

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

Fabio Ricciato, Ugo Monaco

INFO-COM dept. of University "La Sapienza", Rome, Italy

Email: {fabio.monaco}@infocom.uniroma1.it

Abstract—In this work we consider the problem of routing bandwidth-guaranteed flows with time-varying bandwidth profiles on a MPLS network. We assume that each demand is routed in a fixed LSP, and that 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. Such 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 based on shortest-path computation with link weights influenced by the residual *peak* bandwidth. We also provide an ILP formulation for the associated off-line problem, and adopt it as a reference performance bound for the on-line algorithm. The results presented here show that the proposed algorithm, despite its simplicity, closely approximates the optimal solution.

The message of this paper is that the *a priori* knowledge of the per-demand traffic profiles, still within a fixed routing framework, can be exploited to achieve a sensible bandwidth saving, and / or to differentiate bandwidth provisioning (and billing) on a per-hour basis.

I. INTRODUCTION

In this work we consider the problem of routing bandwidth-guaranteed flows with time-varying 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. At the same time we assume that the routing of each LSP is not subject to be changed during the daytime. In other words, our model combines variable (in time) bandwidth reservations with fixed routing.

The per-demand time-of-day profiles can be explicitly declared by the customer and included in the Service Level Agreement (SLA) with the ISP. This would allow the delivering of a more flexible Virtual Private Networks service, namely the "time-varying VPN" (tv-VPN), aimed at better matching the VPN 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".

Alternatively, for non-VPN flows 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 safely predict the time-of-day behavior of single 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 known time-of-day bandwidth profiles. 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. 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 problem and solve it to obtain reference performance bound for the on-line algorithm.

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 $S = \sum_k \max_{\tau} f^k(\tau) - \max_{\tau} \sum_k f^k(\tau)$ 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. 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 *dis-*

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

cretization 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 intervals (or "time slots") for all demands. During the τ -th slot, the generic demand k is associated to a single bandwidth value, say $f^k(\tau)$. This 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 an Integer Linear Programming (ILP) formulation of the associated off-line problem. In section V we provide several numerical results aimed at assessing the goodness of the on-line algorithm. Finally, in section VI we conclude and suggest new directions for further research.

II. RELATION TO PREVIOUS WORK

The problem of on-line routing of guaranteed-bandwidth virtual circuits has been faced in a large number of previous works, for instance [3] [4] [5] [6] 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 [7] and in [8]. Both works where in the perspective of off-line configuration, i.e. global optimization, while here we are mainly interested in the on-line problem. The application contexts were also very different from that considered here: a connection-less network with OSPF/IS-IS routing in [7], and a connection-oriented multi-layer network in [8].

Under the algorithmic perspective the present work has a certain relationship with [3], which first suggested the use of shortest-path routing with link-weights dependent on the residual-bandwidth for load balancing. Let us denote this approach as Residual-bandwidth Influenced Shortest-Path (RISP). Basically, the on-line algorithm proposed in section III is an extension of RISP, 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 [5] and its derivations [6]. 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 previous works [9] [10] we adopted a RISP-like scheme for the on-line routing of *protected demands* against single and dual fault protection, and time-varying reservations could be straightforwardly incorporated into that model. 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 $\tau = 1, 2, \dots, \Theta$ 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 an ingress-egress node pair (s^k, d^k) and to a bandwidth-profile vector denoted by $\{f^k(\tau), \tau = 1, 2, \dots, \Theta\}$, being $f^k(\tau)$ the bandwidth required by the k -th demand during the time slot τ .

Let us introduce the notation used in 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 .
- $v_m = \max_{\tau=1, \dots, \Theta} \{u_m(\tau)\}$ is the *peak link bandwidth* on link m , i.e. the maximum amount of bandwidth reserved across the entire day-time. We 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 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 Network State Database (NSD) is associated to the RSE, collecting the topology information as well as the full profile of reserved bandwidth $\{u_m(\tau), \tau = 1, 2, \dots, \Theta\}$ for each network link m . In case of distributed implementation, the link bandwidth profiles can be disseminated by appropriate extensions to existing flooding protocols (e.g., OSPF-TE [11]). 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 . Therefore, in case that no route exists between the assigned endpoints with sufficient residual bandwidth to support the demand along *all* time slots, the new request is rejected. In a second step, it assigns a link cost w_m to each link m as a non-decreasing function of:

$$x_m = \max_{\tau=1, 2, \dots, \Theta} \{f^k(\tau) + u_m(\tau)\} \quad (1)$$

for example:

$$w_m = \frac{C_m}{C_m - x_m} \quad (2)$$

Note that $w_m < \infty$ because of the previous pruning. We considered several alternative weight functions (see table I). The numerical results showed that in the considered sample scenarios the function given by eq. 2 holds the best performances. Based on the link weights 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.

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 often multiple 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 must 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 (as done in [9] [10]). Such possibilities further enrich the attractiveness of the proposed model.

IV. ILP FORMULATION FOR THE OFF-LINE PROBLEM

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 $\{f^k(\tau), \tau = 1, 2, \dots, \Theta\}$ with the existing profiles of reserved bandwidth on the links $\{u_m(\tau), \tau = 1, 2, \dots, \Theta\}$. 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 can be varied to trade-off between such concurrent objectives. We preferred this approach instead of defining a convex cost function as done in some previous works, e.g. [7] [8]. 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 ¹:

Minimize:

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

Subject to:

$$\sum_{m \rightarrow s^k} r_m^k = 1, \quad \sum_{m \leftarrow s^k} r_m^k = 0 \quad \forall k, \quad (3a)$$

$$\sum_{m \rightarrow d^k} r_m^k = 0, \quad \sum_{m \leftarrow d^k} r_m^k = 1 \quad \forall k, \quad (3b)$$

$$\sum_{m \rightarrow n} r_m^k - \sum_{m \leftarrow n} r_m^k = 0 \quad \forall k, n \neq s^k, d^k \quad (3c)$$

$$u_m(\tau) = \sum_k f^k(\tau) \cdot r_m^k \quad \forall m, \tau \quad (4)$$

$$v_m \geq u_m(\tau) \quad \forall m, \tau \quad (5)$$

¹The notation " $m \rightarrow n$ " [resp. " $m \leftarrow n$ "] identifies the set of directed links m that have node n as source [resp. destination].

$$v_m \leq C_m \quad \forall m \quad (6)$$

$$c_{max} \geq \frac{1}{C_m} \cdot v_m \quad \forall m \quad (7)$$

$$c_{mean} = \frac{1}{M} \cdot \sum_{m=1}^M \frac{1}{C_m} \cdot v_m \quad (8)$$

$$r_m^k \in \{0, 1\} \quad \forall k, m \quad (9)$$

Constraints 3 are classical network flows constraints. In particular 3a and 3b refer to the ingress and egress nodes respectively, while 3c to the remaining intermediate nodes. Eq. 4 defines the reserved bandwidth on link m for each time slot. Eq. 5 defines the *peak* level of bandwidth reservation on link m . Eq. 6 enforces the capacity constraint. Finally, eq. 7 and 8 define respectively the max 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. [12]). The only variant is that the demands 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-B to provide a performance bound to the off-line routing problem, hence to the on-line heuristic algorithm described in section III.

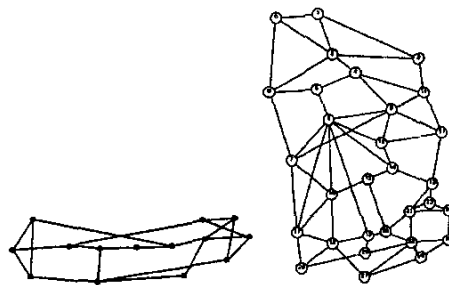


Fig. 1. Test networks.

V. NUMERICAL RESULTS

We implemented the on-line routing algorithm in an ad-hoc simulator. The ILP instances were implemented in AMPL and solved with CPLEX [13]. We run a number of simulations aimed at investigating *i*) the impact of the choice of weight function on the on-line algorithm performances, and *ii*) the allocation efficiency of the on-line algorithm compared to the relaxed off-line problem instance.

All the experiments were run on two test topologies. The former (fig. 1-left, 14 nodes, 21 links) is the celebrated NSF topology which has been extensively considered in several previous works in routing. The latter (fig. 1-right, 30 nodes, 61 links) is the same used in [14], with the arbitrary addition of two links to raise the node degree to 3. 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 $\{0, 1..5\}$ of bandwidth units. We considered $\Theta = 3$ time slots. 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 Gbps, with a maximum requested bandwidth of 100 Mbps.

The demand duration is infinite, so that after a certain number of demands 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 [15]). 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. The replication of this study with different topology/traffic models will be itself an interesting direction for further research.

We considered the following performance metrics:

- The mean and maximum peak link bandwidth across all network links, denoted by c_{mean} and c_{max} respectively.
- The maximum number of allocated demands before the 1st, 10th and 100th 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. Choice of weight function

In a first set of simulations we investigated the impact of the 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. It can be seen that functions "A" and "B" produce the best allocation performances.

Beyond such metrics, it is also important to consider the ability

of the on-line routing scheme to save network resources, i.e. bandwidth. The general objectives of minimizing bandwidth usage and maximizing demand allocation are not in contrast. Instead, 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. In fig. 2 we plot the values of c_{max} and c_{mean} after each allocated demand for a sample input trace, until b_1 , for the weight functions "A" and "B". It can be seen that weight "B" holds a lower c_{max} at the cost of a higher c_{mean} . This means that function "B" tends to produce longer paths than "A". However, under heavy loaded conditions, i.e. after 900 allocated demands, the difference in c_{max} between the two weights diminishes. The decision between the two should depend on the particular application. Depending on whether or not the cost of longer LSPs pays off a slightly higher allocation power, weight "B" might be preferred to "A".

B. Comparison with relaxed off-line optimal configuration

After tuned the on-line algorithm with the choice of a suitable weight function, we were interested in assessing its goodness with respect to a provable bound. To this purpose, we solved the integer relaxation of the ILP formulation of the associated off-line problem instance, as given in section IV. The integer relaxation of the routing variables $r_n^k(\tau) \in [0, 1]$ corresponds to the possibility of arbitrary multi-path routing (also called "splittable traffic" in [5]).

In fig. 2 we compared for a sample input trace the values of c_{mean} and c_{max} obtained with the on-line heuristic with the optimal values obtained by solving the relaxed off-line optimization, for different number of demands. More precisely, the optimal values of c_{mean} (circles in fig. 2) were 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) were obtained by solving the MIN-MAX problem ($\alpha = 1$). We also tried with intermediate values of $0.01 < \alpha < .99$, and we found that in all cases the output values of c_{mean} and c_{max} were extremely close to that reported in fig. 2. 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 [7] with a completely different cost function (convex piece-wise linear). From fig. 2 it can be seen that the on-line algorithm achieves a mean peak bandwidth reservation which is very close to the optimality. The relative distance from the optimal c_{mean} is negligible when the load is low (until 700 demands). After this point, the on-line algorithm with "B" weight holds higher c_{mean} , while "A" follows the optimality within a gap of few percentage points.

Regarding the maximum peak bandwidth c_{max} , the on-line heuristic leads to larger values than the optimal for light load. This is not a major concern, since in such region the reserved bandwidth is faraway from saturating the link

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

link-weight fun.	14-nodes net.			30-nodes net.		
	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)

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 heuristic assigns higher link weights to heavily-loaded links, so that the growth of C_{max} slows down and converge to the optimum. These results globally show that the allocation behavior of the on-line algorithm is within few percentage points from the optimality with respect to bandwidth usage. Additionally, we compared the allocation power of our on-line algorithm - based on the full knowledge of the $f^k(\tau)$ profiles - with a traditional on-line routing scheme where only the peak requested bandwidth $f_{max}^k = \max_{\tau} \{f^k(\tau)\}$ is known for each demand. In the same network scenario considered above, we verified that with $\Theta = 3$ time-slots the profile-based algorithm allocates up to $\approx 50\%$ more demands than the traditional peak-based scheme (figures could not be included here for sake of space limitations).

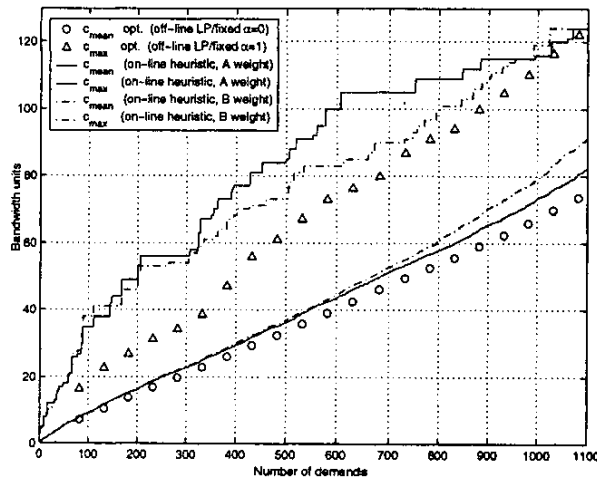


Fig. 2. Mean / maximum bandwidth usage obtained with the on-line algorithm ("A" and "B" weights) and the relaxed off-line optimization.

VI. CONCLUSIONS AND FUTURE WORK

We proposed a simple on-line algorithm for the 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 (RISP). We also provided an ILP formulation for the associated off-line problem, and we used it to obtain reference performance bounds for the on-line algorithm. The results show that the proposed on-line algorithm is highly efficient and closely approaches the optimality in terms of bandwidth usage.

Based on the proposed scheme, ISPs might deliver more flexible connectivity services, for example time-varying VPN with declared profile, or time-dependent billing schemes, in order to better tailor the customer needs in terms of bandwidth provisioning timing. Furthermore, ISPs are solicited to put efforts in the monitoring and prediction of the full time-of-day traffic profiles, rather than only of their peaks. In fact, we showed that such information, where available, can be exploited to achieve a considerable allocation gain.

Beyond its efficiency, the proposed algorithm is attractive because of 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 working directions. Additionally, we are further investigating the performance gap between fixed and variable routing with more sophisticated traffic models.

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 of Virtual Circuits in ATM Networks. *IEEE J-SAC*, 13(7), September 1995.
- [4] R. Guerin, D. Williams, A. Orda. QoS Routing Mechanisms and OSPF Extensions. *Globecom'97, Phoenix*, November 1997.
- [5] T. V. Lakshman K. Kar, M. Kodialam. Minimum interference routing of bandwidth guaranteed tunnels with mpls traffic engineering applications. *IEEE J-SAC*, 18(12), 2000.
- [6] D. Bauer P. R. Warkhede S. Suri, M. Waldvogel. Profile-based routing and traffic engineering. *Computer Communications*, 26:351-365, 2003.
- [7] M. Thorup B. Fortz. Optimizing OSPF/IS weights in a changing world. *IEEE J-SAC*, 20(4), May 2002.
- [8] 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.
- [9] F. Ricciato, S. Salsano, M. Listanti. An Architecture for Differentiated Protection against Single and Double Faults in GMPLS. *to appear in Photonic Networks Journal*.
- [10] F. Ricciato et al. Performance Evaluation of a Distributed Scheme for Protection against Single and Double Faults for MPLS. *2nd Int'l Workshop on QoS in IP Networks (QoS-IP 2003), Milano*, 2003.
- [11] K. Kompella D. Katz. Traffic Engineering Extensions to OSPF Version 2. *draft-katz-yeung-ospf-traffic-10.txt. Work in Progress*, June 2003.
- [12] J. B. Orlin R. K. Ahuja, T. L. Magnanti. *Network Flows: Theory, Algorithms and Applications*. Prentice Hall, 1993.
- [13] AMPL home page: www.ampl.com. CPLEX home page: www.ilog.com.
- [14] 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.
- [15] 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*, 2001.