More adaptive robust stable routing

Mateusz Żotkiewicz
TELECOM & Management SudParis
9, rue Charles Fourier, 91011 Évry, France
Institute of Telecommunications, Warsaw University of Technology
Nowowiejska 15/19, 00-665 Warszawa, Poland
Email: mateusz.zotkiewicz@it-sudparis.eu

Walid Ben-Ameur
TELECOM & Management SudParis
9, rue Charles Fourier, 91011 Êvry, France
Email: walid.benameur@it-sudparis.eu

Abstract—In the paper we deal with the problem of optimal partitioning of a traffic demand polytope using a hyperplane. In the considered model all possible demand matrices belong to a polytope. The polytope can be divided into parts, and different routing schemes can be considered while dealing with traffic matrices from different parts of the polytope. The model can be applied to all networks that support unrestricted routing of bifurcated flows, e.g., MPLS networks or optical networks. In the paper we present an algorithm that solves one of the most practical versions of the considered problem, i.e., reservation vectors on both sides of the hyperplane have to be the same. Moreover, we present another (faster) algorithm that solves a more restricted version of the problem. Finally, we present numerical results proving the applicability of the introduced algorithms.

I. INTRODUCTION

Modern telecommunication networks deal with traffic generated by variety of different applications utilized by a large numbers of users. It makes prediction of traffic patterns increasingly difficult. The introduction of new services and the mobility of customers make the task harder.

Some models have been proposed in the past to address the uncertainty. The first approach consists in building a traffic matrix based on the worst case situation for each traffic component. Routing is then computed based on it. While this approach is simple, it can provide expensive solutions.

A second approach is based on probability information. Probability models are chosen to model the traffic variations. Then, one may look for a routing optimizing a probabilistic criterium: throughput expectation, mean delays, blocking probability, etc. The solution obtained in this way is good on average but can be very bad in some cases. This kind of approach is generally called stochastic programming.

Another general approach called robust optimization (see [5]) is based on a finite number of scenarios. It consists in computing a solution that is compatible with all scenarios. In the networking context, we want to determine a routing scheme such that each traffic matrix belonging to a given finite set of traffic matrices can be carried through the network. Robust optimization can be viewed as an alternative, and more tractable, way to solve the stochastic programming problems.

Another robustness model was proposed in [9] (and independently in [8]). It assumes that the traffic generated in each node is limited. Limitations on the incoming traffic can also be considered. The traffic matrix is then any matrix satisfying this kind of constraints. This model is called Hose model.

A more general polyhedral model is considered in [2]–[4]. The model assumes that each possible traffic matrix belongs to a polytope. In [4] a polynomial time algorithm is proposed to compute a routing scheme that solves this problem. The routing is robust (it is compatible with all matrices) and stable (it does not change when the matrix changes).

A completely different approach to deal with uncertainty consists in allowing the network to change routing in a dynamic way each time there is a change in terms of traffic matrix. This problem was intensively studied in the context of switched networks. Different rules can be applied to connect calls depending on the current situation, e.g., alternate dynamic routing, sequential routing, trunk reservation (see [1] and references therein). While this routing has many benefits, it is generally difficult to implement. Moreover, it is not optimal, because the rules that are used are fixed in advance.

For more detailed discussion of different approaches to the problem of uncertainty proposed in the past readers can consult a recent survey that can be found in [6].

In [4] the following question was raised: given a traffic polytope, is it theoretically easy to compute a fully dynamic routing (routing scheme that depends on the current traffic matrix only). Recently it was proven in [7] that this problem is difficult. Moreover, it is clearly not easy to implement this kind of routing.

Therefore, it is worth to consider something between robust stable routing and fully dynamic routing. Instead of computing a robust solution, it is possible to partition the uncertainty set into some subsets, and compute a robust routing for each of them. This class of problems was considered in [2]. In this paper we present algorithms that solve problems introduced there. Moreover, we present numerical results proving the applicability of our algorithms.

We deal with a class of problems where it is assumed that a direction of a hyperplane dividing the uncertainty set is known but its position is subject to optimization. We make this assumption because some hyperplane’s directions can be implemented easier than others, e.g., an amount of traffic...
generated between a pair of nodes or an amount of traffic generated by a specific node. In both situations a position of a threshold that triggers changes in routing is subject to optimization. In other words, we provide two routings: the first is used when the amount of traffic, which was set as a direction of the hyperplane, is smaller than the threshold, otherwise the second routing is used. Note that when we restrict possible directions to a finite (and tractable) set we can use our algorithm to obtain an optimal solution in reasonable time (by calling it once for each possible direction).

Another possibility is to consider time as a direction of the hyperplane. In this case time is represented by an additional dimension of the uncertainty set. Our goal is to optimize the moment when routing should be changed.

The paper is organized as follows. In Section II we present the notation and formulations of two basic problems. In Section III the presented models are enhanced by introducing the partitioning of the uncertainty set. In Section IV we present an algorithm that solves the introduced problems. Another (faster) algorithm is presented in Section V. Numerical results can be found in Section VI and conclusions in Section VII.

II. NOTATION AND PROBLEM FORMULATIONS

We consider a directed graph \( G = (V, A) \), where \( V \) is the set of nodes and \( A \) is the set of arcs. The graph represents a backbone and the arcs depict the unidirectional transmission links. For each arc \( a \in A \) an installed capacity \( c_a \in \mathbb{R}_+ \) and a routing cost \( w_a \in \mathbb{R}_+ \) for one unit of traffic are given.

Let \( t = (t_{ij})_{i,j \in V} \) be a vector of \( \mathbb{R}^{|V|(|V|-1)} \) that specifies traffic demands (or capacity requirements) between all pairs of nodes of \( V \). This vector will be called a traffic matrix. The traffic matrix is supposed to be variable and can be any point of the traffic demand polytope \( D \). \( D \) is generally defined by some linear constraints involving the variables \( t_{ij} \), for \( i, j \in V \). However, it can be also defined by a set of traffic matrices (convex hull of those matrices) or by an oracle.

To express routing problems as mathematical programs, we introduce the following notation.

- \( \mathcal{P}(i,j) \): finite set of acyclic paths of \( G \) from \( i \) to \( j \) (\( i, j \in V \)).
- \( x_{ij}^p \): proportion of the traffic demand from \( i \) to \( j \) (\( i, j \in V \)) carried through a path \( p \in \mathcal{P}(i,j) \). Note that \( 0 \leq x_{ij}^p \leq 1 \). For a current traffic matrix \( t \in D \) the traffic carried through \( p \) is then given by \( t_{ij} \cdot x_{ij}^p \).
- \( x_a^i \): proportion of traffic from \( i \) to \( j \) flowing through an arc \( a \in A \).
- \( f_a \): maximum amount of traffic carried on an arc \( a \in A \). It depends on the polytope \( D \) and the routing pattern. It can be considered as the minimum capacity that has to be reserved on a link \( a \).
- \( w(D) \): routing cost.

A. Robust-Routing-Problem

The problem of computing the minimum cost robust stable routing of an uncertainty domain \( D \), denoted by Robust-Routing-Problem, can be formulated as follows:

Minimize:

\[
w^{RR}(D) = \sum_{a \in A} w_a f_a
\]

Subject to:

\[
\sum_{p \in \mathcal{P}(i,j)} x_{ij}^p \geq 1 \quad \forall i, j \in V, \quad (1a)
\]

\[
\sum_{p \in \mathcal{P}(i,j), p \geq a} x_{ij}^p \leq x_a^i \quad \forall i, j \in V, \quad \forall a \in A, \quad (1b)
\]

\[
\sum_{i,j \in V} x_a^{ij} t_{ij} \leq f_a \quad \forall a \in A, \quad \forall t \in D, \quad (1c)
\]

\[
f_a \leq c_a \quad \forall a \in A, \quad (1d)
\]

\[
x_{ij}^p \geq 0 \quad \forall p \in \mathcal{P}(i,j), \quad \forall i, j \in V, \quad (1e)
\]

\[
x_a^i \geq 0 \quad \forall a \in A, \quad \forall i, j \in V, \quad (1f)
\]

The objective is to minimize the total routing cost. Inequalities (1a) express the fact that the traffic demand between every pair of nodes may be split among many paths. Every variable \( x_{ij}^p \) is defined by (1b). For a given matrix \( t \) the traffic flowing through a link \( a \) is given by the left-hand side of (1c). Thus, the capacity \( f_a \) that should be reserved on an arc \( a \) must be higher than the traffic carried in all the situations. In other words, inequalities (1c) must be valid for each traffic matrix \( t \in D \). Inequality (1d) indicates that \( f_a \) is lower than the capacity of a link \( a \). Inequalities (1e) and (1f) force the variables to be nonnegative. Note that the variables \( x_a^i \) can be eliminated from the formulation. However, we keep them for the sake of clearness and simplicity.

Presented in this way, Robust-Routing-Problem seems to be difficult. First, the number of paths of \( \mathcal{P}(i,j) \) can be very high. Second, inequalities (1c) are nonlinear. Finally, they have to be satisfied for each \( t \in D \), and \( D \) is generally an infinite set.

Fortunately, the problem was proven to be easily solvable using an algorithm based on constraint generation (see [3], [4]). Considering the current solution of the relaxation of Robust-Routing-Problem (only a finite set of traffic matrices is taken into account instead of the whole polytope \( D \)), we only have to check whether there is an arc \( a \) and a traffic matrix \( t \in D \) such that (1d) is violated. This can be easily done by the following linear program:

\[
\max_{t \in D} \sum_{i \in V} \sum_{j \in V \setminus \{i\}} x_{ij}^p t_{ij}.
\]

If the maximum is larger than \( f_a \), we add a violated inequality.

Routing paths can also be generated in an iterative way by solving shortest path problems where arc weights are given by the values of the dual variables of (1b).

B. No-Sharing-Problem

Another set of problems emerges when we impose harsher rules on capacities’ reservation and forbid different relations to share resources. We denote the core problem of this set as No-Sharing-Problem and formulate it as follows:
Minimize:
\[ w^{NS}(D) = \sum_{a \in A} w_a f_a \]

Subject to:
\[ \sum_{p \in P(i,j)} x_{ij}^{ij} \geq 1 \quad \forall i, j \in \mathcal{V}, \quad (2a) \]
\[ \sum_{p \in P(i,j), p \in a} x_{ij}^{ij} \leq x_a^{ij} \quad \forall i, j \in \mathcal{V}, \quad (2b) \]
\[ x_a^{ij} t_{ij} \leq f_a^{ij} \quad \forall i, j \in \mathcal{V}, \quad (2c) \]
\[ \sum_{i, j \in \mathcal{V}} x_a^{ij} \leq f_a \quad \forall a \in \mathcal{A}, \quad (2d) \]
\[ f_a \leq c_a \quad \forall a \in \mathcal{A}, \quad (2e) \]
\[ x_a^{ij} \geq 0 \quad \forall a \in \mathcal{A}, \quad (2f) \]
\[ x_a^{ij} \geq 0 \quad \forall a \in \mathcal{A}, \quad (2g) \]

In the formulation variables \( f_a^{ij} \) denote the amount of capacity reserved on a link \( a \) by traffic generated between nodes \( i \) and \( j \). In fact, they [like \( x_a^{ij} \) — here and in (1)] are obsolete and can be eliminated from the formulation. However, we keep them for the sake of clarity and simplicity.

Note that the basic No-Sharing-Problem is equivalent to reserving \( t_a^{ij} = \max_{t \in D} t_{ij} \) of flow for each relation. Therefore, it is easily solvable by means of an appropriate linear program facilitated by a path generation mechanism. However, it gets more complicated when we allow \( D \) to be partitioned.

III. PARTITIONING PROBLEMS

Given a traffic demand polytope \( D \) and a normal vector \( \alpha \) that defines a direction of a hyperplane \( \alpha x = \beta \) (\( \beta \) is a variable subject to optimization) we can partition \( D \) into two subsets \( L(D, \beta) = D \cap \{ t, \alpha t \leq \beta \} \) and \( R(D, \beta) = D \cap \{ t, \alpha t \geq \beta \} \), called a left hand side polytope (LHSP) and a right hand side polytope (RHSP), respectively. Then, we can consider a robust routing for each subset. Said another way, instead of having only one routing scheme, we will have two schemes: if the current traffic matrix belongs to the first (second) subset it uses the first (second) routing scheme.

Depending on restrictions imposed on the reservation vectors two different strategies can be considered. The reservation vectors on opposite sides of the hyperplane can be either allowed to be different or required to be identical. The former case models a situation when we are the owners of the network and our task is to minimize the usage of resources. The latter corresponds to a situation when we rent capacities from other operators and we cannot renegotiate those renting agreements each time we change the routing.

A. The reservation vectors can be different

We denote the problem by Different-Reservation-Problem, and for Robust-Routing-Problem we formulate it as follows:
\[ w^{DR/RR}(D) = \min_{\beta} \{ w^{RR}(L(D, \beta)), w^{RR}(R(D, \beta)) \} . \]

This version can be approximately solved within an arbitrary precision using a polynomial algorithm presented in [2]. Knowing that \( w^{RR}(L(D, \beta)) \) is a nondecreasing function of \( \beta \) and \( w^{RR}(R(D, \beta)) \) is a nonincreasing function of \( \beta \) we can find \( \beta^* \) such that \( w^{RR}(L(D, \beta^*)) = w^{RR}(R(D, \beta^*)) \) using a binary search.

In order to formulate Different-Reservation-Problem for No-Sharing-Problem we have to substitute \( w^{NS} \) for \( w^{RR} \). Note that this variant can be solved using the same algorithm.

B. The reservation vectors are identical

Here we assume that the reservation vectors corresponding to the two uncertainty subsets should be the same. We will call it Identical-Reservation-Problem (for Robust-Routing-Problem or No-Sharing-Problem). A version that corresponds to Robust-Routing-Problem can be formulated as follows:

Minimize:
\[ w^{IR/RR}(D) = \sum_{a \in A} w_a f_a \]

Subject to:
\[ \sum_{p \in P(i,j)} x_{ij}^{ij} \geq 1 \quad \forall i, j \in \mathcal{V}, \quad (3a) \]
\[ \sum_{p \in P(i,j), p \in a} x_{ij}^{ij} \leq x_a^{ij} \quad \forall i, j \in \mathcal{V}, \quad (3b) \]
\[ x_a^{ij} t_{ij} \leq f_a^{ij} \quad \forall i, j \in \mathcal{V}, \quad (3c) \]
\[ \sum_{i, j \in \mathcal{V}} x_a^{ij} \leq f_a \quad \forall a \in \mathcal{A}, \quad (3d) \]
\[ f_a \leq c_a \quad \forall a \in \mathcal{A}, \quad (3e) \]
\[ x_a^{ij} \geq 0 \quad \forall a \in \mathcal{A}, \quad (3f) \]
\[ x_a^{ij} \geq 0 \quad \forall a \in \mathcal{A}, \quad (3g) \]

In the formulation non-overlined variables correspond to LHSP, and overlined variables correspond to RHSP.

While dealing with a version of Identical-Reservation-Problem for No-Sharing-Problem we have to replace (3c) and (3f) with:
\[ x_a^{ij} t_{ij} \leq f_a^{ij} \quad \forall i, j \in \mathcal{V}, \quad (4a) \]
\[ x_a^{ij} \leq f_a^{ij} \quad \forall i, j \in \mathcal{V}, \quad (4b) \]

and add the following two sets of constraints:
\[ \sum_{i, j \in \mathcal{V}} f_a^{ij} \leq f_a \quad \forall a \in \mathcal{A}, \quad (5a) \]
\[ \sum_{i, j \in \mathcal{V}} f_a^{ij} \leq f_a \quad \forall a \in \mathcal{A} \]
The way this family of problems (for both Robust-Routing-Problem and No-Sharing-Problem) can be approached is the major novelty of our paper.

IV. DOUBLE BINARY SEARCH ALGORITHM

In this section we present an algorithm that solves Identical-Reservation-Problem. Although it can successfully deal with both versions of the problem (for Robust-Routing-Problem and No-Sharing-Problem), for the sake of simplicity we will concentrate on the former one.

Consider a simpler problem that consists in answering a question if there exists a feasible solution to Identical-Reservation-Problem whose cost is smaller or equal to a given value. We will refer to this simplified problem as Identical-Reservation-Limit. Obviously, knowing the way of solving Identical-Reservation-Limit we can solve Identical-Reservation-Problem relatively fast using a binary search.

First we have to know upper and lower bounds for the cost \( w^\text{L/RHSP}(D, \beta) \) denoted by \( w_{\text{min}} \) and \( w_{\text{max}} \), respectively. In our implementation we set \( w_{\text{min}} \) to the optimal objective of Different-Reservation-Problem, and \( w_{\text{max}} \) to the optimal objective of Identical-Reservation-Problem with \( \beta \) set to the optimal value obtained while computing the lower bound (Identical-Reservation-Problem for fixed \( \beta \) was solved in [2]).

The question arises how to solve Identical-Reservation-Limit. In this section we present an algorithm that can do this for both Robust-Routing-Problem and No-Sharing-Problem. Although it is not polynomial, it performs well and can solve even medium size problems (see the numerical results in Section VI). The key ideas of the approach can be seen in Algorithm 1.

Algorithm 1 Identical-Reservation-Limit(\( w_{\text{lim}} \))

\[
\beta_2 = \beta_{\text{max}} \\
\text{while } \beta_2 > \beta_{\text{min}} \text{ do} \\
\quad \beta_1 = \text{findMax}(\beta_2, w_{\text{lim}}) \\
\quad \text{if } \beta_1 = \beta_2 \text{ then} \\
\quad \quad \text{return YES} \\
\quad \text{else} \\
\quad \quad \beta_2 = \beta_1 \\
\text{end if} \\
\text{end while} \\
\text{return NO}
\]

The algorithm returns YES, if a solution to Identical-Reservation-Problem, whose cost is smaller or equal to \( w_{\text{lim}} \), exists. Otherwise, it returns NO. It uses constants \( \beta_{\text{min}} \) and \( \beta_{\text{max}} \) that define an interval of possible values of \( \beta \). It also uses the function \text{findMax}. The basic idea of the method is to limit the interval of possible positions of the hyperplane. At the begin the hyperplane can be anywhere in \([\beta_{\text{min}}, \beta_{\text{max}}]\). It means that we cannot judge if any \( t \in D \) has to belong either to LHSP or to RHSP (we assume that the interval is sufficiently large). Then, in the loop, we limit the interval of possible positions of the hyperplane. More precisely, we extend the number of matrices that have to belong to RHSP.

Having RHSP defined for a particular loop we can, using \text{findMax} method, calculate the maximal size of LHSP (RHSP is given) which satisfies the cost limit. If the whole polytope \( D \) can be divided between LHSP and RHSP the problem has been solved. If some matrices cannot be added to LHSP for a given RHSP, it means that they have to belong to RHSP. We add them to RHSP and repeat the loop.

Let us now introduce a new problems that will facilitate explanation of \text{findMax} method. Assume that values \( \beta_1 \) and \( \beta_2 \) are given. Two-Known-Planes-Problem consists in finding the optimal routing when LHSP is defined as \( L(D, \beta_1) \) and RHSP is defined as \( R(D, \beta_2) \). Note that \( \beta_1 \) and \( \beta_2 \) are given so the problem can be easily solved using techniques presented in [3], [4] and briefly described in Section II.

Method \text{findMax} returns the maximal value of \( \beta_1 \) that allows Two-Known-Planes-Problem to be solved for a given \( \beta_2 \) and within a given cost limit \( w_{\text{lim}} \). If such a value does not exist it returns \(-\infty\). Note that \( \beta_1 \) is a variable subject to optimization and \( \beta_2 \) is constant. In such a situation the objective function of Two-Known-Planes-Problem is nondecreasing for \( \beta_1 \) as we cannot decrease a cost of a solution by adding new traffic matrices to the problem. Therefore, the method can use (and it does) a binary search.

V. LIMITED NUMBER OF EXTREME POINTS

In Section IV we presented an algorithm that solves Identical-Reservation-Problem for both Robust-Routing-Problem and No-Sharing-Problem. In this section we take advantage of a special feature of No-Sharing-Problem, and present an algorithm that can solve this version of Identical-Reservation-Problem in polynomial time when \( D \) is a convex hull of a given set of traffic matrices.

Assume that the polytope \( D \) and hyperplanes \( \alpha.t = \beta_1 \) and \( \alpha.t = \beta_2 \) \((\beta_1 < \beta_2)\) are given, and the middle polytope \( D_{\text{mid}} = R(D, \beta_1) \cap L(D, \beta_2) \) is not empty but simultaneously does not contain any extreme points of \( D \). We define the left and right polytopes as \( D_{\text{left}} = L(D, \beta_1) \) and \( D_{\text{right}} = R(D, \beta_2) \), respectively. We use the following notation: \( D_{\beta} = D \cap \{t | \alpha.t = \beta\} \).

Lemma 5.1: For any \( \beta \in (\beta_1, \beta_2) \) all points \( t \in D_{\beta} \) can be expressed as \( t = t' + (1-\lambda)t'' \), where \( \lambda = \frac{\beta_2 - \beta}{\beta_2 - \beta_1} \), \( t' \in D_{\beta_1} \), and \( t'' \in D_{\beta_2} \).

Proof: Take any point \( t \in D_{\beta} \). As it belongs to \( D \), it can be expressed as a convex combination of extreme points of \( D \). \( D_{\text{mid}} \) does not contain any extreme points of \( D \), so all of them have to belong either to \( D_{\text{left}} \) or to \( D_{\text{right}} \). We define the left and right polytopes as \( D_{\text{left}} = L(D, \beta_1) \) and \( D_{\text{right}} = R(D, \beta_2) \). It means that we can calculate a convex combination of only those extreme points that belong to \( D_{\text{left}} \) and obtain a point in \( D_{\text{left}} \) (together with its weight). The same operation can be performed on \( D_{\text{right}} \). The considered matrix \( t \) will be a convex combination of those two points. Obviously, those two points can be connected by a line that is contained in \( D \). What is more, this line also contains one point from \( D_{\beta_1} \) (denoted by \( t_{\beta_1} \)) and one point from \( D_{\beta_2} \) (denoted by \( t_{\beta_2} \)). Certainly, \( t = t_{\beta_1} + (1-\lambda)t_{\beta_2} \), and we can take \( t' = t_{\beta_1} \) and \( t'' = t_{\beta_2} \).
When we think about $\beta_2$ as a variable we can write the following lemma.

**Lemma 5.2:** For any $\beta \in (\beta_1, \beta_2)$ all points $t \in D_{mid}$ \( \cap \{t, \alpha.t \leq \beta\} \) can be expressed as $t = \lambda t' + (1 - \lambda)t''$, where $\lambda = \frac{\beta - \beta_1}{\beta_2 - \beta_1}$, $t' \in D_{\beta_1}$ and $t'' \in D_{mid}$ \( \cap \{t, \alpha.t \leq \beta_2\} \).

**Proof:** Take any point $t \in D_{mid}$ \( \cap \{t, \alpha.t \leq \beta\}$. Note that $t \in D_{\beta}$, where $\beta_1 \leq \beta \leq \beta_2$. Now let us take $\beta' = \beta_1 + \frac{\beta - \beta_1}{\beta_2 - \beta_1}(\beta_2 - \beta_1)$. Obviously, $\beta_1 \leq \beta' \leq \beta_2$, so there are no extreme points of $D$ between $\beta$ and $\beta'$. What is more, $\frac{\beta - \beta_1}{\beta_2 - \beta_1}$ is a traffic demand polytope.

Let us now assume that the optimal solutions of\( D, \beta_1 \) and $t' \in L(D, \beta_1)$ and $t'' \in L(D, \beta_2)$.

Obviously, it works also in the opposite direction.

**Corollary 5.4:** For any $\beta \in (\beta_1, \beta_2)$ all points $t \in D_{(\beta, \beta_2)}$ can be expressed as $t = \lambda t' + (1 - \lambda)t''$, where $\lambda = \frac{\beta - \beta_1}{\beta_2 - \beta_1}$, $t' \in R(D, \beta)$ and $t'' \in R(D, \beta_2)$.

Both Corollaries 5.3 and 5.4 can be rewritten taking into account $t_{ij}^{max}$. By $t_{ij}^{max}(A)$ we denote $\max_{a \in A} t_{ij}$, where $A$ is a traffic demand polytope.

**Corollary 5.5:** For any $\beta \in (\beta_1, \beta_2)$ maximal values of $t_{ij}^{max} = t_{ij}^{max}(L(D, \beta_1))$ satisfies an inequality:

\[
\lambda^{t_{ij}^{max}} \leq \lambda^{t_{ij}^{max}} + (1 - \lambda)^{t_{ij}^{max}},
\]

where $\lambda = \frac{\beta_2 - \beta}{\beta_2 - \beta_1}$, $t_{ij}^{max} = t_{ij}^{max}(L(D, \beta))$ and $t_{ij}^{max} = t_{ij}(R(D, \beta_2))$. Moreover, maximal values of $t_{ij}^{max}$ for $t_{ij}^{max}(R(D, \beta))$ satisfies an inequality:

\[
\lambda^{t_{ij}^{max}} \leq \lambda^{t_{ij}^{max}} + (1 - \lambda)^{t_{ij}^{max}},
\]

where $\lambda = \frac{\beta_2 - \beta}{\beta_2 - \beta_1}$, $t_{ij}^{max} = t_{ij}^{max}(R(D, \beta))$ and $t_{ij}^{max} = t_{ij}(R(D, \beta_2))$.

Let us now assume that the optimal solutions of Identical-Reservation-Problem (for No-Sharing-Problem) for given $\beta_1$ and $\beta_2$ are provided, and we want to construct a solution for $\beta \in (\beta_1, \beta_2)$. Variables of the solution for $\beta$ will be denoted as described in Section II. Variables of the solution for $\beta_1$ will be additionally followed by a prim, e.g., $f^{\alpha}_{a'}$. Variables of the solution for $\beta_2$ will be followed by a double prim, e.g., $f^{\alpha}_{a''}$.

Let us now construct routing to the solution in the following way:

\[
x_{ij}^{p} = (\lambda x_{ij}^{p}\frac{t_{ij}^{mid}}{t_{ij}^{max}} + (1 - \lambda)x_{ij}^{p}\frac{t_{ij}^{mid}}{t_{ij}^{max}}) / t_{ij}^{max}
\]

\[
x_{ij}^{p} = (\lambda x_{ij}^{p}\frac{t_{ij}^{mid}}{t_{ij}^{max}} + (1 - \lambda)x_{ij}^{p}\frac{t_{ij}^{mid}}{t_{ij}^{max}}) / t_{ij}^{max}
\]

**Lemma 5.6:** The routing constructed using (6) is feasible, i.e., satisfies (3a) and (3d).

**Proof:** Let us focus on the first equality. We use the first inequality from Corollary 5.5, substitute $u$ for $\lambda t_{ij}^{max}$ and substitute $v$ for $(1 - \lambda)t_{ij}^{max}$. Summing the obtained inequalities for all $p \in P(i,j)$ we receive the following formula:

\[
\sum_{p \in P(i,j)} x_{ij}^{p} \geq \frac{u}{u + v} \sum_{p \in P(i,j)} x_{ij}^{p} + \frac{v}{u + v} \sum_{p \in P(i,j)} x_{ij}^{p}.
\]

Variables $x_{ij}^{p}$ and $x_{ij}^{p}$ satisfy (3a), so $x_{ij}^{p}$ also has to satisfy it. Similar proof can be used to show that $x_{ij}^{p}$ satisfy (3d).

Knowing that the routing expressed by (6) is feasible, and using (4) in order to substitute $f^{a}_{a'}$ we obtain the following corollary.

**Corollary 5.7:** If routing of the solution for $\beta$ satisfies (6) then $f^{a}_{a'} \leq \lambda f^{a}_{a'} + (1 - \lambda)f^{a}_{a''}$ and $f^{a}_{a''} \leq \lambda f^{a}_{a'} + (1 - \lambda)f^{a}_{a''}$, for all $a \in A$ and all $i, j \in V$.

Note that according to (3) (modified for No-Sharing-Problem) cost function $w^{IR/NS}(D, \beta)$ depends on $f_{a}$ only. What is more, according to (5) $f_{a}$ depends on $f_{a}^{i,j}$ only. Therefore, from Corollary 5.7 we can deduce that:

**Corollary 5.8:** Function $w^{IR/NS}(D, \beta)$ is convex for $\beta \in [\beta_1, \beta_2]$, if $D_{mid} = R(D, \beta_1) \cap L(D, \beta_2)$ does not contain any extreme point of $D$.

Let us now return to the algorithm. We can easily calculate a list of increasing values of $\beta$ that describe all hyperplanes which contain extreme points of $D$ (note that the polytope is a convex hull of a given set of demand matrices). We can formally define the list as $B = \{\beta_1, \beta_2, \ldots, \beta_n\}$ such that $\beta_i < \beta_{i+1}$, for $i = 1, 2, \ldots, n - 1$, and each extreme point of $D$ belongs to $\cup_{i=1}^{n} D_{\beta_i}$. Knowing from Corollary 5.8 that function $w^{IR/NS}(D, \beta)$ is convex for $\beta \in [\beta_i, \beta_{i+1}]$ and $i = 1, 2, \ldots, n - 1$, we can calculate optimal $w^{IR/NS}(D, \beta)$, for $i = 1, 2, \ldots, n - 1$, using any known polynomial time algorithm that can find the minimum of a convex function (e.g., golden ratio or Fibonacci search). Obviously, optimal $w^{IR/NS}(D) = \min_{i=1,2,\ldots,n-1} w^{IR/NS}(D, \beta_i)$.

**Proposition 5.9:** Identical-Reservation-Problem for No-Sharing-Problem can be solved in polynomial time, if $D$ is defined as a convex hull of a given set of traffic matrices.

**Proof:** The problem can be solved using the presented algorithm. The algorithm is polynomial as it invokes a polynomial method finding the minimum of a convex function. The method is invoked a polynomial number of times.

**VI. NUMERICAL RESULTS**

We tested our algorithms on an Intel 2.4 GHz CPU with 3.25 GB RAM, using a linear programming solver CPLEX 11.0 [10]. We built our example cases using real world networks available in SNDlib [11]. We used altanta and france topologies. Both were tested using two different sets of active nodes (we will refer to them as VLANS) and three different traffic demand polytopes defined for each of the VLANS. The first polytope for each VLAN satisfied restrictions of Hose model presented in [8], i.e., traffic can originate and terminate in active nodes only and both outgoing and incoming traffic are limited by given bounds. Another two traffic demand polytopes were convex hulls of a number (3 or 10) of extreme points of the first polytope.
Our test cases represent congested network configurations and our task is to minimize additional costs incurred by the congestion. Therefore, we introduced additional uncapacitated links between all pairs of nodes.Routing costs of these additional links were set to 1, while routing costs of links from the original network were set to 0. We can consider those additional links as possibilities to rent capacity from other operators. Each Hose model polytope was scaled in such a way that the minimal congestion on the most heavily utilized link for the polytope built on its 3 extreme points was 1.1. For each topology and each polytope we used up to five different directions $\alpha$ of the hyperplane.

In Table 1 we present networks used in our experiments, possible gains when different strategies of dividing a traffic demand polytope are implemented (we present results obtained for the best case out of 10 randomly generated test cases), and running times of the algorithms that are capable of solving the considered scenarios. The first five columns describe networks used in the simulations: network – name of the topology in SNDlib, $|V|$ – number of nodes, $|A|$ – number of directed links, $\dim D$ – number of $t_{ij} > 0$ (dimension of a traffic demand polytope), variant – way traffic demand polytopes were created (either a full Hose model polytope, or a convex hull built on 3 or 10 extreme points of the full Hose model polytope). The following eight columns present results obtained for four different variants of the problem. By gain we understand the percentage of rented capacity that can be saved by using the partitioning of the traffic demand polytope.

Both versions of Different-Reservation-Problem were solved by means of the method of Section III-A. Identical-Reservation-Problem for Robust-Routing-Problem was solved using the double binary search algorithm of Section IV, while Identical-Reservation-Problem for No-Sharing-Problem was solved using both the double binary search method and the algorithm of Section V. That is why the execution times for this version of the problem are presented as $a/b$, where $a$ is time consumed by the double binary search algorithm and $b$ is time consumed by the algorithm of Section V.

The 9000-second time limit has been hit once. Identical-Reservation-Problem for Robust-Routing-Problem has not been solved to optimality for the last test case. Therefore, in this case the gain is presented as $a - b$, where $a$ is the gain of the best solution found and $b$ is the upper bound for it. In other words, the optimal gain belongs to $[a, b]$.

VII. Conclusions

In the paper we presented two algorithms: the double binary search algorithm that solves Identical-Reservation-Problem, and another (faster) algorithm that solves Identical-Reservation-Problem for No-Sharing-Problem when the traffic demand polytope is given by a set of its extreme points. The algorithms were thoroughly tested and proven to be applicable to networks of practical sizes. We also showed possible savings while migrating from the robust stable routing in a direction of the dynamic routing.

The presented algorithms can be easily extended to solve a problem when the traffic demand polytope is divided by many hyperplanes of the same direction. However, a more general problem that consists in providing an optimal set of hyperplanes of unknown directions still remains a challenge.

References