On-line Routing of MPLS Tunnels with Time-Varying Bandwidth Profiles

Fabio Ricciato, Ugo Monaco, Marco Listanti
INFO-COM dept. of University "La Sapienza" of Rome, Italy
Email: {fabio,monaco,listanti}@infocom.uniroma1.it

***Abstract*— In this work we consider the problem of routing bandwidth-guaranteed flows with time-variable bandwidth profiles on a MPLS network. We assume that each demand is routed in an explicitly routed LSP, and the amount of bandwidth that must be reserved along the LSP varies during the day according to a piece-wise mask which is known in advance. The time-of-day bandwidth profiles can be explicitly declared by the VPN customers in the SLA, or alternatively predicted by the ISP based on past measurements.**

In this framework, we propose a simple on-line algorithm for optimal selection of LSP paths. We also provide a ILP formulation for the associated off-line problem, and adopt it as a reference performance bound for the on-line algorithm.

Additionally, we compare the performances of *fixed* and *variable routing* in presence of time-variable bandwidth profiles. The results presented here suggest that the *a priori* knowledge of the per-demand traffic profiles can be exploited to achieve a fixed routing configuration which can be marginally improved by variable reconfigurations. We relates our findings with a couple of previous works that in different application contexts achieved similar results.

[1]

I. Introduction

In this work we consider the problem of routing bandwidth-guaranteed flows with time-variable bandwidth profiles on a MPLS network. We assume that each flow (demand) is routed in a dedicated Label Switched Path (LSP), and that the amount of bandwidth that must be reserved for a LSP varies during the day according to a piece-wise mask which is known in advance, denoted as *bandwidth profile*. The per-demand time-of-day bandwidth profiles can be explicitly declared by the customers and included in the Service Level Agreement (SLA) with the ISP. This would allow the delivering of a more flexible services, namely the "time-varying Virtual Private Networks" (tv-VPN), aimed at better matching the customer requirements in terms of bandwidth reservation timing. With this approach, the customer requirement might resemble something like:

> "*I want a pipe from site S to site D with assured* 10 *Mbps from 8.00 AM to 6.00 PM, with* 5 *Mbps from 6.00 AM to 10.00 PM and only 1 Mbps from 10.00 PM to 8.00 AM*".

On the customer side, the difficulty in estimating her/his bandwidth needs in the form of a time-of-day vector is not far from that associated to the estimate of a single bandwidth value, as is done in current SLAs. On the operator side, the possibility to distinguish the customer bandwidth provisioning on a per-hour basis opens the way to the introduction of differentiated billing models (e.g., night-time connectivity cheaper). Such models have been extensively used in traditional circuit switched telephone networks, and demonstrated helpful to the operator to modulate the customer behavior in order to optimize and distribute (in time) the usage of the network infrastructure.

Alternatively, for non-VPN flows (e.g., POP-to-POP public traffic) the time-of-day traffic profile might be derived from past measurements. In fact, it has been established that data traffic presents high periodicity at the time-of-day scale (see for instance [1] and reference therein, and [2]). Therefore, it should be possible to predict the time-of-day behavior of traffic demands based on past measurements.

In all such cases (tv-VPN and predicted traffic) the network has to route new LSPs taking into account their time-of-day bandwidth profiles, which is known in advance. With respect to such routing problem, we distinguish the *on-line* and the *off-line* problem instances. In the former, demands are allocated as they arrive, and there is no rearrangement of previously allocated paths. In the latter, the entire set of demands is known in advance and all the routes are computed jointly in a global optimization process. We also distinguish between *fixed* and *variable* routing, depending on whether or not the route of each demand can be varied during the time-of-day. Our interest is admittedly for fixed routing, and we will consider instances of the variable routing problem in order to derive reference performance bounds. Incidentally, we found that the optimal solution to the fixed routing problem with complete knowledge of bandwidth profiles holds about the same resources usage than the optimal variable routing.

In this work we propose a simple on-line algorithm for routing fixed LSPs with declared time-of-day bandwidth profiles. Our algorithm relies on a simple shortest-path computation, where the link weights are a function of the residual *peak bandwidth*. This approach basically extends what proposed in [3] for fixed bandwidth demands. We also provide Integer Linear Programming (ILP) formulations for the associated off-line problems, in case of fixed and variable routing, and solve them to obtain reference performance bound for the on-line algorithm.

[1]This work was supported by the Italian Ministry for University and Scientific Research through the TANGO Project (PNR 2001-2003, FIRB).

The rationale behind this work is that the *a priori* knowledge of the per-demand time-of-day bandwidth profiles can be exploited to minimize the overall bandwidth reservation. In fact, by coupling demands with increasing and decreasing profiles on a same link, it is possible to achieve a certain degree of bandwidth saving. More formally, by denoting with $f^k(\tau)$ the amount of bandwidth required by demand k at time τ on a generic link, a total amount of bandwidth equal to $\max_\tau \sum_k f^k(\tau)$ is sufficient to support the bundle of demands. On the other hand, in absence of the exact knowledge of the full bandwidth profile but its peak, the reserved bandwidth would be $\sum_k \max_\tau f^k(\tau)$.
The potential saving:

$$S = \sum_k \max_\tau f^k(\tau) - \max_\tau \sum_k f^k(\tau) \quad (1)$$

depends on several factors. First, as it comes from the mutual compensation of increasing and decreasing bandwidth profiles, the saving is virtually null in case all demands have parallel behavior, i.e. they all rise and fall in a synchronized fashion. In case of tv-VPN profile synchronization can be counteracted by means of differentiated billing profiles. Secondly, the potential bandwidth saving depends on the routing algorithm and its ability to fit demands with mutually-compensative profiles on the same links. Our on-line routing algorithm is specifically targeted to achieve this goal.

In our model the per-demand bandwidth profiles are piece-wise constant, holding a certain bandwidth value for a certain time interval. It can be expected that some degree of *discretization* in time would be beneficial for a better fitting of compensative demands on the same links, and consequently to increase the potential bandwidth saving. Discretization in time means that the network operator will define a common set of time-of-day intervals (or "time slots") for all demands. During the τ-th slot, the generic demand k is associated to a single bandwidth value, say $f^k(\tau)$. In other words, the discretization in time enforces the synchronization of changes in bandwidth reservations at the time slot boundaries.

The rest of the paper is organized as follows. In section II we relate this work with the existing literature. In section III we describe our on-line algorithm for fixed routing, while in section IV we provide Integer Linear Programming (ILP) formulations of the off-line problem instances, both for fixed and variable routing. In section V we provide several numerical results assessing the goodness of the on-line algorithm, included a comparison with reference bounds as provided by the solution of the of-line optimization instances. In section VI we relate our results on fixed and variable routing with those found in two previous works, namely [4] and [5]. Finally, in section VII we conclude and suggest new directions for further research.

II. Relation to previous work

The literature on routing in connection-oriented packet networks, basically ATM and MPLS, is extremely vast. The problem of on-line routing of guaranteed-bandwidth virtual circuits has been faced in a large number of previous works, for instance [3] [6] [7] [8] to cite only few samples. Quite surprisingly, none of them ever considered the case of traffic demands with time-varying bandwidth profile.
To the best of our knowledge, the problem of optimal network configuration in presence of known time-varying traffic only appeared in [4] and in [5]. Both works where in the perspective of off-line configuration, i.e. global optimization, while in this paper we are mainly interested in the on-line problem. The application contexts were also very different from the one considered here: a connection-less network with OSPF/IS-IS routing in [4], and a connection-oriented multi-layer network in [5]. Despite the considerable differences in the approach and in the application scenario, we believe there is a common fundamental relationship between our findings and the experimental results reported in that previous works. Section VI in this paper is devoted to enlighten such commonalities and suggest a broader direction for further research.

Under the algorithmic perspective the present work has a certain relationship with [3]. In fact, it was the first one to suggest the shortest-path routing with link-weights dependent on the residual-bandwidth (let us denote this approach as Residual Influenced Shortest-Path, RISP). Basically, the on-line heuristic proposed in section III is an extension of the RISP approach, where the residual bandwidth becomes a vector in τ but the link-weight remains a scalar.
One might wonder why we preferred RISP to alternative approaches, for example MIRA [7] and its derivations [8]. We believe that the main attractiveness of the RISP scheme is in its extreme simplicity, so that it can be easily extended to be applied to more articulated problems. For instance, in a previous work [9] [10] we adopted a RISP-like scheme for the on-line routing of *protected demands* against single and dual faults. As a further step, we are currently working to incorporate the time-varying extensions of RISP into that scheme, so as to achieve a global model for the on-line allocation of protected demands with time-variable bandwidth profiles. At the same time, we believe that the extension of a MIRA-like model to the case of time-varying traffic demands is an interesting topic for future research.

III. Proposed on-line algorithm

We consider the 24 hours day-time partitioned into a total of Θ time slots, not necessarily of equal duration. The index $\tau = 1, 2...\Theta$ will denote the generic time slot, while the indices k and m will refer to demands and links respectively. We assume that connection requests (i.e., *demands*) arrive randomly to the network. The generic k-th demand is associated to a ingress-egress node pair (s^k, d^k) and to a bandwidth-profile vector denoted by $\left\{f^k(\tau)\,,\, \tau = 1, 2...\Theta\right\}$, being $f^k(\tau)$ the amount of bandwidth required by the k-th demand during the time slot τ. Let us introduce the notation that will be used throughout the rest of the paper:

- r_m^k is the fraction of traffic from the k-th demand that is routed over link m ($0 \leq r_m^k \leq 1$).

- $u_m(\tau)$ is the amount of bandwidth reserved on link m during the time slot τ.
- $v_m = \max_{\tau=1..\Theta}\{u_m(\tau)\}$ is the *peak link bandwidth* on link m, i.e. the maximum amount of bandwidth reserved across the entire day-time. Unless differently specified, we will assume that the peak bandwidth is the relevant metric accounting for link resource usage, so that occasionally v_m will be simply referred to as *link bandwidth*.
- C_m is the capacity of link m.
- w_m is the link-weight associated to link m. Its value is dynamically computed for each new request according to the profiles of reserved and requested bandwidth.

The generic k-th request arrives at the Route Selection Engine (RSE), which computes the most convenient route between the ingress-egress pair (s^k, d^k). We assume that the new demand will be routed without rearranging the already established ones. The RSE can be either duplicated in each edge-node, like in the distributed MPLS-TE architecture, or centralized in a single route server. In both cases, we assume that a database is available to the RSE, collecting the topology information as well as the full profile of reserved bandwidth $\{u_m(\tau)\,,\,\tau = 1, 2...\Theta\}$ for each network link. In case of distributed implementation, the link bandwidth profiles can be disseminated by appropriate extensions to existing flooding protocols (e.g., OSPF-TE [11]). It is not in the scope of this paper to compare the centralized vs. distributed implementation of on-line routing, and the interested reader is referred for instance to [12] and [10] for more material on this issue. For each new request, the RSE prunes from the network topology graph those links without enough available bandwidth to accommodate the request, i.e. those for which $f^k(\tau) + u_m(\tau) \geq C_m$ for some τ. In a second step, it assigns a link cost w_m to each link m as a non-decreasing function of:

$$x_m = \max_{\tau=1,2...\Theta}\left\{f^k(\tau) + u_m(\tau)\right\} \tag{2}$$

for example:

$$w_m = \frac{C_m}{C_m - x_m} + \epsilon \tag{3}$$

Note [2] that $w_m < \infty$ because of the previous pruning. We tried several alternative weight functions (see table I). The numerical results showed that in the considered sample scenarios the function (3) holds the better performances. Based on the link weight w_m, the RSE produces a weighted directed graph, and on that it runs Dijkstra to find the minimum cost path between the assigned ingress-egress pair for the new demand.

[2]The random term $\epsilon << 1$ has a marginal role, namely to scramble the route selection for parallel demands (i.e., with same ingress-egress pair) in case of a poorly loaded network. In fact, in this case a large number of network links have null reserved bandwidth, resulting in a quasi-uniform link-weight assignment. If multiple paths of equal hop-length exist between a given ingress-egress pair, the addition of a small randomization in the link-weight prevents from preferential selection of the same path for parallel demands.

This scheme is basically an extension of RISP, and it is well suited to incorporate some additional features that are of great importance in real networks. For instance it is easy to incorporate multiple-destination demands, as found for example in the routing of inter-AS flows, where usually more than one Border Routers are candidate egress points for the same flow. This case can be handled with very simple graph transformations. Additionally, if a pair of disjoint paths have to be allocated for some demand in order to apply end-to-end path protection, the Dijkstra algorithm can be replaced by the Suurballe algorithm that returns the shortest-pair of disjoint paths (see [9] [10]). Such possibilities enrich the attractiveness of the proposed model, but have been left out of the scope of this paper.

IV. ILP FORMULATION

The allocation mechanism described above is heuristic, and there is no assurance that it is effective in optimally fitting the bandwidth profile of the new request $\left\{f^k(\tau)\,,\,\tau = 1, 2...\Theta\right\}$ with the existing profiles of reserved bandwidth on the links $\{u_m(\tau)\,,\,\tau = 1, 2...\Theta\}$. In order to evaluate the goodness of our approach, we need to consider a performance metric and compare with a reference performance bound. To this purpose, we consider a Integer Linear Programming (ILP) formulation to the problem of allocating a given set of demands with assigned time-varying bandwidth profiles in a capacitated network, with the general objective of minimizing the peak reserved bandwidth v_m on the network links.

We will consider MIN-MAX, MIN-MEAN and mixed optimization objectives: the factor $0 \leq \alpha \leq 1$ in the objective function to be minimized can be varied to trade-off between these concurrent objectives. We preferred this approach instead of defining a convex cost function as done in some previous works, e.g. [4] [5]. With the convex-cost approach, one applies an increasing penalty on the link load, and in our case we do not need to introduce such a penalty. In fact, we use ILP to solve the off-line instance of the routing problem, therefore assuming full knowledge of the set of demands under optimization, and no penalty is needed to defer the emergence of bottlenecks.

The optimization problem can be formulated as follows [3]:

Minimize:

$$c = \alpha \cdot c_{max} + (1 - \alpha) \cdot c_{mean}$$

Subject to:

$$\sum_{m \to s^k} r_m^k = 1, \quad \sum_{m \leftarrow s^k} r_m^k = 0 \quad \forall k, \tag{4a}$$

$$\sum_{m \to d^k} r_m^k = 0, \quad \sum_{m \leftarrow d^k} r_m^k = 1 \quad \forall k, \tag{4b}$$

$$\sum_{m \to n} r_m^k - \sum_{m \leftarrow n} r_m^k = 0 \quad \forall k, n \neq s^k, d^k \tag{4c}$$

[3]The notation "$m \to n$" [resp. "$m \leftarrow n$"] identifies the set of directed links m that have node n as source [resp. destination].

$$u_m(\tau) = \sum_k f^k(\tau) \cdot r_m^k \quad \forall m, \tau \tag{5}$$

$$v_m \geq u_m(\tau) \qquad \forall m, \tau \tag{6}$$

$$v_m \leq C_m \quad \forall m \tag{7}$$

$$c_{max} \geq \frac{1}{C_m} \cdot v_m \quad \forall m \tag{8}$$

$$c_{mean} = \frac{1}{M} \cdot \sum_{m=1}^{M} \frac{1}{C_m} \cdot v_m \tag{9}$$

$$r_m^k \in \{0, 1\} \quad \forall k, m \tag{10}$$

Constraints 4 are classical network flows constraints. In particular 4a and 4b refer to the demand ingress and egress nodes respectively, while 4c to the remaining intermediate nodes. Eq. 5 defines the reserved bandwidth on link m for each time slot. Eq. 6 defines the *peak* level of bandwidth reservation on link m. Eq. 7 enforces the capacity constraint. Finally, eq. 8 and 9 define respectively the maximum and mean value of link load, which are the two terms in the cost function to be minimized.

The above ILP formulation has the same structure of a classical multicommodity-flows formulation (ref. [13]). The only variant is that the demand size and the associated constraints are defined for each time slot. This formulation can not be resolved for the considered network under test with a large number of demands (in the order of 10^3). Its integer relaxation with continuous routing variables $0 \leq r_m^k(\tau) \leq 1$ will be used in section V-C to provide a performance bound to the off-line routing problem, hence to the on-line heuristic algorithm described in section III.

The routing model expressed by the above formulation, as well as of the heuristic on-line algorithm, is an example of *fixed routing*, in that each demand route is fixed in time. In fact, a single routing variable r_m^k is defined across the full set of time-slots. In our work we were interested in evaluating the potential gain in bandwidth saving that can be achieved by a *variable routing* model, where the route of each demand is allowed to change from time-slot to time-slot. The ILP formulation for the variable routing problem with time-varying demands only requires a minor adaptation to the above formulation: the substitution of the routing variables r_m^k with $r_m^k(\tau)$, and the replacing of constraints 4 and 5 with the following ones:

$$\sum_{m \to s^k} r_m^k(\tau) = 1, \quad \sum_{m \leftarrow d^k} r_m^k(\tau) = 0 \quad \forall k, \tau \tag{11a}$$

$$\sum_{m \to d^k} r_m^k(\tau) = 0, \quad \sum_{m \leftarrow d^k} r_m^k(\tau) = 1 \quad \forall k, \tau \tag{11b}$$

$$\sum_{m \to n} r_m^k(\tau) - \sum_{m \leftarrow n} r_m^k(\tau) = 0 \quad \forall k, \tau, n \neq s^k, d^k \tag{11c}$$

$$u_m(\tau) = \sum_k f^k(\tau) \cdot r_m^k(\tau) \quad \forall m, \tau \tag{12}$$

In practice, the implementation of variable routing would add complexity and overhead to the network architecture, and the global rearrangement of demand routes at each time-slot boundary is something that any ISP would dislike. Therefore, the variable routing model is not attractive unless the cost of the additional complexity is payed-off by a sensible resource saving. The comparison between the bandwidth consumption with the fixed and variable routing formulations, given in section V-D, provides a helpful insight into this issue.

V. Numerical Results

We implemented in a ad-hoc simulator the on-line routing mechanism described in section III. The ILP instances where implemented in AMPL [14] and solved with CPLEX [15]. We run a number of simulations targeted at investigating the following topics:

- Impact of the choice of link-weight function on the on-line algorithm performances.
- Allocation efficiency of the on-line algorithm compared to the relaxed off-line problem instance.
- Comparison between variable and fixed routing.
- Impact of time-slot granularity.

All the experiments were run on two test topologies of 14 and 30 nodes. The former (fig. 1) is the celebrated NSF topology which has been extensively considered in several previous works in routing. The latter (fig. 2) is the same used in [16], with the arbitrary addition of two links to raise the node degree to 3.

We considered the following performance metrics:

- The mean and maximum values of the peak link bandwidth across all the network link, denoted by c_{mean} and c_{max} respectively.
- The maximum number of allocated demands before the 1^{st}, the 10^{th} and the 100^{th} rejection, denoted respectively by b_1, b_{10} and b_{100}.

Given that all demands have identical average profiles, it is meaningful to consider the number of allocated demands as a metric for network load. In this perspective b_1 is taken as the boundary indicator of the non-saturated region, that constitutes the network operational region in the practical cases. Instead, b_{10} and b_{100} are taken as indicators of the quasi-saturated and fully-saturated region boundaries.

A. Traffic model

We considered a random arrival process of connection requests to the network. For each ingress-egress pair (i, j) demands arrive according to a poisson process of intensity λ_{ij}. We adopted a flat spatial distribution of traffic intensities, i.e. $\lambda_{ij} = \lambda, \ \forall i, j$.

For each demand k, its bandwidth profile is built randomly by extracting Θ independent samples, one for each time slot, out of the discrete set of bandwidth units $\{0, 1..5\}$. In case of null profile (all 0s) the extraction is repeated. The link capacity is set to 125 units in each direction. By considering a bandwidth unit of 20 Mbps, this correspond to a link capacity of 2.5

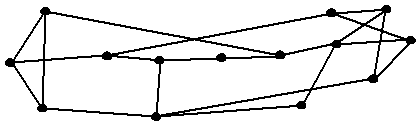

Fig. 1. 14-nodes test network (21 links).

Gbps, with a maximum requested bandwidth of 100 Mbps.
The demand duration is infinite, so that after a certain number of demands have been allocated the network approaches saturation and starts to reject new requests. In the following we will denote a sample sequence of requests as an "input trace". When comparing different routing schemes, we run parallel experiments with exactly the same input traces.
The traffic model adopted in this work is admittedly arbitrary and very simple. It does not pretend in any way to be representative of real traffic distribution in space nor in time. The synthesis of a convincingly representative traffic model is still an open point for research, despite the recent considerable achievements towards the comprehension of real traffic dynamics (see for example [1] and [17]). In any case, such model can not be independent from the underlying topology structure, given that in most cases the topology building process *follows* the evolution of the traffic distribution. Perhaps, in force of such intimate correlation, the research community should pursue the definition of a unique topology/traffic model. So far, there is no established and widely-accepted model for traffic generation at the macroscopic scale. Therefore, in our study we had to arbitrary choose a traffic model, and we gave preference on purpose to one that is as "neutral" as possible: flat spatial distribution ($\lambda_{ij} = \lambda$) and bandwidth samples uncorrelated in time. Moreover, having in mind that the direct application of such algorithm is to support the delivery of tv-VPN services, an additional obstacle to the adoption of a realistic model of bandwidth profiles is the fact such service have not been deployed to date. The replication of this study with different topology/traffic models will be itself an interesting direction for further research.

B. Choice of the link-weight function

In a first set of simulations we investigated the impact of the link-weight function on the performances of the on-line routing algorithm. We considered several different types of function, among the others those given in table I. For each of them, we run 500 simulations and reported in the table the mean number b_n of allocated demands before the n^{th} blocked request, with $n = 1, 10, 100$. In particular, the value of b_1 directly expresses the capacity of the on-line routing algorithm to fill the network before that service degradation appears in the form of request blocking.
Beyond such metrics, it is also important to consider the ability of the on-line routing scheme to save network resources, i.e. bandwidth. In fig. 3 we plot the values of c_{max} and c_{mean} after each allocated demand for a sample input trace, until b_1. The general objectives of minimizing bandwidth usage and maximizing demand allocation are not in contrast. Instead,

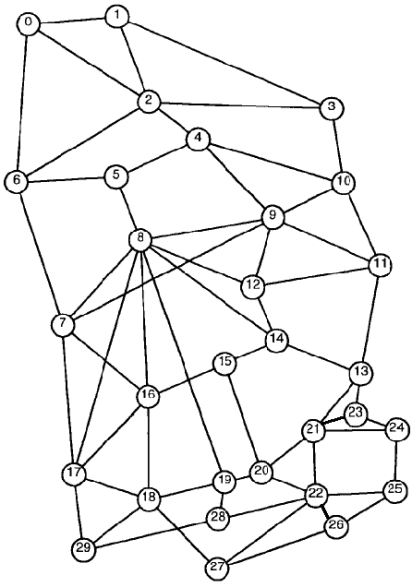

Fig. 2. 30-nodes test network (61 links).

TABLE I
COMPARISON BETWEEN DIFFERENT LINK-WEIGHT FUNCTIONS FOR THE ON-LINE ALGORITHM: MEAN VALUES OF b_n OVER 500 TRIALS (STD.DEV. IN BRACKETS).

		14-nodes net.			30-nodes net.		
link-weight fun.		b_1	b_{10}	b_{100}	b_1	b_{10}	b_{100}
A	$w_m = \frac{C_m}{C_m - x_m}$	629.7 (33.3)	664.3 (29.9)	742.7 (22.5)	1106.0 (55.4)	1160.8 (50.7)	1303.9 (35.2)
B	$w_m = e^{\frac{C_m}{C_m - x_m}}$	640.2 (28.4)	671.1 (27.1)	746.7 (22.5)	1181.0 (51.8)	1235.2 (48.6)	1371.7 (30.8)
C	$w_m = 1$	599.0 (31.6)	631.9 (29.3)	719.9 (22.4)	1097.7 (51.8)	1148.2 (47.5)	1278.2 (32.1)
D	$w_m = x_m$	622.2 (31.3)	654.5 (28.2)	738.6 (23.3)	1116.2 (54.8)	1161.8 (51.2)	1296.0 (34.4)
E	$w_m = e^{x_m}$	228.5 (17.4)	264.0 (17.1)	387.2 (20.5)	403.1 (25.1)	441.9 (25.4)	628.5 (28.0)

the minimization of mean and maximum link bandwidth are often in contrast. In fact, a preferential selection of shorter paths tends to minimize c_{mean} at the cost of a potentially larger c_{max}. On the contrary the preferential minimization of c_{max} pushes towards longer detours.

Our results show that the link-weight function "A" and "B" produce the best performances. The allocation capacity of "A" is always higher than "C" and "D" (b_1 in table I), with a lower c_{max} in the full range of demands (fig. 3). The function "B" allocates more demands than "A" (approximately $+2\%$ for the 14-nodes and $\approx +7\%$ for the 30-nodes), while keeping a lower c_{max} at the cost of a higher c_{mean}. This means that function "B" tends to produce longer paths than "A". Between "A" and "B" we give preference to the former, as we believe it is a good compromise between allocation power (high b_1) and global bandwidth usage (low c_{mean}), and unless differently specified in the rest of this paper we will adopt it in the on-line algorithm.We recognize that such preference might appears like "a matter of taste", on the other hand there is no objective metric telling how much

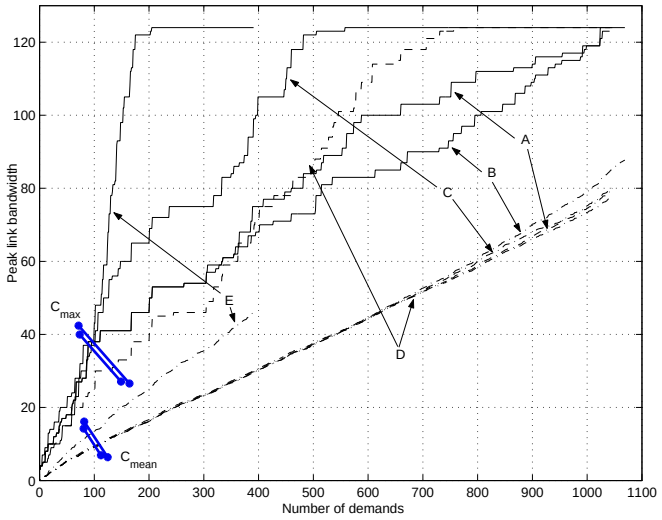

Fig. 3. Comparison between c_{mean} and c_{max} for different link-weight functions for a sample input trace.

an operator is accepting to trade path lengths for allocation power, the decision being driven by external factors (e.g. the policy of link resizing). We remark that in general it is not difficult to tailor the on-line algorithm to the specific preferences of a network operator, for example by including additional constraints on the maximum path length, or a very high penalty for exceeding some link bandwidth watermarks, etc. It is out of the scope of a research paper to elaborate the details and possibilities of such "customization" process. Rather, we are interested in providing a proof-of-concept that the basic underlying algorithm is efficient, and simple enough to be adapted to a broad range of practical scenarios.

C. Comparison with LP off-line

After tuned the on-line algorithm with the choice of a suitable link-weight function, we were interested in evaluating the goodness of our on-line algorithm with respect to a provable bound. To this purpose, in section IV we developed a ILP formulation to the problem of allocating a given set of demands with time-varying traffic profile in a capacitated network (off-line problem). We solved its integer relaxation, by letting the routing variables be continuous in $0 \leq r_m^k(\tau) \leq 1$. This corresponds to the possibility of arbitrary multipath routing (also called "splittable traffic" in [7]).

In fig. 4 we reported the c_{mean} and c_{max} curves as obtained with the on-line algorithm for a sample input trace. Both the cases of "A" and "B" weight function are plotted, and compared with the optimal values obtained by solving the off-line problem instance with the relaxed LP formulation, for different number of demands. More precisely, the optimal values of c_{mean} (circles in fig. 4) where obtained by solving the MIN-MEAN form of the LP/off-line problem, i.e. with $\alpha = 0$ in the objective function. Conversely, the optimal values of c_{max} (triangles) where obtained by solving the MIN-MAX problem ($\alpha = 0.99$). We also tried with intermediate values of $0.01 < \alpha < 0.99$, and we found that in all cases the output values of c_{mean} and c_{max} where extremely close to that reported in in fig. 4. In other words, the quality of the solutions with respect to the dual metric c_{mean} and c_{max} was quite robust to changes in the coefficient of the objective function. Interestingly, a similar result was reported in [4] with a completely different cost function (convex piece-wise linear).

From fig. 4 it can be seen that the the on-line algorithm achieves a mean peak bandwidth reservation which is very close to the optimum, especially in case of "A" link-weight function. The distance from the optimal c_{mean} is negligible when the load is low (until 700 demands). After this point, the on-line algorithm with the "B" function produces higher c_{mean}, while with the "A" function it more closely follows the optimality. In any case, the deviation from the optimality is of few percentage points.

Regarding the maximum peak bandwidth c_{max}, the on-line algorithm leads to larger values than the optimal for light load. This is not a major concern, since in such region the peak link bandwidth is faraway from saturating the link capacity, therefore the risk of emerging bottleneck links that could potentially block new demands is still negligible. On the other hand, when the network load augments, the on-line algorithm assigns higher link weights to heavy-loaded links, so that the growth of c_{max} slows and its values get closer to the optimal ones. The convergence of c_{max} towards the optimum occurs earlier for the "B" link-weight function, but at the cost of a slightly larger c_{mean}: this clearly indicates a stronger preference for longer but less loaded paths.

The above results globally show that the allocation behavior of the on-line algorithm is within few percentage points from the optimality with respect to mean and total bandwidth usage, with acceptable performances in terms of maximum link load. Our conclusion is that the proposed on-line scheme, still very simple, can be regarded as highly efficient. This lets strict margins of further improvement. Hence, it seems there is small room for further refinements and/or additions to the algorithm which are likely to come along with an increase in system complexity and/or reduced scalability, unlikely to be payed-off by few percentage points of performance improvement.

D. Fixed vs. variable routing

The on-line routing scheme proposed in section III couples time-variable bandwidth reservations with *fixed routing*. In fact, the amount of bandwidth reserved by a generic LSP on a link follows a time-varying profile, but the LSP route is fixed in time and does not change.

Removing the fixed routing constraint would mean to allow the demand route to change from time-slot to time-slot. This would add more flexibility to the allocation model, and potentially improve the global bandwidth saving by allowing the route configuration to adapt to the specific traffic distribution in each time-slot, irrespective of what happens in the others. We will refer to such scheme as *variable routing*, as opposite to fixed routing. The main drawback of variable routing is that any such scheme would require additional network capabilities and fatally add complexity to the network protocols, for

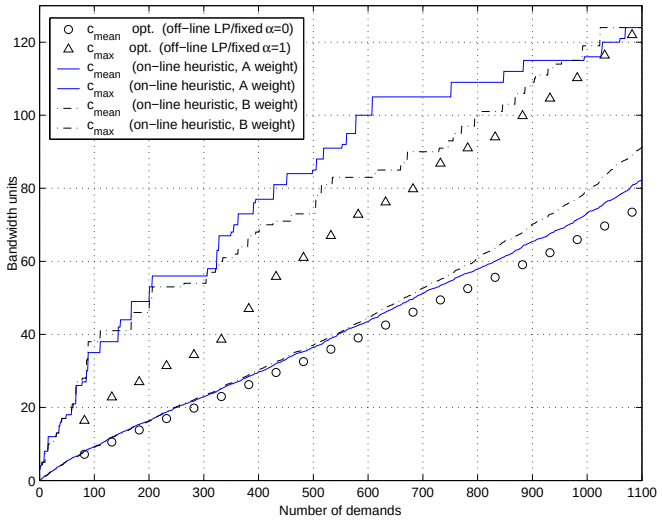

Fig. 4. Comparison between mean and max bandwidth usage with the on-line algorithm ("A" and "B" weight functions) and the optimal values obtained with the off-line relaxed problem instance.

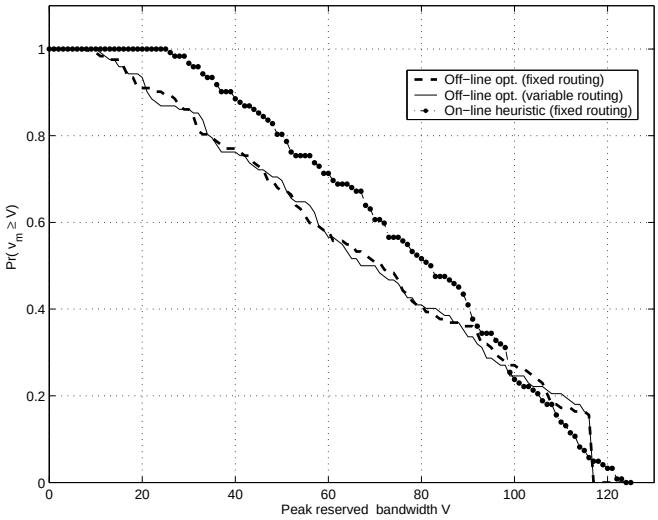

Fig. 5. CDF of the peak link load after 1030 demands (30-nodes network). Comparison between optimal solution with variable / fixed routing (relaxed LP with $\alpha = 0.1$) and on-line algorithm.

instance on the control plane.

In our investigations we were interested in evaluating the potential bandwidth saving of variable routing versus fixed routing, in the context of time-varying bandwidth profiles. To this purpose, we derived in section IV a ILP formulation for the associated off-line problem: given a set of demands and associated bandwidth profiles, find the best *set of routes* (one for each time-slot) for each demand. Again, the objective function to be minimized was the peak link bandwidth, through the metrics c_{mean} and c_{max}, and there was no penalty for route changes.

We solved the relaxed off-line instances of variable and fixed routing for the same set of input traces, for different number of demands, for both the considered topologies, and with different values of α. Quite unexpectedly, in all cases the distance between the values of c_{mean} and c_{max} obtained with variable and fixed routing was extremely close, and in most cases indeed negligible. In order to get a deeper insight into the phenomenon, we plotted in fig. 5 the full complementary distribution function (CDF) of the peak link load over all the network links, obtained with the variable and fixed routing versions of the off-line problem with relaxed routing variables and $\alpha = 0.1$, on the 30-nodes network and for a total of 1030 demands. It can be seen that the two distributions are very close to each other. In fig. 5 we also reported the CDF obtained by the on-line algorithm (with the "A" function), loaded with a sequence of the same set of demands. Again, the closeness of this curve to the others constitutes a further confirmation of the effectiveness of our on-line algorithm.

Similar results hold for the 14-nodes network. In that case, we verified for a sample input trace that the relaxed off-line problem becomes unfeasible after 650 demands, both with fixed and variable routing, while the on-line algorithm accepted 635 demands *before the first rejection*, i.e. only -2%. Such results globally show that the optimal solution obtained with relaxed (splittable) variable routing holds the same bandwidth usage than relaxed fixed routing. In other words, the fixed routing constraint has a marginal impact on the quality of the optimal solution.

This conclusion could seem counterintuitive. In fact, it is easy to provide simple counterexamples - small graphs with a small number of demands - where the gap between fixed and variable routing is considerable. On the other hand, in networks with a large size and/or a large number of demands the statistical impact of such pathological cases is likely small. Admittedly, a similar explication was reported by Fortz and Thorup in [4]. We believe that this argument might explicate not only the closeness of the optimal values for fixed and variable routing, but also the the closeness of our on-line algorithm to the optimality.

Our findings are consistent with the results found independently by Fortz and Thorup [4] and by the same authors [5] in other application contexts. A comparative discussion between the three works is given in the section VI.

E. Impact of time-slot granularity

In the above sections we showed that the proposed on-line algorithm is effective in exploiting the *a priori* knowledge of bandwidth profiles to minimize the overall bandwidth usage. In this section we are interested in evaluating the gain achievable by the proposed scheme versus the traditional reservation strategy, which assumes knowledge of the peak bandwidth only. At the same time, we are interested in assessing the dependence of such gain with respect to the granularity of the time-discretization, i.e. the number of time-slots Θ.

In the traditional reservation scheme (hereafter referred to as "peak-based") the demand size is represented by a single value, namely the peak requested bandwidth $\overline{f^k} = \max_{\tau=1..\Theta} \left\{ f^k(\tau) \right\}$. Hence, the bandwidth reserved on each

link is also a scalar, and will be denoted as $\overline{v_m}$. Instead, in our scheme both the demand size and the link reserved bandwidth are vectors of size Θ in the time-slot index τ.

To compare the two schemes, we implemented a "peak-based" on-line algorithm that reserves an amount of bandwidth equal to $\overline{f^k}$ for each demand along its path and for the entire day-time. The route selection is similar to that escribed in section III, with a link-weight function that is inversely proportional to the residual bandwidth ("A" function), the only difference being in the definition of the residual bandwidth itself. Formally:

$$w_m = \frac{C_m}{C_m - (\overline{v_m} + \overline{f^k})} + \epsilon \tag{13}$$

The peak-based on-line algorithm was compared with that proposed in section III, referred here as "profile-based". The comparison was made for different values of time slot granularity, from $\Theta = 2$ (12-hours slots) to $\Theta = 24$ (1 -hour slots). The case $\Theta = 1$ was taken as a reference, since in this case there is no distinction between the two approaches. Twin experiments were run with peak-based and profile-based algorithms for the same input trace. The figure 6 plots the number of allocated demands before the 1^{st}, 10^{th} and 100^{th} rejection (b_1, b_{10} and b_{100}), for the 30-nodes network. Each point is the average over 500 simulations. As expected the peak-based algorithm is sensibly less efficient: its allocation power is between 36% and 82% less than the profile-based algorithm.

Furthermore, the performance of the peak-based strategy become less and less efficient with increasing number of time slots, while for the profile-based allocation the performance degradation is marginal. This was expected and can be easily explained in the light of the adopted traffic model, by considering the average per-demand bandwidth consumption for the two schemes. In fact, the average amount of bandwidth that is reserved by the peak-based strategy in each time slot is ($E(\cdot)$ denotes the sample average):

$$E\left(\max_{\tau=1..\Theta}\left\{f^k(\tau)\right\}\right)$$

that is increasing with Θ given that the $f^k(\tau)$ samples are extracted independently. Instead the profile-based strategy allocates on average $E\left(f^k(\tau)\right)$, which is independent from Θ. This is confirmed by fig. 7, wherein we compared the growth of reserved bandwidth (c_{mean} and c_{max}) with the number of demands for the two algorithms, for two different values of Θ. The slopes of such curves roughly represent the average per-demand bandwidth consumption. It can be seen that a considerable difference hold between $\Theta = 3$ and $\Theta = 8$ with the peak-based algorithm. Instead with the profile-based approach the variation with Θ is minimal.

A certain performance degradation of the profile-based scheme with increasing Θ was expected, since an increase in the number of time slots directly results in a higher variability of the per-demand profile due to the independent extraction of the $f^k(\tau)$ samples. In turn this would make the matching between mutually compensative demands more problematic. The fact that the expected degradation is extremely slow (approximately -6% from $\Theta = 2$ to $\Theta = 24$) is a further indicator of the power of the proposed algorithm.

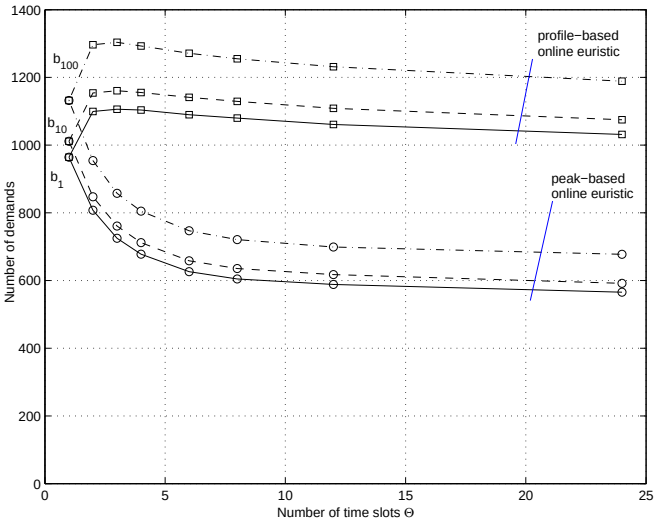

Fig. 6. Comparison between "peak-based" and "profile-based" on-line algorithms: number b_n of allocated demands before the n-th rejection versus number of time slotsΘ (n=1,10,100, average over 500 iterations).

VI. Discussion on the results

In this section we expand the discussion about the experimental results reported in section V-D, and relate them with the experimental results of two previous works [4] and [5]. The three works refer to different application scenarios: configuration of a connection-less network [4], configuration of a connection-oriented multi-layer network (namely IP over WDM) [5], and on-line routing in a MPLS network (this paper). Despite such differences, all these papers share some fundamental commonality in the results. In fact, they all consider the problem of network configuration under the condition that the input traffic is variable in time - with time-of-day periodicity - and known *a priori*. Given the different application contexts, the terms "network configuration" and "input traffic" assume different meanings. A solution of network configuration is a set of OSPF/IS-IS link weights in [4], a logical topology plus a set of LSP routes in [5], and a set of LSP routes here. The input traffic is a node-to-node traffic matrix in [4], and a set of point-to-point demands in our works. In all the three works some method was proposed that explicitly takes into account the amount of traffic at different times - assumed known in advance - to produce a fixed (or static) network configuration solution that can accommodate the traffic and its variability. In all the three works it is found that the goodness of such a fixed solution is very close to that achievable with independent optimizations at different times (an approach referred to as "time-specific" in the following). More precisely, Fortz and Thorup considered two different traffic matrices (night and day), and found that the quality of their *sub-optimal* fixed solution is within 10% from the *optimal*

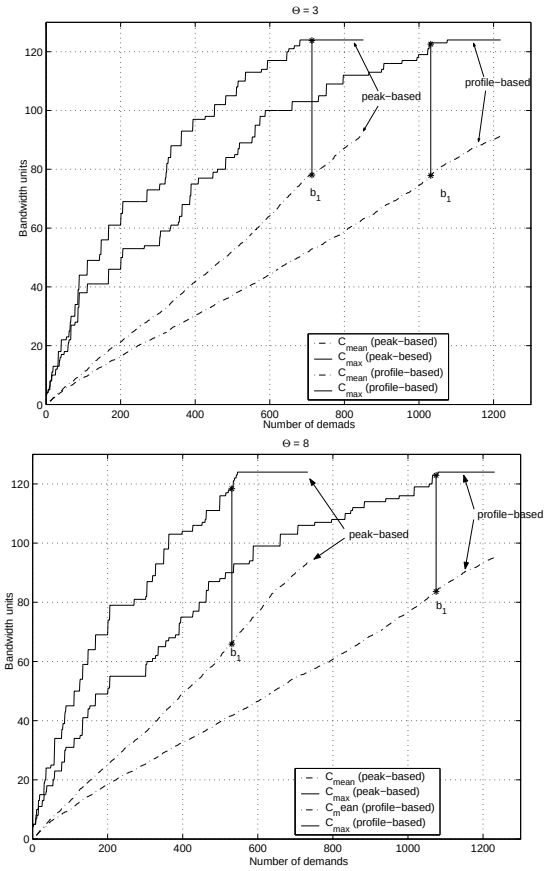

Fig. 7. Comparison between "peak-based" and "profile-based" on-line algorithms: c_{mean} and c_{max} versus number of demands for $\Theta = 3$ (top) and $\Theta = 8$ (bottom).

time-specific solution. Analogously, in [5] the authors found a similar distance between the *sub-optimal* fixed solution (called JCET therein) and the *best found* time-specific solution (called ICET). In fact, in that context it was not possible to give a provable optimality bound because the associated optimization problem was intrinsically non-relaxable. Again, in this paper we found that the quality of the *sub-optimal* fixed solution (as given by the on-line heuristic) is very close to the *optimal* time-specific solution (as given by the relaxed off-line instance of the variable routing problem). Additionally, we found that the quality of the *optimal* fixed solution (as given by the relaxed off-line instance of the fixed routing problem) closely approximates the *optimal* time-specific solution.

The similarity between these results is indeed remarkable, considered that they were obtained in very different application contexts, with different traffic models, parameters, topologies, cost functions etc.

In summary, the common message behind such works is that in several networking scenarios with variable input traffic there is no need of variable network configurations, as far as the traffic variations are known in advance or can be predicted to some extent. We believe that this is an important contribution of the three works as a whole, that indicates a promising direction for further research, with both academical and practical implications. On the academical side, further research efforts should be spent in investigating whether there is some more general theoretical principle underlying such results, that can explicate them at a broader level than the specific application scenarios in which they were conceived.

VII. Conclusions and future work

In this work we proposed a simple on-line algorithm for the fixed routing of demands with variable bandwidth requirements, to be applied in virtual-circuit platforms like MPLS. Basically, the algorithm proposed here is an extension in the time domain of an existing dynamic shortest-path approach. We also provided ILP formulations for the associated off-line problem, both for fixed and variable routing, and we used them to obtain reference performance bounds for the on-line algorithm. The results show that our algorithm is highly efficient and closely approaches the optimality in terms of bandwidth usage.

Beyond its efficiency, an important source of attractiveness resides in its extreme simplicity, that makes it suitable to be adapted in more advanced routing scenarios. For example, considering the problem of routing fault-protected demands, it can be straightforwardly incorporated in the model proposed in [9] [10] in order to provide a global scheme for single and dual fault protection, with and without bandwidth sharing, in presence of demands with time-varying bandwidth requirements. This is one of our current working directions.

Based on the proposed scheme, ISPs might find convenient the delivering of more flexible services, for example time-varying VPN, that better match the customer needs in terms of bandwidth provisioning timing. Also, from the above considerations ISPs are solicited to monitor and put efforts in the prediction of the full time-of-day traffic profiles (e.g., on a 12- or 8- hours time-slot granularity) and not only of their peaks. In fact, we showed that with the proposed scheme such full information can be directly translated into a considerable gain in resource saving and allocation power.

In a broader perspective, the results presented here suggest that the *a priori* knowledge of the per-demand traffic profiles can be exploited to achieve an optimal static routing configuration which cannot be further improved by dynamic reconfigurations. This lets small room to the usefulness of dynamic reconfiguration schemes *provided that future traffic profiles are known or can be predicted in advance*. Therefore, the installation of rerouting functionality into the network for tracking changes in the traffic pattern might be avoided at all, or at least such mechanisms might be relegated to handle the unpredictable components of the global traffic.

Admittedly, our results are limited to the assumptions made in this work, particularly about the choice of the traffic model. We recognize that a conclusive quantitative assessment of our scheme necessitates a topology/traffic model more thoroughly bound to reality. On the other hand, the fact that very similar results were obtained in other works [4] [5], carried in different contexts and for different topology/traffic models is

encouraging about the validity of our conclusions.
We are currently investigating our scheme under different traffic models, for instance with non-flat spatial traffic distribution. This second phase of investigations is in a very preliminary stage. Still, to date we only found further confirmations to the findings reported in this paper. Our feeling is that the general results reported here are *robust with respect to the choice of the topology/traffic scenario*. The verification of this statement needs of course further investigations, but it seems itself an interesting directions for further research.

REFERENCES

[1] Z. L. Zhang C. Diot K. Papagiannaki, N. Taft. Long-Term Forecasting of Internet Backbone Traffic: Observations and Initial Models. *IEEE INFOCOM'03, San Francisco*, April 2003.

[2] C. Lund N. Reingold J. Rexford A. Feldmann, A. Greenberg and F. True. Deriving traffic demands for operational IP networks: Methodology and experience. *IEEE/ACM Trans. on Networking*, June 2001.

[3] S. Plotkin. Competitive Routing ov Virtual Circuits in ATM Networks. *IEEE Journal on Selected Areas in Communications*, 13(7), September 1995.

[4] M. Thorup B. Fortz. Optimizing OSPF/IS-IS weights in a changing world. *IEEE Journal on Selected Areas in Communications*, 20(4), May 2002.

[5] F. Ricciato, S. Salsano, A. Belmonte, M. Listanti. Off-line Configuration of a MPLS over WDM Network under Time-Varying Offered Traffic. *IEEE INFOCOM'02, New York*, June 2002.

[6] R. Guerin, D. Williams, A. Orda. QoS Routing Mechanisms and OSPF Extensions. *Globecom'97, Phoenix*, November 1997.

[7] T. V. Lakshman K. Kar, M. Kodialam. Minimum interference routing of bandwidth guaranteed tunnels with mpls traffic engineering applications. *IEEE Journal on Selected Areas in Communications*, 18(12), 2000.

[8] D. Bauer P. R. Warkhede S. Suri, M. Waldvogel. Profile-based routing and traffic engineering. *Computer Communications*, 26:351–365, 2003.

[9] F. Ricciato, S. Salsano, M. Listanti. An Architecture for Differentiated Protection agains Single and Double Faults in GMPLS. *to appear in Optical Networks Magazine*, 2003.

[10] A. Belmonte D. Perla F. Ricciato, M. Listanti. Performance Evaluation of a Distributed Scheme for Protection against Single and Double Faults for MPLS. *2nd Int'l Workshop on Quality of Service in Multiservice IP Networks (QoS-IP 2003), Milano*, February 2003. Published in Lecture Notes in Computer Science, vol. 2601, Springer, pp. 218-232.

[11] D. Yeung D. Katz, K. Kompella. Traffic Engineering Extensions to OSPF Version 2. *draft-katz-yeung-ospf-traffic-10.txt. Work in Progress*, June 2003.

[12] R. Ramamurthy J. F. Labourdette S. Chaudhuri K. Bala G. Ellinas, E. Bouillet. Routing and Restoration Architectures in Mesh Optical Networks. *Optical Networks Magazine*, 4(1), January/February 2003.

[13] J. B. Orlin R. K. Ahuja, T. L. Magnanti. *Network Flows: Theory, Algorithms and Applications.* Prentice Hall, 1993.

[14] www.ampl.com.

[15] www.ilog.com.

[16] Rainer R. Iraschko, M. H. MacGregor, and Wayne D. Grover. Optimal capacity placement for path restoration in STM or ATM mesh-survivable networks. *IEEE/ACM Trans. on Networking (TON)*, 6(3):325–336, 1998.

[17] J. Jetcheva N. Taft S. Bhattacharyya, C. Diot. Pop-level and access-link-level traffic dynamic in a tier-1 POP. *Proc. of the ACM SIGCOMM Internet Measurement Workshop (IMW 2001). San Francisco*, November 2001.