A Fast Route Planning Algorithm for MPLS-TE

Katia Sarsembagieva, Georgios Gardikis, George Xilouris, Anastasios Kourtis
Institute of Informatics and Telecommunications
NCSR Demokritos
Agia Paraskevi, Greece
emails: {katias; gardikis; xilouris; kourtis}@iit.demokritos.gr

***Abstract*—MPLS and MPLS-TE are playing a key role in resource management of contemporary packet networks and also in the provisioning of end-to-end virtualized network services. In this context, route planning and resource allocation in MPLS networks, conforming to MPLS-TE requirements, is of vital importance for optimizing network usage. This paper proposes an MPLS-TE-conformant route planning algorithm which is significantly fast – and thus suitable for online use – while at the same time yields satisfactory results. Performance assessment results are also presented, accompanied with a comparison with existing algorithms.**

Keywords- MPLS; MPLS-TE; route planning; resource optimisation

I. INTRODUCTION

During the past years, MPLS has become the de-facto standard for establishment of end-to-end switched paths over networking infrastructures. In order to enhance the support of Traffic and Resource Management, Traffic Engineering mechanisms were proposed for MPLS (MPLS-TE) 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 [1]. MPLS-TE requirements include, among others, definition of traffic parameters for each Label Switched Path (LSP), explicit path selection, path priority and protection through back-up routes.

Since the efficiency of the algorithm to be applied for the calculation and establishment of an MPLS LSP on a given infrastructure topology is of vital importance for optimizing the usage of network resources, this subject has been extensively been studied in the literature and several algorithms have been proposed. This article contributes towards this direction, proposing an algorithm which exhibits three key advantages; it is fairly simple and easy to implement, it is significantly fast, thus appropriate for on-line use, and it can also accommodate the requirements posed by MPLS-TE. Moreover, performance evaluation results show that the proposed algorithm achieves the uniform distribution of the traffic across the network.

The paper proceeds as follows: Section II presents an overview of some indicative proposals for MPLS and MPLS-TE routing, Section III describes the proposed algorithm, Section IV presents and discusses performance evaluation results and, finally, Section V concludes the paper.

II. BACKGROUND AND RELATED WORK

In the paragraphs that follow, a brief survey of related works and proposed algorithms in the field of the MPLS-TE are presented. The work of [2] presents and evaluates numerous methods, categorizing them into QoS- and Load based approaches. It shows, however, that most of the algorithms join the two optimization objectives into one final goal, thus producing solutions that meet both the Quality requirements as well as the balance of traffic load. Other algorithms also take into account other optimization objectives, such as the energy consumption in core nodes [3].

A rather interesting approach is presented in [4] which proposes a Capacity and Flow Assignment model to calculate the capacity cost for each LSP in means of the buffer size of each LSR, resulting to an optimized flow assignment on links while minimizing the costs of each path. Although the analysis shows a reduction of the congestion as well as an increase of load, the approach is rather theoretical, thus not assessing the capabilities of the proposed method in a real context.

The work published in [5], uses a genetic optimization algorithm to move the load from congested hot spots to less utilized parts of the network, thus balancing the load across the latter and leaving more residual capacity for future load growth. However, while it produces very good results compared to the SPF approach, at the same time being significantly scalable, its goal is concentrated on the congested links of the topology, thus regarding the underused links as of less importance.

A similar choice of linear programming algorithm, though with a completely different approach, is made in [6] whose algorithm also minimizes the load of the most utilized link using a heuristic Genetic Algorithm based on the combination of routing policies and adaptive route movements. Combining the rejection rates and the total number of hops in the objective function, it shows a considerable improvement on the solutions when compared to the traditional SPF algorithm. However, being quite complex, yet scalable, the algorithm needs a significant amount of time to reach the optimum solution, unsuitable for an online, dynamic context.

Targeting towards a stronger conjunction between the QoS provision and the Load Balancing, [7] addresses the problem of link dimensioning and path optimization for MPLS networks providing DiffServ EF and BE traffic classes. The goal of the work proposed is to minimize the total link cost, subject to the performance constraints of both EF and BE classes. Though the applications of the algorithm produce results very close to the

978-1-4673-2781-7/12/$31.00 ©2012 IEEE

optimal, the linear problem itself is very complex, rendering the proposal unfit for online use.

Unlike the previously presented approaches, which seem to be more valid under offline, network planning contexts, other algorithms - faster yet more approximate - yield solutions fit for online, dynamic use. One of the most famous in this field is MIRA (Minimum Interference Routing Algorithm) [8] whose main goal is to route each FEC in such a way as to least interfere with possible future connections. Introducing the concept of "critical links", i.e. those links that are being used by a large number of end-to-end LSRs, MIRA results to an increase of throughput and a significant decrease of connection rejections. However, its great complexity renders it undesirably slow, while at the same time it may lead to an uneven distribution of the traffic load throughout the network topology.

Proposing a solution to the main drawback of MIRA, the Dynamic Online Routing Algorithm (DORA) was designed, also for online execution, based on the current topology of the network to avoid those links that could lead to an increase of the congestion probabilities [9]. Although slightly more complex than MIRA, DORA results to a smaller number of rejections while its execution time is as fast as that of the Open Shortest Path First (OSPF). However, it can be shown that when running in a network with uneven link capacities, DORA could lead to the rapid saturation of the smaller links, leaving the rest underused, thus resulting to an unbalanced load distribution.

Extending the work of DORA and MIRA, the MIH (Minimum Interference Hybrid) algorithm adds yet another objective, thus optimizing the interference, the link utilization and the number of hops per path [10]. Solving a more complex problem than that of DORA, while, in addition, being more configurable, MIH results to a very good balance between the aforementioned objectives in a very short time. However, although very efficient, MIH, DORA and MIRA appear to be using the MPLS merely as a mean of routing, leaving out of consideration the requirements that should be met while implementing TE algorithms and rules.

Answering to the above statement, the proposal of [11] designs a complex algorithm that takes under consideration some of the MPLE-TE requirements, such as the Path Protection, producing almost optimal results in large networks with little reserved link capacities. However, it gives very much importance to the shortest paths, thus leaving the matter of the actual load balancing quite on the side. Finally, while [11] is taking into consideration only some of the MPLE-TE constraints, the authors of [12] design the CSPF (Constrained Shortest Path First) Algorithm – an easy to implement algorithm which considers most of the requirements, such as the Priority Attribute and the Explicit Links definition. However, while it is very efficient resulting to a quite good approximation, its output is rather suboptimal and can lead to numerous connection rejections.

Carrying on the efforts of the aforementioned works and attempting to cover some of their weaker points, this paper presents the TiReD (Traffic Resource Distribution) algorithm, which tries to compute and establish minimum-length and minimum-interference Label Switched Paths from a given set of FECs, targeting towards:

- The fulfilment of most of the MPLS-TE requirements as far as the requests are concerned, namely: path protection, hierarchical explicit links, maximum path lengths, parallel flows restriction of the same initial FEC and so on.
- The balanced distribution of the traffic within a topology in such a way as to avoid critical links, minimize the congestion probabilities and maximize the network flow.

III. ALGORITHM DESCRIPTION

A. Entities

The TiReD algorithm introduces entities that can be defined as follows:

FEC: A FEC (Forward Equivalence Class) is considered as an end-to-end connection with certain MPLS-TE requirements and constraints, which are presented and described in Table I.

TABLE I. FEC REQUIREMENTS

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.
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.

SubResidual Network: The SubResidual network is a sub-graph of the original topology, designed and created per FEC. It consists of all the nodes of the original network while of the links only those that (1) are not defined as “restricted” by the Link Requirements, (2) are not assigned to be parts of the Primary Path of a FEC, in case of a Backup calculation and (3) have adequate residual capacity are present, leaving the rest of them out of the sub-graph. In addition, the SubResidual Network contains only one Source – Destination pair, depicting the Source and the Destination of the demand under examination.

Extended SubResidual Network: The Extended SubResidual Network can be considered as a super-graph of the aforementioned SubResidual Network and a sub-graph of the original topology. Its design and construction follows the exact same rules, with one exception: links that have strictly positive, but not adequate capacities, also participate in the final topology.

Path Discovery History: In order to add some kind of "prior knowledge" to the algorithm, the Path Discovery History stores the latter's failed attempts to calculate an optimum path per FEC, in order to avoid similar attempts in the future. As will be described in the next sections, the TiReD algorithm can make up to 2 calculations per FEC, or one of its parts. If any of the tries does not yield a successful result, the Path Discovery History is updated to hold the needed metrics and other information concerning the FEC. Thus, if another FEC arrives demanding similar or more restrictive path attributes, the algorithm proceeds to the calculation of the Extended SubResidual Network or the rejection of the incoming connection, avoiding a useless search.

B. Inputs and Outputs

The input of the TiReD algorithm consists of the requested FECs set and the network topology. More specifically, all the requested FECs with their corresponding backup flows are considered given and are being ordered, in the initialization phase, according to the following metrics: (1) Priority Level, (2) Sum of the primary and the backup paths' required bandwidth, (3) Size of the Link Requirements Table, (4) Maximum Hops per Path, (5) Hops Deviation and (6) Maximum number of Parallel Paths. As far as the topology is concerned, the Original Network Graph is known and created a priori, containing all the LSRs (Label Switch Routers), LERs (Label Edge Routers), links and their nominal capacities. All the input information is stored in a relational database, designed in such a way as to protect the integrity of the requests as well as their conformance to the rules of the MPLS.

The output of the algorithm consists of the Original Network Graph, updated in such a way as to represent all the needed information about the load distribution, being: which part of which demand with how much bandwidth is traversing each link and what its type is (primary or backup). The output data is also kept in a relational database, which stores detailed records on each Served FEC, the paths assigned to them (or their parts) and the time needed for the execution.

C. Operation

At the phase of the initialization, the TiReD algorithm designs and creates an empty topology and orders the FECs in a "Most Important/Restrictive" list. At each iteration, until the FECs list is empty, it pops out the first demand and examines it to construct the needed graph. If a SubResidual Network was not tried on it and if a record of a failed try to serve a similar or more restrictive FEC does not exist in the Path Discovery History, the SubResidual topology is constructed and searched for a set of paths that conform to the FECs requirements (explicit links, number of hops etc). For each path of the extracted set, a "Path Weight" function is calculated which measures its conformance ratio, being the actual to the required attributes of the path (for example the ratio of the path's links count to the maximum hops threshold required by the FEC). The path with the maximum residual link capacity and the highest value of the Path Weigh is being chosen and assigned to the FEC. The newly served connection is added to the Network and the next FEC is popped out of the list.

If, after the above procedure, the SubResidual Network fails to result to a feasible path, the failed request is stored in the Path Discovery History which keeps the required information of the attributes and constraints that led to the failure. Afterward, if the FEC does not prohibit its division into parallel paths, then it is left at the top of the FECs list to be popped out on the next cycle at which the Extended SubResidual Network will be designed and the above steps repeated. Of course, as becomes apparent from the Extended SubResidual Network definition, if a path is found in the latter topology, it will be, by default, inadequate to fit the entire FEC. Thus, if allowed, the FEC will be partitioned into two parts: The first is assigned the found optimal path while the second is put at the top of the FECs list to be examined in the next round. At this point it must be stressed that if the FEC (1) does not allow its partition, (2) cannot be further partitioned due to the Max Cuts constraint or (3) it requests a Backup path that also cannot be accommodated to the network wholly or partially, then the FEC is fully rejected and deleted from the Original Network (in the case that the primary route or parts of the primary/backup paths where already established), allowing the algorithm to proceed to the examination of the next FEC of the list.

Another aspect that should be noted is that while the MPLS-TE requirements and constraints are assumed as obligatory for the primary path, they are considered as "good to have" in the case of the Backup route. The latter could be more clarified throughout a simple example: Consider a FEC that requires the explicit use of the link (A-B) and the establishment of a Backup Path. If a feasible path is found for it, then it surely contains the mentioned link. However, when the path of the Backup is calculated, it must avoid all the links that are assigned for the primary connection; thus, it must avoid the link (A-B) as well. If no other Link Requirements with a lower level in the hierarchy are present, then the Backup path, sharing the exact same restrictions as the Primary, could not be calculated, falling into a paradox of needing to traverse and avoid the same link at the same time. In order to avoid such a case, the algorithm adds a relaxation to the requirements of the backup route, considering as "feasible" those paths that can fit to the bandwidth needs of the FEC but may not meet all the MPLS-TE requirements of the original FEC.

The activity diagram in Fig. 1. depicts the high-level logic of the TiReD algorithm.

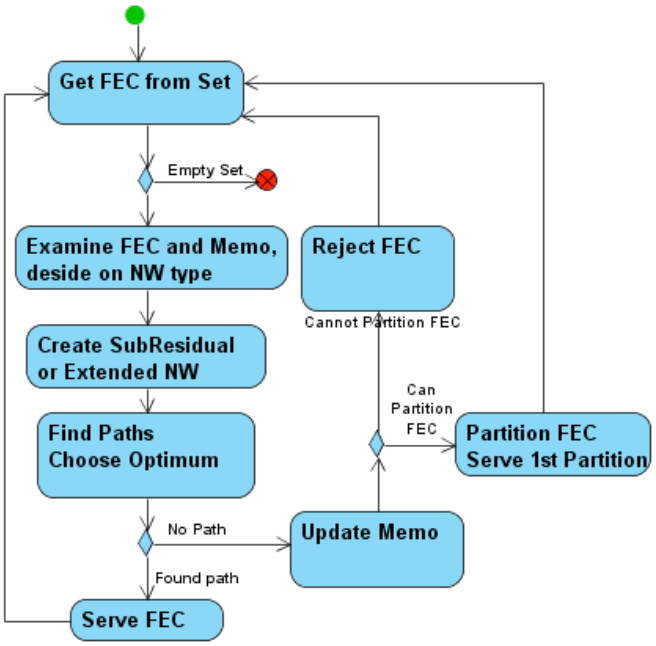

Figure 1. High-level logic of the TiReD algorithm

IV. Performance Assessment

A. Evaluation Procedure

The performance evaluation of the TiReD mechanism was carried out through numerous realistic scenarios and stress tests. In order to be able to perform extensive evaluation, going beyond pre-defined static topology scenarios, a more complex approach was followed for the dynamic generation of evaluation contexts. More specifically, the rapid and easy generation of large network topologies and FEC sets was carried out by a dedicated software module that was designed and implemented, working as a Networks and FEC sets randomizer. Its input data consist of some generic parameters (such as the total number of LSRs/LERs, number of links, maximum/minimum link capacities and FECs bandwidths, number of FECs etc.) while its output produces a network topology and FEC requests with random attributes, which are stored in a MySQL Database to be later extracted as the input data of the TiReD algorithm. The evaluation of the performance that follows, refers to the execution in a laptop system with 2GB of RAM and 1.90 GHz Dual Core CPU.

B. Execution Time

As mentioned in the previous section, various tests were performed so as to examine the performance of the proposed algorithm in terms of optimality of results and execution time. Ten of the most representative scenarios are summarized in Table II, which presents the time needed for the calculation of all paths, depending on the demand load and the complexity of the network (number of LSRs and links).

TABLE II. Execution Times for Several Load Scenarios

Demands	LSRs	Links	Time (sec)
4	5	7	1
6	9	16	2
1	2	1	0,5
1	4	4	1
1	4	4	0,5
1	6	10	1
7	10	38	1
19	30	157	45
44	50	165	52
188	70	241	90

As depicted in Table II, the execution times are mostly dependent on the network complexity and the load of requests – larger topologies that consist of a vast number of links with large nominal capacities increase the execution time when small FECs sets of small bandwidth requirements are demanding the utilization of the network. However, it can be seen that the TiReD algorithm is remarkably scalable and fast, suitable for on-line, real-time use; it needs only 90 seconds to distribute 188 requests in a network of 70 LSRs and 241 links.

C. Flows Distribution and Interference

As far as the connections distribution is concerned, a most representative result of a stress test scenario will be shown in this subsection, whose context consists of 100 LSRs, 197 links and 13 FECs. The FECs have vastly diverse bandwidth requirements and no MPLS-TE constraints except for one: no partition into parallel routes is allowed. Furthermore, each FEC requests its own, dedicated Ingress LER as well as its own Egress router. The TiReD Algorithm execution on the mentioned topology resulted to a connections distribution that can be seen in Figure 2.

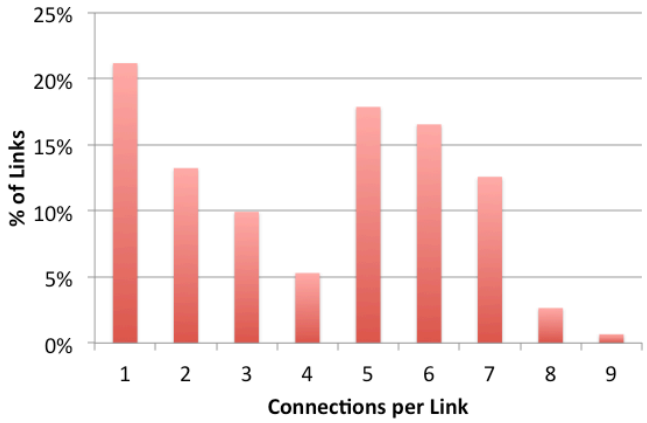

Figure 2. Distribution of Connections per Link

With the vertical axis corresponding to the number of links being used and the horizontal one to the number of connection using the same link, is tan be shown that 83.5% of the links are being traversed by 6 or less connections, with the majority being used by only one. In addition, only one link is being used by as much as 9 out of 13 simultaneous connections, which was designed to lie in the center of the network and passed by the major part of the traffic. Thus, with the vast majority of the

links accommodating almost half of the individual connections in a somewhat dedicated way, it is shown that the TiReD algorithm is quite successful in achieving a non-interfering, balanced distribution of the FECs across the network.

D. Comparison

Compared to the algorithm proposed in [11] which may take as much as 20 hours to calculate the solution, TiReD distributes the load in a topology of the same length in a few tens of seconds, thus producing a sub-optimal yet quite satisfactory solution. In addition, being slightly cognitive and deciding based on a priori and gained knowledge while being able to "remember" past failed attempts to establish a connection, it avoids future pointless searches thus further reducing its calculation time.

The TiReD Algorithm maximizes the overall network utilization while strictly conforming to the MPLS-TE regulations as well as the FEC's MPLS-TE constraints, reaching the given targets by evenly balancing the load within an MPLS domain, minimizing the interference of the established paths. It could be noted that, although it is significantly more complex than MIRA, it results to a better load distribution while at the same time solves the main weakness of DORA by finding the best path with the largest residual capacity, thus avoiding the rapid links saturation. In other words, links with smaller capacities are chosen after long and only in the case that the optimal path has to traverse them, minimizing, this way, the probability of congestion.

V. CONCLUSIONS

This article presented a simple and fast, yet efficient algorithm for MPLS-TE-compliant route planning. The entities and functional sequence of the algorithm was presented. Performance evaluation results followed, showing that complex route calculations can be achieved in significantly short time, yielding at the same time uniform distribution of the traffic across the network.

As a response to the ever increasing need for end-to-end network services based on MPLS, algorithms such as the one proposed, which are the same time relatively simple, fast and efficient, can be of considerable value. Efficient on-line planning mechanisms can significantly contribute to the accelerated roll-out of MPLS network services, provided on-demand and on end-to-end basis across heterogeneous networking infrastructures, thus accommodating a wide variety of users' needs and use case scenarios.

ACKNOWLEDGMENT

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

[2] 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

[3] 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.

[4] 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.

[5] 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

[6] 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

[7] 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.

[8] 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

[9] 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

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

[11] 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

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