An Open Source Traffic Engineering Toolbox

G. Leduc [a,*] H. Abrahamsson [e] S. Balon [a] S. Bessler [b]
M. D'Arienzo [h] O. Delcourt [a] J. Domingo-Pascual [d] S. C. Erbas [g]
I. Gojmerac [b] X. Masip [d] A. Pescapè [h] B. Quoitin [f] S.P. Romano [h]
E. Salvadori [c] F. Skivée [a] H.T. Tran [b] S. Uhlig [f] H. Ümit [g]

[a]*RUN, Université de Liège, Belgium*

[b]*Telecommunications Research Center Vienna (ftw.), Austria*

[c]*Dipartimento di Informatica e Telecomunicazioni, Università di Trento, Italy*

[d]*Departament d'Arquitectura de Computadors, Universitat Politècnica de Catalunya, Spain*

[e]*Computer and Network Architectures Laboratory, SICS, Sweden*

[f]*INGI, Université catholique de Louvain, Belgium*

[g]*POMS, Université catholique de Louvain, Belgium*

[h]*Dipartimento di Informatica e Sistemistica, Università di Napoli Federico II, Italy*

Abstract

We present the software architecture of an open source Traffic Engineering (TE) toolbox and a set of TE methods that we have designed. These methods cover intra-domain and inter-domain TE, IP-based and MPLS-based TE. They are suitable for network optimisation, better routing of traffic for providing QoS, load balancing, protection and restoration in case of failure, etc. The toolbox is designed to be deployed as an on-line tool in an operational network, or used off-line as an optimisation tool or as a traffic engineering simulator.

[*] corresponding author
Email addresses: guy.leduc@ulg.ac.be (G. Leduc), henrik@sics.se (H. Abrahamsson), simon.balon@ulg.ac.be (S. Balon), bessler@ftw.at (S. Bessler), maudarie@unina.it (M. D'Arienzo), olivier.delcourt@ulg.ac.be (O. Delcourt), jordi.domingo@ac.upc.es (J. Domingo-Pascual), cerav@poms.ucl.ac.be (S. C. Erbas), gojmerac@ftw.at (I. Gojmerac), xmasip@ac.upc.es (X. Masip), pescape@unina.it (A. Pescapè), bqu@info.ucl.ac.be (B. Quoitin), spromano@unina.it (S.P. Romano), salvador@dit.unitn.it (E. Salvadori), fabian.skivee@ulg.ac.be (F. Skivée), tran@ftw.at (H.T. Tran), suh@info.ucl.ac.be (S. Uhlig),

1 Introduction

Today the usual way of providing a suitable level of service in an enterprise intranet or an Internet Service Provider is to overprovision the network with respect to the real needs. With the increase in bandwidth demand, this approach is less and less tenable economically. An alternative way is to deploy traffic engineering techniques. However, most of the problems that are encountered in this field are combinatorial and of large size, which implies to find efficient and near optimal heuristics.

The objective of the E-NEXT[1] task force on traffic engineering is to set up an open source Toolbox Of Traffic Engineering Methods (TOTEM) that would federate many independent software pieces designed by the E-NEXT partners. The resulting toolbox is expected to include more functionality than existing commercial ones, and is clearly designed to be open, i.e. incrementally extensible.

This paper presents the software architecture of the toolbox and a set of complementary methods that are already (expected to be) integrated. Our traffic engineering methods can be classified along several axes: intra-domain versus inter-domain, IP versus MPLS (MultiProtocol Label Switching), online versus off-line, or centralized versus distributed. They are suitable for network optimisation, better routing of traffic for providing quality of service, load balancing, protection and restoration in case of failure, etc.

The design of the toolbox also considers different possible use cases. For example, it can be deployed as an on-line tool in an operational network, or used off-line as an optimisation tool or as a traffic engineering simulator.

The paper is structured as follows. Section 2 reviews related work and existing tools. Section 3 presents the role of the toolbox, its typical use cases and its software architecture. Section 4 describes our traffic engineering methods classified in three categories: intra-domain IP-based, inter-domain IP-based, and MPLS-based.

2 Related work

Traffic engineering consists of all the available techniques whose purpose is to directly or indirectly adapt the traffic to achieve certain objectives. Traffic en-

umit@poms.ucl.ac.be (H. Ümit).

[1] E-Next is a Network of Excellence (NoE) funded by the European Commission under the 6th IST programme

gineering has received a lot of attention during the last few years [9]. Initially, traffic engineering was considered as a solution to allow large tier-1 service providers to optimize the utilization of their network. In these large networks, there are typically several possible paths to reach a given destination or border router. Ideally, to achieve a good network utilization, the traffic should be spread evenly among all the available links. Unfortunately, this does not correspond to the way traditional IP routing protocols behave.

At the opposite of large tier-1 providers, small providers and multi-homed corporate networks have different traffic engineering requirements. Their networks have usually a simple topology and are frequently over-provisioned. The traffic engineering solutions mentioned above are not really useful in such networks. For these networks, the costly resource that needs to be optimised with traffic engineering is their inter-domain connectivity, i.e. the links that connect them to the rest of the Internet.

These two problems refer respectively to intra-domain and inter-domain traffic engineering. Intra-domain TE can be further split into IP-based TE (mainly IGP-weight optimisation) and MPLS-based TE.

IGP (Interior Gateway Protocol) weight optimisation is defined for networks employing SPF (Shortest Path First) protocols, e.g. OSPF (Open Shortest Path First) and IS-IS (Intermediate System-Intermediate System). It aims at avoiding congestion by modifying link weights and hence adapting the routing scheme in the network [27]. Current SPF applications are based on default static link weights, e.g., CISCO suggests these weights to be inversely proportional to the link capacities for OSPF networks. However, the performance of routing can be enhanced with an intelligent weight setting that takes the traffic demand matrix into consideration.

It is also possible to extend the basic model with more complex characteristics of the problem, e.g., consideration of the link failures, multiple demand matrices, etc. [26]. The biggest challenge lying in the application of these extensions is the requirement for periodic weight changes under varying network conditions. Weight changes should be avoided as much as possible, since they bring instability to the network. Thus, obtaining a different weight vector for each possible scenario within the network (e.g., different demand matrices, unavailable links) is not a favourable solution. Robust optimisation techniques should be developed to obtain a single weight setting that performs well for possible scenarios.

Traffic engineering based on MPLS has a better potential than IP-based traffic engineering whose routing is only based on the destination prefix. The fundamental problem with MPLS is to compute routes for the Label Switched Paths (LSPs) which will carry the traffic aggregates associated with the considered Forward Equivalent Classes (FECs). Two well-known solutions are MIRA (Minimum Interference Routing) [37] and PBR (Profile-based Routing) [59]. These methods are more efficient than the more classical WSP (Widest Shortest Path) [28] and SWP (Shortest Widest Path) [71].

MPLS also allows to reroute LSPs, or change their bandwidth reservations, to make room for other more important ones [51], and provides protection/restoration methods in case of failures [58,39,38,49] by setting up backup LSPs.

Inter-domain TE is important economically given the high cost of inter-domain links. This problem is usually solved by configuring the BGP routers manually in a trial-and-error manner [68,54]. Some tools also exist to allow content providers to optimize their outgoing traffic [13]. Earlier works on inter-domain TE are optimisation methods to select the best peerings in a large network [11,41]. Large network operators have also studied their traffic repartition and their impact on inter-domain TE [23,24,15].

More references to related works will be found in their dedicated sections.

Several commercial network optimisation toolboxes already exist, e.g., MATE (Cariden) [72], Netscope (AT&T) [25], Tunnel Builder Pro (CISCO) [77], TSOM (Alcatel) [70], Conscious (Zvolve) [74], IP/MPLSView (Wandl) [73] and SP Guru (Opnet) [75]. All these tools are centralised and propose exact and heuristic optimisation methods. Most tools are suitable to solve "what-if" scenarios that allow a network operator to evaluate the impact of, e.g., an IGP weight change. Beside this simulation mode, MATE and Conscious also provide an IGP weight optimizer. All these tools except Netscope also support optimisation methods for MPLS networks, including for most of them the computation of backup paths for protection and restoration. Most tools rely on the knowledge of link loads and the existing MPLS LSPs, but MATE also provides a method to derive the traffic matrix from the link loads. The main drawbacks of these commercial tools are their lack of detailed technical public information about their algorithms and the impossibility to upgrade them by new research proposals.

To the best of our knowledge the only other open-source network optimisation library is MASCOPT [40]. Their current implementation provides a generic graph model and a basic graphical interface. In the future this library will also contain constraint-based routing algorithms taking failures into account, and grooming algorithms for SDH and WDM networks. By contrast to our approach, MASCOPT only provides a library, not a complete toolbox. In that sense, it is comparable to the generic tools and topology manager present in TOTEM (See Fig. 4).

3 Role and architecture of the toolbox

We present the two use cases of the TOTEM toolbox, as an off-line or on-line platform, its software architecture and external interfaces, its core topology representation based on XML, and its facilities to integrate new tools.

3.1 TOTEM as an off-line tool

By off-line tool we mean a tool which is usually not integrated in a real network and is mainly used as a simulator to assess new TE methods on certain topologies and traffic conditions.

Practically, a comparison of TE methods is often difficult to carry out and is at best very time-consuming, because it requires to run competitive methods or exact solvers on the same data. The software code of these methods is not always available, and re-implementing them is tedious, error-prone and sometimes impossible by lack of detailed descriptions in the literature. The Network Simulator [81] is a solution to this problem for packet-based simulations, but no similar tool exists for solving TE problems, which are mostly optimisation problems or require flow-based simulators. Our objective is to bridge this gap.

A designer of a new TE method would only have to integrate his/her algorithm in TOTEM to benefit from the presence of other methods for comparison purposes. Moreover the toolbox will provide several side services, such as topology/traffic generators, and simulation scenario interpreters, and will contain a repository of existing topologies and traffic matrices.

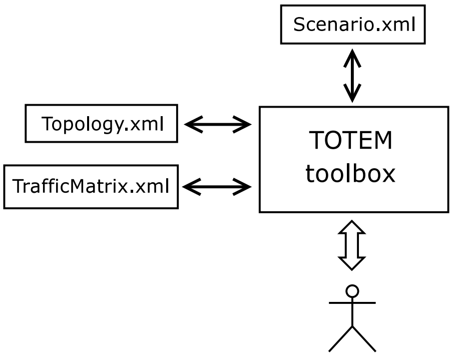

Fig. 1. *TOTEM as an off-line tool*

3.2 TOTEM as an on-line tool

The TOTEM architecture is also designed to be used as an on-line tool, which means that it can be deployed in a real or experimental network. In such a case, the kernel of the toolbox is basically the same as above. However, the topology and traffic generators will advantageously be replaced by a topology discovery tool and a traffic monitoring/measurement tool. These tools can be integrated in TOTEM, but are better considered as external tools which TOTEM can interface with.

The same reasoning applies to external control/provisioning tools used by operators for changing the configuration of their network. Such tools can e.g. modify the IGP weights or create MPLS LSPs.

Therefore, TOTEM can be seen (Fig 2) as a tool that uses information collected by the measurement tool(s) and offers TE services to the provisioning tool(s). The latter can send some request to TOTEM asking for some computations (e.g. give me a route for that LSP). The response would be a(n) (list of) action(s) to be executed on the network (e.g. establish the LSP along a given route and re-route another LSP).

Although some commercial tools offer combined TE and provisioning functionality (e.g. TSOM (Alcatel)[70] or Tunnel Builder Pro (Cisco)[77]), and possibly topology discovery as well, we have opted for a clear separation of these concerns.

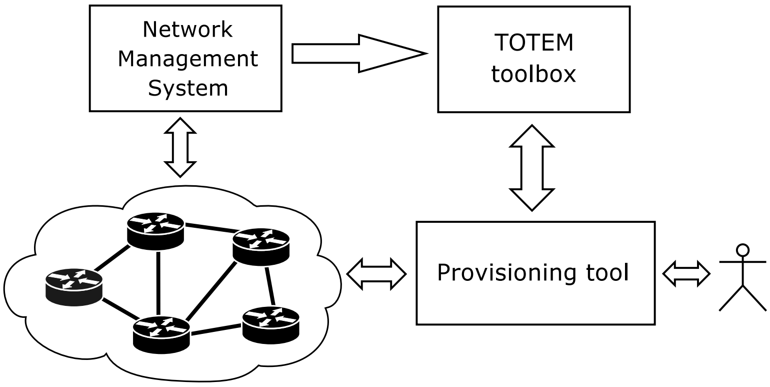

Fig. 2. *TOTEM as an on-line tool*

As an example we illustrate how TOTEM could be integrated in an MPLS-Linux testbed. We briefly present a topology discovery tool and a provisioning tool, and how they could interact with TOTEM.

3.2.1 Interaction with a Topology Discovery tool

Automatic discovery of physical topology information (Fig. 2, Network Management System) plays a crucial role in enhancing the manageability of modern IP networks. Despite the importance of the problem, discovering network topology is an inherently difficult task[32]. The network topology knowledge (i.e. the list of available hosts, routers and subnets) can prove useful in a number of situations such as faults isolation, performance analysis, network planning, services positioning and TE algorithms.

Since there are no standards, any algorithm developed to discover the topology can only use the basic IP primitives. The NeToDi (Network Topology Discovery) architecture [5] represents an adaptive hybrid solution to network topology discovery made by an innovative and efficient composition of active, passive and routing protocol based methodologies. More precisely, it is based on the well-organized combination of:

- Passive Methodology: relying on the use of SNMP (Simple Network Management Protocol) and DNS (Domain Name Server);
- Active Methodology: in this case there is a massive use of tools based on 'ping' and 'traceroute';
- Routing Based Methodology: topology is derived by using the information of routing processes.

Thanks to the use of the hybrid methodology, the NeToDi architecture guarantees to be efficient (i.e. imposes the least possible overhead on the network), fast (i.e. takes the least possible time to complete the job), complete (i.e. discovers the entire topology) and accurate (i.e. makes no mistake). The NeToDi output is provided both in text and XML formats.

3.2.2 Interaction with a Provisioning tool

For testbed experiments, a provisioning tool that can configure an MPLS-Linux testbed [7] nicely complements TOTEM. It is a set of blocks, communicating with each other to configure network nodes. Each network node is a Linux PC with an MPLS-enabled kernel and an RSVP-TE daemon for the setup of explicitly routed LSPs. The interaction between the testbed configuration tool and the TE toolbox would be as follows. Given the network topology and a user request, the TE toolbox engine performs admission control and path selection. The selected path (i.e. the list of IP addresses of its constituent nodes) can then be returned to the testbed configuration tool.

The provisioning tool (Fig. 2) adopts the COPS (Common Open Policy Server) protocol (Fig. 3) to communicate with network elements. The information on

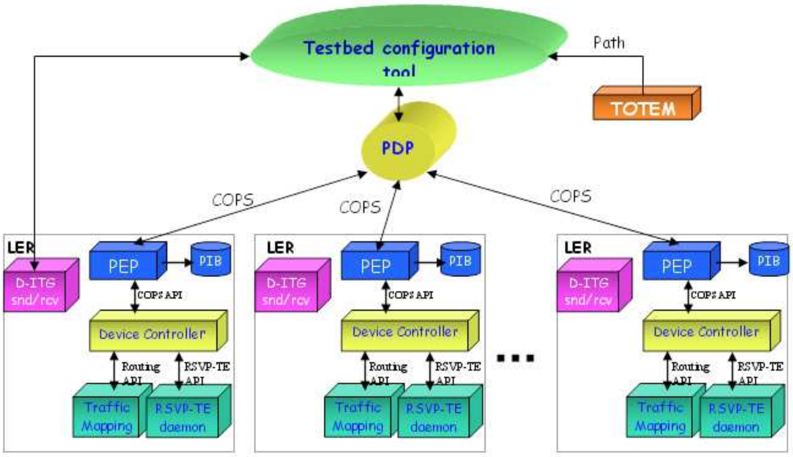

Fig. 3. *TOTEM integration in an MPLS testbed.*

the LSP to be established and the traffic to be mapped on it is received and translated by the PDP (Policy Decision Point) in a set of *policies*. Such policies are sent to the PEP (Policy Enforcement Point) running on the ingress node of the LSP. The policies related to the setup of the LSP are used to appropriately drive the RSVP-TE daemon. The policies related to the traffic mapping are used to install filters that make the specified traffic flow across the corresponding LSP. An ad-hoc LSP tree made of already established LSPs allows to quickly determine if a new flow has been mapped on an existing LSP. In such a case, the PDP avoids sending policies related to the setup of the LSP. Only a traffic filter has to be installed. This simple scheme enhances the scalability property of the testbed configuration tool.

In addition, for experimental purpose, real traffic can be generated across the LSPs using D-ITG (Distributed Internet Traffic Generator)[6] which allows for a remote control of the sender component (running on each network ingress node). Therefore, the PDP, after receiving the acknowledgment of the LSP setup and traffic mapping, directs the D-ITG sender to generate the requested traffic. For each flow, it is then possible to retrieve information on the experimented throughput, delay, jitter and packet loss.

3.3 The TOTEM Architecture

The kernel of the toolbox is the repository of TE methods (see Fig 4) grouped into several categories:

- IP: algorithms using only IP information (e.g. IGP weight optimisation)
- MPLS: algorithms using MPLS TE functionalities (e.g. LSP primary or

backup computation algorithms)
- BGP: inter-domain algorithms (e.g. traffic redistribution)
- Generic: classical optimisation and search algorithms useful for other parts of the toolbox (e.g. tabu search framework)

Besides this kernel, the topology manager contains all the topological data (i.e., node, link, IGP, BGP and MPLS information). This module is the reference access point to the topology representation in the toolbox. The configuration manager configures the global toolbox parameters and the different algorithms. Finally the web-service interface module provides the standard interface for interoperability with existing external tools.

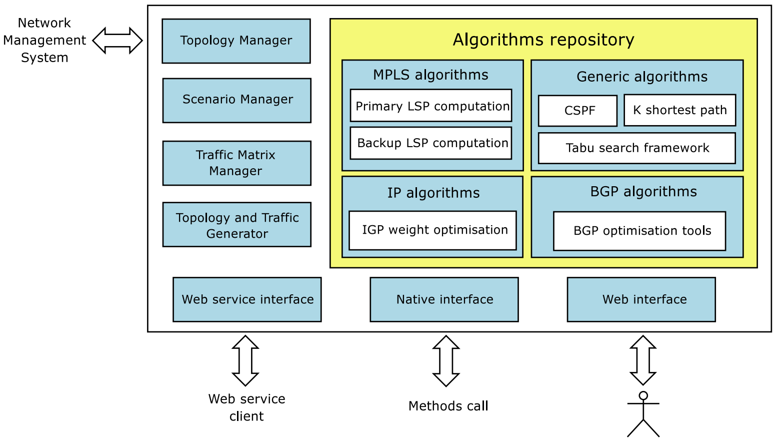

Fig. 4. *TOTEM architecture*

3.4 A standard format for a network topology representation

A common aspect of all the TE methods is that they use a topology representation (as input and/or output). We have chosen the XML language because it is widely used and many tools exist for dealing with this language. So, the XML network topology format can be seen as a common interface between diverse algorithms. We will also provide some tools to convert this format into other common formats (e.g., the BRITE[76], ns-2[81], gt-itm[78], INET[79] formats) and vice versa.

We have developed some tools that can parse network information from routers of a real network (e.g., *show isis* and *show mpls* commands executed on a router via the CLI) and return a file representing the network in our XML format. We can also provide some tools taking the XML topology format as input and producing some results on it (e.g. a graphical representation of the topology). A topology editor[19] could also be used on this format to allow the creation and manipulation of large and complex network simulations scenarios.

Another tool can verify the consistency of the topology. For example, it is possible to verify that all the links are connected to nodes present in the topology, or that the identifiers are unique in the whole file. In the same vein, we have also created an XML Schema[83]. The schema allows us to validate a topology file so that we are sure that an XML instance satisfies the data structure and some basic constraints on the format.

Our XML format is designed to be a single access point where all the different formats and tools converge, reinforcing the collaboration between these tools, see Fig 5.

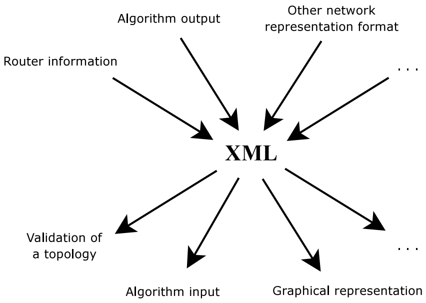

Fig. 5. The XML topology format as a common interface

Obviously, not all the algorithms of the toolbox will use the same topology information. So we decided to define a flexible data format. It can be extended and almost all the attributes and elements are optional. An algorithm using an XML network file as input can simply eliminate the information it does not need.

3.5 Integration of a new algorithm in the toolbox

We have developed the toolbox in Java because it allows rapid and structured development. Moreover, the JNI (Java Native Interface)[80] library allows us to integrate C and C++ algorithms in the toolbox.

The toolbox has been designed to facilitate the integration of new algorithms by providing different generic services. It provides topology information (nodes, links, LSPs,. . .) to the algorithm to be integrated. It also provides a scenario execution service. This service parses an XML file describing a scenario (for example, a sequence of LSP computation requests) and then calls the appropriate algorithm to execute the scenario. This is useful for simulation purposes.

To be integrated in the toolbox, every algorithm must implement two methods called *start* and *stop*. The former is used to instantiate and configure the algorithm and send it all the information related to the current state of the topology, while the *stop* method is used to terminate the algorithm. Depending on the type of algorithm, additional methods must be implemented. For example, for MPLS routing algorithms, a route method must be implemented (the method called by the generic scenario execution service). This route method is susceptible to return a list of actions (addLSP, preemptLSPs,. . .).

4 TE algorithms of the toolbox

In most cases, the IP routing protocol is not aware of the load on the various parts of the network and selects for each destination the shortest path based on static metrics such as the hop count or the delay. This destination-based routing creates an uneven distribution of the traffic that may lead to periods of congestion in the network. Several techniques have been proposed to better spread the load throughout the entire network [9]. A first solution is to select appropriate link metrics based on a known traffic matrix [27]. This solution can provide some interesting results if the traffic matrix is known and stable. A second solution is to rely on a connection-oriented layer-2 technology [10] such as ATM, MPLS or one of the emerging optical technologies. In this case, layer-2 connections can be established statically or dynamically between distant routers and the layout of these connections can be optimised to achieve an even distribution of the traffic inside the network [9]. It is also possible to dynamically create new layer-2 connections in order to quickly respond to link failures or changes in the traffic pattern [9].

This section will summarize some Traffic Engineering methods that we have designed recently. They are classified into three categories: (1) intra-domain IP-based, (2) inter-domain IP-based, and (3) MPLS-based.

4.1 Intra-domain IP-based traffic engineering algorithms

Over the last several years, many different approaches have been proposed for traffic engineering in IP networks. Most proposals can roughly be assigned to two distinct groups: approaches based on off-line optimisation, and approaches based on algorithms which operate in the control plane of the network. Global link weight optimisation for a given traffic demand matrix is representative of the former group, whereas enhancements to current routing protocols, like e.g. the Optimised Multi-Path (OMP)[82] algorithm, are representative of the latter. Both philosophies have specific benefits and drawbacks: approaches based on optimisation necessitate knowledge of the traffic demand matrix and they usually require additional network management efforts, whereas OMP requires sophisticated data structures in the nodes, and produces non-deterministic signalling overhead.

4.1.1 IGP weight optimisation algorithms

The basic model in the weight optimisation problem assumes a given static topology and a fixed demand matrix. The network is represented by a directed graph $G = (N, A)$ where N and A denote the set of routers and links connecting them, respectively. The objective is to maintain the utilization of links within given link capacities. For this reason, a convex piecewise linear cost function increasing with the utilization rate is defined for each link. The idea behind the cost function is that the penalty for assigning an additional load to the link grows with the load on the link.

In a general routing problem, it is assumed that there is no restriction on the distribution of flows over alternative paths. However, in SPF applications, a flow is either distributed evenly among all the shortest paths of an $(s, d) \in N \times N$ pair, referred as equal-cost multipath [50], or routed through a single shortest path. Given these conditions regarding traffic splitting, the IGP weight optimisation problem becomes NP-hard and therefore efficient heuristics are needed to tackle this weight setting procedure.

The search procedure includes a heuristic algorithm based on tabu search [29]. A solution is represented with a weight vector, $(w_a)_{a \in A}$. Two types of neighbourhood functions are defined as follows:

- Single weight change: The weight of a single link is swapped at each iteration.
- Evenly balancing flows: The weight vector is adjusted in a way such that the flows targeted to the router t going through the router u are distributed evenly among the links leaving u.

On a more technical side of the search algorithm, special hash functions are used to facilitate the tabu aspect of the heuristic, as well as to improve the running time. As observed in [27], OSPF performs well with optimised weights in realistic network topologies. The results have shown that the max-utilization rate in OSPF networks with optimised weights is generally close to the one in the ideal case where the traffic is split freely.

Employing an efficient solution technique for this highly complex problem is of great importance for practical purposes. Comparing performance qualities of several heuristic techniques may provide better solutions in shorter CPU times. In order to realize this, a generic software system would be extremely effective.

4.1.2 Optimised multi-path routing algorithms

Optimised Multi-Path (OMP) routing can divide the traffic unequally among multiple parallel paths. We first propose a method based on flow optimisation applicable when the traffic matrix is known, and then an adaptive distributed method.

A multi-path routing algorithm based on flow optimisation

The general problem of finding the best way to route traffic through a network can be mathematically formulated as a multi-commodity flow optimisation problem. With a flow optimisation the network capacity constraints and overall traffic characteristics are taken into account. The input to the optimisation is the network topology, the link capacities and an estimate of the traffic demand between each pair of edge nodes in the network. The output of the optimisation is a routing that gives the optimal flow on each link, according to a cost function.

In [1] an intra-domain routing algorithm based on multi-commodity flow optimisation is presented and an optimising routing architecture where this algorithm fits in is outlined. The algorithm is computationally tractable for on-line optimisation, it requires only small modifications to the packet forwarding mechanisms used today, and it enables a load sensitive routing over several paths that is optimal according to some traffic engineering objective.

By modelling the routing problem in such a way that all traffic to a certain egress node in the network is aggregated into one commodity the number of commodities is reduced to N, the number of nodes. This way of modelling the problem both makes the optimisation computationally tractable and also makes the output from the optimisation well suited for packet forwarding in the routers. The output tells each router how traffic to a certain egress node in the network should be divided between its set of outgoing links. So, if a mapping between destination addresses and egress nodes is added to the forwarding process then the traffic can be distributed over multiple links using a hashing mechanism similar to the one already in use today for the equal cost multi-path extension to OSPF.

The result of the optimisation, how the traffic is distributed in the network, very much depends on the objectives expressed in the cost function that is part of the optimisation. Since one of the main goals with traffic engineering is to avoid congestion it is desirable to balance the load in the network and distribute it in such a way that no link becomes overloaded. Here a cost function is used which allows a network operator to choose a maximum desired link utilisation level. The optimisation then finds the most efficient solution satisfying this constraint. Efficient here means that the traffic takes the shortest paths possible.

The Adaptive Multi-Path Algorithm

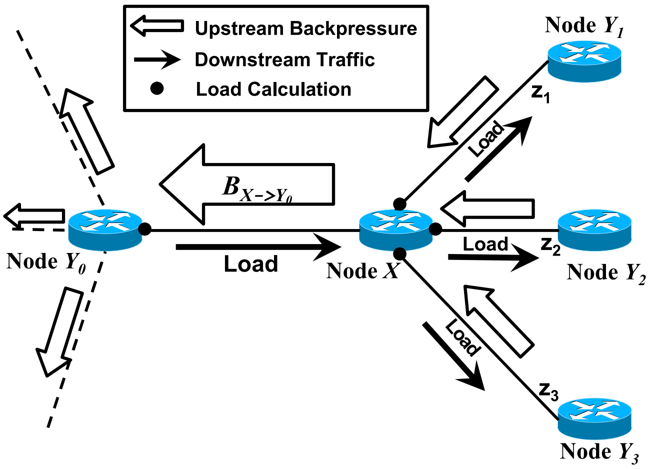

Fig. 6. *Example for a BackPressure Message sent from X to Y_0*

As an alternative to multipath routing solutions based upon global flow optimisation, the Adaptive Multi-Path algorithm (AMP) [30,31] aims at performing traffic engineering by employing only a local view of the network in each node (see Figure 6). With AMP, congestion on a generic link z_i does not result in a multitude of nodes reacting immediately to this change by off-loading some of their paths containing z_i. In contrast, only X as the end node of z_i is concerned and tries to shift away as much traffic as possible onto alternative paths. Additionally, X informs its neighbour nodes Y_j, $j \neq i$, about their contribution to congestion on link z_i by sending them so-called backpressure messages. Figure 6 depicts an example backpressure message sent from X to Y_0, summarizing the congestion on links $z_1, z_2, z_3, ...$, where Y_0 in turn reacts by offloading its link towards X (in case Y_0 is significantly contributing to congestion). At the same time Y_0 sends similar backpressure messages to its neighbour nodes, informing them about their respective contributions to congestion, etc.

This quasi-recursive signalling architecture of AMP achieves seemingly contrary goals: the signalling of load information is restricted only to neighbour nodes, and at the same time load information is propagated throughout the entire network domain. AMP operates autonomously in the control plane of the network, without requiring any manual interventions, it does not require complex data structures and produces low and deterministic signalling overhead. With AMP, the traffic distribution in the network eventually converges to an equilibrium fix-point for any given traffic demand matrix.

AMP has been implemented in the *Network Simulator* (ns-2) [81], and has been simulated for real ISP topologies (AT&T-US network and German B-WiN Research Network) and realistic traffic patterns (Web traffic with spatial distribution according to the gravity model). The performance investigations have shown significant performance improvements, e.g., reductions in Web page response time of up to 43%, compared to the currently used static routing schemes like shortest path routing (SPR) and equal cost multi-path routing (ECMP).

4.2 Inter-domain IP-based Traffic Engineering

The current state-of-the-art in inter-domain traffic engineering is primitive [9]. Operators change their routing policies and the BGP attributes of the routes manually without a proper understanding of such changes on the flow of the traffic. Many problems arise due to misconfigurations in the routers [46]. The current practice in BGP-based traffic engineering is often "trial-and-error" [22], i.e. an operator changes the BGP attributes of some routes that were observed to carry a large amount of traffic and observes the effect on the inter-domain traffic. For large transit ISPs, inter-domain traffic engineering is a complex problem even for outbound traffic due to interactions between BGP and the IGP [3]. In the case of stub ASes on the other hand, the reason for the absence of a proper engineering of BGP is mainly a lack of understanding of the working of BGP and its effect on the traffic. In this section, we present a new BGP simulator aimed at reproducing the routing of large ISP networks, and we describe the architecture of a tool that leverages this simulator for inter-domain TE purposes. Finally, we propose an overlay architecture for inter-domain TE.

4.2.1 C-BGP: A new BGP simulator

For the purpose of evaluating how BGP behaves in the global Internet, we developed a new and efficient open-source BGP simulator, *C-BGP* [53]. A new simulator was required because the other available open source simulators [52,62] are not able to model networks as large as the Internet. The reason is that these simulators are general purpose packet-level simulators and as soon as the size of the simulated topology increases, the simulation quickly becomes untractable. Therefore, the simulation results available from the literature are often based on small topologies composed of only up to a few tens of BGP routers. By contrast, *C-BGP* has been specifically written for the purpose of simulating BGP. *C-BGP* is written in C, has been released under the LGPL license and has been used to perform simulations with more than 15.000 BGP routers.

Simulating BGP in a topology similar to the global Internet is challenging. The BGP decision process is complex by nature because of its rules which define different sometimes contradictory orderings on the routes. Moreover, most BGP decisions are local but can affect the information available to all the other routers. In addition to this, when BGP policies come into play, things become even more intricate. There is thus no easy shortcut in simulating BGP as it is the case for a link state protocol like OSPF where a Dijkstra search in a graph is possible. The most efficient and straightforward method to simulate BGP is to build a realistic implementation of the decision and filtering processes and to follow the propagation of messages.

In *C-BGP*, each BGP router is modelled as a data structure containing its RIB, Adj-RIB-IN and Adj-RIB-OUT [33]. Each simulated BGP router is configured by specifying its physical interfaces, its eBGP and iBGP peers and the filters that are used on the sessions with these peers. *C-BGP* supports filters similar to those used on normal BGP routers. *C-BGP* simulates the BGP messages that are used to advertise and withdraw prefixes over BGP sessions. These BGP messages can contain any valid BGP attribute. When a simulated BGP advertisement is received, this message is placed in the Adj-RIB-IN of the simulated router and the appropriate import filter is used. The BGP decision process is then run and a new BGP message is sent if a change in the best route occurred. In addition to this, *C-BGP* models a simplified session establishment protocol. For scalability reasons, *C-BGP* does not model the other BGP messages (KEEPALIVE, . . .), the underlying TCP connection and the various BGP timers (MRAI, HoldTimer, BGP dampening). Those mechanisms are important when evaluating transient issues such as the convergence of BGP but do not influence the selection of the best route with the standard BGP decision process [55].

In addition to be a simulator, *C-BGP* can be used as a tool to evaluate what-if scenarios. *C-BGP* is able to load real routing tables provided in the widely used MRT format. It can also process UPDATE/WITHDRAW messages collected on real routers. *C-BGP* can thus be used by a network operator to evaluate what-if scenarios, based on information collected on its routers and without impacting the real traffic. For instance, *C-BGP* can be used to evaluate the impact of different policies on the routing choices and on the propagation of the routes in a real network. Another utilization of *C-BGP* in a real operational environment is to evaluate the impact of the failure of an intra-domain link or of a peering link. Indeed, many decisions taken by BGP depends on the IGP cost of intra-domain paths. Changes in these costs can have a dramatic impact on the BGP choices and on the traffic eventually.

4.2.2 BGP-based outbound TE algorithms

To compute the BGP tweaking to perform traffic engineering, our solution is to rely on C-BGP [53] to precisely reproduce the routing inside the AS and on a heuristic that interacts with C-BGP to compute the tweakings of the BGP routes.

Figure 7 illustrates the architecture of our solution. The central component is a script that manages the different inputs and communicates with C-BGP. The script receives as input the BGP RIB's and BGP updates received from the external peers, as well as the traffic statistics. The main script also needs the internal topology, IGP weights, and BGP routing policies enforced by each BGP router of the AS.

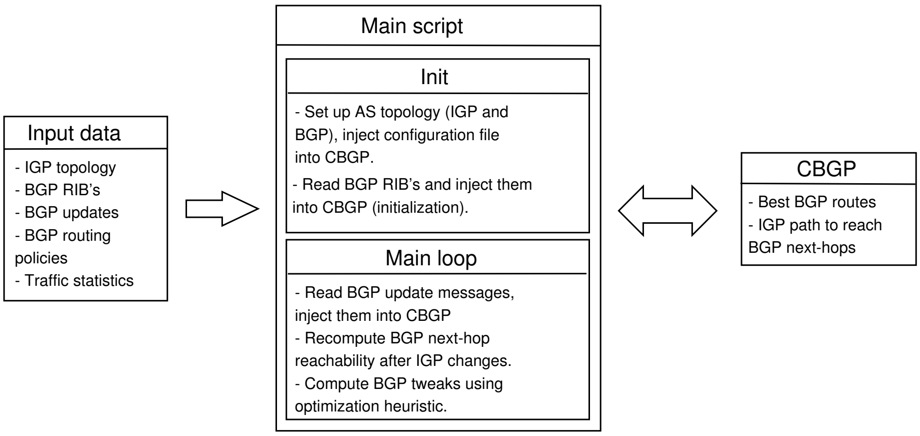
Input data
- IGP topology
- BGP RIB's
- BGP updates
- BGP routing policies
- Traffic statistics
Main script
Init
- Set up AS topology (IGP and BGP), inject configuration file into CBGP.
- Read BGP RIB's and inject them into CBGP (initialization).
Main loop
- Read BGP update messages, inject them into CBGP
- Recompute BGP next-hop reachability after IGP changes.
- Compute BGP tweaks using optimization heuristic.
CBGP
- Best BGP routes
- IGP path to reach BGP next-hops

Fig. 7. Interaction between components for BGP-based inter-domain TE.

With this information, the script builds the C-BGP configuration file it will inject into C-BGP. Then, the RIB's of the border routers having peerings with other ASes are injected into C-BGP to populate the BGP routing tables of all BGP routers inside the AS. This finishes the initialization phase of the script.

The second phase is to compute the tweakings needed for the traffic engineering using an optimisation heuristic. The script then interacts with C-BGP to maintain an up-to-date state of the BGP information of each ingress point of the AS towards each destination prefix. As the traffic engineering does not need to care about the prefixes towards which too small an amount of traffic is sent, we maintain into C-BGP only the BGP routes towards popular destination prefixes. [63,56,65] have shown that most of the traffic is sent to a limited fraction of the destination prefixes.

The heuristic we designed to compute the traffic engineering changes is based on evolutionary optimisation and has been described in details in [64]. Based on this heuristic, we have developed solutions that tweak BGP routes both in the case of stub ASes [66] and transit ASes [67].

4.2.3 An overlay architecture for inter-domain TE

This approach uses BGP to establish a static provisioning (see Fig. 8). For instance, based on the communities attribute of BGP [54], AS1 could request AS4 to prepend its own AS three times before announcing C1 to AS2, to prepend it two times before announcing C1 to AS6, and to perform no prepending operation at all when announcing this block to any other neighbouring AS. Therefore, the advertisements that AS2 receives under this scenario are: {AS4, AS4, AS4, AS1}; {AS6, AS5, AS1}. Then AS2 chooses to forward C1 through AS6. Nevertheless, once this is done, the best path chosen by BGP is completely unaware of any kind of dynamic TE requirements or constraints between AS1 and AS2.

Let us assume now that the link between AS2 and AS6 becomes loaded, while the path {AS4, AS1} through R22 does not. Despite these unequal network conditions, TE-BGP will still prefer the path through R21. The distributed Overlay Architecture approach allows the Overlay Entity (OE) within AS2 to become conscious of these conditions and dynamically reroute its outbound traffic of C1 through R22. An advantage of this approach is that BGP updates could be completely avoided if, for example, the LOCAL PREFERENCE (LOCAL_PREF) is used when reallocating this traffic. These kinds of complementary solutions become perfectly suitable when inter-domain traffic patterns need to dynamically adapt and rapidly react to medium or high network changing conditions, where the in-band TE solutions seems impracticable at the present time.

This mechanism allows OEs to influence the underlying BGP routing layer to take rapid and accurate decisions to bypass some network problems such as link failures, or low-grade service for a given Class of Service (CoS). For this reason the Overlay Architecture may also be used for QoS Routing on top of a mix of QoS-aware and QoS-unaware BGP routers [69].

4.3 Traffic Engineering with MPLS

One of the most interesting applications of MPLS in IP-based networks is Traffic Engineering[8]. The main objective of TE is to optimise the performance of a network through an efficient utilization of the network resources. The optimisation may include the careful creation of new Label Switched Paths (LSPs) through an appropriate path selection mechanism, the re-routing of existing LSPs to decrease the network congestion and the splitting of the traffic between several parallel LSPs.

According to IETF RFC 3272[9], TE schemes for congestion control can be classified according to their response time scale and their congestion manage-

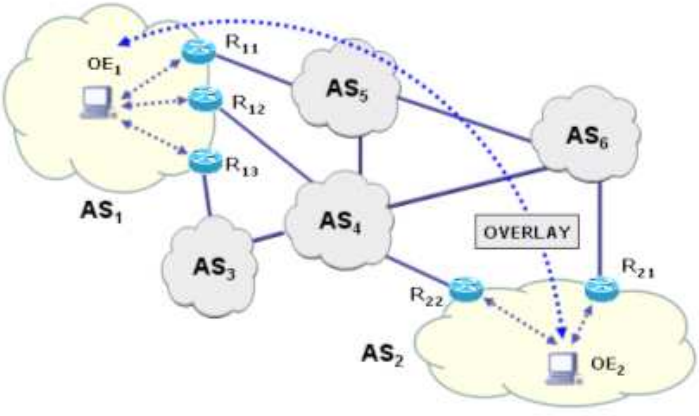

Fig. 8. *Inter-domain scenario where Overlay Entities (OE) are used for dynamic QoS provisioning among remote multi-homed ASes.*

ment policies (reactive or preventive).

Most of the proposed schemes are *preventive*, they allocate paths in the network to achieve certain QoS, to balance the traffic load or to prevent congestion. Two known mechanisms in MPLS networks are Constraint-Based Routing (CBR) and traffic splitting. Preventive methods will be described in section 4.3.1.

The preventive behaviour is not sufficient: when LSPs are set up and torn down dynamically, these schemes can lead to inefficiently routed paths and to future blocking conditions over specific routes. Therefore, preventive methods are complemented by reactive ones, such as LSP re-routing and LSP bandwidth adaptation, which will be presented in section 4.3.2.

4.3.1 Preventive methods

Two main basic classes of methods will be described: Constraint-based routing of LSPs, and Routing of backup LSPs for fast restoration.

Constraint-based Routing

We take for granted the capability of routing flows along explicitly calculated routes. This possibility is actually offered by MPLS networks, when advanced label distribution protocols (e.g. RSVP-TE) are employed. Furthermore, we consider LSP Service Level Specifications (SLS)[20] composed essentially of a bandwidth demand, a QoS class and a pre-emption level. Under these assumptions, the traffic engineering problem is: given a well-defined Service Level Specification for the LSP, find the path that guarantees the SLS, while at the same time optimising network resource usage. There are however several ways to optimise the resources.

One possibility is to minimize the probability that new LSP requests are rejected due to the lack of resources. Such minimization obviously results in an optimisation of network provider's revenues. One of the approaches towards the above objective is the application of load-balancing techniques.

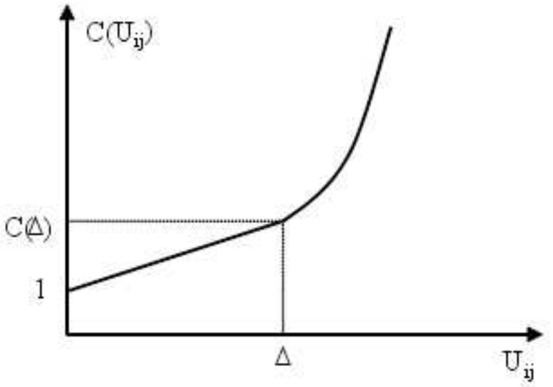

Fig. 9. *Cost function*

In [21] this issue is addressed by assigning appropriate weights to the network links. The main contribution resides in having devised a solution relying on a link weight that depends on the link utilization in a non-linear fashion. More precisely, a link weight is a function which takes into account both the available bandwidth and a bandwidth threshold 'Δ', whose value can depend on both traffic profile and network topology (see Figure 9). The weight assignment algorithm uses a cost function that exhibits, for each link, the following behaviour:

- it grows linearly as long as the percentage of already-allotted bandwidth on the link is less than Δ;
- as soon as such a percentage exceeds Δ, it assumes an exponential profile.

By using the Network Simulator, tests have been carried out to analyze the behavior of a traffic-engineered MPLS network for several different Δ values, while leaving unchanged both traffic load and network topology. Packet losses and traffic distribution have been measured in order to evaluate the network behaviour. Results of such tests are reported along with an analysis of the performance achieved by the network, in terms of SLS acceptance ratio.

The DAMOTE (Decentralized Agent for MPLS On-Line Traffic Engineering) tool [49,16,17], also addresses this issue of Constraint-Based Routing. DAMOTE computes a primary path like the classical CSPF (Constraint Shortest Path First), but generalizes it in several ways. While CSPF is a simple SPF on a pruned topology, obtained by removing links that have not enough resources to accept the new LSP, DAMOTE can perform much clever optimisations based on the minimization of a network-wide score function. Examples of such functions are: resource usage (thus leading to a traditional shortest path), load balancing, hybrid load balancing (where long detours are penalized), preemption-aware routing (where LSP reroutings are penalized). DAMOTE is generic in the sense that this score function is a parameter of the algorithm. For example, DAMOTE can mimic the previous method by choosing link weights that are inversely proportional to the unreserved capacity and by minimizing the network resource usage. Like in CSPF, constraints can be taken into account, but here again the constraints can be parametrized quite freely. Typical constraints refer to the available bandwidth on links per class type (CT), or to pre-emption levels. For example, it is possible to specify that an LSP of a given CT can only be accepted on a link if there is enough unreserved bandwidth for this CT by counting only the resources reserved by LSPs at higher preemption levels. This allows to preempt other LSPs if needed. In that case, DAMOTE can also calculate the "best" subset of LSPs to preempt.

DAMOTE computes efficiently a near optimal solution, it can cope with various network-wide score function and types of constraints and is compatible with the MAM (Maximum Allocation Model)[44] model proposed in the IETF framework of MPLS/DiffServ[43].

In the decentralized mode the LSP computation is done at the ingress node, which requires to have enough information about all link states at all edge nodes. This is usually achieved by using extensions of link-state routing protocols like OSPF-TE or ISIS-TE, which flood the network regularly with updated link-states.

However, there is a trade-off between the amount of routing information exchanged among routers and the accuracy of the routing information database. As control traffic must be kept to a minimum some routing decision may cause extra connection blocking and non-optimal path selection. In [48] new mechanisms are proposed to reduce the effects on global network performance when selecting explicit paths under inaccurate routing information.

For IP/MPLS networks the proposed routing mechanism is called BBR (Bypass Based Routing). According to this dynamic bypass concept, whenever an intermediate node along the selected path (unexpectedly) does not have enough resources to cope with the incoming MPLS demand, it has the capability to reroute the set-up message through alternative pre-computed paths (bypass-paths).

A new parameter is introduced in the working path selection process to represent the routing inaccuracy. An Obstruct-Sensitive Link (OSL) is a link that potentially is unable to support the traffic requirements according to a certain link definition. This decision is made using the standard routing information while looking for the working path at LSP set-up time. Once the working path is selected and the Obstruct-Sensitive Links are identified the Bypass Discovery Process (BDP) starts. A Bypass Path, if any, is an alternative and disjoined route between the edge nodes of the OSL. In the figure 10 the working path goes via N1-N2-N3-N4 and two OSL are found, namely N1-N2 and N3-N4.

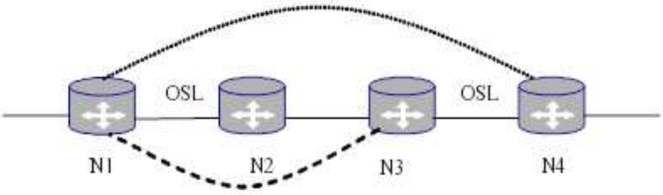

Fig. 10. *Obstruct-Sensitive Link*

Then the DBP finds two bypass paths from N1 to N3 and from N1 to N4. They are used as alternative paths in case the corresponding OSL cannot cope with the incoming traffic. As they are pre-planned alternative paths, the change is made without any problem. As expected, as the number of computed bypass-paths per route increases, blocking probability is reduced.

Fast restoration

Recent surveys on the performance of protection algorithms and MPLS multi-level protection may be found in [18] and [47].

We consider an MPLS network with protected LSPs and rerouting mechanisms based on pre-planned backup LSPs in case of failure. In order to reduce the restoration time and the packet resequencing, a Fast Rerouting Mechanism is recommended. Several schemes have been proposed: (a) A primary LSP is protected by a disjoined edge-to-edge backup LSP, (b) Each link (or node) is protected by a local bypass LSP, (c) Each primary LSP is protected by a series of local detour LSPs.

The mechanism proposed in [34–36] is based on solution (a) and uses a reverse LSP along with the protected LSP so that traffic may be returned to the ingress node and can be re-routed to the alternative (edge-to-edge) disjoined LSP. An extension is the Reliable Fast Rerouting (RFR) mechanism that provides zero packet loss in case of LSP failure and restoration (See Fig. 11).

Finally, a new mechanism is proposed, namely the Optimal and Guaranteed Alternative Path (OGAP), which tries to remove the drawback of pre-planned alternative LSPs and looks for new optimal alternative paths while the protected path is active. This proposal uses an hybrid of fast rerouting and a

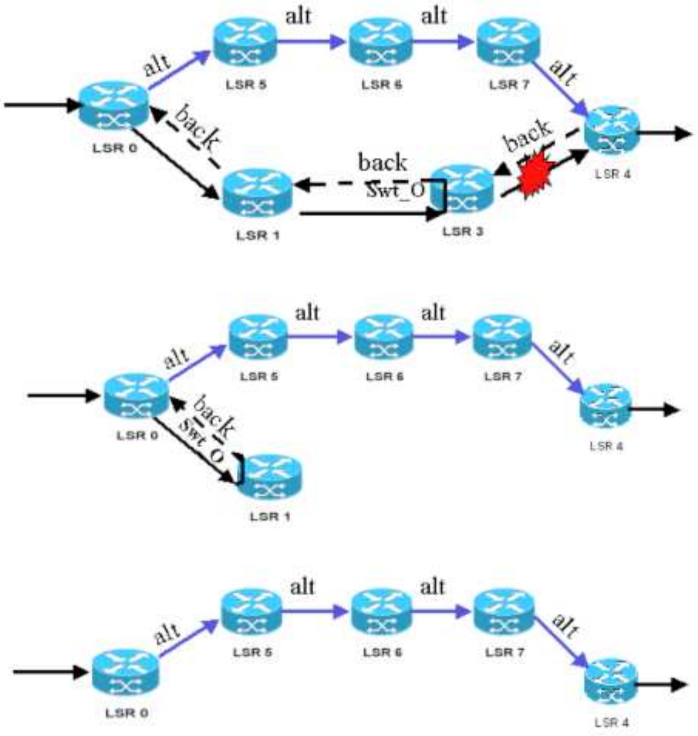

Fig. 11. *Reliable Fast Rerouting*

dynamic approach to establish the optimal alternative LSP while rerouting the protected traffic using the pre-planned alternative LSP. This hybrid approach provides the best of the fast rerouting and the dynamic approaches. As the originally protected path becomes in fact unprotected from additional failures after the traffic has been rerouted, a dynamic approach is used to establish a new alternative pre-planned path. Furthermore, if the new alternative LSP is better, in terms of QoS guarantees, than the current LSP, roles are swapped and the former LSP becomes the working path while the latter becomes the alternative path again.

The method embedded in the DAMOTE tool is based on solution (c). In this approach [49] each primary LSP is typically protected by a series of detour LSPs, each of them originating at the node immediately upstream of any given link on the primary path. Those detour LSPs thus protect the downstream node (if possible) or the downstream link and merge with the primary LSP anywhere between the protected resource and the egress node (inclusive). Those many LSPs have to be pre-established for fast rerouting in case of failure, and provisioned with bandwidth resource. In terms of bandwidth consumption, this scheme is only viable thanks to the fact that detour LSPs are allowed to share bandwidth among themselves (Fig. 12) or with primary LSPs (Fig. 13) under the single-failure hypothesis.

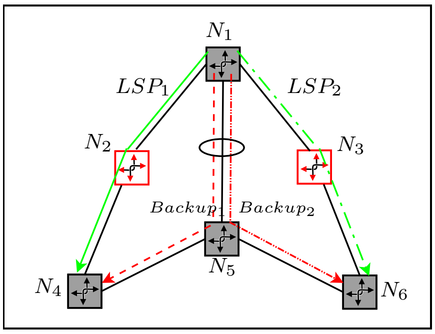

Fig. 12. $Backup_1$ protects LSP_1 from failure of node N_2. $Backup_2$ protects LSP_2 from failure of node N_3. Since $Backup_1$ and $Backup_2$ will never be used simultaneously, they can share bandwidth on link $N_1 - N_5$.

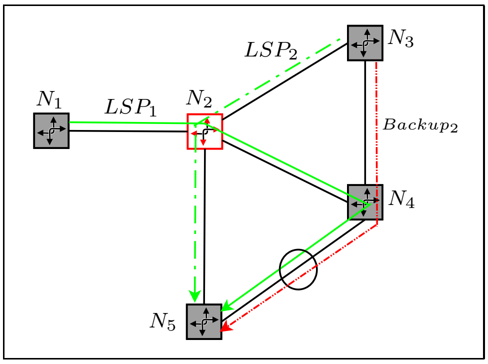

Fig. 13. The two primary LSPs (LSP_1 and LSP_2) will fail together when N_2 fails. $Backup_2$, protecting LSP_2, can share bandwidth with LSP_1 on link $N4 - N5$, since $Backup_2$ and LSP_1 will never use this link simultaneously. $Backup_1$, protecting LSP_1, is not shown on the figure.

DAMOTE achieves full protection of all primary LSPs against link and node failures with a resource over-consumption of 30 to 70% of the resources reserved for primary LSPs, depending on the topology. By contrast SDH/SONET protection leads to 100% of over-consumption without protecting the nodes. Regarding scalability, if the ingress LSPs have to compute all detour LSPs, they need to have access to a substantial amount of information about the states of all links in the network. The solution consisting of flooding this information with OSPF-TE, though possible, scales poorly. Therefore another scalable scheme is proposed [12] that consists of distributing the computation of the series of detour LSPs among the nodes on the primary path. The idea is that each node on the primary path will compute the detour LSP protecting itself (or its upstream link).

4.3.2 Reactive methods

Traffic Engineering schemes based on *reactive* policies have been proposed in the literature recently [4,2,57]. Reactive methods can either re-route LSPs or adapt the bandwidth of LSPs. We will address these two issues in turn.

Two novel schemes are proposed in [57] to reduce the congestion in an MPLS network by using load balancing mechanisms based on different *Local Search* heuristics. The key idea is to efficiently re-route LSPs from the most congested links in the network, in order to balance the overall links load and to allow a better use of the network resources. Network congestion can be detected in two main ways: either when the load on some network links is dangerously close to the link capacity, or when a new LSP demand request cannot be satisfied.

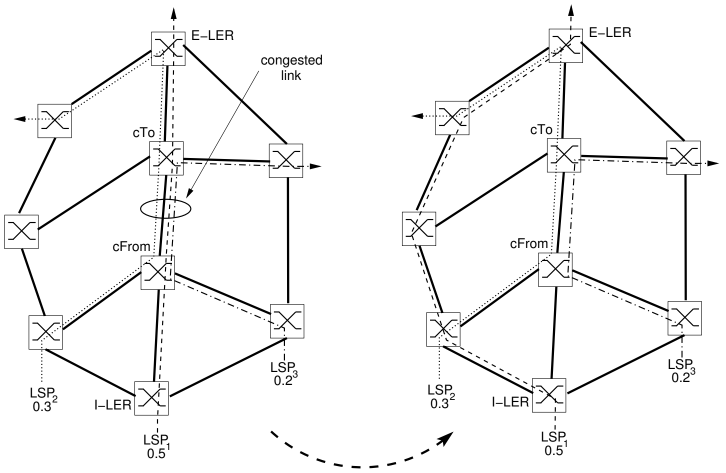

Fig. 14. *The rerouting mechanism of the load balancing algorithms: (left) when link congestion is detected, (right) after LSP_1 rerouting*

Figure 14 shows an example of the rerouting process of the proposed algorithms. Each link of the depicted network has capacity equal to 1. The bandwidth demand for each LSP is a fraction of the link capacity. In the left plot, the link (cFrom, cTo) is detected as congested, and the algorithm triggers the Local Search over the LSPs crossing the link. The LSP whose alternate path guarantees the maximum available capacity in the network (i.e. the minimum network congestion) is LSP_1, so the ingress router I-LER reroute the related traffic over this new path (see Figure 14 (right plot)).

Experiments under a dynamic traffic scenario show a reduced rejection probability especially with long-lived and bandwidth consuming connection requests, thus proving a better network resource utilization compared to existing CBR schemes in MPLS networks, while guaranteeing a reduced computational complexity.

On the other hand, adaptive bandwidth provisioning schemes are proposed in [60,61]. Unlike a number of previous works that simply use a link utilization threshold as a basic factor for provisioning without specifying the role of this threshold on the QoS perception, our schemes pay explicit attention to the packet level QoS. More specifically, they decide the required capacity based on the target QoS constraint $P(packet\_delay > D) < \epsilon$, where D and ϵ are the given delay bound and violation probability, respectively. The input of the provisioning schemes is the aggregate traffic load measured in a slot-by-slot manner, while the bandwidth upgrades are initiated (if necessary) in a window-by-window manner. One resizing window comprises a certain number of slots.

The aggregate input traffic is assimilated by a Gaussian process that has parameters estimated from the traffic dynamics gained from the measurement trace. These parameters can also be derived based on the predicted traffic dynamics. Different prediction rules thus result in different provisioning alternatives. The use of the Gaussian traffic model provides a quantitative relation between the delay violation probability and the required capacity. For the details and performability investigations of the provisioning alternatives, we refer to [60,61].

The schemes can be applied to adaptively resize the bandwidth of LSPs conveying *traffic having a high degree of aggregation* to meet the required QoS of the conveyed aggregate. The high degree of aggregation is necessary to make the Gaussian traffic assumption reasonable.

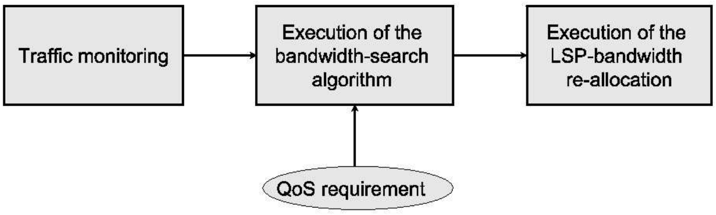
Traffic monitoring
Execution of the bandwidth-search algorithm
Execution of the LSP-bandwidth re-allocation
QoS requirement

Fig. 15. *The building blocks of the implementation of the adaptive resizing schemes*

The implementation of the above LSP resizing schemes comprises building blocks shown in Figure 15 with the following descriptions:

- *Traffic monitoring*: the ingress router of the LSP has to monitor periodically the average traffic rate coming to the LSP. Note that the granularity of monitoring (i.e. the length of measurement slots) and of resizing (i.e. the length of resizing windows) should be configurable parameters.
- *Search algorithm execution*: this is the point where the search for the required bandwidth is executed, using the measured traffic load trace and the QoS requirement (the target delay violation probability). A binary search can be embedded to the router software.
- *Re-allocation of bandwidth*: a signalling protocol is involved to resize the bandwidth of the LSP whenever bandwidth adjustment is needed. By means of RSVP-TE protocol, the procedure can proceed as described in RFC 3209.

In [14], the bandwidth of the LSPs is adapted at periodical time intervals in the range of minutes to hours. In order to reduce the signalling traffic arising at such a high frequency, there is a need to minimize the number of LSP size changes. This requirement leads to a modified LSP dimensioning problem (REOPT), based on a multi-commodity flow problem with multiple explicit paths calculated in advance.

The model consists of a network with U directed links and K commodities (expressed as the tuple source node, destination node, traffic class). The traffic demand d^k is divided between $p(k)$ parallel LSPs. c_j^k represent signalling costs for changing the capacity of pipe x_j^k, and e_j^k are the revenues from routing a bandwidth unit through the network. The objective is to maximize the net revenue $\sum_{k=1}^{K} \sum_{j=1}^{p(k)} e_j^k x_j^k - c_j^k y_j^k$, where the binary decision variables y account for those paths which have to be resized:

$$y_j^k = \begin{cases} 0 \text{ if } x_j^k = x0_j^k \\ 1 \text{ otherwise} \end{cases}$$

The novel part of the formulation consists of the new nonlinear constraint above, where $x0_j^k$ is the current capacity. The remaining two constraints state that the flow on each link should not exceed the reserved link capacity and that the calculated flows sum up in the best case to the traffic demand d^k. The (given) routing information is denoted by P_j^k, which is a binary vector with $|U|$ components having a value of one if the j-th path for commodity k uses the link $u \in U$, and zero otherwise. The QoS paths P_j^k are actually calculated considering the maximal delay constraint for each traffic class and a maximally disjointness metric [42].

In Figure 16, the optimisation component is integrated in a closed-loop provisioning system, in which traffic monitoring and forecasting are essential, as mentioned above. The LSP resizing and the path generation algorithm are written in AMPL and use the CPLEX solver. The LSP changes are reinforced at the ingress routers via the RSVP-TE protocol.

5 Conclusion

We strongly believe that an open source traffic engineering toolbox like the one proposed in this paper would be a suitable alternative to existing commercial tools, both for operators who could benefit from quite a large set of TE methods and select those to deploy, and for academics who could assess their methods against existing ones and on benchmarked data. Used in the latter mode, the toolbox would complement existing simulators (e.g. ns-2), by adding optimisation tools and by offering flow-based simulations instead of packet-based ones.

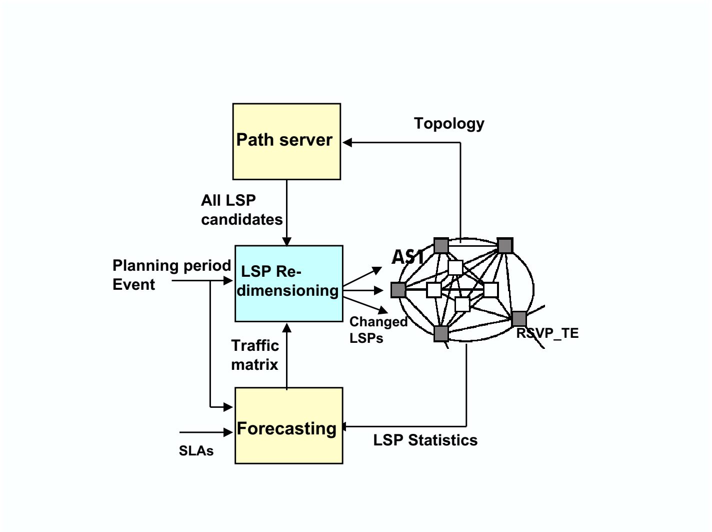

Fig. 16. *Provisioning architecture*

6 Acknowledgment

This work has been partially supported by the European Commission (IST E-Next NoE) and the Walloon Region (TOTEM project).

References

[1] H. Abrahamsson, J. Alonso, B. Ahlgren, A. Andersson, P. Kreuger. A Multi Path Routing Algorithm for IP Networks Based on Flow Optimisation. In *Proceedings of Third COST 263 International Workshop on Quality of Future Internet Services*, QoFIS, Zurich, Switzerland, 2002.

[2] S. Acharya, B. Gupta, P. Risbood, A. Srivastava. MPLS network tuning: enabling hitless network engineering. In *Proceedings of IEEE ICC*, volume 2, pages 1499–1503, Anchorage - Alaska, 2003.

[3] S. Agarwal, C. Chuah, S. Bhattacharyya, C. Diot. The Impact of BGP Dynamics on Intra-Domain Traffic. In *proceedings of ACM SIGMETRICS*, June 2004.

[4] T. Anjali, C. Scoglio, J. de Oliveira, I. Akyildiz, G. Uhl. Optimal Policy for Label Switched Path Setup in MPLS Networks. *Computer Networks*, 39(2):165–183, June 2002.

[5] S. Avallone, S. D'Antonio, M. Esposito, A. Pescapè, S. P. Romano. A Topology Discovery Module based on a Hybrid Methodology. Inter-domain Performance and Simulation Workshop (IPS 2004) - March 2004, Budapest (Hungary).

[6] S. Avallone, D. Emma, A. Pescapè, G. Ventre. Performance evaluation of an open distributed platform for realistic traffic generation. *To appear in Performance Evaluation: An International Journal.*

[7] S. Avallone, M. Esposito, A. Pescapè, S.P. Romano, G. Ventre. An experimental analysis of Diffserv-MPLS interoperability. In *Proc. of ICT2003*, March 2003.

[8] D. Awduche, B. Jabbari. Internet Traffic Engineering using Multi-Protocol Label Switching (MPLS). *Computer Networks*, (40):111–129, Sept. 2002.

[9] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, X. Xiao. Overview and Principles of Internet Traffic Engineering. Internet Engineering Task Force, RFC3272, May 2002.

[10] D. Awduche, J. Malcom, B. Agogbua, M. O'Dell, J. McManus. Requirements for Traffic Engineering Over MPLS. Internet RFC 2702, September 1999.

[11] D. Awduche, J. Agogbua, J. McManus. An Approach To Optimal Peering Between Autonomous Systems in the Internet. In *Proceedings of International Conference on Computer Communications and Networks (IC3N'98)*, Lafayette, Louisiana, October 1998.

[12] S. Balon, L. Mélon, G. Leduc. A scalable and decentralized fast-rerouting scheme with efficient bandwidth sharing. *Submitted to Computer Networks.*

[13] J. Bartlett. Optimizing multi-homed connections. *Business Communications Review*, vol. 32, N. 1, 2002.

[14] S. Bessler. Label switched paths re-configuration under time-varying traffic conditions. In *Proceeding of the 15th ITC Workshop*, Würzburg, 2002.

[15] S. Bhattacharyya, C. Diot, J. Jetcheva, N. Taft. Geographical and Temporal Characteristics of Inter-POP Flows: View from a Single POP. *European Transactions on Telecommunications*, 13(1):5-22, Feb 2002.

[16] F. Blanchy, L. Mélon, G. Leduc. A Preemption-Aware On-line Routing Algorithm for MPLS Networks. *Telecommunication Systems*, vol. 24, nbs. 2-4, pages 187–206, Oct-Dec 2003.

[17] F. Blanchy, L. Mélon, G. Leduc. An efficient decentralized on-line traffic engineering algorithm for MPLS networks. In *proceedings of ITC, Providing QoS in Heterogenous Environments*, vol. 5a, pages 451-460, Aug-Sep 2003.

[18] E. Calle, J. L. Marzo, A. Urra. Protection Performance Components in MPLS Networks. *Computer Communications Journal*, Elsevier 2004.

[19] R. Canonico, D. Emma, G. Ventre. Extended NAM: An NS2 Compatible Network Topology Editor for Simulation of Web Caching Systems on Large Network Topologies. European Simulation and Modelling Conference (ESMc 2003) - October 2003 Napoli (Italy).

[20] G. Cortese, R. Fiutem, P. Cremonese, S. D'Antonio, M. Esposito, S.P. Romano, A. Diaconescu. CADENUS: Creation and Deployment of end-user Services in Premium IP Networks. *IEEE Communications Magazine*, January 2003.

[21] L. D'Alessandro, E. Atteo, S. Avallone, S.P. Romano, G. Ventre. A link weight assignment algorithm for Traffic-engineered Networks. Submitted to the Journal of Communications and Networks (JCN).

[22] N. Feamster, J. Borkenhagen, J. Rexford. Guidelines for inter-domain traffic engineering. ACM SIGCOMM Comput. Commun. Rev., vol 33, number 5, pages 19–30, October 2003.

[23] N. Feamster, J. Borkenhagen, J. Rexford. Techniques for inter-domain traffic engineering. AT&T Research Technical Report 011106-02, November 2001.

[24] N. Feamster, J. Rexford. Network-wide BGP route prediction for traffic engineering. In *Proceedings of SPIE ITCOM Workshop on Scalability and Traffic Control in IP Networks*, August 2002.

[25] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, J. Rexford. NetScope: Traffic engineering for IP Networks. IEEE Network Magazine, 14:11–19, March/April 2000.

[26] B. Fortz, M. Thorup. Optimizing OSPF/IS-IS weights in a changing world. *IEEE Journal in Selected Areas in Communications*, vol 20, nb 4, pages 756-767, 2002.

[27] B. Fortz, M. Thorup. Internet traffic engineering by optimizing OSPF weights. In *Proceedings of IEEE INFOCOM2000 (2)*, pages 519-528, March 2000.

[28] R. Guérin, D. Williams, A. Orda. QoS Routing Mechanisms and OSPF Extensions. In *Proceedings of IEEE Globecom*, 1997.

[29] F. Glover, M. Laguna. Tabu Search. Kluwer Academic Publisher, 1997

[30] I. Gojmerac, T. Ziegler, F. Ricciato, P. Reichl. Adaptive Multipath Routing for Dynamic Traffic Engineering. In *Proceedings of IEEE Globecom 2003*, San Francisco, USA, pages 3058-3062, December 2003.

[31] I. Gojmerac, T. Ziegler, P. Reichl. Adaptive Multipath Routing Based on Local Distribution of Link Load Information. In *Proceedings of the 4th COST 263 International Workshop on Quality of Future Internet Services*, QoFIS'03, Stockholm, Sweden, pages 122-131, October 2003.

[32] R. Govindan, H. Tangmunarunkit. Heuristics for internet map discovery. In *IEEE INFOCOM 2000*, March 2000 Tel Aviv (Israel).

[33] B. Halabi. Internet Routing Architectures (2nd Edition). Cisco Press, 2000.

[34] L. Hundessa, J. Domingo-Pascual. Fast Rerouting mechanism for a protected Label Switched Path. *The Tenth IEEE International Conferencen on Computer Communications and Networks (ICCCN 2001)*, pp. 527-530. October 15-17, 2001. Phoenix, Arizona, USA. ISBN 0-7803-7128-3.

[35] L. Hundessa, J. Domingo-Pascual. A Reliable QoS Provision and Fast Recovery Method for Protected LSP in MPLS-based Networks. *IEEE International Conference on Networking (ICN 2002)*, Atlanta, Georgia, USA. August 26-29, 2002. Proceedings ICN2002, pp. 307-318. ISBN 981-238-127-9.

[36] L. Hundessa, J. Domingo-Pascual. Reliable and Fast Rerouting Mechanism for a Protected Label Switch Path. *IEEE Global Telecommunications Conference*, GLOBECOM 2003, The World Converges. Taipei. Taiwan, 17-21/11/2002. Proceedings of Globecom'03, pp. 1608-1612. ISBN 0-7803-7632-3.

[37] K. Kar, M. Kodialam, T.V. Lakshman. Minimum Interference Routing of Bandwidth Guaranteed Tunnels with MPLS Traffic Engineering Applications. *IEEE Journal on Selected Areas in Communications*, December 2000.

[38] K. Kar, M. Kodialam, T.V. Lakshman. Routing Restorable Bandwidth Guaranteed Connections using Maximum 2-Route Flows. In *Proceedings of IEEE INFOCOM 2002.*

[39] M. Kodialam, T.V. Lakshman. Dynamic Routing of Locally Restorable Bandwidth Guaranteed Tunnels using Aggregated Link Usage Information. In *Proceedings of IEEE INFOCOM 2001.*

[40] J-F. Lalande, M. Syska, and Y. Verhoeven. Mascopt - A Network Optimisation Library: Graph Manipulation. INRIA Technical Report RT-0293, April 2004, Sophia-Antipolis, `http://www-sop.inria.fr/rapports/sophia/RT-0293.html`.

[41] F. Lam. Inter-domain router placement and traffic engineering. In *Proceeding of ICC2002.*

[42] S.S. Lee, M. Gerla. Fault Tolerance and Load Balancing in QoS Provisioning with multiple MPLS-Paths. *Lecture Notes on Computer Science*, 2092, IWQoS, 2001.

[43] F. Le Faucheur, W. Lai. Requirements for Support of Differentiated Services-aware MPLS Traffic Engineering. RFC 3564, July 2003.

[44] F. Le Faucheur, W. Lai. Maximum Allocation Bandwidth Constraints Model for Diff-Serv-aware MPLS Traffic Engineering.
Draft `draft-ietf-tewg-diff-te-mam`, September 2003.

[45] G. Li, D. Wang, C. Kalmanek, R. Doverspike. Efficient Distributed Path Selection for Shared Restoration Connections. In *Proceedings of IEEE INFOCOM 2002.*

[46] R. Mahajan, D. Wetherall, T. Anderson. Understanding BGP Misconfigurations In *Proceedings of ACM SIGCOMM*, September 2002

[47] J. L. Marzo, E. Calle, C. Scoglio, T. Anjali. QoS On-Line Routing and MPLS Multilevel Protection: a Survey. *IEEE Communication Magazine*, vol. 41(10), pp. 126-132, October 2003.

[48] X. Masip-Bruin, S. Sànchez-López, J. Solé-Pareta, J. Domingo-Pascual. QoS Routing Algorithms under Inaccurate Routing Information for Bandwidth Constrained Applications. In *proceedings of the IEEE International Communications Conference* (ICC'03). Vol. 3, pp. 1743-1748. Achorage, Alaska, USA, (11-15/5/2003). ISBN 0-7803-7802-4.

[49] L. Mélon, F. Blanchy, G. Leduc. Decentralized Local Backup LSP Calculation with Efficient Bandwidth Sharing. In *proceedings of IEEE ICT*, pages 929–937, Papeete - Tahiti, February 2003.

[50] J. Moy. OSPF Version 2. RFC 2328, April 1998

[51] J.C. de Oliveira, C. Scoglio, I.F. Akyildiz, G. Uhl. A new preemption policy for diffserv-aware traffic engineering to minimize rerouting. In *Proceedings of IEEE Infocom*, June 2002.

[52] B. J. Premore. SSF Implementations of BGP-4. 2001.
Available from `http://www.cs.dartmouth.edu/~beej/bgp/`

[53] B. Quoitin. C-BGP, an efficient BGP simulator. `http://cbgp.info.ucl.ac.be`, September 2003.

[54] B. Quoitin, S. Uhlig, C. Pelsser, L. Swinnen, O. Bonaventure. Inter-domain traffic engineering with BGP. IEEE Communications Magazine, May 2003.

[55] Y. Rekhter, T. Li, S. Hares. A Border Gateway Protocol 4 (BGP-4). Internet draft, draft-ietf-idr-bgp4-22.txt, work in progress, April 2003.

[56] J. Rexford, J. Wang, Z. Xiao, Y. Zhang. BGP Routing Stability of Popular Destinations. In *Proceedings of the second ACM SIGCOMM Internet Measurement Workshop*, November 2002.

[57] E. Salvadori, R. Battiti. A Load Balancing Scheme for Congestion Control in MPLS Networks. In *Proceedings of IEEE ISCC*, volume 2, pages 951–956, Antalya - Turkey, 2003.

[58] D. Stamatelakis, W. D. Grover. IP Layer Restoration and Network Planning Based on Virtual Protection Cycles. *IEEE Journal on selected areas in communications*, 2000.

[59] S. Suri, M. Waldvogel, D. Bauer, P.R. Warkhede. Profile-Based Routing and Traffic Engineering. *Computer Communications*, February 2003.

[60] H. T. Tran, T. Ziegler. Adaptive Bandwidth Provisioning with Explicit Respect to QoS Requirements. In *LNCS 2811 in Proceedings of the QoFIS'03 conference*, pages 83-92, Sweden, October 2003.

[61] H. T. Tran, T. Ziegler. On the Adaptive Bandwidth Provisioning Schemes. In *Proceedings of the IEEE ICC'04 conference*, France, June 2004.

[62] H. Tyan. Design, realization and evaluation of a component-based compositional software architecture for network simulation. Ohio State University, 2002.

[63] S. Uhlig, O. Bonaventure. Implications of Interdomain Traffic Characteristics on Traffic Engineering. *European Transactions on Telecommunications*, January 2002.

[64] S. Uhlig. A multiple-objectives evolutionary perspective to inter-domain traffic engineering in the Internet. Workshop on Nature Inspired Approaches to Networks and Telecommunications (NIANT) in PPSN04, Birmingham, UK, September 2004.

[65] S. Uhlig, V. Magnin, O. Bonaventure, C. Rapier, L. Deri. Implications of the Topological Properties of Internet Traffic on Traffic Engineering. In *Proceedings of ACM SAC'04*, March 2004.

[66] S. Uhlig, O. Bonaventure. Designing BGP-based outbound traffic engineering techniques for stub ASes. *ACM SIGCOMM Computer Communication Review*, volume 34, number 5, 2004

[67] S. Uhlig, B. Quoitin. BGP-based inter-domain traffic engineering for transit ASes. Under submission, `http://cbgp.info.ucl.ac.be/apps.html#section_transit_te`.

[68] I. Van Beijnum. BGP : Building reliable networks with the Border Gateway Protocol. O'Reilly and associates, 2002.

[69] M. Yannuzzi, A. Fonte, X. Masip, E. Monteiro, S. Sánchez, M. Curado, J. Domingo. A proposal for inter-domain QoS Routing based on distributed overlay entities and QBGP. In *Proceedings of the First International Workshop on QoS Routing (WQoSR)*, LNCS 3266, Barcelona, Catalunya, Spain, October 2004.

[70] S. Van den Bosh, C. Wittoeck. Traffic Engineering, Global Route Optimisation - a mechanism to increase the traffic-handling capability of existing MPLS/IP networks. Alcatel White Paper.

[71] Z. Wang, J. Crowcroft. Quality of Service Routing for Supporting Multimedia Applications. *IEEE JSAC*, 14(7):1288-1234, September 1996.

[72] Cariden MATE `http://www.cariden.com/products/`

[73] IP/MPLSView (Wandl) `http://www.wandl.com/html/mplsview/MPLSview_new.cfm`

[74] Conscious (Zvolve) `http://www.zvolve.com/conscious.html`

[75] Opnet products `http://www.opnet.com`

[76] Brite Topology Generator `http://www.cs.bu.edu/brite/`

[77] Cisco Tunnel Builder Pro `http://www.cisco.com/en/US/products/sw/netmgtsw/ps4731/`

[78] GT-ITM: Georgia Tech Internetwork Topology Models `http://www.cc.gatech.edu/projects/gtitm/`

[79] INET topology generator `http://topology.eecs.umich.edu/inet/`

[80] Java Native Interface `http://java.sun.com/j2se/1.4.2/docs/guide/jni/`

[81] Network Simulator ns-2 `http://www.isi.edu/nsnam/ns/`

[82] OMP `http://www.fictitious.org/omp`

[83] XML Schema `http://www.w3.org/XML/Schema`