A centralized Path Computation System for GMPLS transport networks: design and performance studies

Gino Carrozzo

META Centre
Consorzio Pisa Ricerche
C.so Italia 116, 56125 Pisa, ITALY
Telephone: (+39) 050 915 811
Fax: (+39) 050 915 823
Email: g.carrozzo@cpr.it

Stefano Giordano, Giodi Giorgi,
Federico Rossi, Franco Russo

Dept. of Information Engineering
University of Pisa
via Caruso, 56122 Pisa, ITALY
Telephone: (+39) 050 2217 511
Fax: (+39) 050 2217 522
Email: {s.giordano,g.giorgi,
f.rossi,f.russo}@iet.unipi.it

***Abstract*— The incoming GMPLS standardization is paving the way for new configurable Traffic Engineering (TE) policies and new survivability schemes for transport networks. In this context, a centralized Path Computation System (PCS) has been implemented, suited for transport networks with a GMPLS control plane. After a brief description of the requirements for a PCS in a GMPLS network, some design issues for the proposed implementation are drawn, with particular emphasis on the centralized approach and on the strategies for achieving the connection survivability. Some results of an intensive testing campaign are shown for the validation of the design choices.**

I. Introduction

The international standardization committees (e.g. ITU-T, OIF and IETF) are all converging in the design of an integrated network with a common Generalized Multi-Protocol Label Switching (GMPLS) control plane. GMPLS will manage all the network data planes [1][2], providing the required automation in the computation, the setup and the recovery of circuits for next-generation Optical Transport Network (OTN). GMPLS is an extension to devices capable of performing switching in time, wavelength and space domains of the MPLS control plane architecture. The core GMPLS architecture is based on a set of extensions to protocols for routing (e.g. OSPF and IS-IS) and signalling (e.g. RSVP), just available in IP networks. In the GMPLS context, other signalling protocols have been proposed (e.g. LDP, CR-LDP). Moreover, a new link management protocol (i.e. LMP) has been designed from scratch in order to handle correctly the distinction between the data plane and the control plane; indeed, they might not share the same link connection as in the IP networks (e.g. SONET/SDH networks, DWDM networks, etc.). From a routing perspective, the GMPLS extensions provide new information for circuit computation and they enable configurable Traffic Engineering (TE) policies and new recovery strategies. In such a context, this paper takes aim at describing the implementation of a centralized Path Computation System (PCS) suited for transport networks with a GMPLS control plane. In details, Sec. II is focused on the requirements for the PCS in a centralized GMPLS network scenario. In Sec. III the design issues for the proposed implementation are drawn, with particular emphasis on the centralized approach and on the strategies for achieving the connection survivability. Some results of an intensive testing campaign are shown in Sec. IV while conclusions and future directions are given in Sec. V.

II. Requirements for a GMPLS PCS

The upcoming standardization for GMPLS architecture is focused on protocols objects and mechanisms, while only high level requirements are proposed for traffic engineering (TE) and survivability. Traffic Engineering is fundamental for load balancing in the transport network, in order to avoid the overloading (up to saturation) of some network resources and the sub-utilization of others. Path computation systems for standard IP networks are generally based on distributed, fast and simple routing algorithms (e.g. of the Shortest Path First class -SPF-[3]), integrated into the routing protocol module. This algorithms walks along a graph derived from the real network. The graph contains only the routing-capable nodes (a.k.a. vertices) and the links between them (a.k.a. edges) with an appropriate the link metric. In the GMPLS context, a link connecting two ports of neighbouring nodes may consist of more than one consecutive physical resources (e.g. fibres), possibly crossing routing-incapable devices (e.g. regenerators, optical amplifiers, optical mux/demux, etc.). For this reason a set of properties is assigned to each link for routing purposes (e.g. TE metric, available/used bandwidths, resource colours, SRLG list, inherent protections, etc.), transforming the traditional links in Traffic Engineering links (TE-links). A GMPLS path calculator is expected to return Label Switched Paths (LSPs), i.e. sequences of nodes, TE-links and labels, which try to match some constraints derived from the TE information above. Once an LSP is computed, it describes univocally a unidirectional or bi-directional connection (electrical and/or optical) between a source and a destination node. The standard SPF algorithms are not suited for such a computation, as they cumulate only the link metric along the graph. A modified SPF algorithm is needed, called Constraint-based Shortest Path First (CSPF), as routes should be the shortest among those which satisfy the required set of constraints [4]. Another

important issue, raised by the great traffic amount carried on a LSP, is the connection survivability after a fault. The solution for this problem depends on the recovery strategy implemented in the network. New recovery strategies are enabled by the GMPLS control plane according to the overall taxonomy sketched in Figure 1 (ref. [5][6] [7] for details).

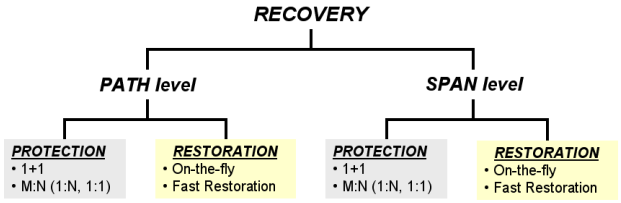

Fig. 1. Recovery taxonomy.

Two recovery classes may be distinguished:

- *path-level*, in which a failure notification is propagated till the end nodes of the affected LSP and there solved (a.k.a. end-to-end);
- *span-level*, in which a failure is notified and solved at intermediate nodes, next to the failed resource (a.k.a. local repair).

For each of this two classes a further distinction may be done between:

- *protection strategies*, which perform pre-calculation and pre-allocation of a backup LSP or set of spans;
- *restoration strategies*, in which a new LSP (or set of spans) is dynamically allocated only at time of failure (On-the-fly) or it is pre-computed and only booked for a future restoration (Fast Restoration).

Because of the sub-optimality of the resulting backup paths, span level strategies are prone to waste resources in the network, whereas end-to-end recovery strategies are more efficient, because they provide the computation of the best end-to-end backup path in the network. Moreover, restoration fits better the dynamical assignment/release of the network resources with respect to protection; but, in case of a fault, a higher blocking probability for the restoring traffic might be experimented, due to the failure handling by control plane mechanisms instead of hardware ones (e.g. detection, notification and mitigation). A common requirement for all the recovery strategies shown above is the disjointness between the resources (links or nodes) used by the primary route and by its backup. This is needed to minimize the blocking probability of the dynamical recovery action in case of fault. In the GMPLS architecture different levels of disjointness for LSPs are defined [4][7]:

- *node*, in which different nodes (and different links) are crossed by the primary-backup pair of LSPs;
- *link*, in which only different links are crossed by the two LSPs;
- *SRLG*, in which the Shared Risk Link Group lists of the two LSPs have no intersection.

The first two kinds of disjointness are related to the logical layer of the transport network, e.g. the TE-topology. On the contrary, the third type of disjointness (e.g. SRLG one) is related to the physical layer, as SRLGs may be used to locate conduits, fibres or general risks for physical resources. Node disjointness is a more stringent condition than the link one; so, two node disjoint paths are also link disjoint. The two levels are distinguished because a graph might block vertex disjoint search, but might still provide edge disjointness.

III. Design Issues for a GMPLS PCS

In the GMPLS architecture no specification is available for the implementation of a PCS module, as this issue is considered implementation-dependent. Moreover, no preference can be derived by the standards on the choice of a centralised or a distributed implementation, besides of the intrinsic distributed approach of the GMPLS control plane.

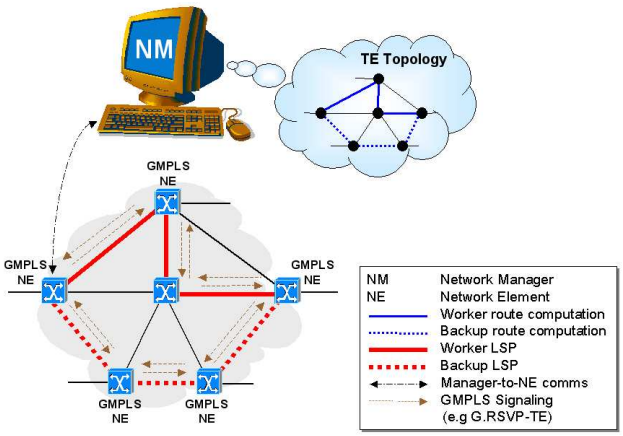

Fig. 2. GMPLS network model with a centralized PCS.

The architectural choice we made in the context of the TANGO project is for the implementation of the PCS module inside a centralized network manager (NM). This solution promises to be the most effective for a full and flexible handling of traffic engineering and survivability into the network, particularly when these requirements need to be extended to a multi-area (or multi-domain) scenario. In our implementation the PCS acts as a path computation server for the GMPLS network (ref. Figure 2), receiving from the GMPLS Network Elements (NE) the topology information and the computation requests across a single- or multi-area/AS. The LSPs computed by PCS (if any) are communicated to the ingress GMPLS NE, triggering a standard GMPLS signalling session (e.g. via G.RSVP-TE). The communications between the NM and the NEs are carried out by means of COPS protocol with proper extensions (ref. [8] for details). Focusing on the PCS module, we based our routing engine on an implementation of the Dijkstra SPF algorithm. In order to handle Constrained SPF computations, we added a TE-constraints validation step to the well-known Dijkstra flow [3]. One of these constraint validations is the check on the bandwidth availability of the candidate link. Moreover, in order to let the algorithm converge

towards an optimal SPF solution (e.g. between those which satisfies the required TE-constraints), the metric we chose to minimize during computation is bandwidth-dependent, according to the following equation:

$$link\_weight = std\_metric + TE\_metric + F(avail\_bw) \quad (1)$$

where the *std_metric* is the standard OSPF link metric, the *TE_metric* is the GMPLS metric for TE purposes, *avail_bw* is the available bandwidth on the TE-link and *F(x)* is a proper Traffic Engineering function designed to balance load (e.g. bandwidth consumption) in the topology. The main feature of our PCS is in the processing of LSP requests with survivability requirements. Based on the recovery taxonomy at path-level and on the requirements shown in Sec. II, we identified four Classes of Recovery (CoR) for the LSP requests, distinguishing the survivability of the connections as:

- *Gold*, when the lowest blocking probability for the recovery and the fastest reaction times (e.g. around 50ms) are required. This CoR applies to the protection strategies; it implies the computation of a pair of maximally disjoint paths in order to guarantee the total resource redundancy and uncorrelation.
- *Silver*, when fast reaction times (e.g. less than 1s) are required. This CoR applies to Fast Restoration strategies; it implies the computation of a pair of disjoint paths in order to guarantee a level of resource uncorrelation.
- *Bronze*, when reaction times around 1s are acceptable. This CoR applies to On-the-fly restoration strategies; it implies the computation of an optimal worker path when the request is received and the computation of an optimal backup one at failure occurrence.
- *Unprotected*, when no survivability is requested for the traffic of the LSP (e.g. it may be preemptable or best-effort traffic).

The bronze service is the slowest approach because of the time spent for failure detection, localisation, notification and mitigation. The silver service is sub-optimal as well, because the backup paths do not take into account any modification of the network load occurred in the meanwhile. So, we chose to compute the backup LSP trying to satisfy the required link/node disjointness, but with no guarantee for success and for optimality (i.e. the resulting backup might not be maximally disjoint w.r.t its primary and the pair might not have the cumulative shortest cost in the network). For this kind of computation we used the Two Step Approach (TSA) algorithm, which is based on a Dijkstra SPF [3] and on a simple temporary network transformation for avoiding links/nodes of the worker path. The main advantages of such an algorithm are in the easiness of implementation and in the limited complexity both of the SPF algorithm (e.g. the Dijkstra complexity in our implementation) and of the network transformation. The gold service is the most exacting in terms of optimality of the computation, because a least cost pair of disjoint paths is required, providing that it is also the maximally disjoint pair in the network. Many algorithms have been proposed in literature for such a computation [9] [10], most of them in the general context of the K-shortest path theme. We focused on the work by R. Bhandari because it promised lower theoretical complexity w.r.t. other algorithms (e.g. the famous Suurballe's one), when only $K = 2$ shortest paths are searched. The Bhandari's algorithm implemented in this work is the optimal counterpart to the sub-optimality of TSA one. However this optimality is paid for a higher complexity introduced by the required network transformation, by a modified SPF running also on negative graphs and by an interlacing/re-ordering procedure for the final paths.

IV. Performance studies

This section is aimed at highlighting the performances of the routing scheme adopted when computing a pair of maximally disjoint shortest paths with respect to links or to nodes. Measures have been collected on different topologies with increasing meshing degrees:

- an interconnected rings topology, with 64 nodes in 8 rings and meshing degree 3.14 (ref. Figure IV), derived from an actual European transport network;
- two simple Manhattan topologies with 49 and 100 nodes and meshing degrees 3.43 and 3.60, respectively (ref. Figure 4a);
- six half-meshed Manhattan topologies with 16, 25, 36, 49, 64 and 100 nodes and meshing degrees 4.125, 4.48, 4.72, 4.9, 5.03 and 5.22, respectively (ref. Figure 4b);
- six meshed Manhattan topologies with 16, 25, 36, 49, 64 and 100 nodes and meshing degrees 5.23, 5.76, 6.11, 6.37, 6.56 and 6.84, respectively (ref. Figure 4c).

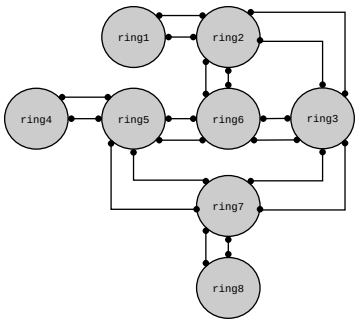

Fig. 3. An 8-interconnected rings topology.

All these topologies have been modelled with generic nodes, configurable as SDH 4/4 Cross-Connects. Nodes have been assumed to be fully connectable, i.e. any of their ingress port may be cross-connected to an egress one. Adjacent nodes have been connected by a TE-link with random values for its TE-information (e.g. TE metric, available/used bandwidths, resource colours, SRLG list, etc.). All the TE-links have been assumed to be bi-directional, each configured with 4 STM-64 ports VC4-multiplexed; so, a maximum of 256 allocable labels has been obtained in each adjacency. The computational environment for all the tests is based on a Pentium III 800MHz

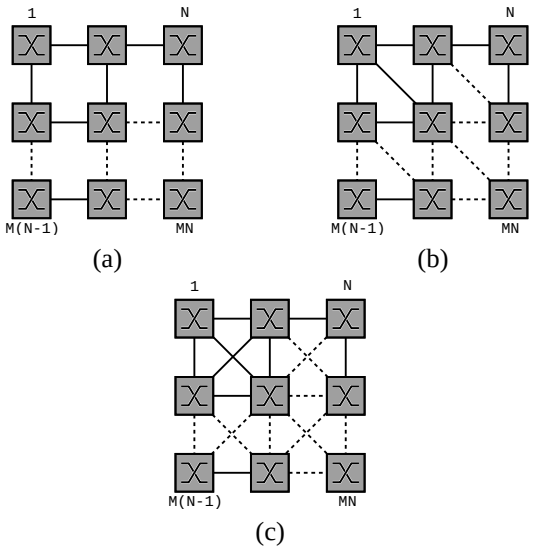

Fig. 4. Generic lattice topologies: (a) simple Manhattan; (b) half-meshed Manhattan; (c) meshed Manhattan.

PC with Linux RedHat 7.1 OS. A large number of LSP computations (*req_path*) have been requested on each topology (e.g. up to a connection request from each node towards all the others), trying to establish a stress condition for the algorithm operations. All the requests have been configured for bi-directional LSPs, as this is a common requirement for circuits in a transport network. For each topology have been observed:

- the number of computed LSPs (*comp_path*);
- the number of totally link-/node-disjoint pairs of LSPs (*disjoint*);
- the mean computation time for each LSP request (*time*).

The overall performance of the two types of algorithms on each topology has been measured by evaluating a Global Performance Factor (GPF), defined as:

$$GPF = \frac{disjoint}{req\_path} \cdot \frac{comp\_path}{req\_path} \quad (2)$$

The first term is related to the algorithm's effectiveness in creating maximally disjoint paths while the latter represents the algorithm's effectiveness in computing valid paths, according to the resource availability on TE-links. This latter term has no effect on GFP in case of a theoretical infinite resource availability. In Figure IV and Figure IV the GPF is drawn at the different meshing degrees, showing a mean higher performance for the Bhandari's algorithm w.r.t. the TSA one ($+0.57\%$ for link- and $+2.79\%$ for node-disjointness). The higher complexity in Bhandari's algorithm is responsible for an increase in mean computation times for each LSP request ($+3.59\%$ for link and $+4.14\%$ for node-disjointness) as shown in Figure IV only for the link because the two trends are similar. This is due to the higher complexity introduced by Bhandari's algorithm (e.g. the graph transformation, the modified SPF for negative graphs and, last but not least, the interlacing/re-ordering procedure for the final paths).

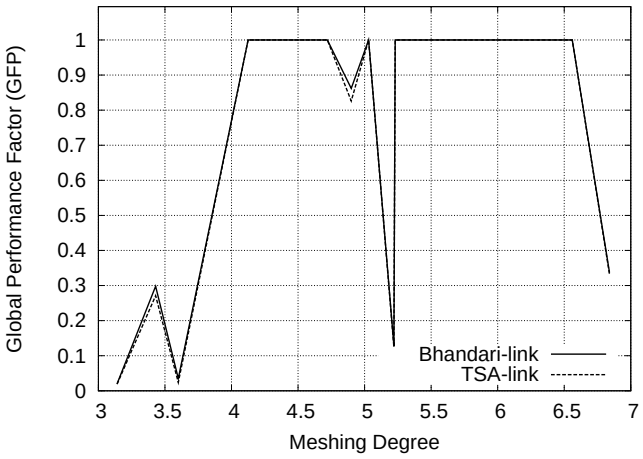

Fig. 5. GPF for the two algorithms in case of link-disjointness computation

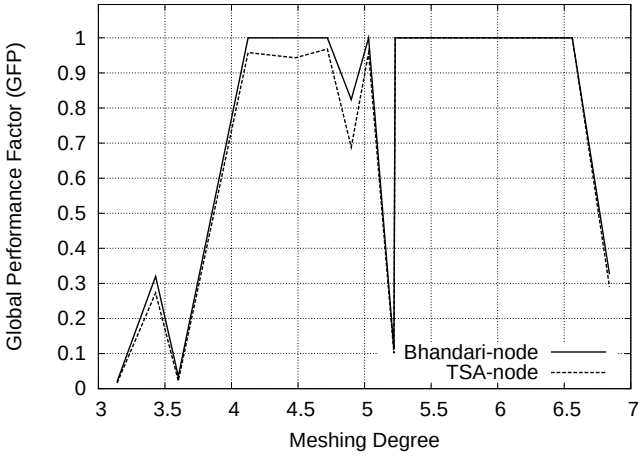

Fig. 6. GPF for the two algorithms in case of node-disjointness computation

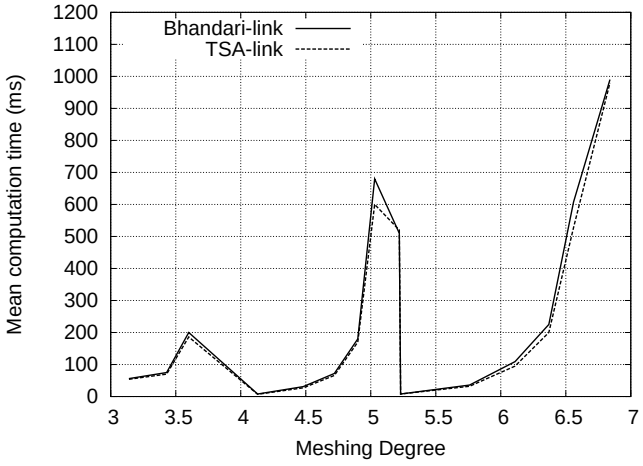

Fig. 7. Mean Computation Time (ms) for an LSP request (link-disjointness)

Results highlight an alignment between the Bhandari's algorithm and the TSA one when meshing degree increases. This is due to the higher resource availability in the network.

However, this trend is strictly related to the variance of the TE-metric values for the TE-links: actually, other tests, not presented here, showed that the more TE-metrics were different between TE-links, the more the Bhandari's algorithm achieved the best performance w.r.t the TSA one. Moreover, these results confirmed that node disjointness proves to be a more exacting requirement than the link one, whatever algorithm is used (e.g., Bhandari's or TSA). In fact, link-disjoint algorithms provided generally a higher number of disjoined paths. The higher number of path computation failures at lower meshing degrees (e.g. interconnected rings and simple Manhattan) is strictly related to the resource availability on TE-links. In case of the TSA algorithm, this issue merges with the intrinsic sub-optimality of this algorithm. This assertion is supported by the results obtained for the 8-interconnected rings and the Manhattan topologies in case of theoretical infinite resource availability (ref. Figure IV). We observed that only the Bhandari's algorithm provided disjoint paths for every request, with an acceptable increase of the main computation time.

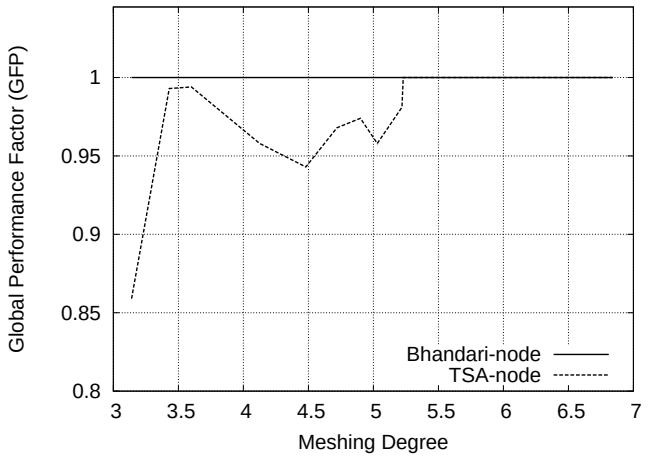

Fig. 8. GPF in case of node-disjointness and theoretical infinite resource availability

V. Concluding remarks

This paper describes the implementation of a centralized Path Computation System (PCS) suited for transport networks with a GMPLS control plane. The requirements and the major design issues for the PCS are drawn, with particular emphasis on the centralized approach and on the strategies for achieving the connection survivability. Some results of an intensive testing campaign are given in support of the design choices w.r.t. survivability. Other tests are going to be completed to point out the effects of different Traffic Engineering functions on the load balancing into the GMPLS network.

Acknowledgements

This work has been supported in part by the Italian Ministry of Education, University and Research through the project TANGO (MIUR Protocol No. RBNE01BNL5).

References

[1] E. Mannie (Editor) et al., *Generalized Multi-Protocol Label Switching (GMPLS) Architecture*, draft-ietf-ccamp-gmpls-architecture-07.txt, Internet Draft, Work in progress, May 2003.

[2] L. Berger (Editor) et al., *Generalized Multi-Protocol Label Switching (GMPLS) Signaling Functional Description*, RFC 3471, Jan. 2003.

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

[4] J. Strand et al., *Issues for Routing in the Optical Layer*, IEEE Communications Magazine, pages 81-87, Feb. 2001.

[5] G. Mohan-C and S. Ram Murthy, *Lightpath Restoration in WDM Optical Networks*, IEEE Network, pages 24-32, Nov./Dec. 2000.

[6] G. Carrozzo et al., *A Pre-planned Local Repair Restoration Strategy for Failure Handling in Optical Transport Networks*, Photonic Network Communication, vol. 4 (no. 3-4), pages 345-355, Jul./Dec. 2002.

[7] P. Lang (Editor) et al., *Generalized MPLS Recovery Functional Specification*, draft-ietf-ccamp-gmpls-recovery-functional-00.txt, Internet Draft, Work in progress, Jan. 2003.

[8] *Traffic models and Algorithms for Next Generation IP networks Optimization (TANGO) Project*, http://tango.isti.cnr.it/.

[9] R. Bhandari, *Survivable Networks - Algorithms for Diverse Routing*, Kluwer Academic Publisher, 1999

[10] J.W. Suurballe, *Disjoint Paths in a Network*, Networks, vol 4., pages 125-145, Jul. 1-3th 1974.