wiley, New York, 1996.