Max-Min Fairness and Its Applications to Routing and Load-Balancing in Communication Networks: A Tutorial

Dritan Nace and Michał Pióro

Abstract—**This tutorial is devoted to the notion of Max-Min Fairness (MMF), associated optimization problems, and their applications to multi-commodity flow networks. We first introduce a theoretical background for a generic MMF optimization problem and discuss its relation to lexicographic optimization. We next present resolution algorithms for convex MMF optimization, and analyze their properties. In the second half of the tutorial we discuss its applications to communication networks, in particular to routing and load-balancing. We state several properties with respect to each of the studied problems and analyze the behavior of the algorithms.**

Index Terms—**Max-min fairness, multi-commodity flow, routing, load-balancing, optimization.**

I. Notion of max-min fairness

THIS TUTORIAL focuses on *max-min fairness* (MMF) and its applications to multi-commodity flow networks. It is intended to give a comprehensive introductory description of this topic to people interested in the MMF theory and its applications to communication networks, especially to PhD students. No particular background is required, except for a basic knowledge on routing/design in communication networks as well as on mathematical programming.

Generally speaking, MMF is applicable in situations where it is desirable to achieve an equitable distribution of certain resources, shared by competing demands. MMF is closely related to max-min or min-max optimization problems, widely studied in the literature. With respect to applications of MMF to communication networks, a lot of work related to rate adaptation and congestion control in TCP (*Transmission Control Protocol*) networks has already been done. Still, it is our conviction that not enough has been done to make people understand the relations of MMF and the multi-commodity flow related applications.

The tutorial is composed of two parts. The first part is dedicated to theoretical aspects of MMF, while the second part treats its applications to communication networks. Hence, we first introduce a theoretical background of the notion of MMF and its relations to lexicographic optimization. We present general formulations of the MMF optimization with a special emphasis on convex problems. Several resolution algorithms follow to complete the theoretical framework of MMF. Some simple examples are also provided to illustrate both the MMF notion and the presented algorithms. In the second part we focus on the way the MMF algorithms are applied to communication networks described in terms of multi-commodity flow framework. After a brief description of the state of the art in MMF, we present its application to routing. This application is thoroughly analyzed and several related properties, including its robustness on traffic variations, are provided. The second application considered in this tutorial is load-balancing in networks. We study several load functions associated with links and especially provide results on a special widely used function, i.e., the Kleinrock function.

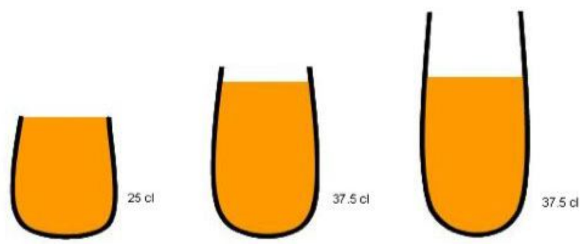

Fig. 1. Example of fair beer sharing between three persons.

A. A simple MMF allocation problem

Let us consider a very simple example. Three persons want to share 100cl of beer in three glasses of different volumes, respectively 25, 40, and 50cl. Obviously the most fair sharing given the volumes of glasses would be the one given in the following Figure 1.

This example, although very simple, illustrates main aspects of MMF. In real-life applications, for example in communication networks, the available resources (as link bandwidth) have to be shared among the actors (traffic demands), as the beer volume among the glasses in our example. Probably the most commonly known problem in communication network traffic

Manuscript received January 24, 2007; revised September 2, 2007. Dritan Nace is supported by Orange Labs. Michał Pióro is supported by the Polish Ministry of Science and Higher Education (MNiSW, grants N517397334 and 3T11C0057), and by the Swedish Research Council (Vetenskapsrådet, grant 621-2006-5509).

Dritan Nace is with Laboratoire Heudiasyc UMR CNRS 6599, Université de Technologie de Compiègne, 60205 Compiègne Cedex, France (e-mail: nace@utc.fr).

Michał Pióro is with Institute of Telecommunications, Warsaw University of Technology, Poland, and Department of Electrical and Information Technology, Lund University, Sweden (e-mail: mpp@tele.pw.edu.pl).

Digital Object Identifier 10.1109/SURV.2008.080403.

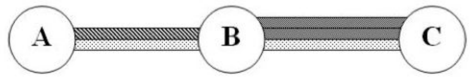

Fig. 2. A simple MMF allocation problem.

engineering related to the notion of MMF is the flow control problem studied by, among others, Bertsekas and Gallager in Section 6.5.2 of their book "Data Networks" [1]. The problem, referred to as MMF/SAP (MMF Simple Allocation Problem) in the sequel, deals with max-min fair bandwidth allocation among connections in a packet network with given link capacities.

The problem is as follows. Consider a network composed of a set $\mathcal{V}$ of nodes, a set $\mathcal{E}$ of links with given capacities C_e ($e \in \mathcal{E}$), and a set $\mathcal{D}$ of connections (demands). Each connection d ($d \in \mathcal{D}$) is assigned a predefined path $\mathcal{P}_d$ between its end nodes; each such path is identified with the set of links it traverses, i.e., $\mathcal{P}_d \subseteq \mathcal{E}, d \in \mathcal{D}$. Now let x_d denote the *bandwidth* allocated to path $\mathcal{P}_d$ and $\boldsymbol{x} = (x_d : d \in \mathcal{D}) \in \mathbb{R}^{|\mathcal{D}|}$ be the corresponding *allocation vector*. We are interested in finding a feasible allocation vector $\boldsymbol{x} \in X$ (the set of all feasible allocation vectors will be denoted by X) which is *fair*.

Clearly, vector $\boldsymbol{x}$ is feasible if $\boldsymbol{x} \geq \boldsymbol{0}$ and

$$\sum_{d \in \mathcal{D}: \mathcal{P}_d \ni e} x_d \leq C_e, \quad e \in \mathcal{E}. \tag{1}$$

Constraint (1) is called the capacity constraint; it assures that for any link e its load does not exceed its capacity. Yet, it is not at all obvious, how to define a fair flow allocation vector $\boldsymbol{x}$. To illustrate this problem, consider the following simple example of a network consisting of 3 routers A, B, C and 2 links AB and BC with the capacities set respectively to 2MB/sec and 3MB/sec. Assume there are 3 connections: $A \to B, B \to C$ and $A \to C$, routed respectively through $A - B$, $B - C$ and $A - B - C$. The resulting (fair) allocation of connections is depicted in Figure 2. The connection $A \to B$ is clearly saturated by link AB; similarly, the connection $B \to C$ is saturated by link BC. We intuitively see that if we start to equally increase the amount of bandwidth allocated to each of the three connections, the first link to become saturated will be AB, with a resulting allocation of 1 unit for each connection. Going one step further, the bandwidth of connection $B \to C$ can still be increased by one unit before the second link BC also becomes saturated. The final MMF rates assigned to the connections are given by the vector $(1, 2, 1)$. This solution is fair according to the common sense.

In the above example we have used the *progressive filling* algorithm presented in [1]. More formally this algorithm can be formulated as follows:

Algorithm 1 (MMF/SAP)

Input: Link capacities ($C_e : e \in \mathcal{E}$), connection paths ($\mathcal{P}_d : d \in \mathcal{D}$).

Output: MMF solution $\boldsymbol{x}^0 \in X$.

- **Step 1:** Set $\boldsymbol{x}^0 = \boldsymbol{0}$ and $k = 0$ (k is the iteration counter).
- **Step 2:** Put $k := k + 1$. Set $\tau = \min_{e \in \mathcal{E}} \frac{C_e}{|\{d \in \mathcal{D}:\, e \in \mathcal{P}_d\}|}$ and put $C_e := C_e - \tau \times \mid \{d \in \mathcal{D} : e \in \mathcal{P}_d\} \mid$ for $e \in \mathcal{E}$. Put $x_d^0 := x_d^0 + \tau$ for $d \in \mathcal{D}$. Remove all saturated links (i.e., all links with $C_e = 0$). Together with each removed link e, remove all connections d for which their paths $\mathcal{P}_d$ use the removed link (i.e., all d with $e \in \mathcal{P}_d$).
- **Step 3:** If there are no connections left then stop, otherwise go to Step 2.

Note that $\mid \{d \in \mathcal{D} : e \in \mathcal{P}_d\} \mid$ is the number of connections that use link e and hence the above algorithm starts from the zero allocation and uniformly increases the individual allocations until one (or more) link(s) gets saturated. Then the connections that cannot be improved are removed from the network, link capacities modified (decreased), and the process continues for the remaining connections. Thus, after the first execution of Step 2, the current vector $\boldsymbol{x}^0$ allocates simultaneously as much bandwidth as possible to *all* connections. Then, the connections for which it is not possible to further increase the allocated bandwidth are removed, and the process continues because in general it is still possible to increase the bandwidth for a subset of connections (for those which do not use saturated links). Denoting the set of the connections removed in iteration k by $\mathcal{D}_k$ we observe that in the final solution the set $\mathcal{D}_1$ of the most "handicapped" connections (i.e., connections which can get the least bandwidth) will be assigned the maximum they can get, then the set $\mathcal{D}_2$ of the next most handicapped connections will get the maximum, and so on. Note that the value of counter k returns the number of times Step 2 is executed, and that there are exactly k distinct values in the final vector $\boldsymbol{x}^0$.

In fact, Bertsekas and Gallager characterize the vector $\boldsymbol{x}^0$ obtained by means of MMF/SAP in a different way. They define an allocation vector $\boldsymbol{x}^0 \in X$ to be max-min fair if it is not possible to increase the allocated bandwidth x_d^0 of any connection d (connections are called sessions in [1]) only at the expense of connections whose allocated bandwidths are greater than x_d^0, i.e., such an increase is possible only if some connections with the allocated bandwidth less or equal to x_d^0 are decreased. More precisely, an allocation vector $\boldsymbol{x}^0$ is said to be max-min fair in X if $\boldsymbol{x}^0 \in X$ and $\boldsymbol{x}^0$ fulfills the following property.

Property 1. *For any allocation vector $\boldsymbol{x} \in X$ and for any connection d such that $x_d > x_d^0$ there exists a connection d' such that $x_{d'} < x_{d'}^0 \leq x_d^0$.*

Besides Property 1, MMF/SAP has several other important properties which are directly implied by the construction of $\boldsymbol{x}^0$ in Algorithm 1. Below we list two more of them (see [2] for details).

Property 2. *The optimal solution $\boldsymbol{x}^0$ is unique.*

Property 3. *For each connection d there exists a saturated (bottleneck) link e on path P_d (i.e., $e \in P_d$ and $C_e = \sum_{d: \mathcal{P}_d \ni e} x_d^0$) such that x_d^0 is at least as large as the bandwidth allocated to any other connection using link e.*

Note that this means that the number of times Step 2 is executed is at most the number of links in E, i.e., $k \leq |\mathcal{E}|$. We note also that in the literature one can find another notion closely related to the MMF resource allocation described above, namely *max-min fair sharing*. It is easy to see that in Figure 2 each connection is necessarily saturated on one of its links, called a bottleneck link. Its (end-to-end) assigned bandwidth is then equal to the minimum of the shares offered by each link in the path. This can also be expressed as the share offered by the bottleneck link to connections saturated on this link[1]. This share is sometimes referred to as *max-min fair share*. Certainly, for the network in Figure 2, the MMF shares of the links are given by vector $(1, 2)$.

In the balance of this tutorial we will introduce a general MMF optimization problem which will generalize MMF/SAP, and next we will present algorithms for the convex case of the MMF problem as well as their applications to routing and load-balancing.

B. General formulations for MMF optimization

There are several ways for formally introducing the idea of max-min fairness. A general way to do it is to use the notion of lexicographical order of ordered outcome vectors, i.e., the way followed in [2], [3], [4] and others.

Consider a vector of real-valued functions defined on a set $X \subseteq \mathbb{R}^n$ of real n-vectors $\boldsymbol{x} = (x_1, x_2, .., x_n)$ (2). Vector $\boldsymbol{f}$ will be referred to as a vector of outcomes, objectives, or criteria.

Definition 1. *Vector $\boldsymbol{y} \in \mathbb{R}^m$ is called lexicographically greater than vector $\boldsymbol{z} \in \mathbb{R}^m$, $\boldsymbol{y} \succ \boldsymbol{z}$, if there exists $j \in \{1, ..., m\}$ such that $y_i = z_i$, for all $i \in \{1, ..., j-1\}$ and $y_j > z_j$. If $\boldsymbol{y} \succ \boldsymbol{z}$ or $\boldsymbol{y} = \boldsymbol{z}$ then we write $\boldsymbol{y} \succeq \boldsymbol{z}$.*

Definition 2. *The lexicographical maximization problem (LXM/P) for given X and $\boldsymbol{f}$ is denoted by*

$$\textbf{\textit{lexmax}}_{\,\boldsymbol{x} \in X}\ \boldsymbol{f}(\boldsymbol{x}), \tag{3}$$

and consists in finding a vector $\boldsymbol{x}^0 \in X$ for which $\boldsymbol{f}(\boldsymbol{x}^0)$ is lexicographically maximal over X, that is, for all $\boldsymbol{x} \in X$, $\boldsymbol{f}(\boldsymbol{x}^0) \succeq \boldsymbol{f}(\boldsymbol{x})$.

A natural way of solving LXM/P is to first maximize $f_1(\boldsymbol{x})$ over X, then, denoting the maximum value of $f_1(\boldsymbol{x})$ with f_1^0, maximizing $f_2(\boldsymbol{x})$ for all vectors $\boldsymbol{x} \in X$ such that $f_1(\boldsymbol{x}) \geq f_1^0$, and so on. Hence, the resolution algorithm for LXM/P is as follows.

Algorithm 2 (General algorithm for LXM/P)

Input: Optimization space X and criteria $\boldsymbol{f}$.
Output: Solution $\boldsymbol{x}^0 \in X$ and the optimal (lexicographically maximal) criteria vector $\boldsymbol{f}^0 = (f_1^0, f_2^0, ..., f_m^0)$.

- **Step 1:** Set $j = 1$ and $X_1 = X$.
- **Step 2:** Solve the following single-objective problem

$$\textbf{\textit{max}}_{\,\boldsymbol{x} \in X_j}\ f_j(\boldsymbol{x}) \tag{4}$$

  and denote an optimal solution and the optimal solution value by $\boldsymbol{x}^0 \in X_j$ and f_j^0, respectively. If $j = m$ then stop: $(\boldsymbol{x}^0, (f_1^0, f_2^0, ..., f_m^0))$ is an optimal solution to LXM/P.
- **Step 3:** Set $X_{j+1} = X_j \cap \{\boldsymbol{x} : f_j(\boldsymbol{x}) \geq f_j^0\}$ and $j = j+1$. Go to Step 2.

In LXM/P, maximization of the first outcome f_1 is the most important, maximization of the second outcome f_2 is the next important, and so on. In consequece, the optimization scheme given in the above algorithm is quite simple.

For the MMF optimization problems, i.e., for the optimization problems considered in this tutorial, the resolution algorithms are more complicated because we do not assume any sequence of priorities of the objectives. We rather treat them equally and try to first maximize the minimum outcome (or all the minimum outcomes, if there are more than one), whichever it is. Hence, the first step in the MMF optimization is to solve the problem:

$$\textbf{\textit{max}}\ \ \tau \tag{5}$$

$$f_j(\boldsymbol{x}) \geq \tau, \quad j = 1, 2, ..., m \tag{6}$$

$$\boldsymbol{x} \in X. \tag{7}$$

After solving problem (5-7) there may be room for further increasing some of the outcomes (but not all). How to do this in general is, however, not obvious and the notion of MMF optimization has to be formally introduced for this purpose. Let $\langle \boldsymbol{y} \rangle = (\langle y \rangle_1, \langle y \rangle_2, ..., \langle y \rangle_m)$ denote a version of vector $\boldsymbol{y} = (y_1, y_2, ..., y_m) \in \mathbb{R}^m$ ordered in the non-decreasing order (i.e., for some permutation φ on the set $\{1, 2, ..., m\}$ it holds that $\langle y \rangle_j = y_{\varphi(j)}$ for $j = 1, 2, ..., m$ and $\langle y \rangle_1 \leq \langle y \rangle_2 \leq ... \leq \langle y \rangle_m$).

Definition 3. *The max-min fairness optimization problem (MMF/OP) for given X and $\boldsymbol{f}$ is as follows*

$$\textbf{\textit{lexmax}}_{\,\boldsymbol{x} \in X}\ \langle \boldsymbol{f}(\boldsymbol{x}) \rangle. \tag{8}$$

Hence, MMF/OP consists in lexicographical maximization of the ***sorted outcome vector*** *$\boldsymbol{f}(\boldsymbol{x})$ over X. Any optimal solution vector $\boldsymbol{x}^0 \in X$ of (8) is called max-min fair on set X (MMF on X) with respect to criteria $\boldsymbol{f}$.*

Such a MMF vector is called *leximin maximal* in [5].

Suppose that $\phi : \mathbb{R} \to \mathbb{R}$ is a strictly increasing function and let $f_j' = \phi \circ f_j, j = 1, 2, ..., m$ be a superposition of ϕ and f_j. Then the following proposition holds [6].

Proposition 1. *A vector $\boldsymbol{x}^0 \in X$ is MMF on set X with respect to criteria $\boldsymbol{f}$ if, and only if, $\boldsymbol{x}^0$ is MMF on X with respect to criteria $\boldsymbol{f}'$.*

The above result will be useful when dealing with non-linear link load functions in the load-balancing problem considered in Section 2.3.

Let us now turn back to Problem MMF/SAP considered in Section I-A. Using the general notion of MMF/OP (8), Problem MMF/SAP takes the following form:

$$\textbf{\textit{lexmax}}_{\,\boldsymbol{x} \in X}\ \langle (x_1, x_2, ..., x_D) \rangle, \tag{9}$$

[1]In other words, the capacity that remains on a link after satisfying the bandwidth requirements of connections routed through the link but saturated elsewhere in their paths is shared equitably among the rest of connections (i.e., saturated on this link) and this share determines the bandwidth allocated to each of these connections.

$$\boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), ..., f_m(\boldsymbol{x})), \quad \text{where} \;\; f_j : X \rightarrow R, \; j = 1, 2, ..., m \tag{2}$$

where $\mathcal{D} = \{1, 2, ..., D\}$ and X is the set of all feasible allocation vectors with respect to the given routing paths. It can be easily shown (see Section 8.1.2 in [2]) that Algorithm 1 does indeed solve problem (9). However, as we will soon learn, general resolution algorithms for MMF/OP are not so simple as Algorithm 1. In fact, describing such general algorithms and their applications to a class of routing problems in communication networks is the main purpose of this tutorial.

C. An alternative definition of MMF

The notion of MMF given in Definition 3 is general, i.e., applicable to all sets X and criteria $\boldsymbol{f}$. Some authors, however, adopt a less general (and in our opinion less intuitive) definition of MMF that is related to Property 1, [1], [5], [7].

Definition 4. *A vector $\boldsymbol{x}^0$ is max-min fair on set $X \subseteq \mathbb{R}^m$ if, and only if, $\forall\ \boldsymbol{x} \in X,\ \exists\ k \in \{1, ..., m\},\ (x_k > x_k^0) \Rightarrow (\exists\ j \in \{1, ..., m\},\ x_j < x_j^0 \leq x_k^0)$.*

In other words, the above definition states that it is not possible to increase (the value of) some component in the vector $\boldsymbol{x}^0$ without decreasing some component with a lower value. Certainly, Definition 3 and Definition 4 cannot be directly compared because the latter does not take into account any criteria function f. To make this possible, Definition 4 can be rewritten as follows:

Definition 5. *A vector $\boldsymbol{x}^0$ is max-min fair on set $X \subseteq \mathbb{R}^m$ with respect to criteria $\boldsymbol{f}$ if, and only if, $\forall\ \boldsymbol{x} \in X,\ \exists\ k \in \{1, ..., m\},\ (f_k(x) > f_k(x^0)) \Rightarrow (\exists\ j \in \{1, ..., m\},\ f_j(x) < f_j(x^0) \leq f_k(x^0))$.*

We note that Definition 3 and Definition 5 are in general not equivalent. Still, they are equivalent whenever X is compact and convex, and functions $f_j, j = 1, 2, ..., m$ are concave. Moreover, the two definitions are always equivalent when the solution of problem (8) is unique.

To further illustrate the relation between the definitions of MMF, consider the allocation Problem MMF/SAP (introduced in Section 1.1) for a simple network composed of just one link (e) with capacity $C_e = 1$, and two distinct demands ($d = 1, 2$) between the end nodes of the link. With respect to the above MMF definitions we have $f_1(x) = x_1$ and $f_2(x) = x_2$, i.e., each criterion is determined by the flow value assigned to a demand. Hence, the criteria vector is (x_1, x_2), while the set X is determined by the capacity constraint: $x_1 + x_2 \leq 1$ (and by $x_1, x_2 \geq 0$). The solution of MMF/SAP is clearly $x_1^0 = x_2^0 = 0.5$, and the resulting vector $\boldsymbol{x}^0$ is MMF according to both definitions. Still, if we assume integrality of the demand flows (i.e., that the flows must be integers) we arrive at two different MMF vectors according to Definition 3: $\boldsymbol{x}' = (1, 0)$ and $\boldsymbol{x}'' = (0, 1)$. None of these vectors is MMF according to Definition 5. Notice that in the integral case the solution space is not convex and the MMF vector according to Definition 5 does not exist at all.

D. An algorithm for convex MMF optimization problems

Throughout this subsection we assume that the set X is compact and convex, and that all criteria functions $f_j(\boldsymbol{x}), j = 1, 2, ...m$, are concave. With these assumptions Problem MMF/OP will be called convex MMF/OP (MMF/CXOP in short). As we shall see, the assumed convexity/concavity will ensure that all the single-objective optimization sub-problems encountered in the sequel of this section are convex. The convexity/concavity assumption is quite strong; in particular it implies the following proposition.

Proposition 2. *Suppose X is convex and $\boldsymbol{f}$ are concave, and let $\boldsymbol{x}'$ and $\boldsymbol{x}''$ be two different optimal solutions of MMF/OP (8). Then*

$$\boldsymbol{f}(\boldsymbol{x}') = \boldsymbol{f}(\boldsymbol{x}''). \tag{10}$$

Proposition 2 states that for the convex case not only $\langle \boldsymbol{f}(\boldsymbol{x}') \rangle = \langle \boldsymbol{f}(\boldsymbol{x}'') \rangle$, which is obvious, but also that the MMF solution is unique in the criteria space. In other words convexity/concavity ensures the uniqueness of the MMF vector (when such a vector exists). However, we need to add compactness for the set X to guarantee both existence and uniqueness, and hence the equivalence of Definitions 3 and 5 for this case.

Now we shall present a general algorithm for MMF/CXOP. Suppose B is a subset of the index set $M = \{1, 2, ..., m\}$, $B \subseteq M$, and let $\boldsymbol{t}^B = (t_j \;:\; j \in B)$ be a $|B|$-element vector. Also, let B' denote the set complementary to $B: \; B' = M \setminus B$. For given B and $\boldsymbol{t}^B$ we define the following (convex!) mathematical programming problem in variables $\boldsymbol{x}$ and τ, $\mathcal{P}(B, \boldsymbol{t}^B)$:

$$\begin{aligned}
&maximize && \tau && && \text{(11a)}\\
&subject\ to && f_j(\boldsymbol{x}) \geq \tau && j \in B' && \text{(11b)}\\
& && f_j(\boldsymbol{x}) \geq t_j^B && j \in B && \text{(11c)}\\
& && \boldsymbol{x} \in X. && && \text{(11d)}
\end{aligned}$$

It is clear that the solution τ^0 of problem $\mathcal{P}(\emptyset, \emptyset)$ defined by (11) for empty set B and empty sequence $\boldsymbol{t}^B$ will yield the smallest value of the corresponding MMF criteria vector. This observation suggests the following algorithm for solving Problem MMF/CXOP.

Algorithm 3 (General algorithm for MMF/CXOP)

Input: Compact convex optimization space X and concave criteria $\boldsymbol{f}$.
Output: MMF solution $\boldsymbol{x}^0 \in X$ and the (optimal) non-sorted MMF criteria vector $\boldsymbol{t}^B$.

- **Step 1:** Set $B = \emptyset$ (empty set) and $\boldsymbol{t}^B = \emptyset$ (empty sequence).
- **Step 2:** If $B = M$ then **stop** $(\boldsymbol{x}^0, \langle \boldsymbol{t}^B \rangle$ is the optimal solution of Problem MMF/OP). Else, solve $\mathcal{P}(B, \boldsymbol{t}^B)$ and denote the resulting optimal solution by $(\boldsymbol{x}^0, \tau^0)$.

- **Step 3:** For each index $k \in B'$ such that $f_k(\boldsymbol{x}^0) = \tau^0$ solve the following test problem $\mathcal{T}(B, \boldsymbol{t}^B, \tau^0, k)$:

$$\begin{aligned} &\mathit{maximize} && f_k(\boldsymbol{x}) && && \text{(12a)}\\ &\mathit{subject\ to} && f_j(\boldsymbol{x}) \geq \tau^0 && j \in B' \setminus \{k\} && \text{(12b)}\\ & && f_j(\boldsymbol{x}) \geq t_j^B && j \in B && \text{(12c)}\\ & && \boldsymbol{x} \in X. && && \text{(12d)} \end{aligned}$$

  If for optimal $\boldsymbol{x}'$, while solving test $\mathcal{T}(B, \boldsymbol{t}^B, \tau^0, k)$ we have $f_k(\boldsymbol{x}') = \tau^0$ (i.e., when criterion $f_k(\boldsymbol{x})$ cannot be further increased), then we put $B := B \cup \{k\}$ and $t_k^B := \tau^0$. Go to Step 2.

The above algorithm constructs the MMF criteria vector solution in the increasing order of its components and find in the final iteration the $\boldsymbol{x}^0$ solution.

It is natural to ask what is the relation of the general algorithm given above and Algorithm 1 for Problem MMF/SAP in Section 1.1. Certainly, MMF/SAP is a convex MMF problem so Algorithm 3 applies to it. Notice that the value of τ computed (in a direct way) in Step 2 of Algorithm 1 is just equal to τ^0, and hence it is an optimal solution of a consecutive problem $\mathcal{P}(B, \boldsymbol{t}^B)$ in Step 2 of Algorithm 3. On the other hand, the general tests $\mathcal{T}(B, \boldsymbol{t}^B, \tau^0, k)$ are not used in Algorithm 1 since in the case of MMF/SAP they become very simple: we test the saturation of links instead.

E. Accelerating the MMF algorithm

Clearly Algorithm 3 presented above can be time consuming due to an excessive number of instances of the problems (11) and (12) that have to be solved in the iteration process. On the other hand, we have already seen in Algorithm 1 that in the particular case of the routing Problem MMF/SAP, the tests (12) may become very easy to perform. Below we shall show that in general the tests $\mathcal{T}(B, \boldsymbol{t}^B, \tau^0, k)$ can be made much more efficient provided that optimal dual variables for problems $\mathcal{P}(B, \boldsymbol{t}^B)$ can be effectively computed. Although the following results derived for dual variables hold for MMF/CXOP, the most important application of their use is LP since for this case the optimal dual variables are readily obtained (e.g., by using the Simplex method).

Suppose $\boldsymbol{\lambda} = (\lambda_j : \ j \in B')$ denotes the vector of dual variables (multipliers) associated with constraints (11b) in the linear case, that is when criteria $f_j(x)$ are linear ($j = 1, 2, ..., m$), and when the set X is described by a system of linear equations/inequalities. With respect to the dual formulation, the following proposition can be proved [2].

Proposition 3. *Let $\boldsymbol{\lambda}^0$ be the vector of optimal dual variables solving the corresponding dual problem. Then*

$$\sum\nolimits_{j \in B'} \lambda_j^0 = 1 \tag{13}$$

and if $\lambda_j^0 > 0$ for some $j \in B'$, then $f_j(\boldsymbol{x})$ cannot be improved, i.e., $f_j(\boldsymbol{x}^0) = \tau^0$ for every optimal primal solution $(\boldsymbol{x}^0, \tau^0)$ of (11).

The above result suggests to simplify the step 3 of Algorithm 3 as follows:

- **Step 3:** Set $B := B \cup \{j \in B' : \ \lambda_j^0 > 0\}$. Go to Step 2.

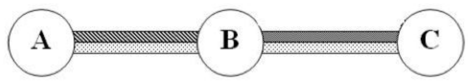

Fig. 3. Example of existence of a blocking demand with null dual coefficient.

This variant of Algorithm 3, called in the sequel *dual-based MMF Algorithm*, is used in Section 2 when addressing the MMF routing problem. Observe that if for some $j \in B'$ with $\lambda_j^0 = 0,\ f_j(\boldsymbol{x})$ cannot be further improved, then in Step 2 the value of τ^0 will not be improved; still at least one such index j will be detected (due to property (13)) and included into set B in the next execution of Step 3.

When using the simplex method for solving LP problems, the optimal dual variables can be obtained directly from the simplex tableau, (in fact the simplex tableau was a basis of early implementations of the MMF solution for LP problems [8], [9], [10]). Note that in general the inverse of the second part of Proposition 3 does not hold: $\lambda_j^0 = 0$ does not necessarily imply that $f_j(\boldsymbol{x})$ can be improved. Indeed, let us take a similar network to this given in Figure 2, that is composed of three nodes A, B and C, and set the capacity of both links to two. We consider three demands, one for each pair of nodes and denote with $\boldsymbol{x}_1$ ($\boldsymbol{x}_2$, $\boldsymbol{x}_3$) the throughput associated with demand $A \rightarrow B$ (respectively $B \rightarrow C$, $A \rightarrow C$). Obviously, there is a single route for each demand and the corresponding MMF allocation is given in Figure 3. The mathematical program related to the first step of MMF computing is then as follows (see also [11], [2]):

$$\begin{aligned} &\mathit{maximize} && \tau && && \text{(14a)}\\ &\mathit{subject\ to} && \boldsymbol{x}_j \geq \tau, && j = 1, 2, 3 && \text{(14b)}\\ & && \boldsymbol{x}_1 + x_3 \leq 2 && && \text{(14c)}\\ & && \boldsymbol{x}_2 + x_3 \leq 2. && && \text{(14d)} \end{aligned}$$

There are then two dual solutions λ with respect to constraints (14b) in the above formulation: $\lambda' = (0, 1/2, 1/2)$ and $\lambda'' = (1/2, 0, 1/2)$. Clearly all demands are blocking at the first step. Hence, none of them is a non-blocking one despite the null value of its dual coefficient.

In other words, we cannot be sure that all blocking/non-blocking indices $j \in B'$ are detected in Step 3. To remedy this, a first idea would be to maximize simultaneously all the slack variables s_i associated with constraints i, hoping that all these corresponding to non-blocking indices will reach strictly positive values. We thus obtain the following LP program:

$$\begin{aligned} &\mathit{maximize} && \textstyle\sum_{j=1}^m s_j && \text{(15a)}\\ &\mathit{subject\ to} && f_j(\boldsymbol{x}) - \tau^0 - s_j \geq 0, \quad j = 1, 2, ..., m && \text{(15b)}\\ & && \boldsymbol{x} \geq \boldsymbol{0},\ 0 \leq s_j \leq \varepsilon, j = 1, 2, ..., m && \text{(15c)} \end{aligned}$$

However, at this stage we are not able to provide the right ε value, which would guarantee that the concerned slack variables will take the maximal value (i.e., ε). We can remedy this through introducing an additional variable $\gamma \geq 1$ that multiplies each constraint, and replace $\gamma \boldsymbol{x}$ by $\boldsymbol{y}$ (see for instance [12]). Next, we bound the slack variables s_j from

$$\begin{array}{lll} maximize & \sum_{j=1}^{m} s_j & \text{(16a)} \\ subject\ to & f_j(\boldsymbol{y}) - \gamma\tau^0 - s_j \geq 0, j = 1, 2, ..., m & \text{(16b)} \\ & \gamma \geq 1, \boldsymbol{y} \geq \boldsymbol{0},\ 0 \leq s_j \leq 1,\ j = 1, 2, ..., m & \text{(16c)} \end{array}$$

above by 1. The resulting problem is as in (16). If index j is not blocking for the initial problem $\mathcal{P}(B, \boldsymbol{t}^B)$, then we can choose a value of γ large enough in the transformed problem so that $s_j = 1$. So, index j for (11b) is not blocking if, and only if, $s_j = 1$ in an optimal solution for LP (16a) - (16b).

Another way to handle this issue is to use methods computing strictly complementary optimal solutions (i.e., when there is only one zero value for each complementary pair of primal-dual variables). Let us recall that the complementary slackness property, one of the most fundamental properties in the LP theory, concerns the conditions relating a pair of complementary optimal solutions, that is, primal and dual solution. Let us consider a constraint with index i in the primal problem and a pair of corresponding variables—primal slack variable s_i and dual variable λ_i. The complementary slackness theorem states that at an optimal solution at least one of the variables s_i and λ_i is equal to 0. An optimal solution is called strictly complementary optimal if only one of these variables takes the zero value (while the other is strictly non-zero). Thus a strict complementary optimal solution $\boldsymbol{x}^0 \in X$ has $\lambda_j^0 > 0$ for all blocking indices simultaneously. Finding directly all blocking indices could be obtained by LP solvers that compute strictly complementary optimal solutions, as IPM (Interior Point Method) based on the central trajectory approach (see for instance [13]). With respect to MMF, a general (polynomial-time) approach based on strictly complementary optimal solutions is presented in [14]. Independently, their usefulness in the convergence of MMF algorithm methods is also shown in [2] (Chapter 8, Example 8.4).

F. Min-max fairness

Analogously to the notion of max-min fairness, we can introduce the notion of min-max fairness:

Definition 6. *Vector $\boldsymbol{x}^0 \in X$ is called min-max fair on set $X \subseteq \mathbb{R}^m$ with respect to criteria $\boldsymbol{f}$, if $\langle(\boldsymbol{f}(\boldsymbol{x}^0)\rangle$ is lexicographically minimal over X, that is, lexicographically smaller than $\langle(\boldsymbol{f}(\boldsymbol{x})\rangle$ for all $\boldsymbol{x} \in X$.*

We could similarly introduce the counterpart of Definition 4 for min-max fairness. Clearly, both definitions are related in the same way as for max-min fairness. Also, analogous approaches can be developed for the min-max fair problem. Certainly, the min-max fairness problems arise when one wants to lexicographically minimize the sorted outcome vector (see [14] for details). An application of min-max fairness to communication networks will be described in Section 2.3.

II. MMF APPLICATIONS TO MULTI-COMMODITY FLOW NETWORKS

In mathematics, the notions of lexicographical ordering and MMF appear both in (multi-criteria) optimization and in game theory. In this tutorial we follow the optimization point of view. Applications of MMF have been studied in many contexts, for example in economics and in multi-commodity flow networks, with a particular emphasis on telecommunications. Questions related to fairness in handling packet traffic have arisen with the deployment of Internet and are given an extensive coverage in the literature, see for example [15], [16], [17], [18], [19], [20], [21], [22], [23], [24].

The area concerning fairness in packet traffic engineering is too broad to be fully discussed in a tutorial like this. Consequently, we restrict the discussion to two selected issues in traffic routing and load balancing. So, concerning the general issues, here we only mention that one of the main problems related to fairness in traffic engineering is the rate and congestion control in TCP/IP networks, as discussed in [22]. There, the authors have studied and proposed congestion control schemes in order to prevent the network from entering a congestion collapse[2]. The key issue discussed there is the extent to which the congestion control mechanisms influence resource sharing among the competing sessions, the fairness of resource sharing, and the advantages of assuming fairness as a design objective. Another important result (see [16]) concerning the TCP behavior is that the AIMD (*Additive Increase, Multiplicative Decrease*) algorithm[3] (on which TCP is based) converges to an efficient and fair equilibrium point corresponding to *max-min fair* allocation in a network composed of a single bottleneck link with n users sharing the link. However, this result does not in general hold for networks with multiple bottlenecks, and hence, assuming max-min fair resource allocation obviously does not allow for an accurate description of the network behavior. Still, the MMF assumption yields an acceptable degree of approximation which can lead to useful conclusions for different application settings. Furthermore, such an assumption allows for defining computationally tractable models.

This section is composed of three paragraphs. The first paragraph briefly describes the related work, while the second and third paragraphs describe, respectively, selected applications of MMF to routing and to load-balancing.

A. Related work

Below we briefly discuss work related to application of MMF to traffic routing in multi-commodity flow networks. We start with an early special application (Subsection II-A1) and then, in Subsection II-A2, we proceed to a more general case.

[2]Congestion collapse refers to the state of a packet network when congestion is so high that almost no useful (payload) traffic is carried.

[3]Based on the principle of AIMD, a TCP connection probes for extra bandwidth by increasing its congestion window linearly with time, and on detecting congestion, reducing its window by a factor of two.

1) A problem in multi-commodity flow networks.: An early work on MMF in multi-commodity flow networks is presented in [25] where the author has studied the fair maximum flow problem for single-source multi-sink networks. This problem can be stated as follows: in a network with given link capacities find an MMF flow from a given source to a set of sinks. After demonstrating that the well-known Ford-Fulkerson labeling method for paths' augmenting (proposed for the traditional maximum flow problem) is not applicable, the author presents an elegant method for computing such a MMF single-source flow. He also demonstrates that in this case the MMF solution maximizes the total flow from the source to all the sink nodes (this is left as an exercise for the reader).

2) Applications of MMF in IP routing: MMF applied to routing multi-commodity flows with elastic rates (as flows of best-effort traffic in the Internet) has been the subject of numerous papers, see for example [1], [2], [3], [7], [26], [27], [28], [29], [30], [31], [32]. These papers can be divided into two categories, dealing with off-line (static routing) and on-line (dynamic routing) flow optimization, respectively. In this tutorial we focus on the static routing case.

The dynamic case (when requests for individual connections/sessions come and go, and for each request a connection path must be found on-line) is considered in [29], [30], [31], which examine and compare performance of some strategies of routing combining the length (or cost) with the bandwidth. In the subsequent studies the authors have tried to find exact algorithms achieving fairness and/or maximizing the overall throughput. They are all on-line routing algorithms: that is, for each new connection, the source computes in real-time the appropriate path according to local or collected information.

In the static routing case, connection paths and corresponding traffic flow are to be found off-line (in advance) and used for a long period. This case is treated in [2], [3], [26], [27], [7], [32]. In [26] the authors propose an LP model for the fixed path routing case case, and, in addition, a heuristic extension of their algorithm for computing certain variants of fair routing.

In [7], the problem of max-min fair bandwidth-sharing among TCP/IP connections when routing paths are not fixed has been considered from the off-line optimization point of view. This work has led to some complex models that lack linearity when considering fairness levels. The proposed algorithm can be seen as an extension of the Algorithm 1 (MMF/SAP) given in Section 1, except that the routing is not fixed and at each iteration a new routing is computed while the previously saturated links and the corresponding fair sharing remain fixed until the end of the algorithm. Also, two other approaches, in the spirit of Algorithms 3 and its dual-based variants, enriched by theoretical analysis, are given in [2], [6], [11].

Several variants of the fair (multi)flow problem have been presented and analyzed in [1], [6], [25], [33], etc. One can distinguish here two different cases: splittable and unsplittable fair flows with respect to the number of paths used to transfer the flow (each flow supplied to a sink is transported through a single (resp. multiple) path for the unsplittable (resp. splittable) case). Conversely, the fair single-source unsplittable flow problem is shown to be NP-complete: special cases of the latter problem include several fundamental load-balancing problems, [3]. In this latter work, the authors propose an interesting approximated algorithm for the fair unsplittable single-source flow problem. However, in many application contexts (e.g., transport), problems are generally not limited to single-source networks: several commodities are required to share the same underlying network, which is the context that we focus on here.

Fairness issues find applications also in load-balancing. Some relevant work on balanced networks is presented in [28] where the authors propose an approach for lexicographically-optimal balanced networks. The problem they address is allocating bandwidth between two endpoints of a backbone network so that the network is equitably loaded, and therefore it is limited to the case of single-commodity network. In contrast, our concern here is a more general case, namely the multi-commodity network.

B. MMF routing

One of the most relevant and fruitful applications of MMF to communication networks is fair routing. Generally speaking, the routing problem for telecommunication operators consists of two consecutive tasks: a credible estimation of traffic demands (commodities), and the subsequent determination of the appropriate paths (routes) for flows realizing the demands, i.e., routing of these multiple commodities in the network. For a range of applications, the latter problem can be modeled as a multi-commodity flow problem in a capacitated network for a given traffic demand matrix.

Forecasts of traffic demand (summarized in the demand matrix) are generally expressed as volumes of traffic (e.g., in Mbs/sec) to be transported, or a number of expected connections to be established between pairs of nodes. Routing optimization is a major component of traffic engineering as well as a part of more complex and general problems related to the design and survivability of communication networks. Let us recall that there are two main extremes in routing traffic in a network. At the least constrained end of the spectrum, routing is splittable (or bifurcated or multi-path) if the volume of a traffic demand is allowed to be split among multiple path flows. The opposite is unsplittable (or non-bifurcated or single-path) routing, where each demand has to be routed along only one single path. Although unsplittable routing gives rise to NP-complete problems, from the communication network point of view the single-path routing is a common requirement in today's IP networks, for example using the OSPF (Open Shortest Path First) protocol. On the other hand, network operators can use the MPLS (Multi-Protocol Label Switching) technology to enable the splitting of traffic among several paths, and hence bifurcated routing can be acceptable as well.

An important reason acting in favor of splittable routing is that even if we assume unlimited demand split, a routing solution obtained through multi-commodity flow models will indicate almost unsplittable routing, with only one route used by a great majority of demands; typically only few demands will use more than one path. From a mathematical point of view, the case of splittable demand routing can be viewed as a linear relaxation of the unsplittable case, thus rendering the problem computationally tractable [7].

The MMF routing problem that we study in this section is intended to achieve the fairest allocation of resources via routing, that is, each demand is to be served fairly, as much as the available network resources permit. This is especially advantageous for elastic traffic flows, which account for a major part of Internet traffic. In practice, the traffic demand changes faster than the network topology and resource capacity, and therefore network operators cannot afford updating their routing schemes according to changes in the traffic distribution. As a consequence, one approach frequently used by the operators for seeking to manage resources wisely while confronted with dynamic real-time traffic conditions is to design the best possible (optimized) static network conditions for a set of estimated average or worst-case traffic demands. This supports the use of MMF: an interesting property of MMF routing is its robustness; we will show that MMF routing solutions are, to a certain extent, robust to demand traffic changes.

Finally, we notice that the MMF routing problem can be used by IP network planners wishing to evaluate the impact of a certain degree of fairness on the resource consumption. In this case, MMF can be viewed as an approximation of the real behavior of TCP. We believe that the degree of this approximation is acceptable to derive useful conclusions in this particular context. On the other hand, it may very soon happen that network planners start thinking on imposing a certain level of fairness into their network, using, for instance, the MPLS based mechanisms.

1) Mathematical model and resolution method: In this section we will introduce the basic notation, specify the MMF routing problem, and its resolution method.

a) Notation and formulation of the problem.: Following the notation given before, let a capacitated network $\mathcal{N}$ be given by the network graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with the set of nodes $\mathcal{V}$ and the set of directed links (arcs) $\mathcal{E}$, $\mathcal{E} \subset \mathcal{V}^2$. Each link $e \in \mathcal{E}$ is of the form $e = (v, w)$ for some different nodes $v, w \in \mathcal{V}$. In the sequel the set of all links outgoing from node v will be denoted by $\delta^+(v)$ (i.e., $\delta^+(v) = \{e \in \mathcal{E} : \exists w \in \mathcal{V}, e = (w, v)\}$). Analogously, $\delta^-(v) = \{e \in \mathcal{E} : \exists w \in \mathcal{V}, e = (v, w)\}$ will denote the set of all links incoming to node v. The (fixed) capacity of link $e \in \mathcal{E}$ is denoted by C_e. The set of demands (commodities) is denoted by $\mathcal{D}$. Each commodity $d \in \mathcal{D}$ has the source (originating) node $o(d)$ and the destination (sink) node $s(d)$, and the target demand volume h_d.

Let $f_{ed} \geq 0$ denote the flow of commodity (demand) $d \in \mathcal{D}$ on link $e \in \mathcal{E}$ and consider a vector $\boldsymbol{f}_d = (f_{ed} : e \in \mathcal{E})$ for a fixed demand d. Vector $\boldsymbol{f}_d$ is called a single-commodity flow of commodity d (with the flow value $\varphi_d \geq 0$) in network $\mathcal{N}$ if

$$\sum_{e \in \delta^+(v)} f_{ed} - \sum_{e \in \delta^-(v)} f_{ed} = \left\{ \begin{array}{ll} \varphi_d, & v = o(d) \\ -\varphi_d, & v = s(d) \\ = 0 & \text{otherwise} \end{array} \right\} \quad v \in \mathcal{V}. \tag{17}$$

Now consider a vector $\boldsymbol{f} = (\boldsymbol{f}_d : d \in \mathcal{D})$ composed of single-commodity flows $\boldsymbol{f}_d$, one for each commodity $d \in \mathcal{D}$. Vector $\boldsymbol{f}$ is a *feasible multi-commodity flow* (*multiflow* in short) in network $\mathcal{N}$ if

$$\sum_{d \in \mathcal{D}} f_{ed} \leq C_e, \; e \in \mathcal{E}. \tag{18}$$

The theorem of decomposition [33] assures that any feasible flow f_d can be decomposed into a set of flows assigned to paths between $o(d)$ and $s(d)$. Then a multi-commodity flow becomes a collection of path flows, whose aggregation satisfies the capacity constraint (18). This leads to an alternative mathematical formulation of multi-commodity flow problems, the so called *link-path formulation*, different from *node-link formulation* (17)–(18) (see for example [2]). In the link-path formulation, each path in the network is associated with a path-flow variable. This potentially requires a large (exponential with the size of the network graph) number of variables. However, link-path formulations can be solved using column generation (see for instance [34]), called path generation in the considered context (see [33], [2]). Path generation is in most cases efficient, especially for networks with low connectivity, where the number of paths is limited. From the communication network point of view, link-path formulations are convenient since they allow to restrict the type of routing paths, for example restrict the number of hops (transit nodes) in the paths (see for instance [35]) which is difficult in the node-link formulation.

A feasible multi-commodity flow $\boldsymbol{f}$, with $\varphi_d \geq h_d, d \in \mathcal{D}$, defines an *admissible flow (routing) pattern*. Now, consider a feasible multi-commodity flow $\boldsymbol{f}$ and a vector $\boldsymbol{t} = (t_d \geq 0 : d \in \mathcal{D})$ such that $\varphi_d = t_d h_d$ in (17). If $t_d \geq 1$ for all $d \in \mathcal{D}$ then the routing is admissible, i.e., it fulfills (perhaps with some surplus) the assumed requirement $h_d, d \in \mathcal{D}$. In general this may not be the case and therefore we call the vector $\boldsymbol{t}$ the *demand satisfaction vector* of routing $\boldsymbol{f}$. We may say now that the MMF routing problem for network $\mathcal{N}$ consists in finding a feasible multi-commodity flow $\boldsymbol{f}$ in $\mathcal{N}$ so that the corresponding demand satisfaction vector $\boldsymbol{t}$ is MMF.

The MMF routing problem can also be seen as an extension of the concurrent flow problem. Recall that the concurrent flow problem [36] is defined as a multi-commodity flow problem in a capacitated network $\mathcal{N} = (\mathcal{V}, \mathcal{E})$ and a set of commodities $\mathcal{D}$. For each commodity d, the goal is to send h_d units from the source node $o(d)$ to its destination node $s(d)$. If there is no feasible solution, then the objective is to maximize the satisfaction ratio for each commodity, that is, find the largest α such that for each commodity we can send simultaneously αh_d units. In the MMF routing problem, we look for a satisfaction ratio vector that is MMF, which models the situation in which one wants to maximize the worst satisfaction ratio; then one wants to maximize the second worst satisfaction ratio, and so on.

b) Resolution algorithm.: Using methods presented in Section 1, the MMF routing problem can be solved sequentially by computing consecutive components of the demand satisfaction vector $\boldsymbol{t}$, which are decision variables of our problem. Intuitively, the main idea behind the resolution algorithm is that first the lowest value among the components of $\boldsymbol{t}$ has to be maximized before the second lowest value is maximized, and inductively, the maximization for a component is carried

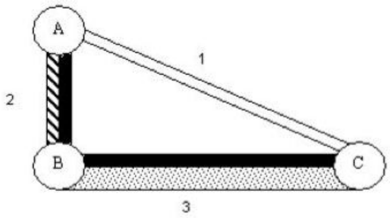

Fig. 4. Example of MMF/SAP application to network $\mathcal{N}_1$.

out after the components whose values are less good than the given value have been maximized. As before, a demand d whose satisfaction value t_d cannot be further increased, is called blocking. Then, the dual-based MMF Algorithm can be applied to the MMF routing problem as follows.

Algorithm 4 (MMF routing)

Input: Link capacities $C_e, e \in \mathcal{E}$, commodities $\mathcal{D}$.
Output: MMF routing $\boldsymbol{f}^*$ and its MMF demand satisfaction vector $\boldsymbol{t}^* = \boldsymbol{t}^B$.

- **Step 1:** Set $k = 0$, $B = \emptyset$ (empty set), and $\boldsymbol{t}^B = \emptyset$ (empty sequence).
- **Step 2:** If $B = \mathcal{D}$ then **stop** ($(\boldsymbol{f}^*, \boldsymbol{t}^B)$ is an optimal solution). Else, set $k := k+1$, solve $\mathcal{P}(B, \boldsymbol{t}^B)$ and obtain an optimal solution $(\boldsymbol{f}^*, \tau^*; \boldsymbol{\lambda}^*)$.
- **Step 3:** Put $\mathcal{D}_k := \{d \in B' : \lambda_d^* > 0\}$, $\boldsymbol{t}_d^B := \tau^*$, $d \in \mathcal{D}_k$, $B := B \cup \mathcal{D}_k$. Go to Step 2.

Above, $\mathcal{D}_k$ denotes the set of blocking demands identified in step k, and, as before, B—the set of all blocking demands detected so far: $B = \bigcup_{1 \le j < k} \mathcal{D}_j$.

c) Formulating and solving routing sub-problem $\mathcal{P}(B, \boldsymbol{t}^B)$*:* .
The sub-problem $\mathcal{P}(B, \boldsymbol{t}^B)$ is as in (19), where $B' = \mathcal{D} \setminus B$ denotes the set complementary to B, constraints (19b) are capacity constraints, and constraints (19c)–(19e) are appropriate flow conservation constraints. At the end of each step we obtain a feasible multi-commodity flow $\boldsymbol{f}^*$, satisfaction ratio τ^*, and optimal dual variables $\boldsymbol{\lambda}^* = (\lambda_d : d \in B')$ associated with constraints (19e).

d) Remarks.: Certainly, as in the general case, after solving problem $\mathcal{P}(B, \boldsymbol{t}^B)$ (19) we do not know which demands out of those who have attained the value τ^* at the end of an iteration (say, iteration number k) are non-blocking, i.e., for which of them we will be able to increase the satisfaction values t_d above τ^* in the next iterations. However, if at each iteration we can achieve a strictly complementary solution, then we can easily identify all blocking demands associated with iteration k since the strict complementary slackness means that $\lambda_d^* = 0$ if, and only if, demand $d \in B'$ is non-blocking (at least for the next iteration). Such a complementary slackness solution can be obtained by means of interior-point methods (IPM) of linear programming [13]. In fact, as explained in Chapter 8 of [2], these methods are basically able to identify non-blocking demands in an even simpler way. An IPM method can produce a so called *analytical center* of the set of all optimal solutions to problem $\mathcal{P}(B, \boldsymbol{t}^B)$.

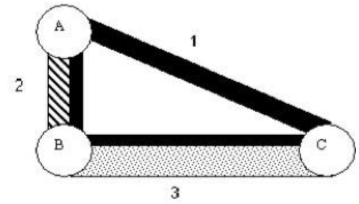

Fig. 5. The resulting resource sharing for MMF routing.

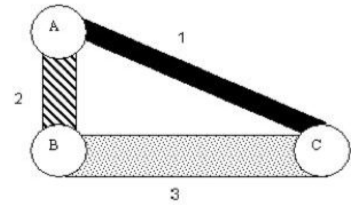

Fig. 6. The resulting routing when maximizing the overall throughput.

Such a solution $(\boldsymbol{f}^*, \boldsymbol{t}^*, \tau^*)$ is a convex combination of all optimal vertices with strictly positive coefficients in the convex combination in question. Certainly, in such a solution $t_d^* > \tau^*$ if, and only if, demand $d \in \mathcal{D}_k$ is non-blocking.

We also mention that frequently the demand satisfaction ratios are bounded by some natural requirements such as the minimal requested bandwidth $\hat{h}_d$ or the maximal bandwidth $\hat{H}_d$ that can be "consumed" by demand $d \in \mathcal{D}$. This leads to an additional constraint

$$\hat{h}_d \le t_d h_d \le \hat{H}_d, \quad d \in \mathcal{D} \tag{20}$$

that has to be included in the problem formulation. Since this constraint is linear it does not affect the resolution algorithms.

2) An example:

Consider the following simple example of a network ($\mathcal{N}_1$) consisting of 3 routers A, B, C and 3 links AB, BC and AC with capacities set respectively to 2MB/sec, 3MB/sec and 1MB/sec. Assume there are 3 connections: $A \to B, B \to C$ and $A \to C$. Let us examine what happens with the MMF flow outcome vector when applying respectively, MMF/SAP algorithm for a given routing (Figure 4), MMF routing algorithm (Figure 5) and maximal flow routing (i.e., routing that maximizes the overall throughput, Figure 6). First, following the example given on MMF/SAP in Section 1, let suppose that the above three connections are routed respectively through $A-B$, $B-C$ and $A-B-C$. The resource sharing obtained by Algorithm 1 is drawn in Figure 4 and reported in Table I (column "Given Routing").

In Figure 5 we show how the routing and the resource sharing is realized when MMF routing algorithm is applied. In the first iteration we obtain $\mathcal{D}_1 = \{A \to B, A \to C\}$ with 1.5. In the second iteration we have $\mathcal{D}_2 = \{B \to C\}$ and flow value equal to 2.5. With respect to network $\mathcal{N}_1$ and the set of given commodities $\mathcal{D}$, the maximal flow is obtained through routing each demand over its corresponding link and using all the available capacity, that is $(1+2+3 = 6)$ (Figure 6). Thus, the total MMF throughput is 5.5 MB/sec, less than maximal throughput, still sharing of bandwidth as well as bandwidth distribution to demands are substantially more fair. Table 1 shows in detail the routing and throughput for each connection

$$
\begin{aligned}
&maximize && \tau && && \text{(19a)}\\
&subject\ to && \textstyle\sum_{d\in\mathcal{D}} f_{ed} \le C_e, && e\in\mathcal{E} && \text{(19b)}\\
& && \textstyle\sum_{e\in\delta^+(v)} f_{ed} - \sum_{e\in\delta^-(v)} f_{ed} = 0 && d\in\mathcal{D}, v\notin\{o(d),s(d)\} && \text{(19c)}\\
& && \textstyle\sum_{e\in\delta^+(o(d))} f_{ed} - \sum_{e\in\delta^-(o(d))} f_{ed} = t_d h_d && d\in B && \text{(19d)}\\
& && \textstyle\sum_{e\in\delta^+(o(d))} f_{ed} - \sum_{e\in\delta^-(o(d))} f_{ed} \ge \tau h_d && d\in B' && \text{(19e)}\\
& && f_{ed}\ge 0 && e\in\mathcal{E}, d\in\mathcal{D}, && \text{(19f)}
\end{aligned}
$$

TABLE I
COMPARISON OF RESOURCE SHARING FOR THREE ROUTING STRATEGIES.

	Given Routing		MMF Routing		Maximal Flow Routing	
Connection	Path	Flow	Path	Flow	Path	Flow
A → B	A-B	1	A-B	1.5	A-B	2
A → C	A-B-C	1	A-B-C + A-C	1.5	A-C	1
B → C	B-C	2	B-C	2.5	B-C	3
Total		4		5.5		6

obtained respectively by MMF/SAP (described in Section 1.1), MMF, and maximal flow routing.

3) Some theoretical properties: In this section we present a theoretical study and highlight certain properties of MMF routing and the MMF demand satisfaction vector. At this stage it should be clear that:

Proposition 4. *The routing solution obtained at the end of the MMF routing algorithm is max-min fair and it can be obtained in polynomial time.*

We can go further and state that any MMF routing solution yields not only MMF allocation of end-to-end rates of connections, but also ensures that the share's link vector is MMF (see [7] for a detailed proof):

Proposition 5. *A feasible routing solution is MMF if, and only if, the max-min fair share's link vector is also MMF with respect to all possible routing solutions.*

In the sequel we will exhibit some other properties related to MMF routing. Let us consider the set of routings satisfying all constraints associated with problem $\mathcal{P}^k(B, \boldsymbol{t}^B)$, which is problem (19) considered in iteration k. Obviously, for each new optimal routing solution obtained at iteration k, there is at least one newly-saturated arc and demand. Then, we can state that all routing solutions obtained at the end of the MMF routing Algorithm saturate demands in the same unique order[4], (see [7] for a detailed proof).

With respect to the number of components of the MMF vector, an immediate result concerning the number of iterations of MMF Algorithm is:

Property 4. *The MMF routing Algorithm terminates in at most $min\{|\mathcal{D}|, |\mathcal{E}|\}$ steps.*

The last result can be improved when dealing with undirected networks [6].

[4]The demands contained in the same $\mathcal{D}_k$ are interchangeable in terms of the saturation order.

Proposition 6. *The MMF routing Algorithm terminates in at most $min\{|\mathcal{D}|, |\mathcal{E}|, |\mathcal{V}|-1\}$ steps when considering undirected networks.*

Sketch of the proof: We first notice that the end nodes of blocking demands in set $\mathcal{D}_1$ are placed in two disjoint complementary sets linked by saturated arcs, thus defining a cut. At the end of the first iteration, let $\mathcal{N}_1$ denote the subnetwork obtained from the initial network $\mathcal{N}$ by removing the saturated arcs. Similarly, let $\mathcal{N}_k$ denote the subnetwork obtained when removing from $\mathcal{N}_{k-1}$ the arcs that are newly-saturated at the end of iteration k. At the end iteration k, the terminal nodes of newly blocking demands are necessarily disconnected in subnetwork $\mathcal{N}_k$, and the number of components in subnetwork $\mathcal{N}_k$ is strictly superior to that of subnetwork $\mathcal{N}_{k-1}$. This is because these demands were non-blocking before executing the most recent iteration, and consequently the respective end nodes were in the same component. This component cannot therefore remain connected in the new subnetwork $\mathcal{N}_k$, and consequently the number of components becomes strictly larger than that obtained at the end of the preceding iteration. So after each iteration, the number of components of the obtained subnetwork is incremented by at least 1. Since at the end of the first iteration the number of components is at least 2, it follows that after the next $k-1$ consecutive iterations the number of components becomes greater or equal to $k+1$. Combining this with the fact that a connected network cannot be decomposed into more than $|\mathcal{V}|$ components, we see that there cannot be more than $|\mathcal{V}|-1$ decompositions of the network, i.e., iterations of the algorithm. □

4) Robustness of MMF routing: It is worth noticing that MMF routing is able to cope with a homogeneous traffic increase/decrease among all demands, and thus is particularly suitable for elastic traffic. Moreover, the MMF routing solution can provide a robust routing scheme even when traffic demands change in a general way and still have to be routed through a given network. This is a highly desirable feature as it is well known that traffic prediction is difficult and estimation errors are common in real-life situations, and that in general routing schemes do not cope well with overload traffic situations.

With respect to a MMF routing solution, we can show that variations in traffic within certain bounds are completely absorbed (provided that routing is feasible). As an illustration, let us suppose that the traffic volumes for all demands $d \in \mathcal{D}$ are increased with a given ratio r_d with respect to the nominal traffic provision (h_d). We can claim that as long as $r_d \leq t_d - 1$, for all $d \in \mathcal{D}$, the computed routing pattern will remain feasible and all demands will be entirely routed. It is then easy to prove, following Proposition 1, that the overload ratio vector given by $\{t_d - 1, d \in \mathcal{D}\}$ is also MMF.

Property 5. *The overload ratio vector associated with an MMF routing solution is also MMF.*

In other words, any MMF routing solution guarantees for each demand a certain ratio of overload, such that it is not possible to do better without decreasing the guaranteed overload ratio of other traffic demands with lower values.

C. Lexicographically load-balanced networks

We shall now consider a problem of load-balancing in a given network, as specified in [28]). We aim at distributing the demand traffic (load) fairly among the network links while satisfying a given set of traffic constraints. More precisely, we aim not only at minimizing the maximal load among links, but also to minimize lexicographically the sorted (non-increasing) load values of network links. In contrast to the MMF/OP problem (see Definition 3 in Section 1.2), the problem considered below concerns fairness in the min-max sense (not max-min). The problem arises in communication networks when the operator needs to define routing with respect to a given traffic demand matrix such that the network load is fairly distributed among the network links. Some relevant work on balanced networks is presented in [28] where the authors propose a strongly polynomial approach for lexicographically-optimal balanced networks. The problem they address is to allocate bandwidth between two endpoints of a backbone network so that the network is equitably loaded, and therefore it is limited to the case of single-commodity network. How to generalize this result to multi-commodity networks is already shown in [14], but we discuss here more general link load functions (especially non-linear, frequently used in telecommunications), and show that each of them can be reduced to the linear case.

1) Link load functions: An arc *load function* $Y_f : \mathcal{E} \rightarrow \mathbb{R}^+$ (linear or not) gives the load associated with each arc for a given flow vector $\boldsymbol{f}$. Such a load function (to be minimized) can for example be expressed (linearly) as $Y_f(e) = \frac{\hat{f}_e}{C_e}$, where $\hat{f}_e = \sum_{d \in \mathcal{D}} f_{ed}$ is the link load induced by given network flow $\boldsymbol{f}$ and C_e is the capacity of link e. Another useful load function (to be maximized) is the residual (unused) capacity $Y_f(e) = C_e - \hat{f}_e$. Global network resources optimization can also involve non-linear link load functions. We will consider two such functions, namely $Y_f(e) = (\alpha - 1)^{-1}(1 - \frac{\hat{f}_e}{C_e})^{1-\alpha}$ and $Y_f(e) = (\alpha - 1)^{-1}(C_e - \hat{f}_e)^{1-\alpha}$, for some $\alpha \in \mathbb{R}^+ \setminus \{1\}$ (see [37], [38]). In the remainder we will show that for both above two non-linear cases it is possible to use one of the two previous linear examples to obtain optimal solutions [6].

a) Link load vector and computation framework.: Similarly as for the demand satisfaction vector, one can define the link load vector with respect to a given routing and a load function.

Definition 7. *Given network flow $\boldsymbol{f}$ and load function Y_f, the vector $\boldsymbol{y} = (y_e : e \in \mathcal{E})$ whose components are the load values associated arcs (i.e., $y_e = Y_f(e)$), is called the link load vector of flow $\boldsymbol{f}$.*

A link load vector is called *feasible* if there exists a feasible routing with this load vector. Let $\mathcal{F}$ denote the set of all feasible flow vectors $\boldsymbol{f}$ in network $\mathcal{N}$ with respect to a given set of traffic demands $\mathcal{D}$. We use the name *lexicographically load-balanced network problem* to refer to the problem of computing a feasible routing such that its link load vector is min-max fair in the set of all feasible load vectors with respect to $\mathcal{F}$. Now, the lexicographically load-balanced network problem with respect to link load function Y_f can be stated as: *given a capacitated network $N = (\mathcal{V}, \mathcal{E})$ and a set of demands $\mathcal{D}$ with traffic volumes $\boldsymbol{h} = (h_d : d \in \mathcal{D})$, find the min-max fair link load vector $\boldsymbol{y}$ and the corresponding feasible routing $\boldsymbol{f}$.*

Computing min-max fair vectors $\boldsymbol{y}$ can be achieved by algorithms similar to those given in Section 1, except that instead of maximizing, one has to iteratively minimize the components of the load vector $\boldsymbol{y}$ sorted in the non-increasing order. This problem has been analyzed in [14] where a similar approach is given and a connection between the two algorithms is established. To resolve the lexicographically load-balanced network problem, we use an algorithm quite similar to the MMF routing Algorithm from Section 2.2.1. At each step we resolve an LP problem similar to $\mathcal{P}(B, \boldsymbol{t}^B)$ used in the MMF routing Algorithm. A link whose load has already been minimized, is called *minimally loaded*. The algorithm terminates when all links are minimally loaded.

Let us consider the (relative) load function $Y_f(e) = \frac{\hat{f}_e}{C_e}$ that we need to minimize. The corresponding problem, called $\mathcal{P}(B, \boldsymbol{y}^B)$, can be then stated as (21), where (21d) and (21e) are load constraints. Similarly to notation used for the previous algorithms, at iteration k the set B is the set of links minimally loaded in the previous $k-1$ iterations of the algorithm. Also, $B' = \mathcal{E} \setminus B$ is the set complementary to B. To complete current iteration, we need to find all links minimally loaded to τ by identifying the blocking indices in (21d). Then, we set $y_e = \tau$ for all the currently minimally loaded links. According to Proposition 3, one can deduce that there is at least one such minimally loaded link at the end of each iteration. Furthermore, one can show that there are at most $|\mathcal{V}| - 1$ distinct load values in the obtained load vector.

b) Non-linear link load functions.: In practice load functions employed by network operators are usually non-linear. A well-known load function, called the Kleinrock function, is given by $\frac{\hat{f}_e}{C_e - \hat{f}_e}$. It can be directly proved that any routing achieving min-max fairness for the relative load function (i.e., $\frac{\hat{f}_e}{C_e}$) achieves also min-max fair load for the Kleinrock function [14].

$$
\begin{aligned}
&\textit{minimize} && \tau && && \text{(21a)}\\
&\textit{subject to} && \textstyle\sum_{d\in\mathcal{D}} f_{ed} \le C_e, && e\in\mathcal{E} && \text{(21b)}\\
& && \textstyle\sum_{e\in\delta^+(v)} f_{ed} - \sum_{e\in\delta^-(v)} f_{ed} = 0 && d\in\mathcal{D}, v\notin\{o(d), s(d)\} && \text{(21c)}\\
& && \textstyle\sum_{d\in\mathcal{D}} f_{ed} = y_e C_e && e\in B && \text{(21d)}\\
& && \textstyle\sum_{d\in\mathcal{D}} f_{ed} \le \tau C_e && e\in B' && \text{(21e)}\\
& && f_{ed}\ge 0 && e\in\mathcal{E}, d\in\mathcal{D} && \text{(21f)}
\end{aligned}
$$

Proposition 7. *Any flow vector* $\boldsymbol{f}$ *that achieves min-max fair load for the linear load function* $Y_f(e) = \frac{\hat{f}_e}{C_e}$ *also achieves min-max fair load for the Kleinrock load function* $Y_f(e) = \frac{\hat{f}_e}{C_e - \hat{f}_e}$*, and vice-versa.*

Sketch of the proof: Let us first show that for a given flow $\boldsymbol{f}$ the order of links sorted according the non-increasing values of their load is the same for both load functions. Let us consider two distinct links, e_1 and e_2, say. Then, we have: $\hat{f}_{e_1}/\,C_{e_1} \le \hat{f}_{e_2}/\,C_{e_2} \Leftrightarrow C_{e_1}/\,\hat{f}_{e_1} \ge C_{e_2}/\,\hat{f}_{e_2} \Leftrightarrow (C_{e_1}/\,\hat{f}_{e_1}) - 1 \ge (C_{e_2}/\,\hat{f}_{e_2}) - 1$.

Consequently, we obtain: $(C_{e_1} - \hat{f}_{e_1})/\,\hat{f}_{e_1} \ge (C_{e_2} - \hat{f}_{e_2})/\,\hat{f}_{e_2} \Leftrightarrow \hat{f}_{e_1}/\,(C_{e_1} - \hat{f}_{e_1}) \le \hat{f}_{e_2}/\,(C_{e_2} - \hat{f}_{e_2})$. We observe also that the set of feasible flows $\mathcal{F}$ is the same for both load functions. As the lexicographical order with respect to criteria $\frac{\hat{f}_e}{C_e}$ is also preserved for criteria $\frac{\hat{f}_e}{(C_e - \hat{f}_e)}$, and vice-versa, one can deduce the thesis of the proposition. □

Instead of considering different link load functions separately, we propose a more general way to deal with the most frequently used link load functions. Let us consider the general link load function: $Y_f(e) = (\alpha-1)^{-1}(1 - \frac{\hat{f}_e}{C_e})^{1-\alpha}$. We observe that the result of Proposition 1 given in Section 1.2 also holds when replacing "max-min fair" with "min-max fair". Then, we obtain the following corollary which is easy to verify since the above function can be expressed as increasing function of $\frac{\hat{f}_e}{C_e}$.

Corollary 1. *Let* $\alpha \in R^+\backslash\{1\}$*. Any flow* $\boldsymbol{f}$ *that achieves min-max fair load for the linear load function* $\frac{\hat{f}_e}{C_e}$ *also achieves min-max fair load for the generalized link load function* $(\alpha - 1)^{-1}(1 - \hat{f}_e/C_e)^{1-\alpha}$*, and vice-versa.*

The second general link load function is $Y_f(e) = (\alpha - 1)^{-1}(C_e - \hat{f}_e)^{1-\alpha}$ $\alpha \in R^+ \setminus \{1\}$. It can easily be seen that the above function is a decreasing function of $(C_e - \hat{f}_e)$. Since $(C_e - \hat{f}_e)$ expresses the residual capacity of the link, it can be seen as an "unusual" load vector, which needs to be maximized lexicographically. Then, the approach is analogous, and the problem $\mathcal{P}(B, \boldsymbol{y}^B)$ can be formulated similarly to above, except changing the objective function to maximize and replacing load constraints (21d) and (21e) by (22a) and (22b), respectively:

$$C_e - \sum_{d\in D} f_{ed} \ge \tau \qquad e \in B' \tag{22a}$$

$$C_e - \sum_{d\in D} f_{ed} = y_e \qquad e \in B. \tag{22b}$$

Now we can use a similar trick as for Corollary 1 based on the following result.

Let function $\phi : \mathbb{R} \to \mathbb{R}$ be decreasing, and let $g'_j = \phi \circ g_j, j = 1, 2, ..., m$. The following proposition holds [6].

Proposition 8. *Some* $\boldsymbol{x}^0 \in X$ *is MMF on set* X *with respect to criteria* $\boldsymbol{g}$ *if, and only if,* $\boldsymbol{x}^0$ *is min-max fair on* X *with respect to* $\boldsymbol{g}'$*.*

Finally, we can conclude with the following corollary which solves the case of the second general link load function.

Corollary 2. *Let* $\alpha \in R^+\backslash\{1\}$*. Any flow vector* $\boldsymbol{f}$ *whose load vector* $\boldsymbol{y}$ *is MMF for the linear load function* $Y_f(e) = C_e - \hat{f}_e$ *achieves also min-max fair load for the generalized link load function* $Y_f(e) = (\alpha-1)^{-1}(C_e - \hat{f}_e)^{1-\alpha}$*, and vice-versa.*

III. Concluding remarks

The presented tutorial is intended to help the reader become familiar with the notion of MMF together with the underlying theory, and to give an introductory description of this topic in the context of multi-commodity flow networks. We have discussed relations of MMF and lexicographic optimization, and developed some general algorithms for the convex case of a basic MMF optimization problem. A particular emphasis has been put on the linear case and its applications to such topics in communication network design as routing and load-balancing. Several theoretical and practical properties have been studied in detail and some examples have been presented to illustrate MMF resolution algorithms, and routing and load-balancing applications.

As the notion of MMF becomes more and more useful in the context of communication networks, we think that the presented tutorial is right in time to present the basics of MMF which are in general not commonly known.

Intentionally, the presentation has been limited to the convex/linear case of a general MMF problem and its resolution algorithms, and to the two selected multi-commodity flow network applications (i.e., routing and load-balancing) relevant for telecommunications. Certainly, the notion of MMF can be studied in a more general context, in particular in the non-convex case related to mixed-integer programming and its applications to more sophisticated communication network applications such as single-path routing, modular flows, modular link capacity, etc. These problems are not commonly known and will be a subject of our future tutorial. In the meantime, the reader is referred to [4] and [39].

ACKNOWLEDGMENTS

The first author wishes to thank L-N. Doan, J. B. Orlin, E. Gourdin, B. Liau, O. Klopfenstein, and A. Bashllari for their precious collaboration and helpful discussions on different topics in MMF. The second author expresses his gratitude for enlightening discussions to W. Ogryczak, an outstanding expert in MMF, to Artur Tomaszewski, and to his PhD students E. Kubilinskas and P. Nilsson.

REFERENCES

[1] D. Bertsekas and R. Gallager. *Data Networks*. Prentice-Hall, Engelwood Cliffs, N.J., 1992.

[2] M. Pioro and D. Medhi. *Routing, Flow and Capacity Design in Communication and Computer Networks*, Morgan Kaufmann Publishers (2004).

[3] J. Kleinberg, Y. Rabani, and E. Tardos. Fairness in routing and load balancing. In *Proc. 35th Annual Symposium on Foundations of Computer Science*, 1999.

[4] W. Ogryczak, M. Pioro and A. Tomaszewski. Telecommunications Network Design and Max-Min Optimization Problem. *J. Telecommunications and Information Technology* 3/2005, pp. 1-14.

[5] B. Radunovic, J.-Y. Le Boudec. A Unified Framework for Max-Min and Min-Max Fairness with Applications. *Proc. 40th Annual Allerton Conference on Communication, Control, and Computing*, Allerton, IL, October 2002.

[6] D. Nace. L.N. Doan, O. Klopfenstein, A. Bashllari. Max-min fairness in multi-commodity flows. *Computers and OR*, Volume 35, Issue 2, February 2008, pp. 557-573, published online in ScienceDirect, doi:10.1016/j.cor.2006.03.020.

[7] D. Nace, L.N. Doan, Eric Gourdin and Bernard Liau. Computing optimal max-min fair resource allocation for elastic flows *IEEE Trans. Networking*, Volume: 14, pp. 1272-1281, Dec. 2006.

[8] F.A. Behringer. A Simplex based algorithm for the lexicographically extended linear maxmin problem. *European Journal of Operational Research* 7 (1981) 274-283.

[9] F.A. Behringer. Linear multiobjective maxmin optimization and some Pareto and lexmaxmin extensions. *OR Spektrum* 8 (1986) pp. 25-32.

[10] R.S. Klein, H. Luss, D.R. Smith. A lexicographic minimax algorithm for multiperiod resource allocation. *Mathematical Programming* 55 (1992) pp. 213-234.

[11] M. Pioro, P. Nilsson, E. Kubilinskas and G. Fodor. On Efficient Max-min Fair Routing Algorithms. *In Proc. ISCC 2003*, Antalya, Turkey, July 2003.

[12] R. M. Freund, R. Roundy, and M.J. Todd. Identifying the Set of Always-Active Constraints in a System of Linear Inequalities by a Single Linear Program. *Working paper 1674-85*, Massachusetts Institute of Technology (MIT), Sloan School of Management, 1985.

[13] R. M. Freund and S. Mizuno. Interior Point Methods: Current Status and Future Directions. In *High Performance Optimization*, H. Frenk et al. (eds.), Kluwer Academic Publishers, pp. 441-466, 2000.

[14] D. Nace, J. B. Orlin. Lexicographically Minimum and Maximum Load Linear Programming Problems. *Operations Research*, Vol. 55, Issue 1, pages 182-187, jan-feb 2007.

[15] F. P. Kelly, A. Maulloo, and D. Tan. Rate control in communication networks: shadow prices, proportional fairness and stability. *J. Operational Research Society*, vol. 49, pp. 237-252, 1998.

[16] D. Chiu and R. Jain. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. *Computer Networks and ISDN Systems*, 17:1–14, 1989.

[17] A. Charny, D. D. Clark and R. Jain. Congestion control with explicit rate indication. In *Proc. IEEE International Conference on Communications*, 1995.

[18] L. Massoulié and J.W. Roberts. Bandwidth sharing: objectives and algorithms. In *Proc. IEEE INFOCOM'99*, 1999.

[19] T. Bonald and L. Massoulié. Impact of fairness on Internet performance. In *Proc. of ACM SIGMETRICS 2001*, 2001.

[20] S. Floyd, K. Fall. Promoting the use of end-to-end congestion control in the Internet. *IEEE/ACM Trans. Networking* 7(4), August 1999.

[21] S. Floyd. Congestion control principle. *IETF draft ¡ draft-floyd-cong-02.txt ¿* March 2000.

[22] V. Jacobson, M. J. Karels Congestion Avoidance and Control In *ACM Computer Communication Review; Proceedings of the Sigcomm'88 Symposium* in Stanford, CA, 1988.

[23] M. Vojvonovic, J.-Y. Le Boudec and C. Boutremans. Global Fairness of additive-increase and multiplicative-decrease with heterogeneous round trip-times. In *Proc. IEEE Infocom'00*, 2000.

[24] J. Y. Le Boudec Rate adaptation, congestion control and fairness: a tutorial. *http://ica1www.epfl.ch/PS_files/LEB3132.pdf*, February 2005.

[25] N. Megiddo. Optimal flows in networks with sources and sinks. *Mathematical Programming*, 7, 1974.

[26] G. Fodor, G. Malicsko, M. Pioro and T. Szymanski. Path Optimization for Elastic Traffic under Fairness Constraints. In *Proc. ITC* 2001.

[27] J. Galtier. Semi-definite programming as a simple extension to linear programming: convex optimization with queueing, equity and other telecom functionals. In *Proc. AlgoTel 2001*. INRIA, 2001.

[28] L. Georgiadis, P. Georgatsos, K. Floros, S. Sartzetakis Lexicographically Optimal Balanced Networks. In *Proc. Infocom'01*, pp. 689-698, 2001.

[29] S. Boulahia-Oueslati. *Qualité de service et routage des flots élastiques dans un réseau multiservice*. PhD thesis, ENST, Paris, France, November 2000.

[30] S. Chen and K. Nahrstedt. Maxmin fair routing in connection-oriented networks. In *Proc. Euro-Parallel and Distributed Systems Conference, pp. 163-168, Vienna, Austria,* July, 1998.

[31] Q. Ma. *Quality-of-Service Routing in Integrated Service Networks*. PhD thesis, Carnegie Mellon Univ., Pittsburgh, USA, Jan. 1998.

[32] P. Nilsson, M. Pioro, Z. Dziong. Link Protection within an Existing Backbone Network *In Proc. INOC 2003, Evry, France, October 2003*, pp. 435-441.

[33] R.K. Ahuja, T.L. Magnanti and J.B. Orlin. *Network Flows: Theory, Algorithms and Applications*. Prentice Hall, New Jersey, 1993.

[34] G. Desaulniers, J. Desrosiers, M. M. Solomon (eds.) *Column Generation*. Springer-Verlag, N.Y, June 2005.

[35] D. Bertsimas and J. N. Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific, 1997.

[36] F. Shahrokhi and D.W. Matula. The Maximum Concurrent Flow Problem. *in Journal of the ACM*, Vol. 37, pp. 318-334, 1990.

[37] W. Ben Ameur, N. Michel, E. Gourdin et B. Liau. Routing strategies for IP networks. *Telektronikk*, 2/3, pp. 145-158, 2001.

[38] L. Kleinrock. *Communications Nets: Stochastic Message Flow and Delay*, McGraw-Hill, New York, 1964.

[39] P. Nilsson *Fairness in communication and computer network design* PhD thesis, ISSN 1101-3931, ISRN LUTEDX/TETS-1080-SE-138P, Lund Institute of Technology, Lund, 2006.

Dritan Nace is a Professor at the University of Technology of Compiègne. He received the degrees of Mathematics in 1991 (University of Tirana, Albania), MSc (DEA) in Computer Science in 1993 and PhD in Computer Science in 1997, both from University of Technology of Compigne. From 1997 to 1998 he worked as research engineer at CNET (Centre National d'Etudes des Télécommunications), the France Telecom research center. From 1998, he is at the University of Technology of Compiègne, first as Assistant Professor, Associate Professor and since 2008 he holds a Full Professor position. His research interests include operations research and in particular network optimization and routing.

Michał Pióro is a Professor and Head of Department of Computer Networks and Switching at the Institute of Telecommunications, Warsaw University of Technology, Poland. At the same time he is a Full Professor at the Department of Electrical and Information Systems, Lund University, Sweden. He received a Ph.D. degree in telecommunications in 1979 and a D.Sc. degree in 1990, both from the Warsaw University of Technology. In 2002 he received a Polish State Professorship. His research interests concentrate on modeling, design, and performance evaluation of telecommunication systems. He is an author of four books (among them *Routing, Flow, and Capacity Design in Communication and Computer Networks*, Morgan Kaufmann Publishers (2004) co-written with Deepankar Medhi) and more than 150 technical papers presented in telecommunication journals and conference proceedings. He has lead many research projects for telecom industry in the field of networking.