On single-path network routing subject to max-min fair flow allocation

Edoardo Amaldi

Dipartimento di Elettronica, Informazione e Bioingegneria
Politecnico di Milano, Milano, Italy
amaldi@elet.polimi.it

Stefano Coniglio

Lehrstuhl II für Matematik
RWTH Aachen University, Aachen, Germany
coniglio@math2.rwth-aachen.de

Luca G. Gianoli, Can Umut Ileri

Dipartimento di Elettronica, Informazione e Bioingegneria
Politecnico di Milano, Milano, Italy
gianoli@elet.polimi.it, ileri@mail.polimi.it

Abstract

Fair allocation of flows in multicommodity networks has been attracting a growing attention. In Max-Min Fair (MMF) flow allocation, not only the flow of the commodity with the smallest allocation is maximized but also, in turn, the second smallest, the third smallest, and so on. Since the MMF paradigm allows to approximate the TCP flow allocation when the routing paths are given and the flows are elastic, we address the network routing problem where, given a graph with arc capacities and a set of origin-destination pairs with unknown demands, we must route each commodity over a single path so as to maximize the throughput, subject
to the constraint that the flows are allocated according to the MMF principle. After discussing two properties of the problem, we describe a column generation based heuristic and report some computational results.

**Keywords:** Max-min fairness, network routing, integer programming, column generation

1 **Introduction**

Telecommunication multicommodity flow problems involving the concept of max-min fairness have been attracting a growing attention (see, e.g., [1] and the references therein).

Consider the task of sharing the network capacity (bandwidth) among \( n \) commodities (users). Let \( \phi \in \mathbb{R}^n \) denote a flow vector where the \( i \)-th component \( \phi_i \) corresponds to the flow allocated to the \( i \)-th commodity. Let \( \sigma \) be a sorting operator permuting the components of \( \phi \) in nondecreasing order, i.e., such that \( \sigma(\phi)_i \leq \sigma(\phi)_j \) whenever \( i < j \).

**Definition:** A flow vector \( \phi \in \mathbb{R}^n \) is *Max-Min Fair* (MMF) if, for any other flow vector \( \phi' \in \mathbb{R}^n \), \( \sigma(\phi) \) lexicographically dominates \( \sigma(\phi') \), i.e., either \( \sigma(\phi)_i = \sigma(\phi')_i \) or there exists an integer \( k \), with \( 1 \leq k \leq n \), such that \( \sigma(\phi)_k > \sigma(\phi')_k \) and \( \sigma(\phi)_l = \sigma(\phi')_l \) for all \( l < k \).

In other words, a flow vector is MMF if there is no way to increase the flow of any commodity without decreasing the flow of a commodity with a smaller or equal flow.

Let us briefly summarize previous work on network routing problems with MMF flow allocation. When the routing paths are given, a simple polynomial-time algorithm, the so-called *Water Filling* algorithm yields an MMF flow allocation [2]. When the routing paths are not known a priori, algorithms to find an MMF flow allocation have been proposed for unsplittable (single path) or splittable routing, see, e.g., [3] and the survey [1]. For the special case with a single source, see [4] for a polynomial-time algorithm which applies to splittable (fractional) flows, and [5] for an \( \mathcal{NP} \)-hardness result and approximation algorithms for unsplittable routing. The reader is referred to [1] for a survey dealing also with some generalizations.

To the best of our knowledge, in the literature the MMF paradigm has only been considered as the objective function. This is in contrast with the fact that network operators aim at maximizing a network utility function (e.g., throughput) and in IP networks, once the routing paths have been chosen by
the IP layer, the transport protocol (e.g., TCP) achieves an average bandwidth allocation which is approximated by MMF [6].

In this work, along the line of [7], we address the following problem:

Max-Throughput Single-Path Network Routing subject to MMF flow allocation (MT-SPNR-MMF): Given a directed graph $G = (V, A)$ with capacities $c_{ij}$ for each $(i, j) \in A$ and a set of commodities $K$ with origin-destination pairs $(s, t)$, route each commodity over a single path so as to maximize the throughput, subject to the constraint that the amount of flow allocated to the commodities be MMF w.r.t. all the flow allocations that are feasible for the paths that have been chosen.

As observed in [7], MT-SPNR-MMF is a bilevel problem where, at the upper level, the leader (network operator) selects a routing path for each commodity and, at the lower level, a follower (TCP protocol) allocates the flows (bandwidth) to the chosen paths according to the MMF paradigm.

2 Properties of MT-SPNR-MMF

Let us start with two simple properties of the problem.

**Proposition 2.1** MT-SPNR-MMF is $\mathcal{NP}$-hard even when $c_{ij} = 1$, $\forall (i, j) \in A$.

**Proof.** By straightforward reduction from the $\mathcal{NP}$-complete problem of deciding whether a given digraph $G$ with $k$ origin-destination pairs admits $k$ edge-disjoint paths. It suffices to observe that $G$ admits $k$ edge-disjoint paths if and only if the MT-SPNR-MMF instance with the same graph $G$, the same $k$ origin-destination pairs, and all $c_{ij} = 1$ admits an MMF flow allocation with a total throughput of value $k$. If it is the case, since each commodity achieves the maximum flow of value 1, $k$ edge-disjoint paths are used. $\square$

Consider the “classical” Single-Level MMF Single-Path Network Routing problem (SL-MMF-SPNR) where we look for a solution whose flow vector is lexicographically undominated w.r.t. all the possible choices of flows and routing paths.

**Proposition 2.2** The gap in terms of throughput or smallest flow allocation between an optimal solution to MT-SPNR-MMF and an optimal solution to SL-MMF-SPNR can be arbitrarily large.

**Proof.** Consider the example in the figure below, with $k$ commodities, capacity 1 for the arcs $(a_i, b_i), i = 2, \ldots, k$, capacity $k(1 + \delta)$ for the arc $(a_1, b_1)$,
with $\delta > 0$, and an arbitrarily large capacity for the other arcs. It is easy to verify that $\sigma(\phi) = (1, \ldots, 1, k(1 + \delta))$, with a total throughput of $k(1 + \delta) + k - 1$, is an optimal solution to MT-SPNR-MMF, while $\sigma(\phi) = (1 + \delta, \ldots, 1 + \delta)$, with a total throughput of $k(1 + \delta)$, is an optimal solution to SL-MMF-SPNR.

For a fixed $\delta$ and an arbitrarily large $k$, these solutions differ by an additive factor of $k - 1$ in terms of throughput. For a fixed $k$ and an arbitrarily large $\delta$, these solutions differ by an additive factor of $\delta$ in terms of smallest flow allocation.

\[ \square \]

3 Column generation

In this section, we describe a column generation method for MT-SPNR-MMF based on a mixed integer programming (MILP) path formulation. For an arc formulation, see \cite{7}.

Let $P_{st}$ be the set of feasible paths for each commodity $(s, t) \in K$ and let the constant $\sigma_{ij}^{pst}$ be 1 if the path $p \in P_{st}$ contains the arc $(i, j) \in A$ and 0 otherwise. Let the variable $\lambda_{ij}^{p} \in \{0, 1\}$ be 1 if the path of index $p$ is chosen for the commodity $(s, t)$ and 0 otherwise. For each commodity $(s, t) \in K$, let $f_{ij}^{st}$ be the amount of flow assigned to it and let $f_{st}$ be its amount on arc $(i, j) \in A$. Let $u_{ij}$ be an upper bound on any flow over the arc $(i, j) \in A$. The variables $y_{ij}^{st}$, which are instrumental in the MMF constraints, are described after the formulation. Then, MT-SNPR-MMF can be formulated as follows:

\[
\text{max} \sum_{(s,t) \in K} \phi_{st} \\
\text{s.t.} \sum_{p \in P_{st}} \lambda_{ij}^{p} = 1 \quad \forall (s, t) \in K \tag{1}
\]

\[
\sum_{(i,j) \in \delta^+(i)} f_{ij}^{st} - \sum_{(j,i) \in \delta^-(i)} f_{ji}^{st} = \begin{cases} 
\phi_{st} & \text{if } i = s \\
-\phi_{st} & \text{if } i = t \\
0 & \text{else} \end{cases} \quad \forall (s, t) \in K, i \in V \tag{2}
\]

\[
\sum_{(s,t) \in K} f_{ij}^{st} \leq c_{ij} \quad \forall (i, j) \in A \tag{3}
\]

\[
f_{ij}^{st} - c_{ij} \sum_{p \in P_{st} \lambda_{ij}^{p}} \leq 0 \quad \forall (i, j) \in A, (s, t) \in K \tag{4}
\]

\[
\phi_{st} \geq \frac{\min_{(i,j) \in A \{c_{ij}\}}}{|K|} \quad \forall (s, t) \in K \tag{5}
\]
Constraints (3)-(4) are standard flow conservation and capacity constraints. Constraints (5) guarantee $f_{ij}^{st} = 0$ if no path $p \in P^{st}$ with $\sigma_{ij}^{p} = 1$ is selected. Constraints (6) impose a valid lower bound on the flow values, achieved when all the flows are routed over the link with minimum capacity.

Constraints (7)-(10), which are expressed in a slightly different w.r.t. [8], are the MMF constraints imposing that the flow vector be MMF for the selected paths. Let us briefly explain why they are sufficient to guarantee an MMF flow allocation. For a given set of paths, an MMF bandwidth allocation can be uniquely determined via the Water Filling (WF) algorithm. Starting from $\phi^{st} = 0$ for all $(s, t) \in K$, WF simultaneously increases all the flows until one or more arcs are saturated (we refer to them as to bottleneck arcs), remains them and the saturating commodities, updates the capacities to their residual values, and iterates until no commodity or arcs are left. For each arc $(i, j) \in A$ and commodity $(s, t) \in K$, we thus introduce the binary variable $y_{ij}^{st}$ which is equal to 1 if $(i, j)$ is a bottleneck arc for $(s, t)$, and to 0 otherwise. Due to the correctness of WF, $\phi$ is MMF for the given set of paths if and only if it satisfies (7)-(10). Constraints (7) ensure that we have at least a bottleneck arc for each $(s, t) \in K$, Constraints (8) guarantee that the bottleneck arcs are saturated, and Constraints (9)-(10) impose that the flow through a bottleneck arc $(i, j)$ for a pair $(s, t)$ be at least as large as the largest flow through $(i, j)$ for all the other commodities.

The restricted master problem is obtained from (1)-(11) by restricting $P^{st}$, for each $(s, t) \in K$, to the paths which have been generated so far. Its LP relaxation is obtained by dropping the integrality constraints on $y_{ij}^{st}$ and $\lambda_{ij}^{p}$. Let $w^{st} \in \mathbb{R}$ and $\pi_{ij}^{st} \geq 0$ be the dual variables associated to, respectively, Constraints (2) and (5). For any commodity $(s, t)$ and a corresponding path of index $p \in P^{st}$, the dual constraint corresponding to $\lambda_{ij}^{p}$ is $w^{st} - \sum_{(i,j) \in A} c_{ij} \sigma_{ij}^{p} \pi_{ij}^{st} \geq 0$. It is violated (or, equivalently, the corresponding column has a nonnegative reduced cost) if $\sum_{(i,j) \in A} c_{ij} \sigma_{ij}^{p} \pi_{ij}^{st} \geq w^{st}$. For each commodity $(s, t)$, the pricing subproblem thus amounts to finding a longest path on the original graph with weights $c_{ij} \pi_{ij}^{st}$ for each arc $(i, j) \in A$. This
can be done by solving a MILP which is obtained by adding to the standard LP formulation for the shortest path problem with nonnegative costs a modified version of the variables and constraints used to prevent subtours in the extended formulation for the TSP by Wong [9]. Let \( z_i \), for \( i \in V \setminus \{s, t\} \), be 1 if the path contains node \( i \) as an intermediate node and 0 otherwise. Let then \( V_s := V \setminus \{s\} \) and let \( q^k_{ij} \geq 0 \) be the variable of an auxiliary flow from node \( s \) to a node \( k \in V_s \). Then, the pricing subproblem can then be formulated as:

\[
\text{max } \sum_{(i,j) \in A} c_{ij}\sigma^{pst}_{ij} z_{ij} \\
\text{s.t. } \sum_{(i,j) \in \delta^+(i)} \sigma^{pst}_{ij} - \sum_{(j,i) \in \delta^-(i)} \sigma^{pst}_{ji} = \begin{cases} 1 & \text{if } i = s \\ -1 & \text{if } i = t \\ 0 & \text{otherwise} \end{cases} \forall i \in V (13)
\]

\[
\sum_{(i,j) \in \delta^+(i)} q^k_{ij} - \sum_{(j,i) \in \delta^-(i)} q^k_{ji} = \begin{cases} z_k & \text{if } i = s \\ -z_k & \text{if } i = k \\ 0 & \text{otherwise} \end{cases} \forall k \in V_s, i \in V (14)
\]

\[
q^k_{ij} \leq \sigma^{pst}_{ij} \forall (i,j) \in A, k \in V_s (15)
\]

\[
z_i \leq \sum_{(j,i) \in \delta^-(i)} \sigma^{pst}_{ji} \forall i \in V_s (16)
\]

\[
\sigma^{pst}_{ij} \in \{0, 1\} \forall (i,j) \in A (17)
\]

\[
q^k_{ij} \geq 0 \forall (i,j) \in A, k \in V_s (18)
\]

\[
z_i \in \{0, 1\} \forall i \in V (19)
\]

To accelerate our column generation, we initialize the column pool with 3 random paths per commodity and, to favor diversity between the paths, we proceed as follows. For each commodity, as long as edge-disjoint paths are found, we generate the new path by solving a shortest path problem w.r.t. uniformly random weights. Whenever a duplicate path is generated, we increase the weight of its arcs by a factor of 2, and then repeat the generation.

To speed up the convergence to an optimal solution of the relaxed master problem, we consider alternative techniques to the “text-book” generation of columns with a maximally reduced cost. Among various options, the best results are obtained with a pseudo-random technique where, in the pricing subproblem, we optimize a linear function with uniformly drawn random coefficients, subject to constraint that the reduced cost be at least \( 10^{-6} \).

Since, in general, the solution found with column generation is not integer, after the algorithm is halted, we simply impose the integrality on \( y^p_{st} \) and \( x^p_{st} \) and reoptimize the restricted master problem to obtain a lower bound. In the following, the resulting method is referred to as \textit{Column Generation based MILP heuristic}. 
4 Some computational results

We consider four network topologies taken from the SND library: polska (|V| = 12, |A| = 36), france (|V| = 25, |A| = 90), nobel-us (|V| = 14, |A| = 42), atlanta (|V| = 15, |A| = 44). From each topology, we generate 20 instances adopting 4 different generation techniques, with an increasing number of commodities and using either randomly generated capacities (10, 5, 2, and 1 Gbps with a probability of, respectively, 0.4, 0.3, 0.2, and 0.1) or uniform ones. For a detailed description, see the full version of the paper.

The computational experiments are carried out with CPLEX 12.3 using the AMPL modeling language on a machine equipped with 2 Intel Xeon E5645 CPUs with 2.40 GHz and 16 GB of RAM. When solving the MILP restricted master problem, we adopt a time limit of 3600 seconds. Whenever solving a MILP, we set the parameter mipemphasis=4.

Due to space restrictions, in the following table we only report the results for a subset of the instances.

<table>
<thead>
<tr>
<th>Network</th>
<th>Inst.</th>
<th>K</th>
<th>Complete formulation</th>
<th>Column generation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>atlanta 1-1</td>
<td>12</td>
<td>48.5</td>
<td>0.0</td>
<td>1.4</td>
</tr>
<tr>
<td>atlanta 1-2</td>
<td>20</td>
<td>62.5</td>
<td>0.0</td>
<td>16.2</td>
</tr>
<tr>
<td>atlanta 1-3</td>
<td>30</td>
<td>96.1</td>
<td>2.1</td>
<td>3593.9</td>
</tr>
<tr>
<td>atlanta 1-4</td>
<td>42</td>
<td>-</td>
<td>- 3594.1</td>
<td>76.0</td>
</tr>
<tr>
<td>atlanta 1-5</td>
<td>56</td>
<td>-</td>
<td>- 3594.1</td>
<td>87.3</td>
</tr>
<tr>
<td>france 1-2</td>
<td>10</td>
<td>52.5</td>
<td>0.0</td>
<td>58.3</td>
</tr>
<tr>
<td>france 1-3</td>
<td>15</td>
<td>56.5</td>
<td>0.0</td>
<td>482.6</td>
</tr>
<tr>
<td>france 1-4</td>
<td>21</td>
<td>76.5</td>
<td>0.9</td>
<td>3600.0</td>
</tr>
<tr>
<td>france 1-5</td>
<td>28</td>
<td>-</td>
<td>- 3600.0</td>
<td>114.8</td>
</tr>
<tr>
<td>france 1-6</td>
<td>36</td>
<td>-</td>
<td>- 3600.0</td>
<td>116.3</td>
</tr>
<tr>
<td>nobel-us 1-3</td>
<td>15</td>
<td>63.5</td>
<td>0.0</td>
<td>62.4</td>
</tr>
<tr>
<td>nobel-us 1-4</td>
<td>21</td>
<td>75.0</td>
<td>0.0</td>
<td>2627.0</td>
</tr>
<tr>
<td>nobel-us 1-5</td>
<td>28</td>
<td>104.5</td>
<td>2.4</td>
<td>3600.0</td>
</tr>
<tr>
<td>nobel-us 1-6</td>
<td>36</td>
<td>-</td>
<td>- 3600.0</td>
<td>91.0</td>
</tr>
<tr>
<td>nobel-us 1-7</td>
<td>42</td>
<td>-</td>
<td>- 3600.0</td>
<td>103.9</td>
</tr>
<tr>
<td>polska 1-3</td>
<td>21</td>
<td>90.0</td>
<td>0.0</td>
<td>20.0</td>
</tr>
<tr>
<td>polska 1-4</td>
<td>28</td>
<td>71.4</td>
<td>0.8</td>
<td>3594.1</td>
</tr>
<tr>
<td>polska 1-5</td>
<td>36</td>
<td>81.8</td>
<td>0.8</td>
<td>3593.9</td>
</tr>
<tr>
<td>polska 1-6</td>
<td>42</td>
<td>-</td>
<td>- 3600.0</td>
<td>118.3</td>
</tr>
<tr>
<td>polska 1-7</td>
<td>56</td>
<td>-</td>
<td>- 3600.0</td>
<td>151.6</td>
</tr>
</tbody>
</table>

The Column Generation based MILP heuristic clearly outperforms the complete arc formulation proposed in [7]. Indeed, in the time limit of 3600 seconds, with the complete formulation no feasible solution is found for 28 instances out of 80 (more than 25%), as opposed to only 4 when using the former one. If we compute the ratio, for each instance, between the value of the solution found with the two methods and then consider their geometric mean, with the column generation based method we have solutions that are worse
than those for the complete formulation by less than 0.01%. Most interestingly, if we compute the geometric mean of the ratios for the computing times, we observe that the column generation based method requires less than 15% the time that is needed when solving the complete formulation.

As to the column generation method without adopting the MILP heuristic, if we compute the gap between its upper bounds and the lower bounds corresponding to the best feasible solution found with either methods, we obtain a value of only 2.75%. Note that, for all the instances but two, the total time invested in the column generation procedure is much smaller than 10 seconds and, overall, it never exceeds 30 seconds. The quality of such bounds suggests that a Branch-and-Price algorithm, which is under development, may be an effective technique to solve the problem to optimality.

References


