A Tabu Search Heuristic for the Off-Line MPLS Reduced Complexity Layout Design Problem

Sergio Beker, Nicolas Puech, and Vasilis Friderikos

École Nationale Supérieure des Télécommunications,
Computer Science and Networks Department,
46, Rue Barrault, 75634 Paris CEDEX 13, France
{beker|npuech|freideri}@enst.fr

Abstract. MPLS paths can be calculated on-line as demands arrive or off-line for an estimate of the traffic demand over time. Off-line calculation has the advantage of allowing a globally optimal network design. From the operational standpoint, a layout design for large networks should consider minimizing the layout complexity, thus reducing the cost of operation. In this paper, we formulate an optimization problem whose objective is to minimize the number of required hops. An end-to-end path delay constraint provides quality of service (QoS) guarantees. The resulting Minimum Path Set and Flow Allocation Problem (MPSFAP), formulated as a Mixed Integer Non Linear Program (MINLP) is $\mathcal{NP}$-complete. As such, it is only possible to solve it exactly on small size networks. To overcome this limitation, we designed a Tabu Search (TS) based algorithm that computes good quality approximate solutions for the MPSFAP problem. Solutions for medium to large size networks in reasonable CPU time and within acceptable optimality tolerances are found using the proposed TS based algorithm.

keywords MPLS, Traffic Engineering, Layout Complexity, Multicommodity Flow, MINLP, Tabu Search, LSP.

1 Introduction

One of the main reasons for present market enthusiasm around MPLS architecture is its capability to implement evolved traffic engineering (TE) functionalities [1] (e.g. required to support the Next Generation Internet). In MPLS based IP networks, all packets in a Label Switched Path (LSP) between source and destination nodes follow a predefined route, enabling TE mecanisms such as load sharing and flow separation. This allows the network operator to offer differentiated services with Quality of Service (QoS) guarantees, while using the available resources cost-effectively. Planning and dimensioning the network under varying traffic conditions then becomes a central issue for the operator.
Given the physical topology, the operator has to design a layout or *virtual* topology (i.e. find an optimal set of paths and a flow distribution over the physical network) to meet a given demand, as well as to adapt the layout to varying

traffic conditions. Additionally, each path must ensure some Quality of Service (QoS) requirements (e.g. end-to-end delay, throughput).

To design the MPLS layout, on-line or off-line approaches have been proposed. On-line methods calculate paths as demands arrive, either by updating the routing protocol metrics for each link [2], or by calculating the route at the source node using available network load information. As the resulting problem of finding one constrained path is usually $\mathcal{NP}$-complete [3], a number of algorithms have been proposed in order to find approximate solutions in times compatible with the on-line operation. Constrained Shortest Path First (CSPF) [4] provides Open Shortest Path First (OSPF) extensions to find the shortest path meeting one or a set of constraints. Minimum Interference Routing (MIRA) [5] proposes an algorithm to find a path avoiding the critically loaded links, and [6] improves MIRA by finding a set of k-minimum interferring paths with bounded length. On-line approaches have the advantage of adapting the network layout to changing load conditions. However, by taking into account the flow dynamics incrementally, an on-line method might produce a sub-optimal resource usage in the long run. Off-Line LSP layout design considers global information about the state of the network and traffic characteristics over time. The network can be optimally engineered and set up around a point of operation on a long term basis. Complementary, on-line decisions can be made as how to route arriving demands, in order to be able to accommodate traffic variations in the short term. Common objective functions for the off-line design aim at minimizing the maximum loaded link or minimizing the average cross network packet delay. However, the operator's main concern is to reduce the cost of operation while honoring the Service Level Agreements (SLAs). The cost of operation is directly related to the number of paths (hence to the layout complexity) to be monitored [7]. A main objective in the network design should then be to minimize the layout complexity, while meeting QoS guarantees.

In this paper we focus on the problem of obtaining an MPLS layout which is optimal w.r.t. the number of required hops in the set of LSP in the layout (indirectly driving to a reduction in the number of required paths). The Minimum Path Set and Flow Allocation Problem (MPSFAP) is formulated as a Mixed Integer Non-Linear Multicommodity Flow Problem (MINLP-MCFP). The objective is to obtain the minimum set of paths and the associated flow distribution under bounded end-to-end path delay for each Class of Service (CoS) for a given traffic demand. We first define the notation used through the paper in Section 2. In section 3 the general MPSFAP problem is set for multiple Classes of Service (CoS). Even in the case when we consider only one class of traffic, the model is intractable and may be solved exactly only for small size networks. This led us to develop algorithms that provide (hopefully good quality) approximate solutions to the considered problem. An adaptation of the Tabu Search (TS) heuristic applied to the MPSFAP problem is presented in section 4. The evaluation of numerical results for the MPSFAP problem with a single CoS are presented in Section 5. Results obtained from exact solvers for small size networks are compared to results obtained from the TS algorithm. The comparison shows that

the approximate TS solutions are close to the exact ones, making the TS based algorithm a good candidate to solve MPSFAP on large problem instances. We then test the TS algorithm on real size networks to assess our approach. Finally, in Section 6 conclusions and future work are presented.

2 Model and Notation

2.1 Network

The physical network is represented by a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of vertices indexed $1, 2, \ldots, N$; $\mathcal{E}$ is the set of directed edges indexed $1, 2, \ldots, M$. The vertices represent the MPLS Label Switching Routers (LSR), and each edge represents a connecting link between two LSRs. Link i has capacity C_i. Let $\mathbf{C}$ be the single column matrix representing the M-dimensional capacity vector.

2.2 Paths

Let the node couples $q = (m, n)$, with $m \neq n$, be indexed $1, 2, \ldots, Q$ where $Q = N(N-1)$. Let the classes of service supported by the network be indexed $1, \ldots, L$. For each source-destination couple, a *commodity* $d_{q,l}$ describes the demand requested for service class l and couple q. The commodity $d_{q,l}$ may be routed via K_q different paths (or routes) binding the node couple q. We denote by K_q the total number of such paths, indexed $a_q^1, a_q^2, \ldots, a_q^{K_q}$. Any path a_q^k can be represented as a vector with entries $a_{q,i}^k = 1$ if path a_q^k lies on link i, and 0 otherwise, for $i = 1, 2, \ldots, M$. The paths are acyclic, i.e. no path traverses twice the same link or the same node. We can represent the $K = \sum_{q=1}^{Q} K_q$ single paths between all pair of nodes as the $M \times K$ arc-path incidence matrix $\mathbf{A}$:

$$\mathbf{A} = \left(A_1 \middle| A_2 \middle| \ldots \middle| A_Q \right)$$

where :

$$A_q = \begin{pmatrix} a_{q,1}^1 & \ldots & a_{q,1}^{K_q} \\ a_{q,2}^1 & \ldots & a_{q,2}^{K_q} \\ \vdots & & \vdots \\ a_{q,M}^1 & \ldots & a_{q,M}^{K_q} \end{pmatrix}$$

for $q = 1, ..., Q$, and where the block A_q is the arc-path matrix corresponding to the K_q paths for the node couple q. Each path a_q^k has an associated weight $w_{q,l}^k$ to carry a portion of commodity $d_{q,l}$. Let $\mathbf{W}$ be the $K \times L$ weight matrix.

2.3 Demands and Flows

The demand matrix is represented by a $Q \times L$ dimensional matrix $\mathbf{D}$ whose entry $d_{q,l}$ stands for the amount of traffic requested for the service class l for the node couple q. Let be the flow assigned to path a_q^k for the service class l denoted by $b_{q,l}^k$. The $K \times L$ matrix $\mathbf{B}$ represents the flow distribution per class and path:

$$\mathbf{D} = \begin{pmatrix} d_{1,1} & \dots & d_{1,L} \\ d_{2,1} & \dots & d_{2,L} \\ \vdots & \vdots & \vdots \\ d_{Q,1} & \dots & d_{Q,L} \end{pmatrix} \qquad \mathbf{B} = \left(\begin{array}{ccc} b_{1,1}^1 & \dots & b_{1,L}^1 \\ \vdots & \vdots & \vdots \\ b_{1,1}^{K_1} & \dots & b_{1,L}^{K_1} \\ \hline \vdots & \vdots & \vdots \\ \hline b_{Q,1}^1 & \dots & b_{Q,L}^1 \\ \vdots & \vdots & \vdots \\ b_{Q,1}^{K_Q} & \dots & b_{Q,L}^{K_Q} \end{array}\right)$$

Let us define $\mathbf{R} = (r_{q,i})$ the $Q \times K$ commodity-path incidence matrix, where entry $r_{q,i} = 1$ if i lies in the range $[K_1 + K_2 + \ldots + K_{q-1} + 1, K_1 + K_2 + \ldots + K_{q-1} + K_q]$ and 0 otherwise, for $1 \leq q \leq Q$ and $1 \leq i \leq K$. Let x_i be the value of the total flow traversing link i, for $1 \leq i \leq M$, and $\mathbf{X}$ be the M dimensional single column matrix of link flows:

$$\mathbf{X} = \mathbf{A} \cdot \mathbf{B} \cdot \mathbb{1}_L \tag{1}$$

where $\mathbb{1}_L$ is the L-dimensional column matrix with all its entries set to 1.

3 Minimum Path Set and Flow Allocation Problem (MPSFAP)

As stated before, the cost of operation for large networks is related to layout complexity. We define layout complexity as the number of paths necessary to transport a given demand matrix $\mathbf{D}$. The MPSFAP problem can be formulated as a MINLP problem, whose objective is to minimize the total number of required paths:

Given:

$$\mathbf{A}, \mathbf{C}, \mathbf{D}, \boldsymbol{\Theta}, \boldsymbol{\Delta}, \epsilon$$

minimize:

$$\sum_{l=1}^{L} {}^t\mathbf{W}^{(\mathbf{l})} \cdot \mathbf{H}^{(\mathbf{l})} \tag{2}$$

subject to:

$$\mathbf{X} = \mathbf{A} \cdot \mathbf{B} \cdot \mathbb{1}_{\mathbf{L}} \leq (1-\epsilon)\mathbf{C} \tag{3}$$

$$\mathbf{R} \cdot \mathbf{B} = \mathbf{D} \tag{4}$$

$$h_{q,l}^k \sum_{i=1}^{M} \frac{\lambda a_{q,i}^k}{C_i - x_i} \leq \theta_{q,l} \qquad l = 1, \ldots, L; \quad k = 1, \ldots, K_q; \quad q = 1, \ldots, Q \tag{5}$$

$$0 \leq b_{q,l}^k \leq h_{q,l}^k \delta_{q,l} \qquad l = 1, \ldots, L; \quad k = 1, \ldots, K_q; \quad q = 1, \ldots, Q \tag{6}$$

$$h_{q,l}^k \in \{0, 1\} \qquad l = 1, \ldots, L; \quad k = 1, \ldots, K_q; \quad q = 1, \ldots, Q \tag{7}$$

where:

- $\mathbf{\Theta}$ is the $Q \times L$ matrix containing the maximum tolerable delay $\theta_{q,l}$ for each commodity $d_{q,l}$;
- $\mathbf{\Delta}$ is the $Q \times L$ matrix containing the maximum acceptable flow $\delta_{q,l}$ on one path for the commodity $d_{q,l}$;
- $\epsilon > 0$, and λ is the average packet size.
- $\mathbf{H}$ is a $K \times L$ matrix, whose entries are binary variables taking the value $h_{q,l}^k = 1$ if path a_q^k is in use by commodity $d_{q,l}$, and 0 otherwise. $\mathbf{H^{(l)}}$ and $\mathbf{W^{(l)}}$ are the single columns corresponding to the l^{th} CoS of $\mathbf{H}$ and $\mathbf{W}$ respectively.

Constraint (4) expresses that traffic demands must be met by allocated flows (no flow loss). Constraint (5) ensures that no packet belonging to class l traversing a path connecting the node pair q will experience a delay greater than $\theta_{q,l}$. The path weights $w_{q,l}^k$ can be used by the operator to fine tune the cost structure of the path layout. Hop-count based weights, for instance, account for the cost of signaling long paths:

$$w_{q,l}^k = \sum_{i=1}^{M} a_{q,i}^k \qquad l = 1, \ldots, L; \quad k = 1, \ldots, K_q; \quad q = 1, \ldots, Q \tag{8}$$

According to the studied objective function, flows are allocated to the paths up to the limit given by the constraints (5) and (6).
The advantage of the flow distribution obtained with the objective function (2) is that commodity splitting is reduced. Its main drawback is that it usually leads to an important path delay difference between the various LSPs. This doesn't have a major impact if a flow-by-flow load sharing strategy is used (e.g. all packets in a TCP flow are forwarded on the same LSP).

In the case of packet-by-packet load sharing strategy, the objective function should include a term to distribute flows according to path delay to avoid significant packet reordering:

$$\alpha \sum_{l=1}^{L} {}^t\mathbf{W^{(l)}} \cdot \mathbf{H^{(l)}} + \beta \sum_{q=1}^{Q} \sum_{k=1}^{K_q} \sum_{l=1}^{L} h_{q,l}^k \sum_{i=1}^{M} \frac{\lambda a_{q,i}^k}{C_i - x_i} \tag{9}$$

where $0 \leq \alpha \leq 1$ and $0 \leq \beta \leq 1$ are factors used to balance the importance of each term in the objective function.

4 Tabu Search: A Metaheuristic Approach

The major drawback of multicommodity flow models is that they are computationally intractable even for small problem instances. Optimization algorithms based on metaheuristics have recently gained interest because they are able to cope with problem instances of large size. Among others, simulated annealing [8], genetic algorithms [9] and Tabu Search (TS) [10, 11] have been proposed to tackle multicommodity flow problems. The computed solution however is not guaranteed to be optimal. That is why it is necessary to check the accuracy of the solutions provided by such a heuristic based algorithm and to compare them with exact solutions on small size problems.

A detailed description of the TS algorithm can be found in [10]. Briefly, TS consists of exploring the space of solutions until a number of iterations is reached or until a specific cost criterion is satisfied. The exploration starts with an initial solution computed by another algorithm (e.g. randomly generated). At each iteration, TS computes a set of solutions or *neighborhood* derived from the current solution via perturbations applied to it. All the solutions of the neighborhood are evaluated (see below) and the best one is selected as the new current solution. In order to prevent the algorithm from cycling along the same series of current solutions, a tabu list is maintained. It contains a number of last visited solutions, which cannot be chosen as long as they belong in this list. This allows the algorithm to choose a solution worse than the current one, allowing it to escape from the local minima encountered during the search. Three problem-specific elements must be defined in order to apply TS to the considered problems:

Initial Solution: An initial solution (i.e. layout) must be computed. It will constitute the starting point for the exploration of the solution space. The initial layout is the one obtained by choosing the shortest path for all the demands in the traffic matrix given as input.

Perturbation mechanism: A perturbation mechanism is necessary to generate a neighborhood of the current solution. Neighbor solutions are calculated by randomly choosing a source-destination pair. The current solution sets a given flow value along each path connecting this node pair. This flow distribution is randomly changed so that a new flow distribution meeting the flow demands but not necessarily meeting the link flow and path delay constraints is obtained. Hence, the neighbor generation process leads to new candidate solutions that may be considered as *valid* (solutions that meet all the constraints) or *invalid* (solutions that do not meet the link flow or path delay requirements). In particular, the flows in the paths belonging to the chosen node pair are rearranged in

a way that current flow traversing each path is taken into account in order not to unnecessary make appear new paths.

A cost function: A cost function $f : S \rightarrow \mathbb{R}$ that maps elements in the solution space S to real numbers must be defined to allow for solution comparison. We used a cost function that evaluates distinctly valid and invalid solutions. For valid solutions, the TS cost function is the function defined by Equation 2 for MPSFAP and defined by Equation 10 for MTDFAP (see Section 5 below for full definition of MTDFAP problem). For invalid solutions, an extra threshold value is added to ensure that the evaluation of an invalid solution is always greater than for any valid solution, along with a term dependent on the overload. This term is a descendent function of the overload proportion on overloaded links, so a slope towards valid solutions is created in the solution landscape. A complete description of the TS algorithm can be found in [12].

5 Experimental Results

MPSFAP belongs to the class of $\mathcal{NP}$-complete problems, as it can be viewed as a constrained shortest path problem between all couples, which is itself a $\mathcal{NP}$-complete problem [3]. In order to find the TS parameters which lead the algorithm to the production of good results, and in order to show that our TS based algorithm provides acceptable approximate solutions, we used a couple of small sized networks, denoted by NET1 and NET2. NET1 is a 4-node, 8-link and 24-LSP network. NET2 is a 4-node, 10-link and 38-LSP network. The simplicity of these networks allowed us to obtain exact results from a numerical solver and to use them as a reference for the evaluation of the TS solutions.

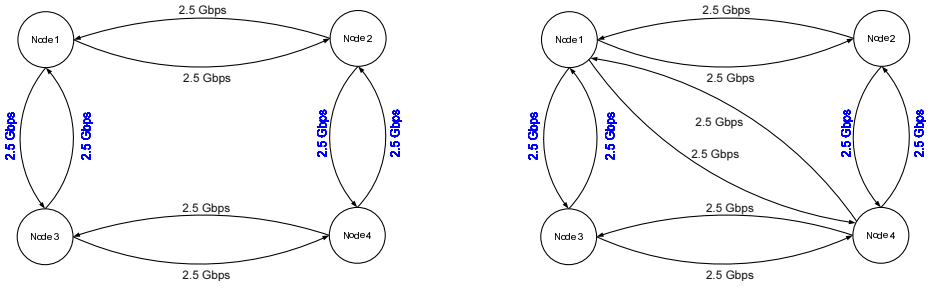

Fig. 1. NET1 and NET2 networks.

The TS algorithm was then tested on two real size networks denoted by VTHD and NSFNET. VTHD, consisting of 9 nodes 24 links and 550 LSPs, is a model version of a French network designed for research purposes, in the context

of the VTHD (Vraiment Très Haut Débit) [13] research project. NSFNET is a 14-node, 42-link and 496-LSP network.

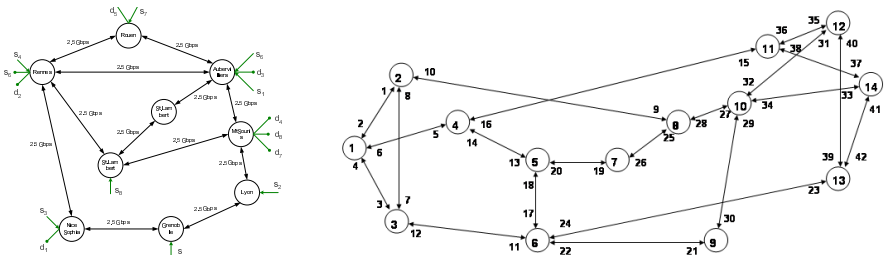

Fig. 2. VTHD and NSFNET networks.

We tested the algorithms in the particular case of a single service class. In order to derive results we had to consider the solutions obtained under various traffic conditions. We generated a series of 25 traffic matrices for the considered test networks. The matrices were obtained randomly according to the following method, which is derived from the method described in [14]. A certain percentage F of the demands for each source- destination node pair q is a random variable uniformly distributed within the interval $[0, \frac{C_q}{a}]$ and the rest $(100 - F)$ is uniformly distributed within the interval $[0, \frac{C_q Y}{a}]$, where C_q is the total capacity connecting the end nodes of couple q. The parameter a is an arbitrary integer which may be 1 or more, and Y is the ratio $\frac{max_q\{C_q\}}{min_q\{C_q\}}$. Parameter a is used to tune the traffic intensity on the network so that the produced matrices can lead to feasible MPSFAP problems, whereas Y allows a specific traffic distribution for some node pairs in the network. Mean packet size is assumed to be 128 bytes. In all cases link capacities are set to the same value of 2.5 Gbps. Results obtained for MPSFAP are compared with the results obtained with a reference objective function minimizing the total delay in the network, which we denote by minimum total delay flow allocation problem MTDFAP, and where the objective is to minimize:

$$\sum_{q=1}^{Q} \sum_{k=1}^{K_q} \sum_{l=1}^{L} h_{q,l}^{k} \sum_{i=1}^{M} \frac{\lambda a_{q,i}^{k}}{C_i - x_i} \tag{10}$$

subject to the same path-delay constraints. MTDFAP is also formulated as a MINLP problem. This objective function is extensively used through the literature and serves as a reference. All numerical results are obtained from the MINLP solver [15], available through the NEOS server on the Internet [16]. In all cases, solutions are obtained with $\epsilon = 0.0001$, and the flow limit for all commodities is $\delta_q = 2.5$ Gbps (i.e. the path flows can use up to the maximum available capacity in any link). Path delay limits are set to $\theta_q = 30$ μsec for NET1 and NET2 and $\theta_q = 50$ μsec for VTHD and NSFNET networks.

Table 1 shows the values obtained for the objective function with the Tabu Search heuristic algorithm and with the exact solver for both the MPSFAP and the MTDFAP problems. The presented values represent the range and the mean value of the objective function evaluations for 25 executions of each method (one execution for each traffic matrix). The chosen parameter values were $a = 4$, $Y = 1$ and $K = 24$ for the matrices generated for the tests with NET1 and $a = 5$, $Y = 1.33$ and $K = 38$ for the tests with NET2. Although the TS algorithm cannot always find the optimal value, the average evaluation difference between the approximate and the exact solution is not very large, since it reaches $13,75\%$ in the worst case. Results can be further improved by sacrificing more CPU time, i.e. by examining a greater number of possible solutions. Figure 3 shows the results obtained by the two algorithms for every matrix with MPSFAP for the NET1 and NET2 networks. Infeasible matrixes show no value in the figure.

Table 1. Comparison of the values (number of hops) obtained by the TS algorithm and the exact solver for NET1 and NET2.

	MPSFAP		MTDFAP ($\times 10^{-3}$)	
Network	Range	Average	Range	Average
	Tabu Search Results			
NET1	16-19	16.48	13.36-24.56	17.65
NET2	14-20	15.48	12.74-122.89	31.10
	Exact Solutions			
NET1	16-16	16	12.7-21.86	15.52
NET2	14-16	14.92	11.65-96.09	29

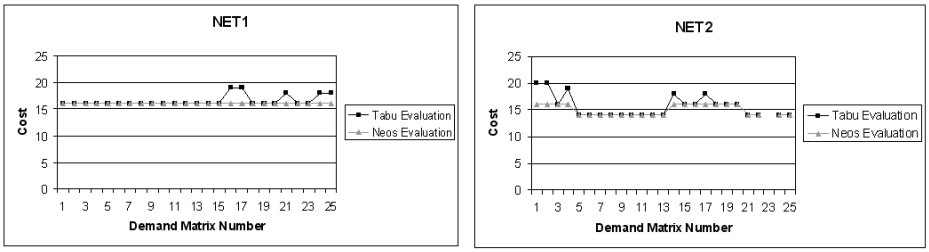

Fig. 3. MPSFAP evaluations obtained by Tabu Search and NEOS Server for NET1 and NET2 networks.

Table 2 presents the total number of paths required by MPSFAP and MTDFAP to allocate the demand imposed by the 25 used traffic matrices. Despite the small size of the tested networks, we can see that the MPSFAP problem leads to solutions with a smaller set of paths than MTDFAP. MTDFAP tends to use

more available paths to balance the network load, in order to lower the total cross network delay. The approximate solutions obtained with the TS algorithm are close to the exact ones in terms of used paths. Nevertheless we must keep in mind that the considered cost functions aim at minimizing the hop count, and as such the TS algorithm sometimes reaches better values than the exact solution in terms of used paths.

Table 2. Comparison of the number of used paths obtained by the TS algorithm and the exact solver for NET1 and NET2.

	MPSFAP		MTDFAP ($\times 10^{-3}$)	
Network	Range	Average	Range	Average
	Tabu Search Results			
NET1	12-13	12.20	12-14	12.52
NET2	12-14	12.64	12-14	12.96
	Exact Solutions			
NET1	12-14	12.32	12-14	12.44
NET2	12-14	12.66	12-14	12.72

From this first series of tests on small size networks NET1 and NET2, we observe that the approximate solutions computed by the TS algorithm are close to the exact ones. This makes our algorithm a good candidate to solve more complex problems. We performed a second series of experiments with TS on real size networks. This time, the deterministic solver is unable to compute (exact) solutions, so reference values will not be further available.

Exact solvers on powerful contemporary computers are not able to give a solution in an acceptable amount of time for these networks, whereas our TS algorithm is able to propose approximate solutions in a reasonable amount of time. The time of execution of the heuristic method has a primary role in the quality of the solutions, which can be clearly noticed in cases where a feasible solution can hardly been found for a particular problem instance.

As before, we generated random sets of 25 traffic matrices for the considered networks. More precisely, in order to take into account various parameter configurations, we generated two different matrix sets for the VTHD network (which leads to test1 and test2). The chosen parameter values were $a = 100$, $Y = 2.6$ and $K = 550$ (respectively $a = 120$, $Y = 2.6$ and $K = 550$) for the matrices generated for the first series (respectively second series) of tests with VTHD, and $a = 30$, $Y = 1.5$ and $K = 496$ for the tests with NSFNET.In the first set of VTHD traffic matrices (test1), parameter a is chosen to produce a quite heavy load in the network, while in the second set (test2) the choice $a = 120$ leads to milder traffic conditions. Table 3 shows the hop count on the network for two set of traffic matrices in VTHD and one in the NSFNET network. Under heavy traffic conditions, more paths are needed to allocate the demand, resulting in a greater number of required hops. Table 4 shows the corresponding number of

required paths for the same sets of traffic matrices for VTHD and NSFNET networks. As seen before, MTDFAP solutions require additional paths with respect to MPSFAP in order to minimize the total delay in the network.

Table 3. Value of the cost function (number of hops in the case MPSFAP) by the TS algorithm for VTHD test1, VTHD test2 and NSFNET.

	MPSFAP		MTDFAP ($\times 10^{-3}$)	
Network	Range	Average	Range	Average
VTHD test1	145-227	162.67	149-214	169.35
VTHD test2	139-160	144	140-162	152.04
NSFNET	390-430	410.6	411-466	428.17

Table 4. Number of used paths computed by the TS algorithm for VTHD test1, VTHD test2 and NSFNET.

	MPSFAP		MTDFAP ($\times 10^{-3}$)	
Network	Range	Average	Range	Average
VTHD test1	71-88	75.76	73-89	78.1
VTHD test2	71-75	72.20	72-77	74.48
NSFNET	182-193	186.83	187-200	191.17

We used a personal computer equipped with a Pentium IV processor at 2.4 GHz and 512 MB of physical memory to perform our tests. The typical processing time for the above given solutions is 5-11min. for the VTHD topology and around 20min. for NSFNET.

6 Conclusions and Future Directions

Our main concern in this paper is to obtain low complexity layouts, thus reducing the cost of operation for large networks, while still providing QoS guarantees for several CoSs. Reducing reconfiguration complexity (i.e. the number of paths that must be changed) helps reducing service disruption times and resource overdimensioning during transition management. We propose a MINLP model to obtain low complexity layouts for the traffic demand, while keeping the QoS guarantees. This model is $\mathcal{NP}$-complete and as such solvers are able to compute (exact) solutions only for small size networks. To overcome this limitation, we designed a Tabu Search based algorithm that provides approximate solutions for the studied problem. We compared the solutions computed by our algorithm to the exact ones for small size problem instances. The comparison showed that the approximate solutions are close to the exact ones, hence making the TS based

algorithm a good one to deal with the MPSFAP problem on real size problems. We presented the results obtained with TS for two different real size networks. The reasonable amount of CPU time required by the TS algorithm to solve these problems make it a good method to tackle the MPLS layout design problem.

Results obtained on small networks show that layout and reconfiguration complexity are significantly reduced when compared with classical approaches. The formulated optimization problems are $\mathcal{NP}$-complete. In order to compute optimal layouts for larger networks, we need to devise efficient algorithms providing approximate solutions. We are currently working on algorithms implementing heuristics which take advantage of the nature and knowledge of the underlying problem, and metaheuristic such as simulated annealing or tabu search.

References

[1] D. Adwuche, G. Malcom, J. Agogbua, M. O'Dell, and J. McManus. Requirements for traffic engineering over MPLS. RFC 2702, IETF, 1999.

[2] B. Fortz and M. Thorup. Internet traffic engineering by optimizing OSPF weights. In *Volume 2, p.519-528*, INFOCOM 2000, 2000. IEEE.

[3] R. Ahuja, T. Magnanti, and J. Orlin. *Network Flows: Theory, Alogorithms and Applications*. Prentice Hall Inc., 1993.

[4] D. Katz, D. Yeung, and K. Kompella. Traffic engineering extensions to OSPF version 2. Draft, IETF, 2002.

[5] M. Kodialam and T. Lakshman. Minimum interference routing with applications to MPLS traffic engineering. In *Volume 2, p.884-893*, INFOCOM 2000, 2000. IEEE.

[6] G. Banerjee and D. Sidhu. Path computation for traffic engineering in MPLS networks. ICN 2001, 2001. IEEE.

[7] S. Beker, D. Kofman, and N. Puech. Off-line reduced complexity layout design for MPLS networks. In *Proceedings of IP Operations and Management*, IPOM 2003, 2003. IEEE.

[8] S. Kirkpatrick, C.D. Jr. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. *Science*, (4598), 1983.

[9] C. Gazen and C. Ersoy. Genetic algorithms for designing multihop lightwave network topologies. *Artificial Intelligence in Engineering*, 13:211–221, 1999.

[10] F. Glover and M. Laguna. *Tabu Search*. Kluwer Academic Publishers, Boston, MA, 1997.

[11] N. Puech, J. Kuri, and M. Gagnaire. Models for the logical topology design problem. In *Proceedings of the 2nd IFIP-TC6 Networking Conference*. Springer-Verlag, May 2002.

[12] S. Beker. *Thesis: Optimization Techniques for the Dimensioning and Reconfiguration of MPLS Networks*. Télécom Paris, 2004 (to appear).

[13] Réseau national de reherche en télécommunications. Technical report, Ministère de l'Economie, des Finances et de l'Industrie de France, 2000-2004. www.vthd.org.

[14] D. Banerjee and B. Mukherjee. Wavelength-routed optical networks: Linear formulation, resource budgeting tradeoffs, and a reconfiguration study. *Networking, IEEE/ACM Transactions on*, 8(5):598–607, 2000.

[15] R. Fletcher and S. Leyffer. Numerical experience with lower bounds for MIQP branch and bound. *SIAM Journal of Optimization*, 8(2):604–616, 1998.

[16] J. Czyzyk, M. Mesnier, and J. Moré. The Neos Server. *IEEE Journal on Computational Science and Engineering*, 5:68–75.