Efficient Planning of Virtual Network Services

K. Sarsembagieva, G. Gardikis,
G. Xilouris, A. Kourtis
Institute of Informatics and Telecommunications
National Centre for Scientific Research "Demokritos"
Aghia Paraskevi, Athens, Greece
{katias;gardikis;xilouris;kourtis}@iit.demokritos.gr

P.Demestichas
Dept. of Digital Systems
University of Piraeus
Piraeus, Greece
pdemest@unipi.gr

Abstract—**This paper addresses the issue of virtual network service provision, i.e. establishment of virtualized connectivity services over heterogeneous network infrastructures and argues that, besides the enabling transport and switching technologies, effective planning methods must also be studied so that such services can be provided on-demand, within reasonable provisioning times. In this context, the paper proposes a relatively simple and fast method for MPLS -particularly- and virtual network –generally- planning, able to map several demands over complex infrastructures within minutes. The proposed method is evaluated with regard not only to its response time, but also to the uniformity and balance of the traffic load distributed to the network.**

Keywords: virtual network services; MPLS planning; MPLS-TE

I. INTRODUCTION

Providing virtualized network services over heterogeneous infrastructures has attracted significant attention from the research community during the past years. At the same time, it is exhibiting an ever increasing market potential, mostly in the form of virtual private network (VPN) service provisioning i.e. the establishment of overlay/tunneled connectivity services over wide-area networking infrastructures with specific topology and QoS constraints. Such services are gaining ground as revenue source for network operators, who have the opportunity to actually monetize on their available network resources, in addition to traditional flat-rate, best-effort connectivity offered to the wide public. Popular use cases include corporate VPNs (interconnecting national or international points of presence), telecom network backhauling and media/content distribution networks.

Such virtualized services can be realized via a multitude of enabling transport/tunneling and switching technologies, such as IP/(G)MPLS or recently Carrier Ethernet. Virtual network connectivity as a service is currently provisioned manually, with considerable provisioning time (i.e. from request to establishment) and thus restricted to long-term usage. However, mechanisms allowing fast and automated on-demand provisioning of virtual networks for short-term exploitation are crucial in order to accommodate a wider spectrum of use cases and open up new market opportunities to network operators. Towards this, several research efforts are currently investigating network management architectures for dynamic on-demand establishment of virtual networks either within a single administrative domain or across multiple ones [1]. At the same time, efficient planning mechanisms are required, able to map the virtual network requests on the underlying network infrastructure and according to the user's topology and QoS requirements with optimal (or near-optimal) exploitation of the underlying network resources. While several algorithms have been proposed for this purpose, most of them are too complex for real-world implementation, either being non-scalable (i.e. able to serve only a limited number of requests) or significantly time-demanding.

In this context, this paper proposes a method for virtual network planning which is relatively simple, fast and scalable, yet remarkably efficient in terms of resource optimization. The method is introduced for IP/MPLS networks, although it can be applied to other transport technologies, such as Carrier Ethernet. The paper proceeds as follows: Section II briefly overviews the MPLS-TE mechanisms and refers to proposed algorithms for traffic engineering. Section III analyses the proposed method and Section IV presents performance evaluation results. Finally, Section V concludes the paper.

II. BACKGROUND AND RELATED WORK

During the past years Multi Protocol Label Switching (MPLS) has become the de-facto standard for traffic engineering and optimisation of IP networks. MPLS establishes Label-Switched Paths (LSPs) based on the fast switching (instead of Layer 3 routing) of packets, which traverse the path. A group of packets which are forwarded in the same manner, over the same LSP, and with the same forwarding treatment is called Forwarding Equivalence Class (FEC). By assigning LSPs and FECs to customers, network operators can exploit MPLS to enable the provision of virtual network services over a multitude of underlying transport technologies, satisfying a wide spectrum of use cases and customer needs. IETF has standardised the MPLS protocol suite as well as various technologies that build on it. In order better support of Traffic and Resource Management, MPLS-based Traffic Engineering (MPLS-TE) mechanisms were introduced, defining certain requirements that should be met, as well as numerous characteristics that must or should be deployed on the entities of an MPLS domain [2]. Some of the requirements associated with MPLS paths are: traffic load (peak/average rates etc), explicit link selection, path priority and requirement for backup (failover) paths.

A key issue for MPLS-TE, critical for the efficient deployment of virtual network services, is finding a mapping policy that optimises the set-up of the Label Switched Paths from a given demand and on a given network topology, with the ultimate goal of optimal user performance and efficient use of network resources. Many works have proposed solutions to tackle this problem either by optimising some custom objective cost functions or by addressing partially some of them (sacrificing completeness to execution speed). In [3] various methods are evaluated following their initial categorisation into QoS- and Load- based and then evaluated. There are also algorithms which take into account energy consumption in the core network [4]. A very interesting approach is presented in [5] which proposes a non-linear model to find a solution for the capacity and flow allocation problem on the LSPs with the main goal being to minimize the link congestion using buffer capacity allocation. However, the solution is approached from a theoretical point of view with no extensive tests of the implementation, thus not assessing the capabilities of the algorithm in a real context.

The works published in [6] and [7] exploit genetic algorithms with the objective being the minimization of the maximum of link utilization. Both algorithms are designed to be used off-line The algorithm proposed in [6], compared to the basic Shortest Path First (SPF) algorithm, appears to be very scalable, producing a good load balance throughout the topology. However, its goal is concentrated on the congested parts of the network, considering the rest of the links to be of less importance. On the other hand, the algorithm proposed in [7] combines the rejection rate and the total number of hops in one objective function, thus balancing the network load by minimizing the load on the most utilized link and calculating the point in which the trade-off curve between LSP delays and link load balancing is achieved. The results show an improvement on the solutions when compared to traditional SPF approaches, though the algorithm itself is significantly complex, resulting in considerable execution times.

Other approaches formulate the problem as an optimization problem which, unlike the previous approaches, take into account MPLS Differentiated Services (DiffServ) traffic classes, targeting towards a stronger interrelation between QoS provision and Load Balancing [8]. Although the execution time is significantly longer, constituting such approaches valid only for off-line network planning, the solution they yield is close to optimal. A much faster algorithm valid for on-line usage is MIRA (Minimum Interference Routing Algorithm) whose key idea is to route an incoming connection over a path which least interferes with possible future requests [9]. Based on the “critical links” and having as the main goal to avoid them, MIRA results to very few connection rejections thus increasing the throughput, though can be considered as very complex, also producing uneven load distribution. Answering to the weaknesses of MIRA, the Dynamic Online Routing Algorithm (DORA) was proposed [10], designed for online execution while relying on the current network topology to avoid links that have bigger congestion probabilities.

Continuing the work of MIRA and DORA, the MIH (Minimum Interference Hybrid) algorithm optimizes three objectives: the number of hops, the link utilization and the interference [11]. Its application results show a very good balance among the three metrics while its execution time is truly efficient for an online use in an operating network. Although very efficient, MIH, as well as the rest of the previously discussed algorithms seem to use the MPLS as a mere mean of routing, without taking into consideration the requirements that should be met while implementing TE rules. Answering to the latter statement, the work of [12] proposes an algorithm that takes into account some of the MPLS-TE requirements, such as the Path Protection, giving good results in very large networks and producing an approximate, yet very close to the optimal, solution. While [12] is considering only some of the MPLS-TE requirements, the authors of [13] propose the Constrained Shortest Path First (CSPF) algorithm, which considers most of them while not being complex and quite easy to implement. CSPF also takes into account topics such as the Priority Attribute and the Explicit Links definition. Although CSPF is quite efficient and yields satisfactory results, it output is rather sub-optimal and it can lead to many requests being rejected.

Continuing the work of the aforementioned research efforts and trying to address to some of the weaknesses of the presented algorithms, this paper proposes the TiReD (**T**raff**i**c **Re**source **D**istribution) algorithm, which attempts to calculate and establish minimum-length LSPs from a given set of requests, also achieving:

- The fulfilment of most of MPLS-TE requirements as far as the FECs are concerned: Backup routes, hierarchical explicit links, maximum hop number per path, and parallel flows.
- The even distribution of the load within a network in such a way as to avoid critical links, minimize the congestion and maximize the overall network flow.

III. Proposed Algorithm

A. *Algorithm entities*

The TiReD algorithm uses the following entities:

a. FEC: A FEC (Forward Equivalence Class), as requested, can be considered as an end-to-end connection with certain attributes, which are listed and described in Table I.

TABLE I. FEC Demand Attributes

Attribute	**Description**
Source	The IP address of the Ingress LER (Label Edge Router) in the MPLS Domain.
Destination	The IP Address of the Egress LER.
Priority Level	The MPLS-TE Priority Attribute of a flow, taking numbers from {0..7}, with the 0 being the highest priority, denoting the most important FEC.
Bandwidth	The nominal bandwidth requested by this FEC
Max Hops	If set, this attribute denotes the maximum length (hop count) that the FEC should exhibit.
Hop Deviation	Relaxation attribute for the “Max Hops” showing the upper and lower thresholds for the hop count.
Backup	Attribute that, if set, means that the FEC requires the establishment of a Backup path as well, in a P+B strategy (where P is the number of the Primary Parallel Paths {P=1 for one unique parallel path} and B is the number of the Backup Parallel Paths {B=1 for one unique backup path}).

Attribute	Description
Max Cuts	Attribute that shows the maximum number of parts that the FEC could be partitioned into, if a single feasible path could not be calculated. A value of “1” means that no flow partitioning is allowed for the particular FEC.
Link Reqs	A table holding a hierarchical representation of the explicit links defined by the FEC, ordered from most to least desired and showing the links that are to be traversed or avoided.

b. SubResidual Network: The SubResidual network is a sub-graph of the original topology, designed in a way that: a) All nodes of the original network are present; b) From the Links of the original network only those that: (1) are not defined as “restricted” by the Link Requirements; (2) are not being used by the Primary Path of the FEC (if a Backup path is being calculated) and (3) have adequate residual capacity, participate in the subgraph.

c. Extended SubResidual Network: The Extended SubResidual Network is a sub-graph of the original topology and a super-graph of the SubResidual Network. It is designed in the exact same way as the latter with the only difference that the chosen links could have inadequate but strictly positive residual capacity.

d. Path Discovery History: The Path Discovery History is a type of “memory” in which the algorithm stores its previous failed attempts to find a path, so as to avoid repeating the same attempt in the future. As will be shown in the next section, the algorithm can execute up to 2 attempts to find a path for a FEC (or one of its parts). If any of the tries does not succeed in a calculation of the optimal path, the Path Discovery History is updated to hold the values of the Source, the Destination and the Bandwidth that resulted to the failure. Thus, when another FEC arrives with the same Source-Destination pair and with a Bandwidth bigger than or equal to one that led up to a failed attempt, the algorithm proceeds to the calculation of the Extended SubResidual Network or the rejection of the FEC's connection, instead of repeating a useless search.

B. *Inputs*

The input data to the TiReD algorithm consist of the network topology and the requested FECs. More specifically, the Original Network Graph is known and created a priori, containing all the LSRs (Label Switch Routers), LERs, links and their nominal capacities. In addition, the set of requested FECs with their corresponding backup flows are considered given and are being ordered, in the initialization phase of the algorithm, according to the following parameters: (1) Priority Level, (2) Sum of the primary and the backup paths, (3) Size of the Link Requirements Table, (4) Maximum Hops per Path, (5) Hops Deviation and (6) Maximum Parallel Paths. All the input data are stored in a relational database with well defined relations and constrains that protect the integrity of the latter and their conformance to the rules of the Networks in general and MPLS in particular.

C. *Outputs*

The output of the TiReD algorithm is the Original Network Graph, updated in such a way so as to represent the information about which part of which demand with how much bandwidth is traversing each link and what is its type (main or backup). Just as the input, the output data is also stored in a relational database, which keeps detailed records of the Served FECs, the paths assigned to them as well as the time of the LSP invocation.

D. *Algorithm logic*

At the initialization phase, the FEC requests are being ordered and put to a “Most Important/Restrictive FEC First” list. At each iteration, the algorithm pops out the first FEC in the list and examines it. If a SubResidual Network was not tried on it and if there is no record of a failed attempt to serve a similar or more restrictive FEC in the Path Discovery History, then the SubResidual Network is constructed, containing only those links that are not marked as “restricted” and have adequate residual capacity to support the new connection.

Then, having constructed the sub-graph, the algorithm goes on to find all the paths from Source to Destination, trying to keep the length of each to a minimum (thus ignoring longer paths). All paths that do not conform to the FEC’s requirements (number of hops, explicit links etc.) are rejected. For each path of the resulting subset, a “Path Weight” function is calculated which measures the conformance ratio of the path (between the required and actual attributes of the path) and of the FEC respectively (for example the ratio of the path’s links count to the maximum hops required by the SLA). The path with the maximum link capacity and the highest value of the Path Weight is being chosen and assigned to the FEC. The new connection is added to the Network and the next FEC is popped from the list.

If, after the above procedure, the SubResidual Network does not result to any feasible paths, the failed request is added as a new record to the Path Discovery History. Furthermore, if the FEC allows its division into parallel paths, then it is left at the top of the list and the Extended SubResidual Network is designed with the above steps being repeated. Of course, if a path in the Extended SubResidual Network is found, it will be, by default, inadequate to fit the entire FEC. Thus, the requested FEC is being partitioned into two parts: The first is assigned the found optimal path while the second is put at the top of the request list to be examined and served as a separate request in the next round. At this point it should be noted that if the FEC i) does not support its partitioning, ii) cannot be further divided due to the Max Cuts restriction or iii) it requests a Backup route that also cannot be served wholly or partially, then the FEC is fully rejected and cleared out of the Original Network, allowing the algorithm to proceed to the next FEC of the list.

Another notable aspect is that while the MPLS-TE requirements are considered to be obligatory for the primary path, they play a mere “nice to have” role in the case of the Backup path. This could be easily understood through the following example: Consider a FEC, which requires the explicit traversal of the link (A-B) and the establishment of a Backup path. If the FEC is served, then its path must contain the mentioned link. When its Backup route is being searched for, it must, by all means, avoid each and every link that the primary connection is using; thus it should avoid the (A-B) link as well. If no other Link Requirements with a lower level in the hierarchy are present, then the Backup route will never succeed into finding a feasible path since it shares the same restrictions as the primary and marks the (A-B) as mandatory. In order to avoid such a deadlock, the algorithm relaxes the requirements of the backup route, considering those paths that have the needed residual capacity but may not meet all the MPLS-TE requirements of the original FEC.

A high-level flow diagram depicting the TiReD algorithm logic is depicted in Figure 1.

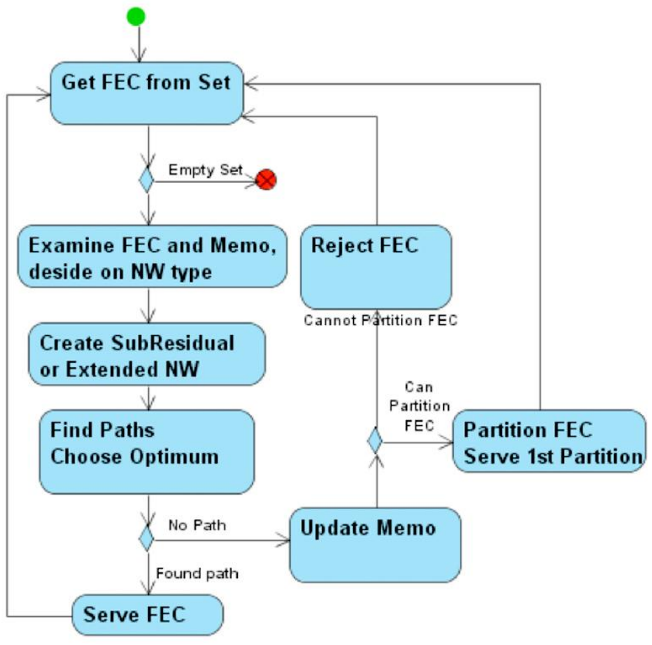
Get FEC from Set
Empty Set
Examine FEC and Memo, deside on NW type
Reject FEC
Cannot Partition FEC
Create SubResidual or Extended NW
Can Partition FEC
Find Paths Choose Optimum
Partition FEC Serve 1st Partition
No Path
Update Memo
Found path
Serve FEC

Figure 1. High-level Flow diagram of the TiReD algorithm logic

IV. Performance Evaluation

A. Test Environment

In order to evaluate the performance of the TiReD mechanism, the algorithm was developed and examined under numerous realistic scenarios, as well as under stress tests. Instead of testing the algorithm in a pre-defined static network topology, a more complex approach was followed for the dynamic generation of evaluation scenarios. For the rapid and easy generation of large network topologies and request sets, a dedicated software module was designed and implemented, working as a Networks and Request Sets Randomizer. It takes as input some generic parameters for the scenarios (such as the number of LSRs/LERs, number of links, maximum/minimum link capacities, number of FECs, maximum/minimum required bandwidths), and produces a network topology and FEC requests with random attributes, which are in turn stored in a MySQL Database to be later extracted by the TiReD algorithm as its input data. The performance evaluation results to follow refer to execution in a desktop system with 1.90 GHz CPU and 2GB of RAM.

B. Execution Times

As already mentioned, numerous tests were performed in order to examine the performance of the algorithm in terms of both execution time and optimality of results. Thirteen of the most representative test scenarios are summarized in Table II, which depicts the execution time for every scenario. A brief notation of the columns is given to shed some light on the presented results:

- Demands - Number of FEC requests in the set
- LSRs - Number of MPLS Routers (LERs and LSRs) in the topology.
- Links - Number of links in the topology
- MinDB/ MaxDB - Minimum/ Maximum Bandwidth of the FEC requests (uniformly distributed)
- MinLC / MaxLC - Minimum/ Maximum Link Capacities (uniformly distributed)

TABLE II. Execution Time under Various Test Scenarios

Dem ands	**LSRs**	**Link s**	**Min DB**	**Max DB**	**Min LC**	**Max LC**	**Time (sec)**
6	9	16	2	5	1	123	2
4	5	7	10	15	20	20	1
1	2	1	2	2	5	5	0,5
1	4	4	2	2	2	765	1
1	4	4	2	2	2	765	0,5
1	6	10	1	1	9	9	1
7	10	38	10	100	72	99	1
19	30	157	2	140	125	199	45
20	50	231	1	20	100	199	5
44	50	165	1	5	100	199	52
188	70	241	1	1	120	200	90
100	100	170	1	1	100	101	69
8	8	15	8	9	10	200	2

As seen in Table II execution time mostly depends on the network complexity and the load of requests. As it can be seen, the algorithm is and remarkably scalable and fast, suitable for on-line, realtime use; it takes only 90 seconds to map 188 requests to a network of 70 LSRs.

C. Traffic Distribution

As far as the load distribution is concerned, a most representative result of a stress test will be shown in this subsection, whose network topology consists of 100 routers and 197 links. The TiReD algorithm is used to map a set of 13 FEC requests on the aforementioned network. Requests have highly diverse bandwidth requirements and no MPLS-TE restrictions save one: each FEC must be assigned one route and no partition is allowed. Each of the FECs requests its own Ingress-Egress pair. The TiReD Algorithm execution on the discussed topology resulted to a load distribution that can be seen in Figure 2.

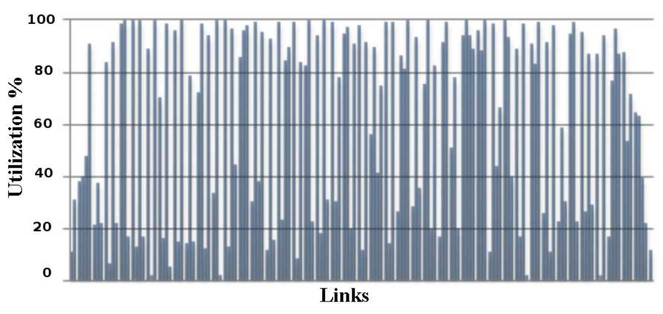

Figure 2. Link utilisation across the network after the assignment of the 13 FECs.

With the vertical axis corresponding to the link utilization and the horizontal one to the individual links of the topology, it can be shown that the Utilization values of the particular test lie between 1.5% and 99.78% with practically zero unused links. As expected for such a large amount of required load which, more importantly, cannot be further distributed into partitioned paths, 59.4% of the links have utilization values higher than 50%, yet do not become saturated. Thus, it is fair to assume that the relatively congested parts are used by the largest flows which, if their partition was allowed, they would be more evenly distributed. However, considering the large network topology and the considerable differences in the FEC request requirements, it can be seen that TiReD algorithm is quite successful in achieving a uniform load distribution across the network links.

D. Comparison

In order to yield more valuable information, several comparison tests, whose results are presented in the next paragraphs, have been run and evaluated. The comparison experiments where ran for two different test-bed configurations: (a) setting the topology of the TiReD execution to a topology similar to that in which another algorithm was evaluated and (b) setting the topology of the TiReD execution to a topology that would reveal the weaknesses of another algorithm.

Compared to e.g. the algorithm proposed in [12] which can take more than 20 hours to reach the optimal (and more accurate) solution, TiReD distributes the load in a network of the same length in a few tens of seconds, producing, thus, a sub-optimal yet satisfactory solution. Furthermore, it introduces a type of “intelligence”, being able to “remember” failed attempts to accommodate traffic in the network, thus reducing future pointless searches.

The proposed algorithm maximizes the overall flow in the network while conforming to the MPLS-TE regulations in general and the FECs’ MPLS-TE restrictions in particular. It reaches the given targets by evenly balancing the traffic and load within an MPLS topology. Thus, it could be noted that although TiReD is more complex than MIRA, it performs better in distributing the load and avoids critical links. Furthermore, it seems to be able to solve the weakness of DORA which could lead to the rapid saturation of links with smaller capacities, since TiReD is always trying to find the best path with the largest residual, so as to balance the utilization ratios. Thus, links with smaller capacities will be chosen after long and only if the optimal path happens to traverse them, minimizing, this way, the congestion probability.

V. Conclusions

The paper presented a simple and fast, yet efficient approach for planning of virtual paths over network infrastructures. The logic of the algorithm was presented, followed by performance evaluation results showing that significantly small decision times can be achieved, yielding at the same time uniform distribution of the traffic across the network. Although the algorithm was designed under the principles of MPLS-TE, it can be used with any underlying transport/switching technology, under numerous application scenarios.

Continuing the presented work, the proposed algorithm will be slightly re-designed in order to reach more optimal solutions by relaxing the hop count criterion in path selection, towards the maximization of the network usage and the minimization of request rejections.

Acknowledgement

The design of the proposed algorithm was initiated in the context of a Master’s thesis of Mrs. K. Sarsembagieva under the supervision of Prof. P. Demestichas, as part of the MSc program “Technology Education and Digital Systems”, direction “Network-Centric Systems”, of the Dept. of Digital Systems of the University of Piraeus. The work described in this paper is also partially supported by the research project ALICANTE, funded by the European Commission within the Seventh Framework Programme under grant agreement n° 248652.

References

[1] T. C. Hung et al, “Advanced Routing Algorithms and Load Balancing on MPLS”, Proc. International Conference on Advanced Communication Technology (ICACT), vol.3, pp.1886-1891, 12-14 February 2007

[2] D. Awduche et al, “Requirements for Traffic Engineering over MPLS”, RFC 2702, http://www.ietf.org/rfc/rfc2702.txt

[3] T. C. Hung et al, “Advanced Routing Algorithms and Load Balancing on MPLS”, Proc. 9th ICACT, 12-14 Feb. 2007, Vol. 3, pp. 1886 – 1891

[4] V. Foteinos, K. Tsagkaris, P. Peloso, L Ciavaglia, P. Demestichas, “Energy Saving with Multilayer Traffic Engineering in Future Core Networks”, accepted for publication in the Journal of Green Engineering, 2012.

[5] M. Huerta, J. Padilla, X. Hesselbach, R. Fabregat, O. Ravelo, "Buffer Capacity Allocation: A method to QoS support on MPLS networks", Proc. EATIS2006 - Euro American Conference on Telematics and Information Systems, 2006, pp. 14-28.

[6] Y. Ling and D. Meng, “A Genetic Optimization Algorithm to Solve the Problem of the Load- Balancing of Network Load”, International Journal of Computer Science and Network Security, Vol.6, No.7B, July 2006, pp. 63-68

[7] A. Oliveira and G. R. Mateus, “Using genetic algorithms to LSP setup in MPLS networks”, Proc. XXIV Simpósio Brasileiro de Redes de Computadores, 2006, pp. 705–721

[8] K. Wu and D. S. Reeves, “Link Dimensioning and LSP Optimization for MPLS Networks Supporting DiffServ EF and BE traffic classes”, Proc, 18th International Teletraffic Congress, 2003.

[9] K. Kar, M. Kodialam and T.V. Lakshman “Minimum Interference Routing of Bandwidth Guaranteed Tunnels with MPLS Traffic Engineering Applications”, IEEE Journal on Selected Areas in Communications, Vol. 18, No. 12, Dec. 2000, pp. 2566 - 2579

[10] R. Boutaba,W. Szeto and Y. Iraqi, “DORA: Efficient Routing for MPLS Traffic Engineering”, Journal of Network and Systems Management, Vol. 10, No. 3, 2002

[11] K. Banglore, “A Minimum Interference Hybrid Algorithm for MPLS Networks”, Master Thesis, Florida State University, 2002

[12] B. G. Jozsa et al, “An Efficient Algorithm for Global Path Optimization in MPLS Networks”, in Optimization and Engineering, Vol. 2(3), pp. 321–347, 2001

[13] K. Manayya, “Constrained Shortest Path First”, Internet Draft, http://www.ietf.org/id/draft-manayya-cspf-00.txt, Feb 20, 2009