On the Convex Piecewise Linear Unsplittable Multicommodity Flow Problem

Bernard Fortz
Département d'Informatique
Université Libre de Bruxelles
Boulevard du Triomphe, CP 210/01
1050 Bruxelles, Belgium
Email: bernard.fortz@ulb.ac.be

Luís Gouveia
Departamento de Estatística e Investigação Operacional
Faculdade de Ciências da Universidade de Lisboa
Bloco C / 2 - Campo Grande, Cidade Universitária
1700 Lisboa, Portugal
Email: legouveia@fc.ul.pt

Martim Joyce-Moniz
Département d'Informatique
Université Libre de Bruxelles
Boulevard du Triomphe, CP 210/01
1050 Bruxelles, Belgium
Email: martim.moniz@ulb.ac.be

***Abstract*—We consider the problem of finding the cheapest routing for a set of commodities over a directed graph, such that: *i)* each commodity flows through a single path, *ii)* the routing cost of each arc is given by a convex piecewise linear function of the load (*i.e.* the total flow) traversing it. We propose a new mixed-integer programming formulation for this problem. This formulation gives a complete description of the associated polyhedron for the single commodity case, and produces very tight linear programming bounds for the multi-commodity case.**

I. Introduction

In this paper, we study the convex piecewise linear unsplittable multicommodity flow (PUMF) problem. Let $G = (V, A)$ be a directed graph, with a set of nodes V, and a set of arcs A. Consider as well the set of commodities K, each $k \in K$ with a given origin o_k, destination d_k, and demand ρ_k. Each arc $a \in A$ has an associated cost function $g_a(l_a)$ of the load flowing through the arc l_a. This cost function is continuous, convex and piecewise linear, with the segments being represented by the finite set $S_a = \{1, 2, ..., |S_a|\}$. Each segment $s \in S_a$ has a lower and upper bound on the flow, represented by the breakpoints b_a^{s-1} and b_a^s. If finite, the breakpoint of the last segment of each arc $a \in A$, $b_a^{|S_a|}$, can be interpreted as the capacity of the arc. However, the case where $b_a^{|S_a|} = \infty$ also stands. A segment is also characterized by the a slope c_a^s and an intercept f_a^s. Here, we consider only the case where the cost functions are convex, so these values must be such that $c_a^1 > 0$, $c_a^s > c_a^{s-1}$ and $f_a^s \leq 0$, $f_a^s < f_a^{s-1}$. We also assume that for every arc $a \in A$, $g_a(0) = 0$, and consequently, $f_a^1 = 0$. The PUMF problem is to find a single path for each commodity, such that the sum of the costs associated to the load of the arcs is minimized.

The PUMF problem can be seen as a variant of the origin-destination integer multicommodity flow (ODIMCF) problem, which was introduced by Barnhart *et al.* [**?**]. The latter has also been referred to as the unsplittable multicommodity flow problem in [**?**]. In the ODIMCF problem, the costs are directly proportional to the load on the arcs. Moreover, there are explicit capacity constraints on the amount of flow traversing an arc. In the PUMF problem, it is the routing costs, given by the piecewise linear functions, that confine the load within the arcs' capacities, and at the same time, help to avoid unnecessary detours in the network. An example of such a cost function was proposed by Fortz and Thorup [**?**], [**?**] and is illustrated in Figure 2.

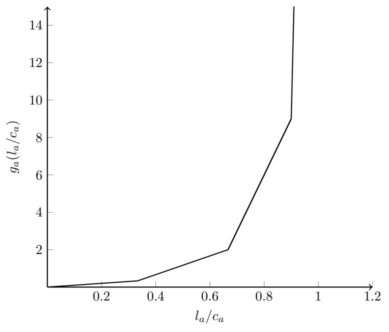

Fig. 1. Example of piecewise linear cost function

In this function, the cost of sending flow is cheap for arcs with a small utilization (ratio between the arc's load and its capacity). However, the price quickly rises when the utilization approaches the arc's capacity. Even though it is possible for the utilization to go above 100%, this is so heavily penalized, that such a solution will likely be avoided.

In [**?**], the authors model this cost function by defining variables $g_a, a \in A$, and imposing the following lower bounds, representing the segments of the piecewise linear function:

$$g_a \geq c_a^s l_a + f_a^s, \qquad a \in A, s \in S_a. \tag{1a}$$

$$g_a \geq 0, \qquad a \in A. \tag{1b}$$

Thus, for each arc $a \in A$, the variable g_a stands for its routing cost, and the objective is then to minimize $\sum_{a \in A} g_a$.

Such objective functions have been widely used in problems related with Traffic Engineering (TE) in internet networks. One of the most important and well known of such objectives is the Kleinrock delay function [**?**]:

$$F = \sum_{a \in A} \frac{l_a}{C_a - l_a}, \tag{2}$$

where C_a denotes the capacity of link $a \in A$. The Kleinrock function helps avoid congestion by penalizing heavily loaded links. Observe that this objective function is convex, and hence can be approximated with a convex piecewise linear function, leading to a linear optimization problem ([**?**], [**?**], [**?**]). Balon et al. [**?**] and Gourdin [**?**] discuss various TE objective functions. These authors evaluate how well different objective functions meet TE requirements, and conclude that piecewise linear objectives provide a good trade-off between different measures of quality of service.

Recently, such an objective function was also used by Papadimitriou and Fortz [**?**], [**?**] in the context of a complex multi-period design and routing problem. Lower bounds resulting from the linear programming (LP) relaxation of the problem are very weak, and part of this weakness is due to the piecewise linear objective function combined with single path routing. Stronger models provided in this paper could be embedded in the models of [**?**], [**?**] to improve the lower bounds and make the problems more tractable.

A related problem to the PUMF problem is the non-convex piecewise linear multicommodity network flow (NCPMF) problem. This problem can be seen as occupying an opposite spectre from the PUMF problem, with respect to both the type of piecewise linear cost function, and the splittability of the flows. Whereas, as far as we are aware, the PUMF problem has not been yet studied in the literature, there are many works dealing with the NCPMF problem. Croxton *et al.* [**?**] review three formulations that had been previously used in the literature for generic problems with non-convex piecewise linear costs. These three formulations use integer variables to model the costs, and are deemed equivalent, with respect to their LP relaxation. In [**?**], the same authors choose one of this formulations, the multiple choice model [**?**], and strengthen it, so as to solve the NCPMF. Due to the common properties of the two problems, these works provide an important basis for our work.

In Section II, we begin by formulating the problem with a simple multiple choice model, as seen in [**?**], [**?**] and [**?**]. Next, we propose a set of valid inequalities that considerably strengthen the linear programming relaxation of this formulation. This is confirmed by the computational experiments, described in Section III. Finally, in Section IV we present the conclusions and discuss future developments.

II. Problem formulation

Consider the notation for the PUMF problem introduced in the beginning of Section I. In addition, let λ_i^k be such that $\lambda_{o(k)}^k = 1$, $\lambda_{d(k)}^k = -1$, and $\lambda_i^k = 0$ for every $i \neq \{o(k), d(k)\}$. We define the binary variables x_a^k, with $x_a^k = 1$ if arc $a \in A$ is on the unique path chosen to route commodity $k \in K$, and $x_a^k = 0$ otherwise; and y_a^s, with $y_a^s = 1$, if arc $a \in A$ contains a non-zero flow on segment $s \in S_a$, and $y_a^s = 0$ otherwise. For the sake of simplicity, if arc a contains a non-zero flow on segment $s \in S_a$, we say that arc a is on segment $s \in S_a$. We also define the continuous variables l_a^s, that indicate the load going through arc $a \in A$, which is on segment $s \in S_a$. The PUMF problem can be formulated with the following mixed integer linear program (MIP), which we refer to as the Basic Model 1:

$$\begin{aligned}
\min \quad & \sum_{\substack{a\in A\\ s\in S_a}} (f_a^s y_a^s + c_a^s l_a^s) && && \text{(3a)}\\
\text{s.t.} \quad & \sum_{a\in\delta^+(i)} x_a^k - \sum_{a\in\delta^-(i)} x_a^k = \lambda_i^k, && i \in V, k \in K, && \text{(3b)}\\
& \sum_{s\in S_a} y_a^s \le 1, && a \in A, && \text{(3c)}\\
& \sum_{s\in S_a} l_a^s = \sum_{k\in K} \rho^k x_a^k, && a \in A, && \text{(3d)}\\
& b_a^{s-1} y_a^s \le l_a^s \le b_a^s y_a^s, && a \in A, s \in S_a, && \text{(3e)}\\
& x_a^k \in \{0,1\}, && a \in A, k \in K, && \text{(3f)}\\
& y_a^s \in \{0,1\}, && a \in A, s \in S_a, && \text{(3g)}\\
& l_a^s \ge 0, && a \in A, s \in S_a. && \text{(3h)}
\end{aligned}$$

The typical multicommodity flow balance constraints (3b) define the path between the origin and destination node of each commodity. Then, constraint sets (3c-3e) identify the segment each arc is on. Naturally, only a single segment per arc may be selected (3c). The choice of segment is implied by the load flowing through the respective arc. This load is calculated in constraints (3d) and its value is assigned to one of the variables l. To ensure that only the appropriate load variable is non-zero, in (3e) we either bound l_a^s by the breakpoints of the corresponding segment, if $y_a^s = 1$; or we force it to zero, if $y_a^s = 0$.

Variables x and y are considered as binary (3f-3g), whilst variables l are regarded as continuous (3h). Note that if the cost function for every arc is convex, it is not necessary to define explicitly y as binary; instead they can simply be defined, as continuous and between 0 and 1. Nevertheless, one reason for defining them as binary will be discussed on Section III.

The PUMF problem can also be modelled using the objective function $\sum_{a\in A} g_a$, together with constraints (1a,1b) and (3b,3d,3f,3h). We refer to this formulation as Basic Model 2. The LP relaxation of these Basic Models, in which we relax the integrality on (3f), can provide very weak lower bounds, even for toy instances. In fact, it is worthwhile to investigate whether or not these two models produce the same LP bound, since we have observed that for all the instances tested, their LP bounds are equal.

As an example to show how weak these bounds are, consider a graph with only two nodes, o and d, connected by three parallel arcs. Assume a single commodity with origin o, destination d, and demand 3. Finally, assume that the cost function on all three arcs is the same, and is characterized by only two segments, such that the breakpoints are $b^0 = 0$, $b^1 = 1$ and $b^2 = 3$; the slopes are $c^1 = 1$ and $c^2 = 10$; and the intercepts $f^1 = 0$ and $f^2 = -9$. It is easy to see that there are three solutions to the PUMF problem on this graph, all with the same cost of 63: sending the commodity flow through each one of the three available arcs. However, in the LP relaxation of the Basic Models, the flow of the commodity can be split among the three arcs, each on the first segment. This results in a LP relaxation optimum value of only 9. By manipulating the structure of the cost functions, this gap can be virtually any value. Therefore, it is important to strengthen this model, in order to solve our problem efficiently.

To this end, we use variable disaggregation, a common technique to strengthen the LP relaxation of MIPs. Thus, consider the binary variables x_a^{ks}, with $x_a^{ks} = 1$ if arc $a \in A$, which is on segment $s \in S_a$, is on the unique path chosen to route commodity $k \in K$, and $x_a^{ks} = 0$ otherwise. We denote as the Disaggregated Model, the following formulation:

$$\min \sum_{\substack{a \in A \\ s \in S_a}} \left(f_a^s y_a^s + c_a^s \sum_{k \in K} \rho^k x_a^{ks} \right) \tag{4a}$$

$$\text{s.t.} \sum_{\substack{a \in \delta^+(i) \\ s \in S_a}} x_a^{ks} - \sum_{\substack{a \in \delta^-(i) \\ s \in S_a}} x_a^{ks} = \lambda_i^k, \qquad i \in V, k \in K, \tag{4b}$$

$$\sum_{s \in S_a} y_a^s \leq 1, \qquad a \in A, \tag{4c}$$

$$b_a^{s-1} y_a^s \leq \sum_{k \in K} \rho^k x_a^{ks} \leq b_a^s y_a^s, \qquad a \in A, s \in S_a, \tag{4d}$$

$$x_a^{ks} \in \{0,1\}, \qquad a \in A, k \in K, s \in S_a, \tag{4e}$$

$$y_a^s \in \{0,1\}, \qquad a \in A, s \in S_a. \tag{4f}$$

In this new model the l-variables are no longer necessary as $l_a^s = \sum_{k \in K} \rho^k x_a^{ks}$ for every $a \in A, s \in S_a$, leading to the new objective function (4a) and variable bound constraints (4d). Constraints (4b) define, for each commodity, a path between the origin and destination, now with the disaggregated x-variables.

It is easy to see that the Disaggregated Model, as it has been defined so far, is equivalent to the Basic Model. However, we can use the disaggregated variables to create new inequalities that considerably strengthen the LP relaxation.

First, note that an arc being traversed by a given commodity cannot be on a segment whose upper breakpoint is smaller than the demand flow. Therefore, we can fix the x-variables as follows:

$$x_a^{ks} = 0, \qquad a \in A, k \in K, s \in S_a : b_a^s < d^k \tag{5}$$

Furthermore, when combined with inequalities (4d), this variable fixing has a strong impact on the values of the y-variables.

A well-known class of valid inequalities, common with this variable disaggregation, are the following:

$$x_a^{ks} \leq y_a^s, \qquad k \in K, a \in A, s \in S_a. \tag{6}$$

These valid inequalities are an obvious choice in cases where the intercepts f_a^s are non-negative (*e.g.* [**?**]), as they lift the y-variables. This is not the case for the PUMF. However, due to (4c), these valid inequalities can still be useful in cutting-off LP solutions; those that for a given arc a have $x_a^{ks} + x_a^{k's'} > 1$, $k' \neq k$, $s' \neq s$.

A related class of valid inequalities, but now making use of the fact that the intercepts f_a^s are negative, is obtained by tightening coefficients in the first inequality in (4d):

$$b_a^{s-1} y_a^s \leq \sum_{k \in K} \min(\rho^k, b_a^{s-1}) x_a^{ks}, \qquad a \in A, s \in S_a. \tag{7}$$

Empirical results reveal that combining the Disaggregated Model with both fixing variables (5) and valid inequalities (7) is highly advantageous, improving both the quality of the lower bounds provided by the LP relaxation, and the computing time of the MIPs. We refer to this strengthening of the Disaggregated Model as the Strong Model. We do not include valid inequalities (6) in the Strong Model, as empirical results also show that they seldom improve the bounds of the LP relaxation, and often cause out-of-memory issues for large instances. Note, however, that they might prove to be useful for specific instances, when generated dynamically.

Finally, we conclude this section by pointing out a relevant result about the Strong Model, namely that its LP feasible set provides a complete description of the associated polyhedron for the single commodity case. Due to space restrictions, we omit the proof in this paper.

III. Computational experiments

In this section, we present the results of computational experiments that were conducted in order to compare the Basic Models 1 and 2, and the Strong Model, for solving instances of the PUMF problem. These experiments were done on randomly generated instances, that follow the motivation of the problem. We created 40 instances and grouped them into 8 classes, according to number of nodes, arcs and commodities. Table I describes each class of instances.

In each of these instances, the distribution of the arcs on the graph is random. Each arc was assigned a capacity of 50, 75 or 100. The traffic demand between the origin and destination node of each commodity was calculated using the following formula, used in [**?**]:

$$\rho^k = \alpha O_{o^k} D_{d^k} R_{(o^k,d^k)} e^{\frac{-L_2(o^k,d^k)}{2\Delta}} \tag{8}$$

For each node i, two random numbers, O_i and D_i were randomly generated in the interval $[0,1]$. These values reflect, respectively, the activeness of each node as a sender and as a receiver. Another value, $R_{(i,j)}$, was generated in the same interval, for each pair of nodes. The parameter α was set to 0.6, for all the experiments; we found that this value lead to instances, where the majority of the arcs used in the optimal solution were not overloaded. The Euclidian distance (L_2) was substituted by the length of the shortest path between each pair of nodes, with respect to the number of links. Δ is the largest distance in the network. The final values were rounded to the nearest integer.

The routing cost on every arc are given by the function shown in Figure 2, and described in [**?**].

Class ID	$\|V\|$	$\|A\|$	$\|K\|$
1	60	2478	200
2	60	2832	150
3	60	2832	200
4	80	316	250
5	80	1896	200
6	80	1896	250
7	80	3160	200
8	80	1896	350

TABLE I
DESCRIPTION OF EACH CLASS OF INSTANCES.

The three formulations were implemented using ILOG CLEX 12.6, on a Intel Core i7 CPU 960 @ 3.20GHz with 12GB of memory with 64 bits, and running Ubuntu $14.04.2$ LTS (GNU/Linux $3.2.0 - 26-$generic x86_64). Tests were done using both the MIP and LP of each instance. Figures 3, 4 and 5 illustrate the results.

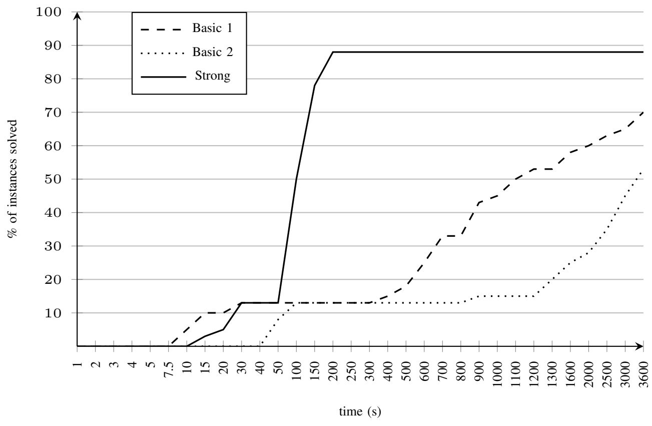

Fig. 2. Performance profile of the solving time of the MIP.

Figure 3 describes the performance profile of the solving time of the MIPs, for each model. This performance profile measures the percentage of instances (spanned over the y-axis) that are solved under different amounts of time, detailed in the x-axis. It reveals that, even though using CPLEX with the Basic Model 1 is faster in solving the "easier" instances (Class 4), for the more "demanding" ones, using CPLEX with the Strong Model far out-performs the latter. Whereas by using the Basic Model 1 we were only able to solve 70% of the instances in the time limit, by using the Strong Model we solved 88%. CPLEX was not able to solve the remaining 5 instances, with the Strong Model, not due to time restrictions, but due to lack of memory. All the other instances were solved fairly quickly, under 200 seconds. Finally, we can also observe that the Basic Model 2 is much slower than the other models, only solving 53% of the instances within the time limit. The disparity between the results of the two Basic Models might be partly explained by our implementation: we defined the y-variables as binary, which might help during CPLEX's branch-and-bound procedure.

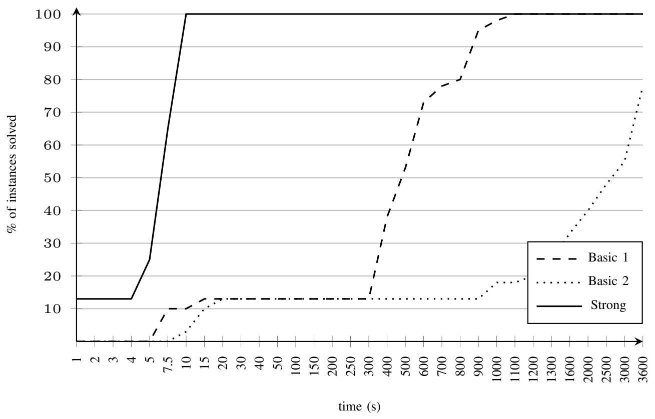

Fig. 3. Performance profile of the solving time of the LP.

When it comes to solving the LP relaxation of each instance, the Strong model is also much faster than the two Basic Models. This can be observed in the performance profile depicted in Figure 4. It is also curious to verify that the Basic Model 2 is only able to solve the LP relaxation of 78% of the instances within the one hour time limit.

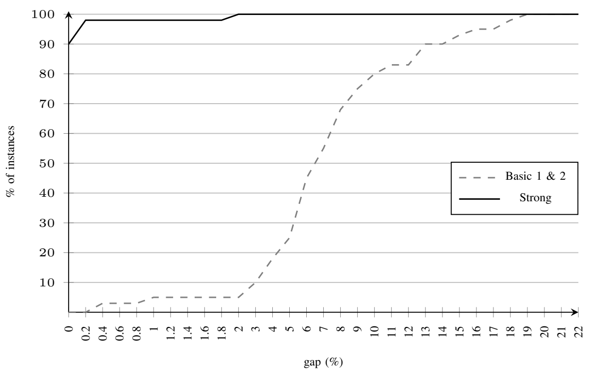

Fig. 4. Performance profile of the LP gap.

Let g^*_{IP} and g^*_{LP} be, respectively, the cost of the best integer solution found, and the cost of the solution of the LP relaxation of each model. We denote as LP gap the ratio $\frac{g^*_{IP}-g^*_{LP}}{g^*_{IP}}$. The performance profile in Figure 5 measures the proportion of our instances whose LP gap is under the the percentages detailed in the x-axis. In it, we can observe that the two Basic Models produce the same LP bound. More importantly, we can see how strong the Strong Model really is. The high percentage (90%) of instances that have 0 LP gap with the Strong Model, contrasts significantly with the results for the Basic Models.

IV. CONCLUSION

In this work, we present a novel, special case of the multicommodity flow problem, for which the flow of each commodity is unsplittable, and the routing costs on the arcs are given by a convex piecewise linear function. We propose four MIP formulations for this problem: the Basic Model 1 and 2, the Disaggregated Model and the Strong Models. We present computational experiments on randomly generated instances. These experiments reveal that the LP relaxation of the Strong Model is able to provide very good lower bounds, far better than the ones provided by the LP relaxation of the Basic Models. In fact, for 90% of the randomly generated instances, the LP gap of the Strong Model is null. For this reason, the Strong Model is also much more efficient than the Basic Models, in solving the MIPs of the instance. When implemented in CPLEX, this model was able to solve 88% of the instances in under 200 seconds. As for the remaining, unsolved instances, the issue was not the time restriction, but lack of CPU memory.

In the future, we want to explore the use of decomposition methods, that could eventually help solve the issue of large memory consumption for the more demanding instances. Moreover, we consider that it would be interesting to delve into a variant of this problem, where the piecewise linear cost functions can be non-convex. This could be seen as a special case of the non-convex piecewise linear multicommodity problem studied in [**?**] and [**?**], where the flows are unsplittable.

Acknowledgment

The work of Bernard Fortz and Martim Joyce-Moniz is supported by the Interuniversity Attraction Poles Programme P7/36 COMEX initiated by the Belgian Science Policy Office. The work of Luís Gouveia is supported by National Funding from FCT - Fundação para a Ciência e a Tecnologia, under the project: PEst-OE/MAT/UI0152.