



**HAL**  
open science

## **FATE: A Polynomial Time Framework for Flow Allocation in MPLS-TE Networks**

Samer Lahoud, Géraldine Texier, Laurent Toutain

► **To cite this version:**

Samer Lahoud, Géraldine Texier, Laurent Toutain. FATE: A Polynomial Time Framework for Flow Allocation in MPLS-TE Networks. LANMAN 2005: 14th IEEE Workshop on Local and Metropolitan Area Networks, Sep 2005, Crète, Greece. hal-02368183

**HAL Id: hal-02368183**

**<https://hal.science/hal-02368183>**

Submitted on 18 Nov 2019

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# FATE: A Polynomial Time Framework for Flow Allocation in MPLS-TE Networks

Samer Lahoud, Géraldine Texier and Laurent Toutain  
GET/ENST Bretagne

Département Réseaux et Services Multimédia

2 rue de la Châtaigneraie - CS 17607 - 35576 Cesson Sévigné Cedex - France  
samer.lahoud@enst-bretagne.fr, geraldine.texier@enst-bretagne.fr, laurent.toutain@irisa.fr

**Abstract**—In this paper, we present a novel polynomial time framework for the off-line flow allocation problem in MPLS-Traffic Engineering (MPLS-TE) networks. This framework, named FATE (Flow Allocation for Traffic Engineering), encompasses different objectives that illustrate the operator needs for traffic engineering and provides advanced control over the quality of the solution. Typically, we consider minimizing congestion, bandwidth consumption and operational costs. FATE is based on linear programming and presents an efficient solution for the problem both in terms of computational complexity and approximation guarantees. Simulation results show that we can get an efficient trade-off between congestion and bandwidth consumption while having a reduced number of Label Switched Paths (LSPs) for various known topologies.

## I. INTRODUCTION

Traffic engineering is a complex process that includes the set of strategies deployed by network operators in order to get maximum return on network capital expenditures (capex) while reducing operational expenditures (opex). Capex is determined in the network design process when capacity is installed in the core links, whereas opex is related to the cost of managing, supervising and tuning the network in order to get optimal performance. Traffic engineering can be implemented as an on-line and/or an off-line process. Off-line traffic engineering tends to optimize the network considering full knowledge of the demands. This is usually expressed as a traffic demand matrix, with point-to-point demands.

In this work, we introduce a novel polynomial time framework, named FATE (Flow Allocation for Traffic Engineering), for the off-line flow allocation problem in MPLS-Traffic Engineering (MPLS-TE) networks. First, FATE considers minimizing the global network congestion. Second, FATE tries to minimize resource consumption, expressed in terms of the total used bandwidth. Particularly, FATE enables to get efficient return on network capex involved in capacity installation by tuning the relevancy of the first two objectives. Finally, FATE provides a flow allocation solution with a minimum number of Label Switched Paths (LSPs) inducing minimum operational costs for an MPLS-TE network.

This paper is organized as follows: in section II, we introduce relevant related work. In section III, we present the different components in the FATE framework. Then, in section IV, we formulate the mathematical problems used in FATE and emphasize the polynomial time complexity. In section V

we present relevant simulation results. Finally, we conclude by presenting future work and possible extensions to the framework.

## II. RELATED WORK

Flow allocation is one of the problems that are well studied in the network flow domain. [1] presents a good overview of the general problem, including different formulations and numerous techniques associated with Linear Programs (LPs) such as relaxations and problem decomposition. Moreover, many articles have studied the application to MPLS traffic engineering. In [2], a multi-objective approach to the problem is introduced. This approach considers the compromise between load balancing, network costs, and the number of active LSPs. The problem is formulated as a single objective Mixed Integer LP (MILP) using a weighted sum of the different objectives and solved by means of standard commercial tools. However, no heuristic is presented in order to reduce the computation time. [3] presents a traffic engineering solution for differentiated reliability services. A single weighted objective is used to combine different optimization criteria such as resource consumption, protection level and rejection. The problem is also formulated as a MILP and the authors present two heuristics: a first heuristic is based on successive approximations by continuous relaxations of the MILP and a second heuristic based on the simulated allocation technique. [4] introduces a multi-objective framework based on utilization and resource usage. The authors propose linear programs that find near optimal paths satisfying the given traffic demand under constraints such as maximum hop count, and preferred node/link list. Moreover, a MILP formulation calculates the traffic split ratio for the multiple paths. However, the size of the problems that can be solved is very limited in practice. [5] introduces a novel heuristic for multi-objective traffic engineering in the Virtual Private Network (VPN) context. A multi-stage approach is used in order to successively reduce congestion and bandwidth utilization. Finally, the authors in [6] introduce a simple polynomial time framework for multi-path routing. They present efficient techniques that reduce the path set for simple separated problems such as minimum congestion or minimum bandwidth utilization. The mathematical tools introduced in [6] are similar to those used in our work. However, we introduce a more general framework adapted for

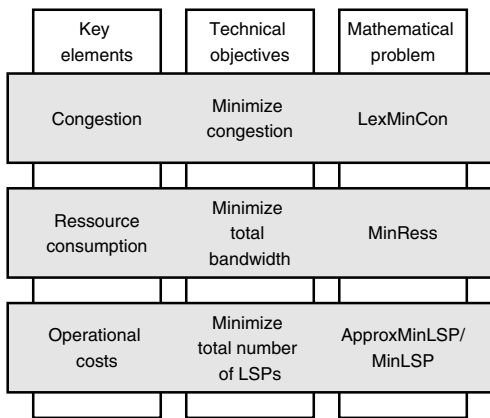


Fig. 1. FATE framework

MPLS traffic engineering. We add a more general concept for minimizing congestion, notably with *LexMinCon*. Moreover, we emphasize the trade-offs between the different elements in our framework, and provide insights on the tuning process that enables to reach a desired flow allocation solution.

### III. FRAMEWORK PRESENTATION AND MOTIVATION

Our main contribution in this article is the introduction of a polynomial time framework for flow allocation in MPLS-TE networks. This framework, named FATE, is illustrated in Figure 1.

The first key element in the FATE framework is congestion. The corresponding technical objective that illustrates the operator need is to minimize the network congestion. Generally, congestion implies service degradation, thus this objective is well suited for next generation networks that promise to provide quality of service. Moreover, a flow allocation solution with minimal congestion tolerates more dynamic traffic changes, thus the flow reallocation process can be done less frequently. Our approach, formulated in *LexMinCon* (section IV-A), leads to a network with minimum global congestion (i.e. all link utilization are reduced to their minimal values). Particularly, this is achieved in the search for a lexicographical minimal congestion solution.

However, approaches based solely on network congestion minimization will lead to high resource consumption. Roughly speaking, minimizing congestion will spread the load over the network links. Demands will take relatively longer paths in order to distribute the load, thus will consume more bandwidth on links. Therefore, the second key element in our framework is resource consumption and the corresponding technical objective is to minimize the total used bandwidth formulated in *MinRes* (section IV-B).

Considering the first two objectives, FATE enables to easily tune their respective relevancy. Particularly, it enables to approximate the optimal congestion in order to get significant reduction in resource consumption.

Finally, the third key element in our framework is operational costs. When there is more than one LSP between two edge Label Switching Routers (LSRs), it may be necessary to perform complex traffic splitting procedures. Moreover,

signaling and measurement become more complex when the overlay involves a big number of LSPs. Therefore, our third and final technical objective is to minimize the operational costs for a given solution expressed in terms of the number of LSPs in use. Particularly, *ApproxMinLSP* (section IV-C) provides an approximation for the minimal number of LSPs. Note that an optimal solution corresponds to an unsplittable multi-commodity flow and is NP-complete [7]. Moreover, an extension is provided in order to get the minimal number of LSPs in *MinLSP* (section IV-C) via tuning network congestion.

### IV. FRAMEWORK COMPONENTS

In this section, we introduce the mathematical formulation and focus on the interaction between the different framework components. We also give insights on the polynomial time performance of the solution. In the following, we consider an undirected network  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  with  $\mathcal{V}$  being the set vertices and  $\mathcal{E}$  the set of edges.  $\mathcal{V}$  and  $\mathcal{E}$  have a respective cardinality of  $n$  and  $m$ . The capacity for each  $e \in \mathcal{E}$  is denoted by  $c_e$ . Moreover, we are given a set of LSP demands denoted by  $\mathcal{D}$  having a cardinality of  $l$ . Each demand  $d \in \mathcal{D}$  is defined by a triple  $(s_d, t_d, h_d)$ , where  $s_d$  is the source node or ingress LSR,  $t_d$  the destination node or egress LSR and  $h_d$  the demand volume or bandwidth request from  $s_d$  to  $t_d$ . Note that we are using the terms node/vertex/LSR, link/edge interchangeably in this paper.

#### A. Congestion

The first objective in FATE is to minimize the global network congestion. This objective is captured in the *LexMinCon* problem formulated in LP 1. In this formulation,  $In(v)$  (resp.

---

#### LP 1 *LexMinCon*

---

$$\text{Lexicographically Minimize } \mathbb{K} \quad (1)$$

Subject to:

$$\sum_{d \in \mathcal{D}} f_{de} \leq k_e c_e, \quad \forall e \in \mathcal{E} \quad (2)$$

$$\sum_{e \in Out(v)} f_{de} - \sum_{e \in In(v)} f_{de} = \epsilon_{dv}, \quad \forall (d, v) \in \mathcal{D} \times \mathcal{V} \quad (3)$$

$$\epsilon_{dv} = \begin{cases} h_d, & v = s_d \\ -h_d, & v = t_d \\ 0, & v \neq (s_d, t_d) \end{cases}, \quad \forall (d, v) \in \mathcal{D} \times \mathcal{V} \quad (4)$$

$$f_{de} \geq 0, \quad \forall (d, e) \in \mathcal{D} \times \mathcal{E} \quad (5)$$

$$k_e \geq 0, \quad \forall e \in \mathcal{E} \quad (6)$$


---

$Out(v)$ ) denote the set of links going in (resp. out of) node  $v$ ,  $f_{de}$  denotes the flow from demand  $d$  routed on link  $e$  and  $k_e$  the congestion on link  $e$ . In other terms,  $k_e$  denotes the ratio between the total bandwidth used by flows traversing link  $e$  and the capacity of the link. Moreover, we collect the  $k_e$  congestion values ( $\forall e \in \mathcal{E}$ ) and define the congestion vector  $\mathbb{K} = (k_1, k_2, \dots, k_m)$ . Technically, the problem consists of finding a lexicographically minimal congestion vector as

defined in Definition 1 while satisfying capacity constraints in (2) for each link and flow conservation constraints in (3). Note that *LexMinCon* is not a mathematical problem since the objective is not a simple maximization or minimization. However, it can be solved in a polynomial time as a sequence of mathematical problems as detailed in the Appendix.

**Definition 1** A congestion vector  $\mathbb{K} = (k_1, k_2, \dots, k_m)$  with elements sorted in a non-increasing order is said to be lexicographically minimal if, for every other congestion vector  $\mathbb{K}' = (k'_1, k'_2, \dots, k'_m)$  with elements sorted in a non-increasing order, an index  $0 \leq i \leq m$  exists, such that  $k_j = k'_j$  for  $0 \leq j \leq i$  and  $k_l \leq k'_l$  for  $i \leq l \leq m$ .

### B. Resource Consumption

The second technical objective in FATE is to minimize the total used bandwidth. The corresponding mathematical problem, *MinRess*, is formulated in LP 2 as an LP with node-link variables <sup>1</sup>. The size of this LP is polynomial and can be solved in a polynomial time as a multicommodity network flow problem [8]. In this formulation, we introduce a new congestion vector  $\mathbb{K}' = (k'_1, k'_2, \dots, k'_m)$  given by  $\mathbb{K}' = \alpha\mathbb{K}$ , where  $\mathbb{K}$  is the output vector of *LexMinCon*. Hence,  $k'_e = \alpha k_e$  corresponds to the maximum allowed congestion on link  $e$ . The choice of  $\alpha$  determines the interaction between the first two components. It enables to approximate the optimal minimal congestion in order to get significant reduction in resource consumption. Particularly, increasing  $\alpha$  enables more efficient resource consumption at the expense of increased network congestion and vice versa. Finally, *MinRess* provides link flow values for a given congestion state given by  $\mathbb{K}'$ , with minimum bandwidth utilization. These link flow values will be used as an input for the third component that deals with network operational costs.

---

#### LP 2 *MinRess*

---

$$\text{Minimize } \sum_{d \in \mathcal{D}} \sum_{e \in E} f_{de} \quad (7)$$

Subject to:

$$\sum_{d \in \mathcal{D}} f_{de} \leq k'_e c_e, \quad \forall e \in \mathcal{E} \quad (8)$$

$$\sum_{e \in \text{Out}(v)} f_{de} - \sum_{e \in \text{In}(v)} f_{de} = \epsilon_{dv}, \quad \forall d \in \mathcal{D} \quad (9)$$

$$f_{de} \geq 0, \quad \forall (d, e) \in \mathcal{D} \times \mathcal{E} \quad (10)$$

$$\mathbb{K}' = \alpha\mathbb{K}, \text{ where } \mathbb{K} \text{ is the output of } \textit{LexMinCon} \quad (11)$$


---

### C. Operational Costs

The final technical objective in FATE is to minimize the operational costs for a given solution expressed in terms

<sup>1</sup>*MinRess* can be formulated with path-link variables using a pre-calculated set of all possible paths. However, this path set may be exponentially large and, even with column generation techniques, the corresponding LP cannot be solved in a polynomial time.

of the number of LSPs in use. Particularly, our framework provides an approximation for the minimal number of LSPs in *ApproxMinLSP*. Moreover, an optional extension is provided in order to tune the number of LSPs in *MinLSP* via changing the network congestion state.

First we should point out that the output of the *MinRess* component is a set of link flow values that represent a network congestion state with minimum bandwidth utilization. This will not give us insights on the LSPs or the paths that demands are actually allocated to. Therefore, we use a standard flow decomposition algorithm [9] in order to get a set of paths with corresponding bandwidth values that saturate link flow values obtained from the *MinRess* problem. Then, the obtained set of paths denoted  $\mathcal{P}_d$  is used in the *ApproxMinLSP* problem. Note that after flow decomposition, the maximal number of paths per demand is equal to  $m$  (this is a straight conclusion from the flow decomposition algorithm). Technically, *ApproxMinLSP*, introduced in LP 3, is a path-link formulation of the *MinRess* problem. This formulation assigns a variable  $f_{dp}$  for the flow routed on path  $p$  for demand  $d$  and uses a constant denoted by  $\delta_{edp}$  (12) indicating whether path  $p$  of the set  $\mathcal{P}_d$  corresponding to demand  $d$  uses link  $e$ .

$$\delta_{edp} = \begin{cases} 0 & \text{if path } p \in \mathcal{P}_d \text{ uses link } e \\ 1 & \text{elsewhere} \end{cases} \quad (12)$$

With *ApproxMinLSP*, we get the same congestion (given by  $\mathbb{K}' = \alpha\mathbb{K}$ ) and resource consumption values as the output of the *MinRess* problem (since we are using the same set of paths  $\mathcal{P}_d$  as an input), but with a reduced set of paths. The main idea behind reducing the number of used LSPs resides in finding a Basic Feasible Solution (BFS) for the *ApproxMinLSP* problem. Particularly, in a BFS, the number of non-zero variables is smaller than the number of constraints. Therefore, we get:

$$\text{Nb of LSPs} \leq l + m \quad (13)$$

Hence, *ApproxMinLSP* provides a  $(1 + \frac{m}{l})$ -approximation for the single path solution. Recall that an optimal solution corresponds to an unsplittable multi-commodity flow and is proved to be NP-complete [7]. Particularly, we can get a

---

#### LP 3 *ApproxMinLSP*

---

$$\text{Minimize } \sum_{d \in \mathcal{D}} \sum_{p \in \mathcal{P}_d} f_{dp} \quad (14)$$

Subject to:

$$\sum_{d \in \mathcal{D}} \sum_{p \in \mathcal{P}_d} \delta_{edp} f_{dp} \leq k'_e c_e, \quad \forall e \in \mathcal{E} \quad (15)$$

$$\sum_{p \in \mathcal{P}_d} f_{dp} = h_d, \quad \forall d \in \mathcal{D} \quad (16)$$

$$f_{dp} \geq 0, \quad \forall (d, p) \in \mathcal{D} \times \mathcal{P}_d \quad (17)$$


---

BFS for this problem in a polynomial time using for instance Tardos's algorithm [8]. Moreover, according to Definition 2, the output of *ApproxMinLSP* in the FATE framework is an  $(\alpha, 1 + \frac{m}{l})$ -approximation for the flow allocation problem.

**Definition 2** A flow allocation solution is an  $(\lambda, \mu)$  approximation if the corresponding congestion is  $\lambda\mathbb{K}$  with  $\mathbb{K}$  being the optimal minimal congestion vector, and if the solution uses at most  $\mu$  times the number of paths in the optimal solution.

Finally, following the technique used in *ApproxMinLSP*, we introduce *MinLSP*, a final optional component that enables to tune the operational costs. *MinLSP* is formulated in LP 4 as an Mixed Integer LP (MILP) with path-link variables and uses the output path set  $\mathcal{P}'_d$  of *ApproxMinLSP*. This formulation introduces integer variables  $z_{dp}$  that denote whether path  $p$  of the set  $\mathcal{P}'_d$  is used by demand  $d$  i.e.  $z_{dp} = 1$  iff  $f_{dp} > 0$ . The latter expression is depicted by constraint (24) that uses the constant  $H$ , an upper bound on the demand volume ( $h_d < H, \forall d \in \mathcal{D}$ ). Therefore, the objective (21) minimizes the number of LSPs in use. Moreover, *MinLSP* introduces a final congestion vector  $\mathbb{K}''$ , given by (18), that enables to tune congestion in order to get a reduction in the number of LSPs.

$$\mathbb{K}'' = \beta\mathbb{K}' \quad (18)$$

Increasing  $\beta$  enables to reduce the number of LSPs at the expense of increased network congestion and vice versa. Note that  $\mathbb{K}'$  is the congestion vector used in *MinRes* (given by  $\mathbb{K}' = \alpha\mathbb{K}$ ). Technically, in order to explain the functionality of *MinLSP*, we introduce a slacked version of the *MinRes* problem by replacing constraint (22) with:

$$\sum_{d \in \mathcal{D}} \sum_{p \in \mathcal{P}'_d} \delta_{edp} f_{dp} + s''_e = k''_e c_e, \quad \forall e \in \mathcal{E} \quad (19)$$

Following the same reasoning as for *ApproxMinLSP*, we get:

$$\text{Nb of LSPs} + \text{Nb of non-zero } s''_e \leq l + m \quad (20)$$

Hence, increasing the number of non-zero  $s''_e$  will reduce the number of LSPs and is equivalent to increasing  $k''_e$  (or  $\beta$ ) in constraint (22) of the original problem. Finally, *MinLSP* is an MILP with a polynomial number of paths and can be efficiently solved by commercial solvers using branch and bound techniques (though no guarantee for polynomial time complexity is given when using this optional component).

---

#### LP 4 *MinLSP*

---

$$\text{Minimize } \sum_{d \in \mathcal{D}} \sum_{p \in \mathcal{P}'_d} z_{dp} \quad (21)$$

Subject to:

$$\sum_{d \in \mathcal{D}} \sum_{p \in \mathcal{P}'_d} \delta_{edp} f_{dp} \leq k''_e c_e, \quad \forall e \in \mathcal{E} \quad (22)$$

$$\sum_{p \in \mathcal{P}'_d} f_{dp} = h_d, \quad \forall d \in \mathcal{D} \quad (23)$$

$$0 \leq f_{dp} \leq H z_{dp}, \quad \forall (d, p) \in \mathcal{D} \times \mathcal{P}'_d \quad (24)$$

$$z_{dp} \in \{0, 1\}, \quad \forall (d, p) \in \mathcal{D} \times \mathcal{P}'_d \quad (25)$$

$$\mathbb{K}'' = \beta\mathbb{K}', \text{ where } \mathbb{K}' \text{ is the congestion in } \textit{MinRes} \quad (26)$$


---

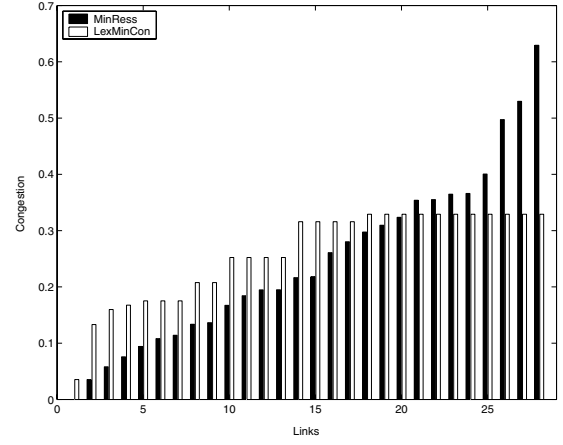


Fig. 2. Congestion state - Single objective approach (KL,  $l = 40$ )

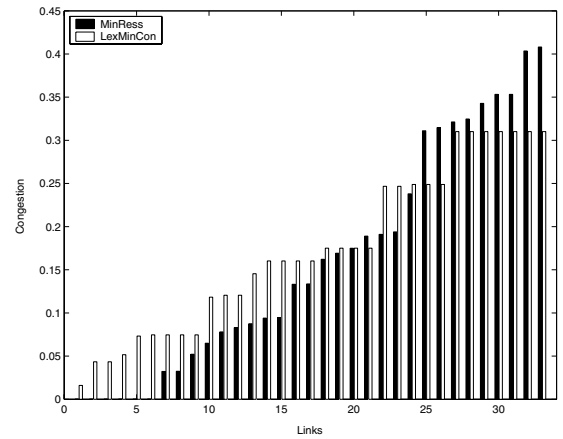


Fig. 3. Congestion state - Single objective approach (MCI,  $l = 30$ )

## V. SIMULATION AND RESULTS

In order to evaluate the performance of our approach, we perform a series of simulations on typical topologies used in the state of the art: the MCI network topology [10] (19 nodes, 33 links), the KL topology (15 nodes, 28 links), the REG topology (14 nodes, 29 links) and the NSF topology (8 nodes, 10 links) [6]. The linear programs were formulated with MATLAB and solved using an external interface (MEX-function) to the lpsolve package [11]. In our simulation scenarios, the traffic demands are uni-directional and the bandwidth for each demand is drawn from a uniform distribution over the range of  $[10, 50]$  bandwidth units. The edge capacities are obtained from a uniform distribution over the range of  $[250, 650]$  bandwidth units.

In our first set of simulations, we show the relevancy of including both congestion and resource minimization components in our framework. Therefore, we run simulations considering each time a single objective optimization chosen between *LexMinCon* and *MinRes*. Figure 2 illustrates the congestion state for the KL network with 40 demands. We can clearly see that *MinRes* induces a highly congested network, while *LexMinCon* has the best performance with minimum congestion over all the network links. This is also valid for other simulation topologies and typical results are illustrated in

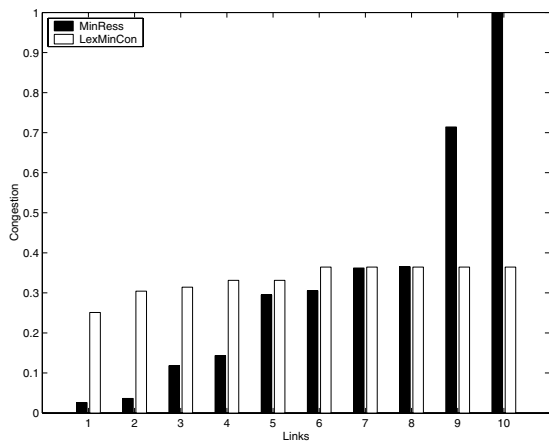


Fig. 4. Congestion state - Single objective approach (NSF,  $l = 25$ )

	MCI $l = 30$	KL $l = 40$	NSF $l = 25$
<i>MinRes</i>	2100	2600	1320
<i>LexMinCon</i>	2400	3000	1590

TABLE I  
BANDWIDTH CONSUMPTION - SINGLE OBJECTIVE APPROACH

Figure 3 and Figure 4. For instance, in the NSF case, *MinRes* leads to a highly congested network, with links approaching the maximal capacity, while *LexMinCon* distributes the load over the network links, thus reduces global congestion. However, if we observe the bandwidth consumption for each of the simulation runs, we can see in Table I that *MinRes* reaches the minimum bandwidth consumption, while *LexMinCon* induces a slightly higher bandwidth consumption. These typical results illustrate the inefficiency of approaches based on a single objective and the need for combining congestion and resource minimization as in the FATE framework.

Hence, our second set of simulations enables to evaluate the combination of the first two components in the FATE framework. Results for bandwidth consumption obtained after executing *LexMinCon* and *MinRes* successively are depicted in Table II. These results show that by increasing the value of  $\alpha$  we can lower the bandwidth consumption. Moreover, we find it useful to compare these results with the performance of *MinRes* used as a single objective (Table III). For instance, considering the REG topology with 60 demands, we get a minimal resource consumption (3303 bandwidth units) by slightly increasing the congestion state ( $\alpha = 1.2$ ).

Our final set of simulations provides insights on the total

$\alpha$	MCI		REG	
	$l = 30$	$l = 50$	$l = 40$	$l = 60$
1	1888	3585	2249	3418
1.2	1765	3404	2138	3303
1.4	1749	3404	2119	3303
1.8	1744	3404	2102	3303

TABLE II  
BANDWIDTH CONSUMPTION - FATE: OUTPUT OF THE *MinRes* COMPONENT

	MCI		REG	
	$l = 30$	$l = 50$	$l = 40$	$l = 60$
<i>MinRes</i>	1744	3404	2102	3303

TABLE III  
BANDWIDTH CONSUMPTION - SINGLE OBJECTIVE *MinRes*

$\alpha$	<i>Min-Res</i>	<i>Approx-MinLSP</i>	<i>MinLSP</i>		
			$\beta = 1$	$\beta = 1.1$	$\beta = 1.2$
1	59	59	59	45	41
1.2	51	45	44	41	41
1.4	47	45	43	41	40
1.8	43	42	40	40	40

TABLE IV  
TOTAL NUMBER OF PATHS - FATE: OUTPUT OF EACH COMPONENT (MCI,  $l = 40$ )

number of paths in our framework. The first two columns of Table IV and Table V illustrate the number of paths obtained at the output of the *MinRes* and the *ApproxMinLSP* components for the MCI and REG topologies (with 40 and 30 demands respectively). We note that when  $\alpha$  increases, the number of paths is reduced. However, there is no significant difference between the two values for *MinRes* and *ApproxMinLSP* since we find a BFS in both cases, thus the number of paths is already reduced for *MinRes*. Moreover, we note that this number is lower than  $l+m$ , validating the theoretical bound. In the last three columns of Table IV and Table V, we evaluate the performance of the optional *MinLSP* component for the same simulation cases. For instance, in the MCI case, we obtain an unsplitable flow allocation solution for  $\beta = 1.2$  and  $\alpha = 1.4$ . Finally, special attention is given to the *MinLSP* solution procedure. In fact, this MILP has a polynomial size (due to the reduced path set) but the optimal solution can still be hard to compute, especially for lower values of  $\alpha$  and  $\beta$ . Therefore, we choose a gap of  $10^{-4}$  as the tolerance of the branch and bound algorithm (this gap is the difference between the best-found solution and the current solution) and an LU decomposition procedure as a basic factorization [11].

## VI. CONCLUSION

In this paper, we present FATE, a novel polynomial time framework for the flow allocation problem in MPLS-TE networks. FATE includes three tunable components that enable to successively reduce congestion, bandwidth consumption and the number of LSPs in use. The solution obtained with FATE is an  $(\alpha, 1 + \frac{m}{l})$ -approximation for the general flow allocation

$\alpha$	<i>Min-Res</i>	<i>Approx-MinLSP</i>	<i>MinLSP</i>		
			$\beta = 1$	$\beta = 1.1$	$\beta = 1.2$
1	46	46	46	35	33
1.2	41	37	34	33	31
1.4	37	34	32	30	30
1.8	34	31	30	30	30

TABLE V  
TOTAL NUMBER OF PATHS - FATE: OUTPUT OF EACH COMPONENT (REG,  $l = 30$ )

problem. Simulation results that are presented in this work provide concrete proof of the importance of each component and insights on the tuning process. Moreover, we show that the flow allocation solution can be obtained in a polynomial time so results can be further extended to larger topologies with a big number of demands. Our future work will study a possible extension for the FATE framework that enables to find efficient flow allocation solutions for survivable flows with end-to-end restoration requirements.

#### APPENDIX

*LexMinCon*, formulated in LP 1, can be solved in a polynomial time as a sequence of mathematical problems following Algorithm 1, introduced in [12]. At each iteration of this

---

#### Algorithm 1 *LexMinCon* algorithm

---

- 1:  $E = \Phi$
  - 2: **while**  $E \neq \mathcal{E}$  **do**
  - 3:   Solve LP 5
  - 4:    $K^* = \min K$
  - 5:   Solve LP 6
  - 6:    $E^* = \{e \in \mathcal{E} - E : s_e = 0\}$
  - 7:    $k_e = K^*, \forall e \in E^*$
  - 8:    $E = E \cup E^*$
  - 9: **end while**
  - 10:  $\mathbb{K} = \{k_e, \forall e \in \mathcal{E}\}$
- 

algorithm,  $E$  captures the subset of  $\mathcal{E}$  including links that has the congestion fixed to the final value (as it will be in the output vector  $\mathbb{K}$ ). First, we solve an elementary problem formulated in LP 5 and get a value of  $K^*$ . Then, we need to

---

#### LP 5 Elementary problem

---

$$\text{Minimize } K \quad (27)$$

Subject to:

$$\sum_{d \in \mathcal{D}} f_{de} \leq K c_e, \quad \forall e \in \mathcal{E} - E \quad (28)$$

$$\sum_{d \in \mathcal{D}} f_{de} = k_e c_e, \quad \forall e \in E \quad (29)$$

$$\sum_{e \in \text{Out}(v)} f_{de} - \sum_{e \in \text{In}(v)} f_{de} = \epsilon_{dv}, \quad \forall (d, v) \in \mathcal{D} \times \mathcal{V} \quad (30)$$

$$f_{de} \geq 0, \quad \forall (d, e) \in \mathcal{D} \times \mathcal{E} \quad (31)$$

$$K \geq 0 \quad (32)$$


---

find the set of binding constraints in this elementary problem; binding constraints are defined as the subset of constraints that verify an equality relation in (28) and correspond to the set of links  $E^*$  given by:

$$E^* = \{e \in \mathcal{E} - E : \sum_{d \in \mathcal{D}} f_{de} = K^* c_e\}. \quad (33)$$

Practically, in order to get  $E^*$ , we solve a slacked version of the problem, formulated in LP 6, using additional slack

variables  $s_e$ . Now the binding constraint set is given by:

$$E^* = \{e \in \mathcal{E} - E : s_e = 0\}. \quad (34)$$

Therefore, for all the links in  $E^*$ ,  $k_e$  takes a final congestion value of  $K^*$  and these links are added to the set  $E$ . After iterating over the set  $E$ , we finally get  $\mathbb{K}$ , the lexicographically minimal vector with all its elements given by the final  $k_e$  values.

---

#### LP 6 Slacked problem

---

$$\text{Maximize } \sum_e s_e \quad (35)$$

Subject to:

$$\sum_{d \in \mathcal{D}} f_{de} + s_e \leq K^* c_e, \quad \forall e \in \mathcal{E} - E \quad (36)$$

$$\sum_{d \in \mathcal{D}} f_{de} = k_e c_e, \quad \forall e \in E \quad (37)$$

$$\sum_{e \in \text{Out}(v)} f_{de} - \sum_{e \in \text{In}(v)} f_{de} = \epsilon_{dv}, \quad \forall (d, v) \in \mathcal{D} \times \mathcal{V} \quad (38)$$

$$f_{de} \geq 0, \quad \forall (d, e) \in \mathcal{D} \times \mathcal{E} \quad (39)$$

$$s_e \geq 0, \quad \forall e \in \mathcal{E} - E \quad (40)$$


---

The polynomiality of *LexMinCon* follows from the fact that Algorithm 1 computes the optimal lexicographically minimum load vector in at most  $(2m - 1)$  LP computations and each of them can be solved in a polynomial time [8]. Note that more sophisticated approaches can be used for this problem, as it is strictly related to the max-min fairness optimization in [1].

#### REFERENCES

- [1] M. Pióro and D. Medhi, *Routing, Flow and Capacity Design in Communication and Computer Networks*. Morgan Kaufmann Publishers (Elsevier), June 2004.
- [2] S. C. Erbas and C. Erbas, "A multiobjective off-line routing model for mpls networks," in *Proc. of the 18th International Teletraffic Congress*, Aug.-Sep. 2003.
- [3] S. Shekhar Srivastava and D. Medhi, "Network traffic engineering with varied levels of protection in the next generation internet," *Performance Evaluations and Planning Methods for the Next Generation Internet*, February 2005.
- [4] Y. Lee *et al.*, "A constrained multipath traffic engineering scheme for mpls networks," in *ICC*, April 2002.
- [5] C. T. Chou, "Traffic engineering for mpls-based virtual private networks," *Computer Networks*, vol. 44, no. 3, pp. 319–333, February 2004.
- [6] V. S. Mirrokni *et al.*, "A simple polynomial time framework for reduced-path decomposition in multi-path routing," in *IEEE INFOCOM*, 2004.
- [7] A. Chakrabarti *et al.*, "Approximation algorithms for the unsplittable flow problem," in *5th International Workshop on Approximation Algorithms for Combinatorial Optimization*, 2002.
- [8] E. Tardos, "A strongly polynomial algorithm to solve combinatorial linear programs," *Operations Research*, vol. 34, 1986.
- [9] T. Ahuja, R. Magnanti and J. B. Orlin, *Network flows: theory, algorithms, and applications*. Upper Saddle River, NJ: Prentice-Hall, Inc., 1993.
- [10] S. Nelakuditi and Z. Zhang, "On selection of paths for multi-path routing," in *IWQoS*, June 2001.
- [11] K. Berkelaar, M. Eikland and P. Notebaert, "Open source (mixed-integer) linear programming system," Version 5.1.0.0 - GNU LGPL (Lesser General Public Licence), May 2004.
- [12] D. Nace and J. B. Orlin, "Lexicographically minimum and maximum load linear programming problems," <http://web.mit.edu/jorlin/www/papersfolder/LexMinLoad.pdf>, August 2004.