

On the integration of Dantzig-Wolfe and Fenchel decompositions via directional normalizations

François Lamothe · Alain Haït ·
Emmanuel Rachelson · Claudio
Contardo · Bernard Gendron

the date of receipt and acceptance should be inserted later

Abstract The strengthening of linear relaxations and bounds of mixed integer linear programs has been an active research topic for decades. Enumeration-based methods for integer programming like linear programming-based branch-and-bound exploit strong dual bounds to fathom unpromising regions of the feasible space. In this paper, we consider the strengthening of linear programs via a composite of Dantzig-Wolfe and Fenchel decompositions. We provide geometric interpretations of these two classical methods. Motivated by these geometric interpretations, we introduce a novel approach for solving Fenchel sub-problems and introduce a novel decomposition combining Dantzig-Wolfe and Fenchel decompositions in an original manner. We carry out an extensive computational campaign assessing the performance of the novel decomposition on the unsplittable flow problem. Very promising results are obtained when the new approach is compared to classical decomposition methods.

Keywords Mixed Integer Linear Programming · Decomposition methods · Unsplittable flows · Dantzig-Wolfe decomposition · Fenchel decomposition

Acknowledgments

This document is the result of a research project funded by the *Centre national d'études spatiales* (CNES) and Thales Alenia Space. C. Contardo and B. Gendron thank the Natural Sciences and Engineering Research Council of Canada

François Lamothe · Alain Haït · Emmanuel Rachelson
ISAE-SUPAERO, Université de Toulouse
E-mail: francois.lamothe@isae.fr

Claudio Contardo
Concordia University, Canada

Bernard Gendron
Université de Montréal, Canada

(NSERC) for its financial support under Discovery Grants no. 2020-06311 and 2017-06054.

1 Introduction

Enumeration-based algorithms are arguably the main algorithmic frameworks to solve mixed-integer linear programs among which the branch-and-bound (B&B) method [33] is perhaps the most efficient and versatile one. It relies on the ability to compute primal and dual bounds on the value of the solutions to the problem at hand. The method is most efficient when quick computing of tight primal and dual bounds is available, resulting in short enumerations. Traditionally, primal bounds are found via heuristics and dual bounds via solving relaxations of the problem. The most classical such relaxation is the so-called *linear relaxation* in which the integrality constraints are ignored. The resulting problem, an ordinary linear program, can be solved efficiently either in polynomial time via interior point methods [30, 38], or via a greedy method — namely the simplex method [19] — of exponential worst-case complexity, but very efficient in practice [10].

A common practice to improve the linear relaxation of mixed integer linear programs is to apply a decomposition method to a subset of the problem’s constraints whose associated polyhedron does not have the integrality property (some vertices of the polyhedron have non-integer components). The decomposition tightens the polyhedron which in turn strengthens the linear relaxation of the problem. One of the most classical decomposition methods is the one by Dantzig and Wolfe [20], known as the Dantzig-Wolfe decomposition. It has proven successful in various applications which explains the attention it has received over the years [22]. The Dantzig-Wolfe decomposition is closely related to the Lagrangian decomposition which has also been successful in many practical applications. The main difference between the Lagrangian and Dantzig-Wolfe decomposition is that in the former only dual information is extracted and exploited in a sub-gradient algorithm whereas the latter also extracts primal information which can be embedded within an enumeration scheme, usually referred to as *branch-and-price* [3]. Because in this work we will consider column generation and cutting plane procedures, we will focus on the Dantzig-Wolfe decomposition.

Our analysis restricts to decompositions that are able to exploit integer subproblems —as opposed to pure linear problems—, and for that reason, we do not consider Benders decomposition [7]. Fenchel decomposition, on the other hand, is a cutting plane method similar to Benders decomposition that can be applied when the sub-problem is integer. It is thus able to improve the relaxation quality of mixed integer linear programs. This technique has had success in several applications such as knapsack problems [11, 29], generalized assignment problems [1], network design problems with unsplittable flow [15] or even stochastic optimization problems [40, 4].

Although Fenchel and Dantzig-Wolfe decompositions have been largely studied in the context of linear relaxation strengthening, several limitations of both decomposition methods have been identified in the literature which may affect their convergence. In particular, the Dantzig-Wolfe decomposition is known to suffer from the degeneracy of its master problem (which happens when the master problem admits several optimal dual solutions). This explains why a considerable effort has been made by the scientific community to improve these decomposition methods and overcome their weaknesses. This work is in this line of contribution. We show that Dantzig-Wolfe and Fenchel decompositions can be seen in a similar light as constructing inner and outer approximations of the polyhedron being decomposed and that the synergy between the two approximations may be beneficial for the entire approach when applied to some large-scale mixed-integer linear programs.

We outline the main contributions of our work as follows:

- We provide geometric interpretations of Dantzig-Wolfe and Fenchel decompositions which may fuel complementary insights on these two approaches compared to purely analytical interpretations.
- We provide a critical overview of normalizations at the core of the separation problems arising in Fenchel decomposition. The normalization impacts the type of cut, its properties, and ultimately the convergence speed of the decomposition method. We provide geometric interpretations of several types of normalization and their properties.
- We introduce a novel approach to the Fenchel sub-problem when a directional normalization is used. The proposed method possesses reduces the numerical instabilities of a direct resolution approach commonly used. We show that the new approach solves the Fenchel sub-problem in finitely many iterations.
- We introduce a new decomposition method inspired by both the Dantzig-Wolfe and the Fenchel decompositions. The proposed method uses a Dantzig-Wolfe master problem and a Fenchel master problem. A Fenchel sub-problem guided with a directional normalization is used to coordinate the two master problems. The resulting method is shown to perform especially well on instances presenting high degrees of degeneracy. We provide a possible explanation of this phenomenon based on our findings.

In the following, we start by giving in Section 2 the context in which we apply the various decomposition methods together with some notation. Then, we present in Section 3 a geometric interpretation of Dantzig-Wolfe and Fenchel decomposition methods. Section 4 is dedicated to an overview of the resolution of the Fenchel subproblem and in particular, the impact of the normalization used. This concept will be used in the next section for the new methods introduced in the paper. We then present in Section 5 a new approach to solving the Fenchel subproblem. In Section 6, we introduce the new decomposition method which integrates both Dantzig-Wolfe and Fenchel master problems as well as a Fenchel subproblem. In Section 7, we illustrate our method by applying it to the unsplittable flow problem. An experimental

evaluation of all proposed methods is then made in Section 8 on small and medium-sized instances of the problem.

2 Context

In the remainder of this paper, we will apply decomposition methods to the following general mixed integer linear program:

$$(P) \quad \max_x \quad cx \quad (1a)$$

$$\text{subject to} \quad A_1x \leq b_1 \quad (1b)$$

$$A_2x \leq b_2 \quad (1c)$$

$$x \in X \quad (1d)$$

where X is a product set $\mathbb{R}^p \times \mathbb{Z}^q$ of appropriate dimension whose linear relaxation \mathbb{R}^{p+q} will be noted \bar{X} . Moreover, to explain the methods used, we will consider the following polyhedra which are assumed to be bounded to simplify the explanations:

$$LR_1 = \{x \in \bar{X} | A_1x \leq b_1\}$$

$$LR_2 = \{x \in \bar{X} | A_2x \leq b_2\}$$

$$Q_2 = \text{conv}(\{x \in X | A_2x \leq b_2\})$$

In order to bound from above the value of the optimal solution of a general mixed integer linear program as (P) one can optimize its objective function over a relaxation containing the set of valid solutions. Usually, the linear relaxation of the set of solutions $LR_1 \cap LR_2$ is used. However, one might want a tighter relaxation to obtain a better upper bound. This can be obtained by using the relaxation $LR_1 \cap Q_2$ which still contains the set of valid solutions while being included in $LR_1 \cap LR_2$. As illustrated in Figure 1, this relaxation usually returns strictly better bounds than the linear relaxation when the polyhedron Q_2 is strictly included in LR_2 which happens when the polyhedron LR_2 does not have the integrality property (some of the vertices of the polyhedron have non-integer coordinates). However, the drawback of Q_2 compared to LR_2 is that one usually only has a representation of Q_2 with an exponential number of variables or constraints which is not manageable directly by a linear programming solver. In compensation, we assume to have at our disposal an efficient algorithm to optimize a linear function on the polyhedron Q_2 . This algorithm is called the optimization oracle and solves the following mixed integer linear program:

$$(O) \quad \max_x \quad \pi x$$

$$\text{subject to} \quad A_2x \leq b_2$$

$$x \in X$$

where $x \mapsto \pi x$ is any linear function. Thus, to be able to optimize over $LR_1 \cap Q_2$, the goal of a decomposition method is to use this oracle to compute an approximation of Q_2 of manageable size. In particular, the Dantzig-Wolfe decomposition iteratively grows an inner approximation of Q_2 while the Fenchel decomposition iteratively refines an outer approximation of Q_2 .

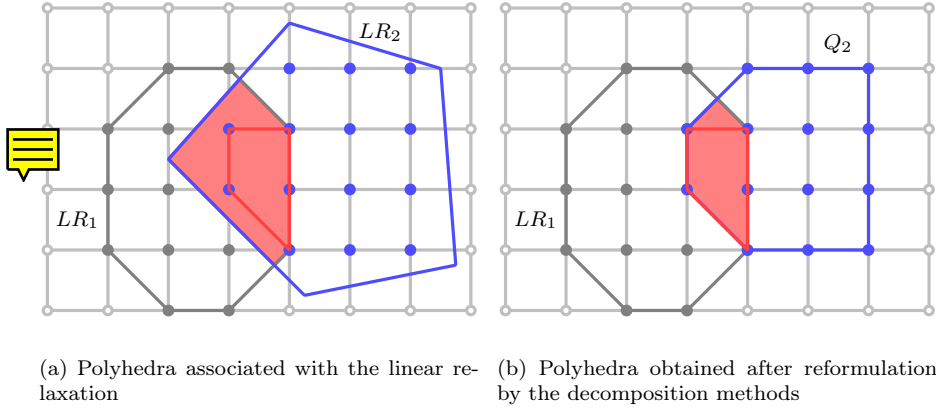


Fig. 1: Representation of solution spaces of the linear relaxation and the models obtained from the decomposition methods

Decomposition methods are known to work very well when the matrix A_2 is block diagonal because it enables decomposing the problem of optimizing over Q_2 into several smaller problems (one per block of the matrix). For instance, in the case of the application considered in this study, the unsplittable flow problem, the capacity constraints lead to a block diagonal structure. In this context, the constraints $A_1x \leq b_1$ would correspond to the flow conservation constraints while the constraints $A_2x \leq b_2$ would correspond to the capacity constraints. However, our work makes use of a different rationale. Let us assume that the matrix A_2 is sparse with non-zero elements for only a few variables; this may happen for instance when A_2 corresponds to one of the blocks of a block diagonal matrix. With this choice, it is possible to apply decomposition methods even in contexts where the main problem (P) does not have any block diagonal structure. For clarity purposes, we will present the methods as if we were decomposing only one polyhedron at a time. However, in practice, one would decompose several polyhedra at the same time; for example, all the blocks of a block diagonal matrix.

3 Geometric interpretation of Dantzig-Wolfe and Fenchel decompositions

In this section, we provide a geometric interpretation of both decomposition methods with the potential to fuel new intuitions regarding their strengths and weaknesses.

3.1 Dantzig-Wolfe decomposition

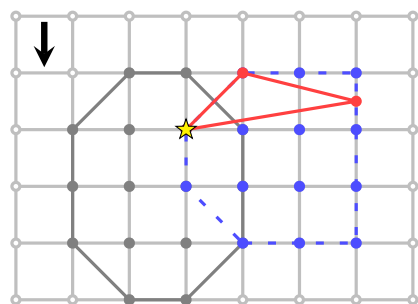
Instead of optimizing the objective function of mixed integer linear program (P) over the solution set $LR_1 \cap LR_2$ of its linear relaxation, Dantzig-Wolfe decomposition allows for the optimization over the smaller set $LR_1 \cap Q_2$. Because one does not usually have a manageable description of the set Q_2 , it is necessary to compute an approximation of Q_2 in order to optimize over the intersection $LR_1 \cap Q_2$. To compute this approximation, we assume the availability of an optimization oracle over Q_2 . The main idea in the Dantzig-Wolfe decomposition is to iteratively grow an inner approximation of Q_2 . One can create such an approximation by obtaining a set of points x_i belonging to Q_2 (usually extreme points of Q_2) and setting the approximation as the convex envelope of these points. We will denote this inner approximation with \hat{Q}_2 . The Dantzig-Wolfe decomposition illustrated in Figure 2 proceeds as follows:

1. Initialize the approximation \hat{Q}_2 with points of Q_2
2. Find the optimal solution \hat{x} over $LR_1 \cap \hat{Q}_2$
3. Search for a point of Q_2 whose addition to \hat{Q}_2 may improve the value of the optimal solution \hat{x}
4. If such a point exist, add it to \hat{Q}_2 and go to Step 2.
5. Else: \hat{x} is the optimal solution over $LR_1 \cap Q_2$. Stop.

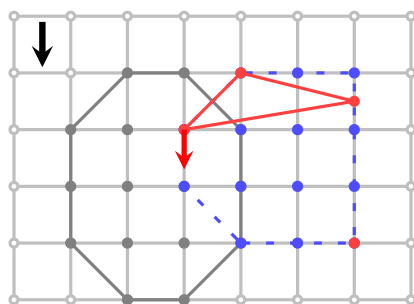
3.1.1 Optimizing over $LR_1 \cap \hat{Q}_2$

To optimize over $LR_1 \cap \hat{Q}_2$, a master linear program is created in which the condition $x \in \hat{Q}_2$ must be enforced. This can be done by rewriting x as a convex combination of its extreme points x_i . To that end, a variable λ_i is introduced for each point x_i used to create \hat{Q}_2 . This variable represents the weight of the vertex x_i in the convex combination. The Dantzig-Wolfe master linear program can then be written as follows:

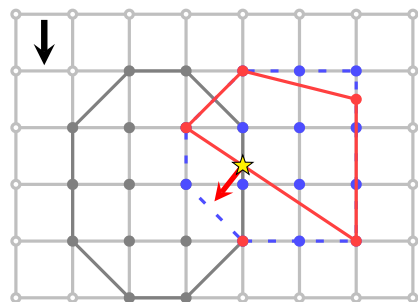
$$\begin{aligned}
 (DW) \quad & \max_{x, \lambda_i} \quad c^T x \\
 & \text{subject to} \quad A_1 x \leq b_1 \\
 & \quad x = \sum_{i \in I} \lambda_i x_i \\
 & \quad \sum_{i \in I} \lambda_i = 1 \\
 & \quad x \in \bar{X}, \quad \lambda_i \in \mathbb{R}^+, \quad \forall i \in I
 \end{aligned}$$



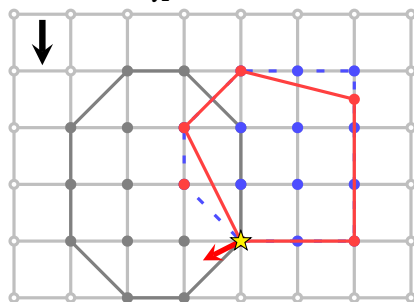
(a) Iteration 1. Initialization of \hat{Q}_2 with points of Q_2 and optimization over $LR_1 \cap \hat{Q}_2$. Black arrow: objective function of the master problem



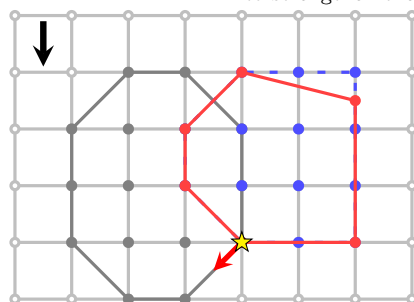
(b) A direction (red arrow) in which to grow the polyhedron \hat{Q}_2 is created with the dual variables of the master problem. The farthest point of Q_2 in this direction is found and added to \hat{Q}_2 .



(c) Iteration 2. Re-optimization over $LR_1 \cap \hat{Q}_2$. The farthest point of Q_2 in the dual direction is found and added to \hat{Q}_2 .



(d) Iteration 3. Even though the optimal solution is found, due to degeneracy, a new point of Q_2 is found in the dual direction to strengthen the dual bound.



(e) Iteration 4. Once the current solution is the farthest point of Q_2 in the dual direction then it is optimal and the method stops.

Fig. 2: Geometric illustration of the Dantzig-Wolfe decomposition

3.1.2 Finding an improving point of Q_2

In the Dantzig-Wolfe decomposition, the master problem returns the farthest point x^* of $LR_1 \cap \hat{Q}_2$ in the direction c and one would like to know whether this point is also the optimal for $LR_1 \cap Q_2$ or if the inner approximation \hat{Q}_2 needs to be improved. The dual point of view of this statement, on which is based the Dantzig-Wolfe subproblem, is that the bound $cx \leq cx^*$ is valid for $LR_1 \cap \hat{Q}_2$ and we would like to know if it also holds true for $LR_1 \cap Q_2$.

Ideas from linear programming duality theory: The following concepts are illustrated in Figure 3. Linear programming duality informs us that because $LR_1 \cap \hat{Q}_2$ is the intersection of two polyhedra, the bound $cx \leq cx^*$ can always be decomposed as the sum of two inequalities, one valid for LR_1 and the other valid for \hat{Q}_2 . Furthermore, the dual variables of the master problem yield such a decomposition of the optimal bound for $LR_1 \cap \hat{Q}_2$. Indeed, let us denote u , π and π_0 the optimal dual variables of the constraints $A_1x \leq b_1$, $x = \sum_i \lambda_i x_i$ and $\sum_i \lambda_i = 1$, respectively. By construction of the dual of the master problem (whose derivation we let to the reader), $uA_1x \leq ub_1$ is valid for LR_1 , $\pi x \leq \pi_0$ is valid for \hat{Q}_2 and these two inequalities sum to the optimal bound $cx \leq cx^*$ over $LR_1 \cap \hat{Q}_2$ (note: the equality $ub + \pi_0 = cx^*$ follows thanks to the strong duality theorem of linear programming [37]).

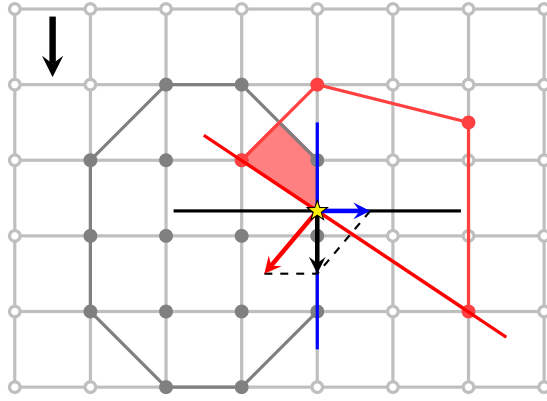


Fig. 3: The optimal bound (black) on the optimization direction (black arrows) over the intersection (red fill) of LR_1 (grey) and \hat{Q}_2 (red) can be decomposed as the sum of a valid inequality (blue) for LR_1 and a valid inequality (red) for \hat{Q}_2 .

The Dantzig-Wolfe subproblem: In order to show that the solution x^* of the master problem is not the farthest point of $LR_1 \cap Q_2$ in the direction c , one must at least prove that the bound $cx \leq cx^*$ implied by the dual variables of the master problem is not valid for $LR_1 \cap Q_2$. However, if the inequality $\pi x \leq \pi_0$ were valid for Q_2 , because we know that $uA_1x \leq ub_1$ is valid for LR_1

and that these two inequalities sum to the bound $cx \leq cx^*$ then this bound would be valid for $LR_1 \cap Q_2$. Thus, the goal of the sub-problem is to show that the inequality $\pi x \leq \pi_0$ is not valid for Q_2 . To that end, the subproblem is tasked to find the farthest point of Q_2 in the direction π . If this point violates $\pi x \leq \pi_0$ then we have found a point of Q_2 violating the inequality. The point can then be added to the master problem to grow the inner approximation \hat{Q}_2 and at least prevent the master problem from yielding the same dual variables again. Otherwise, the current solution is optimal because the bound implied by the dual variables is valid for $LR_1 \cap Q_2$.

3.1.3 A note on degeneracy

We have seen above that the dual variables of the master problem imply a bound on the value of its objective function. This bound can be considered a certificate that the current value of the master problem is optimal. Meanwhile, a linear program such as a Dantzig-Wolfe master problem is said to be degenerate when it has several dual optimal solutions. Each of these dual solutions is a certificate of optimality for the current value of the master problem. Thus, to improve the value of the master problem, one must invalidate each of these certificates. However, the subproblem of the Dantzig-Wolfe procedure only invalidates one of these certificates without any guarantee about the one implied by the other dual solutions. Thus the Dantzig-Wolfe decomposition is susceptible to having many iterations without improvement of the objective function when the master problem is highly degenerate. This can slow down the convergence of the method considerably.

3.2 Fenchel decomposition

In Fenchel decomposition, instead of growing an inner approximation of Q_2 , an outer approximation of the polyhedron Q_2 is refined to enable the optimization over $LR_1 \cap Q_2$. Such an outer approximation can use any collection of inequalities valid for Q_2 . The decomposition is illustrated in Figure 4 and proceeds as follows:

1. Initialize an outer approximation \check{Q}_2 with inequalities valid for Q_2 ; typically one can take $\check{Q}_2 = LR_2$.
2. Optimize over $LR_1 \cap \check{Q}_2$ and recover a solution \check{x} .
3. Search for a cut separating \check{x} from Q_2 .
4. If such a cut exists, add the cut to \check{Q}_2 and go to Step 2.
5. Else: \check{x} is the optimal solution over $LR_1 \cap Q_2$. Stop.

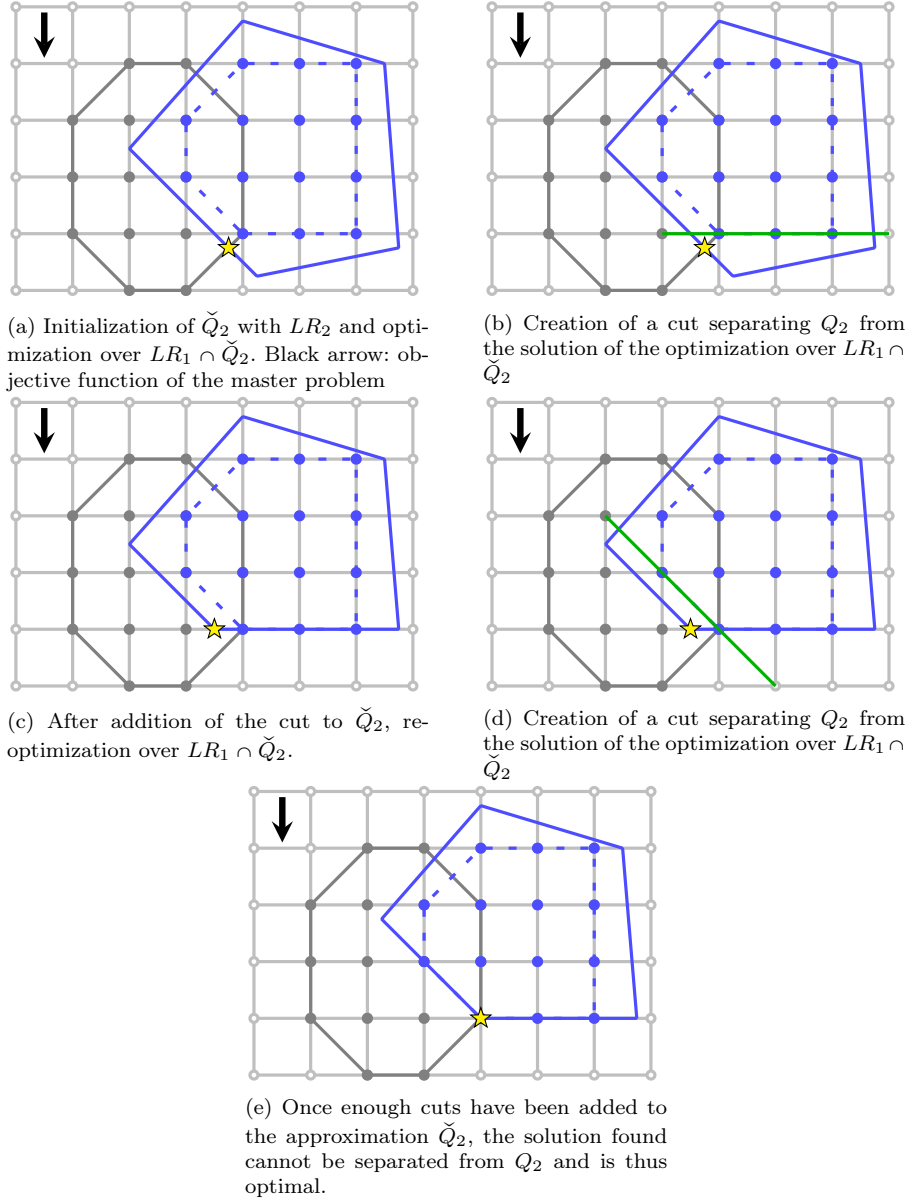


Fig. 4: Geometric illustration of the Fenchel decomposition

In the second step, in order to optimize over $LR_1 \cap \tilde{Q}_2$, the following Fenchel master problem is used:

$$\begin{aligned}
 (F) \quad & \max_x \quad c^T x \\
 & \text{subject to} \quad A_1 x \leq b_1 \\
 & \quad \pi x \leq \pi_0 \quad \forall (\pi, \pi_0) \in \mathcal{C} \\
 & \quad x \in \bar{X}
 \end{aligned}$$

where \mathcal{C} is the set of cuts describing \check{Q}_2 .

The main challenge in the Fenchel decomposition is to generate a cut $\pi x \leq \pi_0$ separating the solution of the Fenchel master problem \check{x} from the polyhedron Q_2 . The classical approach to generating such cuts is based on a linear program as described in Section 4.

4 The Fenchel separation subproblem and its normalizations

In the Fenchel decomposition, a cut $\pi x \leq \pi_0$ separating the solution of the Fenchel master problem \check{x} from the polyhedron Q_2 must be found. Such a cut can be created by finding a solution of non-negative value of the following separation linear program:

$$\begin{aligned} (S) \quad & \max_{\pi, \pi_0} \quad \pi \check{x} - \pi_0 \\ & \text{subject to} \quad \pi x_i \leq \pi_0, \quad \forall x_i \in Q_2 \\ & \quad \quad \quad \pi, \pi_0 \in \mathbb{R} \end{aligned}$$

where the objective maximizes the violation of the generated cut by \check{x} while the constraints ensure that the cut is valid for every point of the polyhedron Q_2 . However, this separation problem has too many constraints to consider them all explicitly. It is thus initially solved with a subset of its constraints ensuring that the generated cut is valid for a few points of Q_2 . New points will progressively be taken into account in the constraints until the validity of the cut for the whole polyhedron can be ensured. To check if a cut $\bar{\pi} x \leq \bar{\pi}_0$ is valid for Q_2 , we search for the point of Q_2 that most violates the cut. This can be done with a call to the optimization oracle (O) for the linear function associated with $\bar{\pi}$. If the point returned by (O) violates the cut, it is added as a constraint to (S). Otherwise, the cut $\bar{\pi} x \leq \bar{\pi}_0$ is the optimal solution of (S) and maximizes the separation of the solution of the Fenchel master problem from Q_2 .

Another issue preventing the direct resolution of the problem (S) is that its solution space is a cone. Indeed, if a cut $\pi x \leq \pi_0$ is valid for all the points of Q_2 then so is the cut $\alpha \pi x \leq \alpha \pi_0$ for any non-negative constant α . Thus, the problem (S) is often unbounded. To prevent this, a normalization process must be performed which classically consists in adding a constraint to the separation problem (S) that will make the set of possible coefficients of the generated cut bounded. The choice of this normalization greatly impacts the generated cut as well as the convergence speed of the Fenchel decomposition. In addition, the normalization may have an effect on the quality of the cut obtained, for instance by favoring (or not) the obtention of facets of the polyhedron Q_2 . As we will see in the following sections, the normalization also impacts the dual problem of (S) which becomes “*Find the point of Q_2 minimizing some criterion of proximity to \check{x}* ”. It is often more intuitive to interpret the impact of a normalization on the dual and therefore we use this approach for our geometric interpretations.

4.1 Normalization $\|\pi\| \leq 1$

Boyd [12] analyzes the normalization consisting in adding to the separation problem the constraint $\|\pi\| \leq 1$ for any norm (*i.e.* a function that satisfies the triangular inequality and for which for every $x \in \mathbb{R}^n$ and $\lambda \in \mathbb{R}$ satisfies $\|\lambda x\| = |\lambda| \|x\|$ and $\|x\| = 0 \Rightarrow x = 0$). With this choice, we can consider that the optimized quantity is $\frac{\pi \tilde{x} - \pi_0}{\|\pi\|}$, *i.e.* the distance from \tilde{x} to the hyperplane $\pi x = \pi_0$. Note that this distance is not measured in the norm $\|\cdot\|$ but in its dual norm $\|\cdot\|^*$: $\|\lambda\|^* = \max_{\|x\| \leq 1} \lambda x$. In particular, note that the norms $\|\cdot\|_1$ and $\|\cdot\|_\infty$ are dual and that the norm $\|\cdot\|_2$ is self-dual. When using this normalization, the dual problem (D) becomes "Find the point of Q_2 closest to \tilde{x} in the sense of the dual norm $\|\cdot\|^*$ ". This point is located on the border of Q_2 and corresponds to the point of contact between Q_2 and the smallest sphere centered in \tilde{x} touching Q_2 . The generated cut is then a tangent cut to Q_2 and to the sphere at the point of contact. One important result of the work of Boyd [12] is that the polyhedron Q_2 can be computed in finite time with this normalization. On the other hand, the generated cuts are not always facets of the polyhedron Q_2 . All these geometric interpretations are illustrated in the $\|\cdot\|_2$ norm in Figure 5.

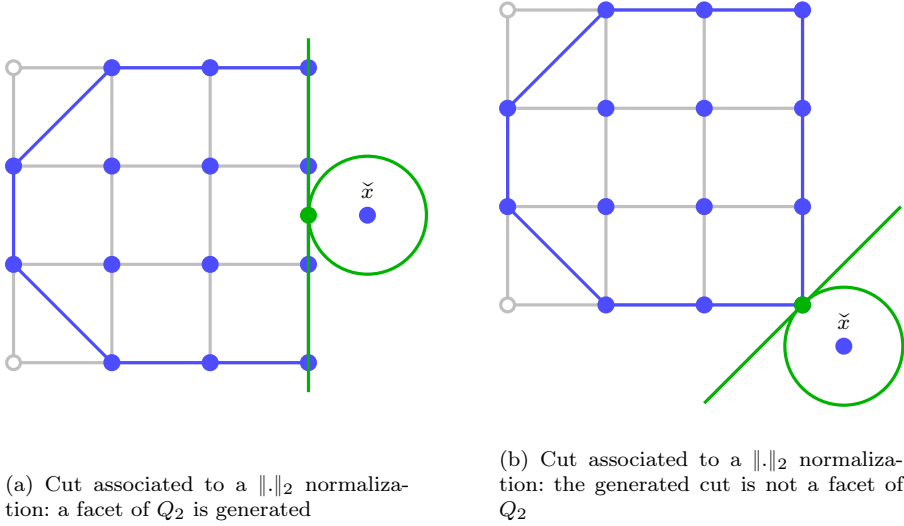


Fig. 5: Example of cuts generated with a $\|\cdot\|_2$ normalization

4.2 Normalizations guaranteeing the generation of facets

We are now interested in normalizations guaranteeing the generation of facets of Q_2 . To do this, we start by presenting a theorem linking the facets of Q_2

and the extreme rays of the cone of the solutions of the separation problem (S) . This theorem can be found in Conforti and Wolsey [17] (Proposition 1). In this theorem, $\text{cone}(X)$ (resp. $\text{vect}(X)$) is the set of all conical (resp. linear) combinations of elements of a set X .

Theorem 1 *Let P be a nonempty polyhedron. Let $(\pi^j x = \pi_0^j)_{j \in J^=}$ be a non-redundant representation of the affine envelope of P and let $(\pi^j x \leq \pi_0^j)_{j \in J^{\leq}}$ be the set of facets of P . Then the set of valid cuts for P is:*

$$\mathcal{C}(P) = \text{cone} \left(\begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} \pi^j \\ \pi_0^j \end{pmatrix}_{j \in J^{\leq}} \right) + \text{vect} \left(\begin{pmatrix} \pi^j \\ \pi_0^j \end{pmatrix}_{j \in J^=} \right)$$

In addition, all these vectors are necessary for the previous representation.

This theorem states that apart from the trivial ray $(0, 1)$, the extreme rays of the cone of the solutions of (S) are associated with the facets of the polyhedron Q_2 . When the polyhedron is not fully dimensional (*i.e.* $J^= \neq \emptyset$), the cuts corresponding to $(\pi^j, \pi_0^j)_{j \in J^=}$ are the improper faces of Q_2 . Let us suppose for the following that Q_2 is full-dimensional. This happens once all the improper faces have been generated.

A sufficient condition ensuring that the cuts generated are facets of Q_2 is to use a normalization condition which can be applied by adding a unique linear constraint to the separation problem. If only one linear constraint is added to (S) , then all the extreme points of the new solution space correspond to old extreme rays and therefore to facets of Q_2 . If the problem (S) is indeed made bounded by the addition of the linear constraint, then it suffices to find an optimal vertex to guarantee the construction of a facet. More generally, it is possible to add several linear constraints, as long as they do not intersect inside the cone of solution of (S) . For example, adding a constraint $-1 \leq f(\pi, \pi_0) \leq 1$, where f is a linear function, corresponds to adding two linear constraints that do not intersect. The cuts generated will then induce facets of Q_2 . When, for reasons specific to the problem, we know that valid cuts satisfy $\pi \geq 0$ (for example if Q_2 is a knapsack polyhedron), then the condition $\|\pi\|_1 \leq 1$ reduces to $\sum_i \pi_i \leq 1$ and can be used to generate facet inducing inequalities.

Normalization of π_0 : A normalization guaranteeing the generation of facets is $|\pi_0| \leq 1$. This normalization is studied in detail in Conforti and Wolsey [17]. It makes the problem (S) bounded if and only if \tilde{x} can be written as a combination of elements in Q_2 using only non-negative coefficients. Indeed, let us look at the impact of normalization on the dual problem (D) which becomes:

$$\begin{aligned} (D) \quad & \min_{\lambda_i, z} && |z| \\ & \text{subject to} && \sum_{i \in I} \lambda_i x_i = \tilde{x} \\ & && \sum_{i \in I} \lambda_i = 1 + z \\ & && z \in \mathbb{R}, \lambda_i \in \mathbb{R}^+, \quad \forall i \in I \end{aligned}$$

In simple words, this problem can be interpreted as follows: find a combination with non-negative coefficients of elements of Q_2 equal to \tilde{x} whose sum of the coefficients is as close as possible to 1. The separation problem (S) is bounded if and only if its dual (D) is feasible. However, the problem (D) is feasible if and only if \tilde{x} can be written as a combination with non-negative coefficients of elements of Q_2 . This condition is naturally reached when the origin is in the interior of Q_2 since the set of combinations with non-negative coefficients of elements of Q_2 is then the entire space. If we know a point in the interior of Q_2 , it is possible to ensure this condition by translating the problem to place the origin on this interior point. Once the origin is inside Q_2 , the generated cut is a facet of Q_2 intersecting the segment connecting \tilde{x} to the origin.

Directional normalization of π : A second normalization, studied by Bonami [9] and guaranteeing the generation of facets, consists in bounding the coefficients of π in a given direction using the constraint $|(\hat{x} - \tilde{x})\pi| \leq 1$ where \hat{x} is an arbitrary point in \mathbb{R}^n . In order to determine when this normalization makes the problem (S) bounded, let us look at its impact on the dual problem (D) which becomes:

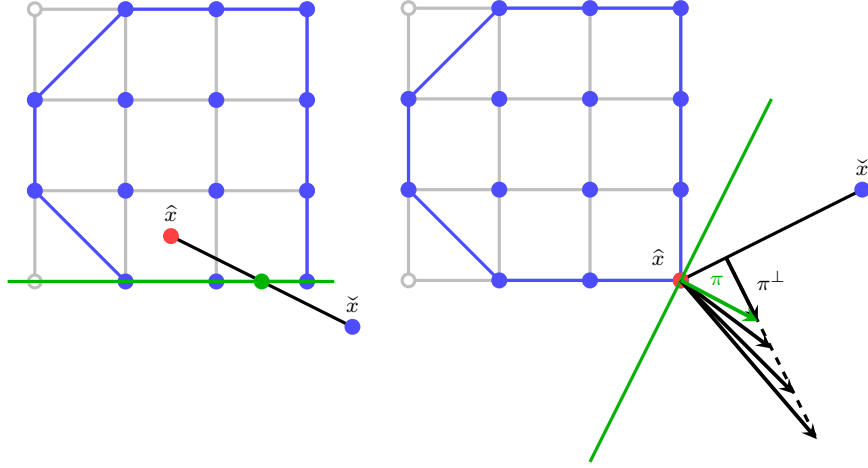
$$\begin{aligned}
 (D) \quad & \min_{\lambda_i, z} \quad |z| \\
 & \text{subject to} \quad \sum_{i \in I} \lambda_i x_i = z\hat{x} + (1-z)\tilde{x} \\
 & \quad \sum_{i \in I} \lambda_i = 1 \\
 & \quad z \in \mathbb{R}, \lambda_i \in \mathbb{R}^+, \quad \forall i \in I
 \end{aligned}$$

This problem can be interpreted as follows: *find the point of Q_2 closest to \tilde{x} on the line containing \hat{x} and \tilde{x} .* The problem (S) is bounded when its dual (D) is feasible which happens when there is a point belonging to both Q_2 and to the previous line. This condition can be fulfilled for instance by choosing \hat{x} equal to a known point of Q_2 . In this case, the generated cut is a facet of Q_2 intersecting the segment connecting \tilde{x} and \hat{x} . This is illustrated in Figure 6a. Unlike the previous normalization, the known point \hat{x} does not need to be in the interior of Q_2 . However, note that when it is on the border of Q_2 , the optimal value of the problem (S) is bounded but the set of optimal solutions may be unbounded. Indeed, let us consider the following example illustrated in Figure 6b.

Example: Suppose that \hat{x} is located on a vertex of Q_2 and that \tilde{x} is located such that the line (\tilde{x}, \hat{x}) intersects Q_2 only in \hat{x} . Let $\pi x \leq \pi \hat{x}$ be an optimal cut separating \tilde{x} from Q_2 thus passing through \hat{x} . Let π^\perp be the projection of π on the orthogonal of the vector space induced by the vector $\tilde{x} - \hat{x}$. Finally let us assume that the cut $\pi^\perp x \leq \pi^\perp \hat{x}$ is valid for Q_2 . This is for instance the case in the 2D example illustrated in Figure 6b as the hyperplane $\pi^\perp x = \pi^\perp \hat{x}$ is, in this case, the line (\tilde{x}, \hat{x}) . Then for all $\alpha \geq 0$, the cut $(\pi + \alpha\pi^\perp)x \leq (\pi + \alpha\pi^\perp)\hat{x}$ is also an optimal cut for the separation problem. Indeed, it is valid for Q_2 as a non-negative combination of valid cuts for Q_2 . Moreover, the violation of this

cuts by \tilde{x} is the same as the violation of $\pi x \leq \pi \hat{x}$ because by construction of π^\perp the product $\pi^\perp(\tilde{x} - \hat{x})$ is equal to zero. Thus, in this example, the separation problem is bounded because its dual is feasible but the set of optimal cuts is unbounded because the norm of $\pi + \alpha\pi^\perp$ approaches infinity when α does the same.

Despite this unbounded set of solutions, only the facets of the polyhedron Q_2 are vertices of the solution space of (S) . Thus, if the algorithm solving (S) always returns a vertex, the generated cut will always be a facet of Q_2 .



(a) Cut associated with directional normalization (b) Different normals of optimal cuts for the directional normalization: the norm of optimal normals can tend to infinity

Fig. 6: Examples of results obtained when using directional normalization.

The above normalizations have been presented several times in the literature and are applicable in the general case. However, certain problems can admit normalizations particularly adapted to their structure. One can find such normalizations for example in the separation problems of disjunctive programming [2]. In addition, the unsplittable flow problem, which will be used in our experimental study, admits a normalization that seems natural and which be presented in Section 7.1.

5 A new approach for the Fenchel sub-problem

In this section, we present a new approach to solve the separation problem of the Fenchel decomposition when the directional normalization presented in Section 4 is used.

5.1 Presentation of the method

The proposed method is described in Algorithm 1 and illustrated in Figure 7. Note that the cut associated with a directional normalization toward \hat{x} is the same for \tilde{x} than for any other point of the segment (\tilde{x}, \hat{x}) not belonging to Q_2 . The underlying idea is to compute intermediate cuts using an alternative normalization. By projecting a point x' (initially equal to \tilde{x}) onto these intermediate cuts, the algorithm gradually shifts the point x' along the segment (\tilde{x}, \hat{x}) in the direction of \hat{x} . Once x' reaches the frontier of Q_2 the procedure stops. Indeed, the last cut generated is the one associated with the directional normalization toward \hat{x} since it contains the intersection between the segment (\tilde{x}, \hat{x}) and the frontier of Q_2 .

Algorithm 1 New approach for solving the Fenchel sub-problem

Input: a polyhedron Q_2 , a point \tilde{x} to separate from Q_2 , a point \hat{x} belonging to Q_2
Output: a cut C separating \tilde{x} from Q_2 and containing the intersection point the frontier of Q_2 and the segment (\tilde{x}, \hat{x}) , a list LS of vertices of Q_2 satisfying the cut C to equality
1: Set x' equal to \tilde{x}
2: **while** $x' \notin Q_2$ **do**
3: $C, LS = \text{Secondary_separation}(Q_2, x')$
4: Set x' equal to the intersection point between the segment (\tilde{x}, \hat{x}) and the cut C
5: **return** C, LS

The advantage of this iterative approach is that the linear separation program (S) is never directly solved with the directional normalization, which often presents numerical instabilities. Indeed, for this normalization, the separation problem (S) seeks a face of the polyhedron Q_2 intersecting the segment (\tilde{x}, \hat{x}) . If, for example, this segment intersected a facet of Q_2 by forming an angle close to 0 with it then a small error on the parameters of the segment or of the facet can induce a large error on the position of the point of intersection. For this reason, the problem (S) is sometimes too numerically unstable to be solved directly. The new method presented above yields an alternative way of computing the cut associated with the directional normalization without ever solving the numerically unstable linear program of the direct method. Unfortunately, note that this iterative algorithm requires the computation of the intersection between the segment (\tilde{x}, \hat{x}) and the cuts returned by the secondary separation problem. If the segment is almost parallel to one of these cuts, the computation of this point of intersection may still be numerically unstable. However, this is only an intermediate point not returned by the separation oracle. The results of the method are a cut and vertices which are computed with a secondary normalization. In practice, this seems sufficient as no numerical instabilities were found during our thorough numerical campaign.

Our experimental section yields insights into the practical performance of this new approach for the separation oracle. From a theoretical standpoint, we provide proofs in A that the method converges in a finite number of iterations in two cases: 1) when the secondary normalization always generates facets of

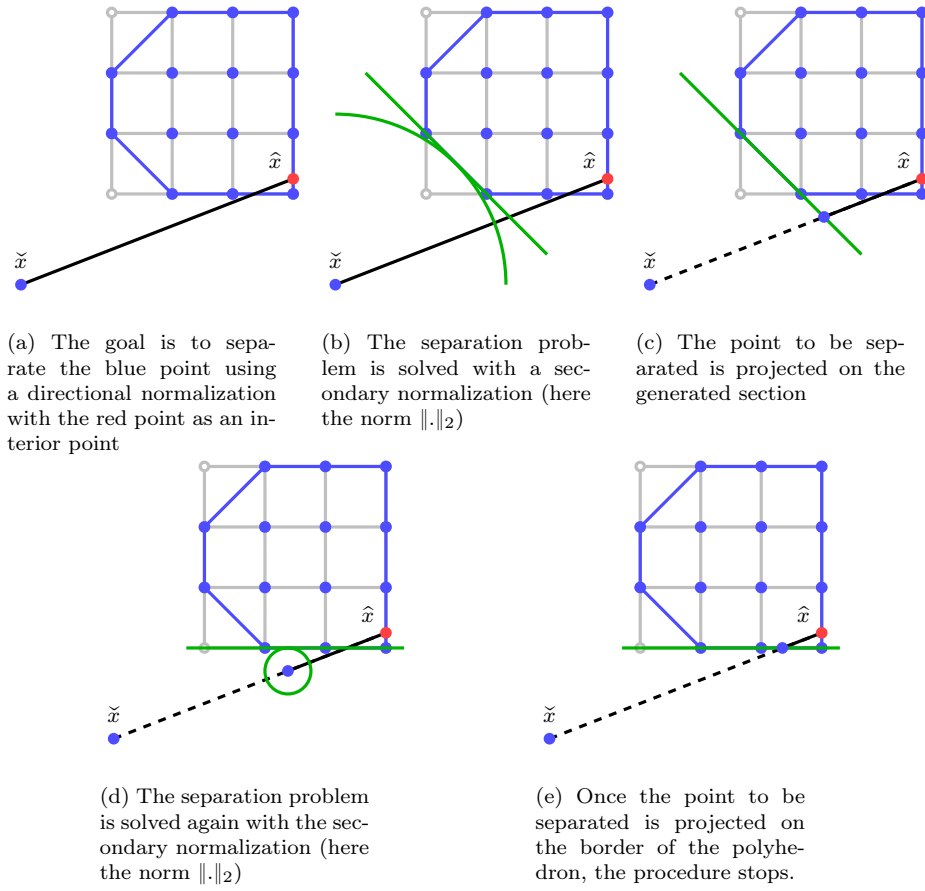


Fig. 7: Example of solving the Fenchel sub-problem for directional normalization using a secondary normalization

Q_2 , and 2) when the secondary normalization is $\|\pi\| \leq 1$ for any norm. These two cases cover most normalizations used in the literature including all those mentioned in Section 4.

6 Coupling the Fenchel and Dantzig-Wolfe decompositions

In this section, we present a decomposition method that integrates Dantzig-Wolfe and Fenchel decompositions. In our experimental campaign, we show that the new method presents a much superior performance than Fenchel decomposition alone, and is competitive against Dantzig-Wolfe on non-degenerate problems. For degenerate problems, again the proposed method shows a superior performance to a classical Dantzig-Wolfe decomposition.

6.1 Presentation of the method

When generating a Fenchel cut, the primal variables of the separation sub-problem (S) are the coefficients of the generated cut and the active constraints correspond to the vertices of the separated polyhedron Q_2 verifying this cut at equality. Thus, the Fenchel sub-problem generates both valid cuts for the polyhedron Q_2 and vertices of this polyhedron.

The main idea of the new decomposition method is to use two master problems operating in tandem. The first corresponds to the Fenchel formulation (F) in which the generated Fenchel cuts are added to improve an outer approximation of Q_2 . The second one is the Dantzig-Wolfe formulation (DW) in which the vertices generated in the separation sub-problem are added to grow an inner approximation of Q_2 . One of the key points of the method is that points \tilde{x} provided by the Fenchel master problem are separated using directional normalization. This normalization requires the knowledge of a point in the polyhedron Q_2 , for which we use the solution \hat{x} computed by the Dantzig-Wolfe master problem.

The steps of the algorithm, illustrated in Figure 8, are as follows:

1. Initialize an inner approximation \hat{Q}_2 of the polyhedron Q_2 as in the Dantzig-Wolfe decomposition and an outer approximation \check{Q}_2 as in the Fenchel decomposition.
2. Optimize over $LR_1 \cap \check{Q}_2$ using a Fenchel master problem: a solution \tilde{x} is obtained whose value is an upper bound of the problem.
3. Optimize over $LR_1 \cap \hat{Q}_2$ using a Dantzig-Wolfe master problem: a solution \hat{x} is obtained whose value is a lower bound of the problem.
4. If the two bounds are equal: end of the algorithm
5. With the Fenchel separation problem (S), separate the point \tilde{x} from Q_2 using directional normalization with \hat{x} as the interior point: a cut is obtained as well as vertices of Q_2 .
6. Add the cut to the outer approximation \check{Q}_2 and the vertices to the inner approximation \hat{Q}_2 . Then, go to Step 2.

Although this new decomposition method uses ideas taken from both the Dantzig-Wolfe and the Fenchel decompositions, the way in which they operate yields a few remarkable observations and interpretations that are worth describing:

- Intuitively, the inner and outer approximations constructed disagree on the location of the boundary of Q_2 on the segment (\tilde{x}, \hat{x}) . The separation problem finds the exact position of the frontier and gives information to the two approximations so that they can approximate exactly this part of the frontier.
- Another way of looking at the method is to say that it gradually improves the point \hat{x} . At each iteration, the algorithm tests a direction of potential improvement $\tilde{x} - \hat{x}$. The method then finds either new vertices of Q_2 allowing to improve \hat{x} or a facet of Q_2 passing through \hat{x} proving that it is not possible to improve \hat{x} in this direction. This interaction between an exterior

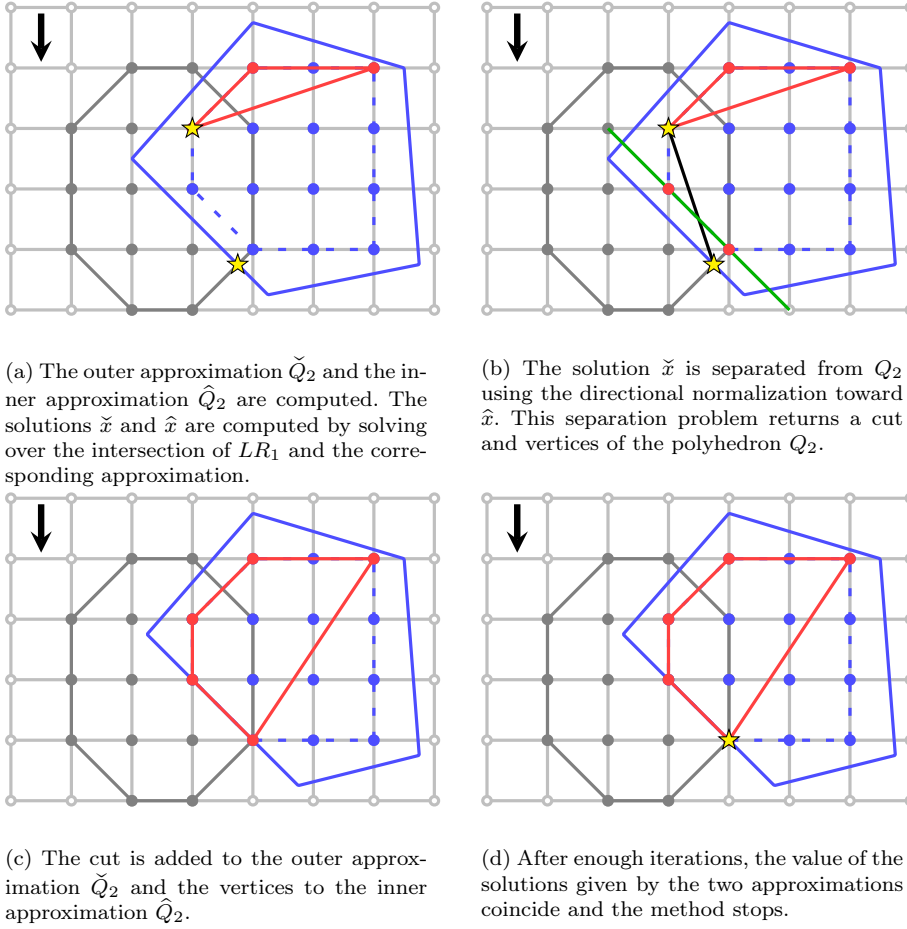


Fig. 8: Illustration of the Dantzig-Wolfe-Fenchel decomposition

point and an interior point of Q_2 is reminiscent of the *in-out* separation proposed for the Benders decomposition [6].

- Compared to a classic Fenchel decomposition, this method concentrates its cut generation around the point \hat{x} . It thus refines the knowledge of the polyhedron Q_2 around this point. On the other hand, a classical Fenchel decomposition might spend multiple iterations searching for cuts in regions of the solution space that end up being far from the optimum.
- Compared to a classic Dantzig-Wolfe method, this method devotes more time to the resolution of its sub-problem which allows it to generate a greater number of vertices to add to the master problem.

6.2 Degeneracy

The Dantzig-Wolfe decomposition is known to present convergence issues when its master problem is highly degenerate. Our experiments reveal that the method presented in this section does not suffer from the same issues. In the following, we give a partial theoretical explanation of the absence of degeneracy issues in the new method.

A linear program is said to be degenerate when it has multiple dual optima. We have seen in Section 3.1 that each dual solution of the Dantzig-Wolfe master problem implies a bound on the value of its objective function $cx \leq c_0$ that certifies the optimality of its current primal solution on the solution set $LR_1 \cap \hat{Q}_2$. Thus, in order to improve the solution of the master problem, one needs to be able to separate each of these dual solutions from the polyhedron $LR_1 \cap \hat{Q}_2$. However, the sub-problem of the Dantzig-Wolfe decomposition may only separate one such dual vector which explains why the Dantzig-Wolfe decomposition is so affected by degeneracy.

The new proposed Dantzig-Wolfe-Fenchel decomposition seems to be unaffected by the degeneracy of its Dantzig-Wolfe master problem. One possible explanation of this behavior is supported in the following observation and proposition. First, let us remark that the new decomposition neither computes nor uses any dual information, therefore showing no sensitivity to the quality of the duals. Second:

Proposition 1 *At each iteration, the sub-problem invalidates either all the dual solutions of the Dantzig-Wolfe master program or none of them.*

Proof The sub-problem finds the farthest point in Q_2 along the segment (\hat{x}, \tilde{x}) . On the one hand, if this farthest point coincides with \hat{x} then it does not invalidate any of the bounds implied by the dual solutions because \hat{x} belongs to $LR_1 \cap \hat{Q}_2$. On the other hand, suppose the farthest point x^* does not coincide with \hat{x} . First, by construction, x^* belongs to Q_2 but also to LR_1 because it is a convex combination of \hat{x} and \tilde{x} , both belonging to LR_1 . Second, assuming that \tilde{x} has a strictly better objective value than \hat{x} (which is always the case except when the method is about to terminate) then x^* also has a strictly better objective value than \hat{x} . Thus, it must invalidate all the bounds implied by the dual solutions of the Dantzig-Wolfe master problem (remember that these bounds certify that no better solution than \hat{x} exists). \square

Therefore, whether the Dantzig-Wolfe master problem admits multiple dual optima or not does not influence the algorithm's capacity to find a strictly improving point.

Although we presented all the decomposition methods as if only one polyhedron was decomposed at once (e.g. a block of a block diagonal matrix), in practice several polyhedra are decomposed at the same time (e.g. all the blocks of a block diagonal matrix). In this case, Proposition 1 does not hold. However, the ideas discussed in its proof may still impact positively the practical computations. In any case, the decomposition method still does not use any

dual information which renders it oblivious to the number of dual solutions of its Dantzig-Wolfe master problem.

7 Application to the unsplittable flow problem

In the unsplittable flow problem (UFP), one is given a weighted directed graph $\mathcal{G} = (V, A, c)$ where c_a is the capacity of the arc a for every $a \in A$. We are also given a family K of commodities, each composed of an origin $o(k)$, a destination $d(k)$, and a demand D_k , for every $k \in K$. Each commodity has to be routed from its origin to its destination through an unique path. We consider the problem where the capacity constraints are soft, meaning that they can be violated at a certain unit penalty. The objective is to design routing paths for every commodity on the network so as to minimize the sum of the violations of the arcs' capacities.

The UFP is an extensively studied NP-hard variant of the classic maximum-flow problem. It has multiple applications, as for instance in telecommunication networks (*e.g.* optical networks, telecommunication satellites [18, 31]), and logistics [23]. The early work of Belaidouni and Ben-Ameur [5] studied the UFP from a polyhedral perspective, proposing cutting planes to strengthen the linear relaxation of a three-index model that uses variables x_{ijk} for every arc (i, j) in the network and every commodity k to be transported. Park, Kim, and Lee [41] also strengthen the linear relaxation by applying the Dantzig-Wolfe decomposition to the capacity constraints of the UFP. The resulting relaxation is as strong as if all the inequalities valid for the capacity constraints of the problem were added. Thus, faster decomposition methods able to compute the Dantzig-Wolfe linear relaxation could yield improvements in the resolution of the UFP.

We will consider an arc-path formulation where the meaning of the variables is the following:

- x_{pk} indicates whether commodity k uses path p to push its flow,
- Δ_a represents the overflow on arc a .

In addition to the decision variables, we also denote, for a given commodity $k \in K$, P_k the set of all $o(k)$ - $d(k)$ -paths in G . For every $p \in P_k$ and arc $a \in A$, we define a constant α_{ap} that takes the value 1 iff p uses the arc a . The path formulation of the UFP is the following:

$$\min_{x_{pk}, \Delta_a} \sum_{a \in A} \Delta_a \quad (9a)$$

$$\text{subject to } \sum_{p \in P_k} x_{pk} = 1 \quad \forall k \in K, \quad (9b)$$

$$\sum_{k \in K} \sum_{p \in P_k, a \in p} x_{pk} D_k \leq c_a + \Delta_a \quad \forall a \in A, \quad (9c)$$

$$x_{pk} \in \{0, 1\}, \Delta_a \in \mathbb{R}^+ \quad \forall p \in P_k, \forall k \in K, \forall a \in A \quad (9d)$$

The objective function minimizes the sum of the overflows on the arcs. Equation (9b) ensures that exactly one path is chosen for each commodity. Equation (9c) corresponds to the soft capacity constraints. It ensures that any overflow on an arc a is recorded on the corresponding variable Δ_a . The fact that $x_{pk} \in \{0, 1\}$ ensures that the flow is unsplittable.

The polyhedron associated with the capacity constraints does not have the integrality property and its relaxation can thus be tightened with any of the previously discussed decomposition methods. If we denote $f_a^k = \sum_{p \in P_k, a \in p} x_{pk}$, this polyhedron can be written as follows:

$$\left\{ f_a^1, \dots, f_a^k \in [0, 1], o_a \in \mathbb{R}^+ \mid \sum_{k \in K} f_a^k D^k \leq c_a + o_a \right\}.$$

Studies have been carried out on the structure, the cut selection, and the strengthening of the linear relaxation of this type of polyhedron by Marchand and Wolsey [36] as well as in the more general framework of linear programs in mixed variables [21, 24, 16]. Moreover, optimization methods on this polyhedron have been studied by Büther and Briskorn [14], Lin, Zhu, and Ali [34], Zhao and Li [46], He et al. [28], and Liu [35].

In the following, we present a specialized normalization for this polyhedron. This normalization guarantees the generation of facets and will be used in our implementation of the Fenchel decomposition and as secondary normalization in the new procedure to solve the Fenchel sub-problem presented in Section 4. Then, we describe our implementation of the oracle that optimizes a linear function on the polyhedron associated with the capacity constraints of the UFP.

7.1 Natural normalization for unsplittable flows

Let $\check{f} = (\check{f}^k)_{k \in K} \in [0, 1]^{|K|}$ be a flow distribution for each commodity which induces an overflow \check{o} on a given arc. In this section, we assume that the arc is fixed and will therefore drop the arc index for the sake of simplicity. In the context of unsplittable flows, vertices of the polyhedron Q_2 correspond to commodity patterns which will be indexed by a superscript g . A naturally occurring question is: how does this distribution break down into a combination of commodity patterns inducing a minimum capacity overflow? This question

can be solved using the following linear program:

$$\begin{aligned}
 (D) \quad & \min_{\lambda^g, z} \quad z \\
 & \text{subject to} \quad \sum_{g \in G} \lambda^g f^g = \check{f} \\
 & \quad \sum_{g \in G} \lambda^g o^g = \check{o} + z \\
 & \quad \sum_{g \in G} \lambda^g = 1 \\
 & \quad \lambda^g \in \mathbb{R}^+, z \in \mathbb{R}^+ \quad \forall g \in G
 \end{aligned}$$

where λ^g is the coefficient in the decomposition associated with a commodity pattern f^g inducing an overflow o^g .

Now the dual of this decomposition program is the following program:

$$\begin{aligned}
 (S) \quad & \max_{\pi, \pi_o, \pi_0} \quad \pi \check{f} + \pi_o \check{o} - \pi_0 \\
 & \text{subject to} \quad \pi f^g + \pi_o o^g \leq \pi_0, \quad \forall (f^g, o^g) \in Q_2 \\
 & \quad \pi_o \leq -1 \\
 & \quad \pi, \pi_0, \pi_o \in \mathbb{R}
 \end{aligned}$$

This program corresponds exactly to the problem (S) of separating the point $\check{x} = (\check{f}_1, \dots, \check{f}_{|K|}, \check{o})$ with a constraint of normalization imposing that the coefficient π_o associated with the overflow variable satisfies $\pi_o \leq -1$. This normalization constraint is what we will call the natural normalization for the unsplittable flow problem. This normalization is very close to a particular case of directional normalization for the direction $\hat{x} - \check{x} = (0, \dots, 0, 1)$. Just like directional normalization, the natural normalization guarantees the generation facets because it is imposed using a single linear constraint.

7.2 Knapsack oracle resolution

All the decomposition methods presented in this work assume that there exists an efficient algorithm capable of optimizing a linear function on the polyhedron Q_2 . In this section, we detail the problem solved by the oracle in the context of unsplittable flows.

In the version of the unsplittable flow problem that we are studying, the capacity constraints do not require that the flow of commodities respect the capacities c_a of the arcs. However, the overflow must be stored in a variable o_a . Thus, the polyhedron of variables satisfying the soft capacity constraint associated with the arc a is written:

$$\left\{ (f_a^k \in \{0, 1\})_{a \in A, k \in K}, o_a \in \mathbb{R}^+ \left| \sum_{k \in K} f_a^k D^k \leq c_a + o_a \right. \right\}.$$

The optimization of a linear function whose coefficients are $(\pi^k)_{k \in K}$ and $-\pi_o$ on this polyhedron can be written as follows:

$$\begin{aligned}
 (O_a) \quad & \max_{f_a^k, o_a} \quad \sum_{k \in K} \pi^k f_a^k - \pi_o o_a \\
 & \text{subject to} \quad \sum_{k \in K} f_a^k D^k \leq c_a + o_a \\
 & \quad f_a^k \in \{0, 1\}, \quad o_a \in \mathbb{R}^+
 \end{aligned}$$

This problem can be solved as a sequence of two 0-1 knapsack problems using a case disjunction. This method was presented by Büther and Briskorn [14] and is recalled in B. In our experiments, we use the MINKNAP algorithm proposed by Pisinger [43] to solve the two associated knapsack problems.

8 Experimental study

In this section, we present an experimental comparison of different decomposition methods. The datasets and code used in this section can be accessed at https://github.com/TwistedNerves/decomposition_paper_code. The code was written in Python 3 and the experiments carried on an Intel Core i9-9900K 3.60 GHz $\times 16$ cores CPU, 60 Gbit of RAM, running Ubuntu 20.10.

8.1 Datasets

An instance of the unsplittable flow problem is composed of a graph and a list of commodities. The method used to create instances in our experiments is the one presented in Lamothe et al. [32]. All the graphs used are strongly connected random graphs. To create demands for the commodities, Lamothe et al. [32] used two formulas. In this work, we use the formula that creates mainly commodities with large demands because it tends to create instances that are harder to solve. Moreover, in each instance, all the commodities can be unsplittably routed without exceeding the arc capacities. Therefore, the lower bound given by the linear relaxation is optimal. In order to create an optimality gap in the instances, we slightly modify the capacities of some arcs as follows a number of times equal to 100 times the number of nodes:

- Randomly select the origin of a commodity.
- Randomly select two arcs coming out of this origin.
- Add 1 to the capacity of one arc and subtract 1 from the ability of the other arc.

Because of the way instances are created, all outgoing arcs from each origin node are saturated in the solutions without overflow while the other arcs are often non-saturated. Therefore, in most cases, transferring some capacity

between outgoing arcs of origins does not change the value of the linear relaxation. On the other hand, this transfer of capacity can have an impact on the value of the best unsplittable solution. Indeed, there is no longer necessarily a combination of commodities whose sum of demands is exactly equal to the capacity of each arc. In this case, the best unsplittable solution has a non-zero overflow.

Another change made to the instances is that the commodities have only access to a restricted set of paths to push their flow. The restricted set of paths of a commodity is chosen to be the k -shortest paths from the origin to the destination of the commodity with $k = 4$. Because this study explores the strengthening of the linear relaxation of the unsplittable flow problem through its capacity constraints, this modification should not change the relative behavior of the tested algorithms but does make the instances much simpler to solve which enables the testing of the different algorithms on larger instances.

The datasets Three different datasets are used during the experiments in which ten instances are generated for each value of the varying parameter.

- *Low maximum demand* dataset: this dataset considers strongly connected random graphs from 50 nodes to 145 nodes. The maximum commodity demand is set at $\tilde{d}_{max} = 100$ and the arc capacity at 1000. This choice of maximum demand implies that a large number of commodities can pass through each arch. However, in our tests, the optimal solution for these instances often does not contain overflow. Therefore, these instances do not contain an optimality gap. We hypothesize that the large number of commodities allows them to rearrange themselves to exactly fill the capacity of each arc.
- *High maximum demand* dataset: this dataset considers strongly connected random graphs from 145 nodes to 1000 nodes. The maximum commodity demand is set at $\tilde{d}_{max} = 1000$ and the arc capacity at 1000. Because of this maximum demand choice, these instances contain only a small number of commodities. However, they generally have an optimality gap which allows us to study the evolution of the lower bounds given by the algorithms.
- *Size of capacities* dataset: this dataset considers strongly connected random graphs of 70 nodes. The maximum demand of the \tilde{d}_{max} commodities is fixed at $1/10$ of the common capacity of the arcs which varies from 100 to 100,000. The knapsack problem is known to have algorithms that are pseudo-polynomial in the capacity of the knapsack. One such algorithm is the MINKNAP algorithm we use. In the case of unsplittable flows, the capacity of the knapsack corresponds to the capacity of the arcs. The instances of this dataset all have the same structure (same graph size, same size of commodities relative to the capacity of the arcs) but varying arc capacities. This impacts the resolution time of the MINKNAP algorithm.

8.2 Comparison of Dantzig-Wolfe variants

A large number of works have proposed variations and improvements to Dantzig-Wolfe decomposition. Indeed, the method presented in Section 3.1 sometimes suffers from a slow convergence which has been associated with the following observations [42]:

- **Dual oscillations:** the dual variables π used to generate the vertices of Q_2 perform large oscillations and do not converge monotonically toward their optimal value.
- **The tailing-off effect:** during the last iterations, the space of the dual solutions is only marginally reduced and the dual bound progresses very slowly.
- **Degenerate primal and equivalent dual solutions:** The master problem (DW) is regularly degenerate because it has several dual optimal solutions. The method iterates between equivalent dual solutions without making progress on the value of the objective function.

In order to overcome these difficulties, stabilization methods for the dual variables have been considered. These methods can be classified into three categories [42]:

- **Penalization:** the penalization methods are best interpreted by considering the dual of the problem (DW). In order to stabilize the dual variables π , a penalty $f(\|\pi - \bar{\pi}\|)$ is added to the objective function of the dual. In this penalty, $f(\cdot)$ is an increasing function, which pushes π to stay close to a value $\bar{\pi}$ which evolves slowly during the algorithm. Typically, $\bar{\pi}$ is one of the values taken by the dual variables during the previous iterations. A widely studied special case is to penalize proportionally to $\|\pi - \bar{\pi}\|_2^2$, which is done in the *Bundle* methods [13].
- **Smoothing:** in smoothing methods, the dual variables π of the problem (DW) are not used directly in the sub-problem in order to generate new vertices of Q_2 . We note, for the iteration j of column generation, π_j the values of the variables resulting from the dual of the problem (DW) and $\bar{\pi}_j$ the values used in the sub-problem. A smoothing method proposed by Neame [39] uses the following formula: $\bar{\pi}_j = \alpha \bar{\pi}_{j-1} + (1 - \alpha) \pi_j$. This method amounts to adding an *momentum* effect to the dual variables. Another method, proposed by Wentges [44], performs a convex combination with a fixed dual value $\bar{\pi}$, i.e. $\bar{\pi}_j = \alpha \bar{\pi} + (1 - \alpha) \pi_j$.
- **Centralization:** The idea of centralization methods is that it is more efficient to use in the sub-problem dual values located inside the dual polyhedron rather than on an extreme vertex of the dual polyhedron. On the other hand, such interior points are more expensive to compute than extreme points. The interior point used in the Primal-Dual Column Generation [26] is obtained by approximately solving the problem (DW) by an interior point method. Another classic point is the analytical center used in the analytical center cutting plane method [25].

In order to have a suitable comparison for the decomposition methods we proposed in this work, we experimentally compared the following three variations of Dantzig-Wolfe decomposition:

DW: Dantzig-Wolfe decomposition method. No stabilization of the dual variables is used. The lower bounds are computed using the dual variables and the value of the solution of the knapsack sub-problems.

DW-momentum: similar to the *DW* method except that the dual variables are stabilized by smoothing using the formula of Neame [39], $\bar{\pi}_t = \alpha\bar{\pi}_{t-1} + (1 - \alpha)\pi_t$ with the coefficient α set to 0.8.

DW-interior-point: similar to the *DW* method except that the Dantzig-Wolfe master problem is solved with an interior point method in order to return a non-optimal but centered solution. To that end, we ask the [27] solver to solve the linear program using an interior point method with a precision of 10^{-3} and without using its *crossover* method. However, the first time the sub-problem fails to generate a new negative reduced cost variable, the solver Gurobi is reset to its default settings to ensure an exact computation of the last reduced costs. With the default parameters, the generation of columns is no longer stabilized.

These methods are compared in Figure 9. For the rest of the experiments, we will use the variation based on the interior point solver as it always returns the best results in our tests.

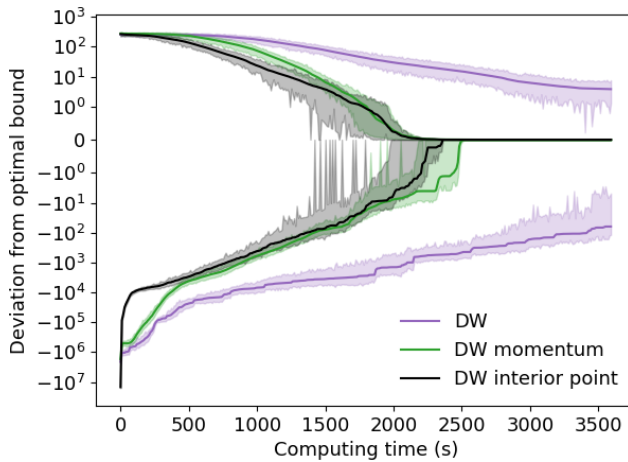


Fig. 9: Compararison of Dantzig-Wolfe decomposition variations, *Low maximum demand* dataset, 70 nodes

8.3 The decomposition methods studied

In the following, we experimentally compare the following decomposition methods:

Fenchel: Fenchel decomposition method. The cuts generated are added to the linear relaxation while the generated vertices are added to a Dantzig-Wolfe formulation. The Fenchel sub-problem is solved with the natural normalization presented in Section 7.1. Therefore, the two master problems do not act in tandem.

DW-Fenchel: method combining the Fenchel and Dantzig-Wolfe decompositions presented in Section 6, the cuts generated are added to the linear relaxation while the generated vertices are added to a Dantzig-Wolfe formulation. The Fenchel sub-problem is solved with directional normalization with the optimal point of the Dantzig-Wolfe formulation as the interior point. The use of this normalization couples the two formulations.

DW-Fenchel-iterative: similar to the **DW-Fenchel** method except that the Fenchel sub-problem is solved using the iterative method presented in Section 5.

DW-interior-point: Dantzig-Wolfe decomposition introduced in Section 8.2 where the master problem is solved with an interior point method in order to return a non-optimal but centered solution.

8.4 Algorithms' parameters

Authorized paths Because the focus of this work is on the capacity constraints and not on how to generate the paths for each commodity, each commodity is restrained to a small set of allowed paths. This set is made up of the four shortest paths between the origin and destination of the commodity as well as the path used to create the commodity in the method of Lamothe et al. [32].

Algorithm termination condition The decomposition methods considered are stopped when the absolute difference between their bounds is 10^{-3} .

Pre/post-processings for the sub-problem of Fenchel: Solving directly a Fenchel subproblem is sometimes too computationally expensive to be integrated into a decomposition method. However, the resolution time of this sub-problem can be greatly reduced with pre/post-processing steps. Indeed, Boccia et al. [8] showed that it is possible to solve the Fenchel separation problem by focusing on a sub-polyhedron of Q_2 of far lesser dimension which decreases the computing time. However, the generated cut is not directly valid for Q_2 and one must use a *lifting* procedure to create a cut valid for Q_2 . These concepts are explained in C. Together, the techniques of *Dimensionality reduction* and *Lifting* induce a drastic decrease in the resolution time of the Fenchel subproblem. This fact was confirmed by our experiments and the results we present in our experimental study reflect this algorithmic choice.

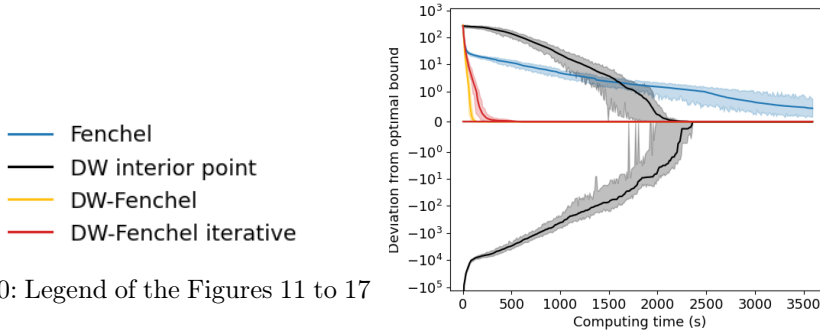


Fig. 10: Legend of the Figures 11 to 17

Fig. 11: *Low maximum demand dataset, 70 nodes*

8.5 Experimental results

We now present the results of our experimental campaign for the different decomposition methods presented in this work. In each figure, we display the evolution of the lower and upper bounds achieved by the algorithms as a function of the computational time (in seconds). Note that all the displayed values are not directly the bounds but their deviation from the value of an optimal solution of the Dantzig-Wolfe reformulation. The plotted curves represent the average results of the algorithms aggregated on instances using the same parameters while the confidence intervals at 95% for the mean are plotted in semi-transparency around the main curve. A problem encountered when generating these curves is that the algorithms only return bounds at the end of each of their iterations but these iterations take a variable time for the same algorithm depending on the instance. It is therefore not possible to directly aggregate the curves using the points given at the end of each iteration because they do not correspond to the same computing time. To obtain points on which we can appropriately average the values, the points defining the curves are replaced with points sampled every ten seconds by considering that the bounds evolve linearly between two iterations. The confidence intervals are created using the statistical method called Bootstrapping with a number of resamplings equal to 1000. Because the Bootstrapping method is applied independently for every ten seconds of the curves the resulting confidence intervals have jitters. These jitters can be interpreted as the uncertainty on the bound of the confidence intervals due to the Bootstrapping method.

Solving the Fenchel sub-problem with a secondary normalization. The new method of solving the Fenchel sub-problem presented in Section 5 is used in the DW-Fenchel-iterative method. This method appears to be slightly slower than the DW-Fenchel method which uses a direct approach for the sub-problem. On the other hand, the direct approach sometimes fails to solve the sub-problem because of numerical instabilities which prevent the decomposition method

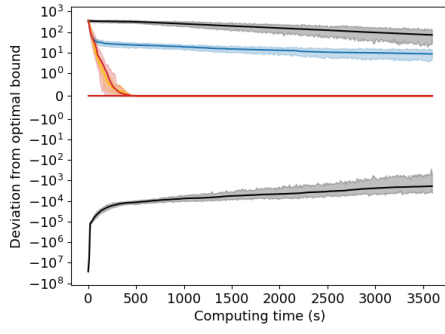


Fig. 12: *Low maximum demand* dataset, 90 nodes

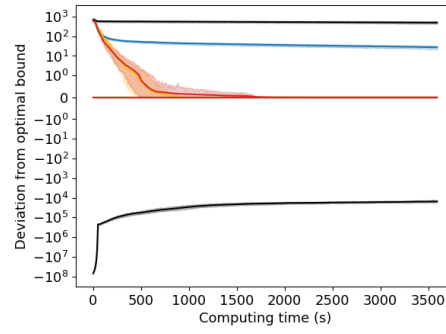


Fig. 13: *Low maximum demand* dataset, 145 nodes

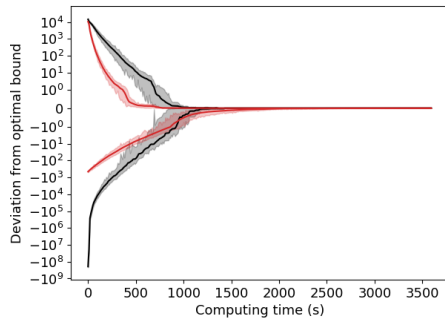


Fig. 14: *High maximum demand* dataset, 250 nodes

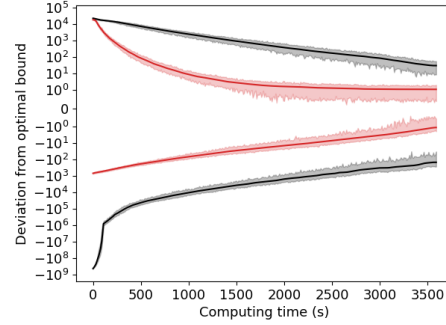


Fig. 15: *High maximum demand* dataset, 400 nodes

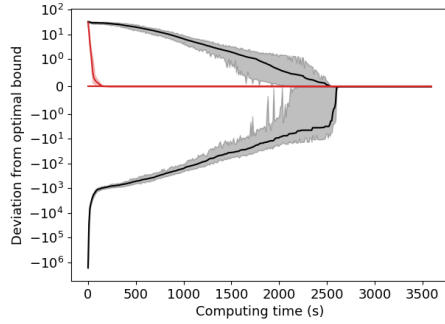


Fig. 16: *Size of commodities* dataset, capacity 100

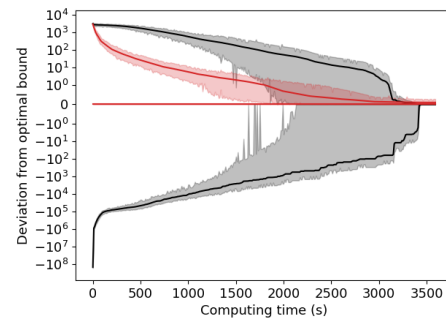


Fig. 17: *Size of commodities* dataset, capacity 10000

from converging. For instances with 145 nodes of the *Low maximum demand* dataset, this happens every 10 to 20 instances. The 10 instances presented in Figure 13 did not suffer from instability in this set of experiments thus it does not appear in the figure. However, we were able to identify seeds for which the instability appears. Unfortunately, these seeds appear to be hardware-dependent thus researchers trying to reproduce the results will have to find their own seeds.

Impact of coupling the two master problems using directional normalization. The DW-Fenchel and DW-Fenchel-iterative methods couple the Dantzig-Wolfe and Fenchel master problems using a directional normalization in their Fenchel sub-problem. The impact of this coupling can be studied by comparing these methods to the Fenchel method whose only difference is to use the natural normalization of the unsplittable flow problem in its sub-problem. As illustrated in Figure 12, one notes that all the methods are similar during the first iterations. However, the Fenchel method stalls rapidly. On the other hand, this is not the case with the methods using directional normalization which yield much better results. Our interpretation of this phenomenon is as follows. There are many equivalent optimal solutions of the linear relaxation of the unsplittable flow problem. When a cut is generated with the natural normalization, it only cuts a subset of these solutions, and the new solution to the Fenchel master problem is in a completely different place in the solution space. The method then fails to cut all the solutions because of this large number of symmetries. In contrast, the cuts generated using directional normalization focus on the optimal solution of the Dantzig-Wolfe master problem and try to prove its optimality. By focusing on a sub-part of the solution space, this method avoids the problem of symmetries which improves its convergence. This hypothesis is supported by the results of a preliminary study on a variation of the unsplittable flow problem where a path is favored. Indeed, because of the presence of a privileged path for each commodity, this variant does not have as many symmetries. In this context, there is a smaller difference between the methods based on the two normalizations.

Comparison between DW-interior-point and DW-Fenchel-iterative. In the case of the *low maximum demand* dataset where the number of commodities is high, the DW-Fenchel-iterative method behaves a lot better than the Dantzig-Wolfe methods. We assume that this is because the large number of commodities implies a greater degeneracy of the master problem which should less bother the DW-Fenchel methods. Indeed, this degeneracy seems to be the cause of the rather slow start of the Dantzig-Wolfe methods on these instances. In contrast, for the *high maximum demand* dataset, the DW-interior-point and DW-Fenchel-iterative methods show more similar results. On these instances, the DW-Fenchel-iterative method shows a faster start of convergence, but slows down at the end of convergence, in particular for the lower bound.

Impact of capacity size. The knapsack problem is known to have pseudo-polynomial resolution algorithms in the knapsack capacity such as the MIN-KNAP algorithm that we use. In the case of unsplittable flows, this capacity of the knapsack corresponds to the capacity of the arcs. In Figure 16 and 17,

we vary the capacities of the arcs. Note that the results for capacities of 1000 are given in Figure 11. This variation in capacities impacts the computation time of the two methods making them slower. However, the DW-Fenchel-iterative method is much more impacted because the methods having a Fenchel sub-problem spend more time in their sub-problem than the Dantzig-Wolfe methods. This emphasizes the fact that having a fast optimization oracle for the polyhedron Q_2 is much more important for the methods based on a Fenchel sub-problem than those based on a Dantzig-Wolfe sub-problem.

General comments. The new methods presented that couples the Dantzig-Wolfe and Fenchel decompositions shows very promising results. In particular, they seems to be far less affected by degeneracy than the Dantzig-Wolfe decomposition and possess better convergence than the Fenchel decomposition. On the other hand, they can end up converging slightly less rapidly than the Dantzig-Wolfe decomposition on instances where degeneracy is not an issue. The new methods are particularly effective when the optimization oracle (O) can be implemented by a fast algorithm.

9 Conclusions

In this work we have revisited Dantzig-Wolfe and Fenchel decompositions for some hard combinatorial problems with block structures. We have provided geometrical and intuitive interpretations of several types of normalizations used in the literature to stabilize the sub-problems associated. This intuition has fueled the development of a novel methodology capable of coupling both decomposition approaches acting in tandem via a directional normalization. We have conducted a thorough computational campaign to demonstrate the effectiveness of the newly proposed approach for the unsplittable flow problem. We have observed that on problems suffering from high degrees of degeneracy, the new approach is superior to its competitors. Meanwhile, it is also competitive with the classical approaches on the less degenerate cases. We also proposed a new approach to solve the Fenchel subproblem with directional normalization by using an alternative normalization. We provide theoretical guarantees for the finiteness of this new approach for several classes of alternative normalizations and our experimental campaign revealed that it presents far less numerical instabilities.

A likely lead for future research will therefore be to investigate the performance of this new method in different contexts than the unsplittable flow problems. Moreover, one of the central points of this new method is the use of directional normalization in the Fenchel sub-problem. It would be interesting to use this normalization inside other decomposition methods.

References

- [1] Pasquale Avella, Maurizio Boccia, and Igor Vasilyev. “A computational study of exact knapsack separation for the generalized assignment prob-

- lem". In: *Computational Optimization and Applications* 45.3 (2010), pp. 543–555. DOI: 10.1007/s10589-008-9183-8.
- [2] Egon Balas and Michael Perregaard. "Lift-and-project for mixed 0–1 programming: recent progress". In: *Discrete Applied Mathematics* 123.1–3 (2002), pp. 129–154. DOI: 10.1016/S0166-218X(01)00340-7.
- [3] Cynthia Barnhart et al. "Branch-and-Price: Column Generation for Solving Huge Integer Programs". In: *Operations Research* 46.3 (1998), pp. 316–329. DOI: 10.1287/opre.46.3.316.
- [4] Eric Beier et al. "Stage-and scenario-wise fenchel decomposition for stochastic mixed 0-1 programs with special structure". In: *Computers & Operations Research* 59 (2015), pp. 94–103. DOI: <https://doi.org/10.1016/j.cor.2014.12.011>.
- [5] Meriema Belaidouni and Walid Ben-Ameur. "On the minimum cost multiple-source unsplittable flow problem". In: *RAIRO-Operations Research* 41.3 (2007), pp. 253–273. DOI: 10.1051/ro:2007023.
- [6] Walid Ben-Ameur and José Neto. "Acceleration of cutting-plane and column generation algorithms: Applications to network design". In: *Networks: An International Journal* 49.1 (2007), pp. 3–17. DOI: 10.1002/net.20137.
- [7] JF Benders. "Partitioning procedures for solving mixed-variables programming problems." In: *Numerische Mathematik* 4 (1962), pp. 238–252. DOI: 10.1007/BF01386316.
- [8] Maurizio Boccia et al. "A cut and branch approach for the capacitated p-median problem based on Fenchel cutting planes". In: *Journal of mathematical modelling and algorithms* 7.1 (2008), pp. 43–58. DOI: 10.1007/s10852-007-9074-5.
- [9] Pierre Bonami. "Etude et mise en œuvre d’approches polyédriques pour la résolution de programmes en nombres entiers ou mixtes généraux". PhD thesis. Paris 6, 2003.
- [10] Karl Heinz Borgwardt. *The Simplex Method: a probabilistic analysis*. Vol. 1. Springer Science & Business Media, 2012. DOI: /10.1007/978-3-642-61578-8.
- [11] E Andrew Boyd. "Generating Fenchel cutting planes for knapsack polyhedra". In: *SIAM Journal on Optimization* 3.4 (1993), pp. 734–750. DOI: 10.1137/0803038.
- [12] E Andrew Boyd. "On the convergence of Fenchel cutting planes in mixed-integer programming". In: *SIAM Journal on Optimization* 5.2 (1995), pp. 421–435. DOI: 10.1137/0805021.
- [13] Olivier Briant et al. "Comparison of bundle and classical column generation". In: *Mathematical programming* 113.2 (2008), pp. 299–344. DOI: 10.1007/s10107-006-0079-z.
- [14] Marcel Büther and Dirk Briskorn. "Reducing the 0-1 knapsack problem with a single continuous variable to the standard 0-1 knapsack problem". In: *International Journal of Operations Research and Information Systems (IJORIS)* 3.1 (2012), pp. 1–12. DOI: 10.4018/joris.2012010101.

- [15] Liang Chen et al. “An exact separation algorithm for unsplittable flow capacitated network design arc-set polyhedron”. In: *Journal of Global Optimization* (2021), pp. 1–31. DOI: 10.1007/s10898-020-00967-z.
- [16] Vašek Chvátal, William Cook, and Daniel Espinoza. “Local cuts for mixed-integer programming”. In: *Mathematical Programming Computation* 5.2 (2013), pp. 171–200. DOI: 10.1007/s12532-013-0052-9.
- [17] Michele Conforti and Laurence A Wolsey. ““Facet” separation with one linear program”. In: *Mathematical Programming* 178.1 (2019), pp. 361–380. DOI: 10.1007/s10107-018-1299-8.
- [18] D. Coudert and H. Rivano. “Lightpath assignment for multifibers WDM networks with wavelength translators”. In: *Global Telecommunications Conference, 2002. GLOBECOM '02. IEEE*. Vol. 3. 2002, 2686–2690 vol.3. DOI: 10.1109/GLOCOM.2002.1189117.
- [19] George B Dantzig. “Maximization of a linear function of variables subject to linear inequalities”. In: *Activity analysis of production and allocation* 13 (1951), pp. 339–347.
- [20] George B Dantzig and Philip Wolfe. “Decomposition principle for linear programs”. In: *Operations research* 8.1 (1960), pp. 101–111. DOI: 10.1287/opre.8.1.101.
- [21] Sanjeeb Dash. “Mixed integer rounding cuts and master group polyhedra”. In: *Combinatorial Optimization*. IOS Press, 2011, pp. 1–32. DOI: 10.3233/978-1-60750-718-5-1.
- [22] Guy Desaulniers, Jacques Desrosiers, and Marius M Solomon. *Column generation*. Vol. 5. Springer Science & Business Media, 2006. DOI: 10.1007/b135457.
- [23] Judith M Farvolden, Warren B Powell, and Irvin J Lustig. “A primal partitioning solution for the arc-chain formulation of a multicommodity network flow problem”. In: *Operations Research* 41.4 (1993), pp. 669–693. DOI: 10.1287/opre.41.4.669.
- [24] Ricardo Fukasawa and Marcos Goycoolea. “On the exact separation of mixed integer knapsack cuts”. In: *Mathematical programming* 128.1 (2011), pp. 19–41. DOI: 10.1007/s10107-009-0284-7.
- [25] Jean-Louis Goffin and Jean-Philippe Vial. “Convex nondifferentiable optimization: A survey focused on the analytic center cutting plane method”. In: *Optimization methods and software* 17.5 (2002), pp. 805–867. DOI: 10.1080/1055678021000060829a.
- [26] Jacek Gondzio, Pablo González-Brevis, and Pedro Munari. “New developments in the primal–dual column generation technique”. In: *European Journal of Operational Research* 224.1 (2013), pp. 41–51. DOI: 10.1016/j.ejor.2012.07.024.
- [27] LLC Gurobi Optimization. *Gurobi Optimizer Reference Manual*. 2020. URL: <http://www.gurobi.com>.
- [28] Yichao He et al. “Encoding transformation-based differential evolution algorithm for solving knapsack problem with single continuous variable”. In: *Swarm and Evolutionary Computation* 50 (2019), p. 100507. DOI: 10.1016/j.swevo.2019.03.002.

- [29] Konstantinos Kaparis and Adam N Letchford. “Separation algorithms for 0-1 knapsack polytopes”. In: *Mathematical programming* 124.1 (2010), pp. 69–91. DOI: 10.1007/s10107-010-0359-5.
- [30] L.G. Khachiyan. “Polynomial algorithms in linear programming”. In: *USSR Computational Mathematics and Mathematical Physics* 20.1 (1980), pp. 53–72. ISSN: 0041-5553. DOI: 10.1016/0041-5553(80)90061-0.
- [31] François Lamothe et al. “Dynamic unsplittable flows with path-change penalties: New formulations and solution schemes for large instances”. In: *Computers & Operations Research* 152 (2023), pp. 106–154. ISSN: 0305-0548. DOI: 10.1016/j.cor.2023.106154.
- [32] François Lamothe et al. “Randomized rounding algorithms for large scale unsplittable flow problems”. en. In: *Journal of Heuristics* (2021). DOI: 10.1007/s10732-021-09478-w. (Visited on 09/10/2021).
- [33] AH Land and AG Doig. “An Automatic Method of Solving Discrete Programming Problems”. In: *Econometrica* 28.3 (1960), pp. 497–520. DOI: 10.1007/978-3-540-68279-0_5.
- [34] Geng Lin, Wenxing Zhu, and M Montaz Ali. “An exact algorithm for the 0–1 linear knapsack problem with a single continuous variable”. In: *Journal of global optimization* 50.4 (2011), pp. 657–673. DOI: 10.1007/s10898-010-9642-5.
- [35] Hongtao Liu. “An exact algorithm for the biobjective 0-1 linear knapsack problem with a single continuous variable”. In: *2017 18th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT)*. IEEE. 2017, pp. 81–85. DOI: 10.1109/PDCAT.2017.00022.
- [36] Hugues Marchand and Laurence A Wolsey. “The 0-1 knapsack problem with a single continuous variable”. In: *Mathematical Programming* 85.1 (1999), pp. 15–33. DOI: 10.1007/s101070050044.
- [37] Jiří Matoušek and Bernd Gärtner. *Understanding and using linear programming*. Vol. 33. Springer, 2007. DOI: 10.1007/978-3-540-30717-4.
- [38] Sanjay Mehrotra. “On the implementation of a primal-dual interior point method”. In: *SIAM Journal on optimization* 2.4 (1992), pp. 575–601. DOI: 10.1137/0802028.
- [39] Philip James Neame. *Nonsmooth dual methods in integer programming*. University of Melbourne, Department of Mathematics and Statistics, 2000.
- [40] Lewis Ntaimo. “Fenchel decomposition for stochastic mixed-integer programming”. In: *Journal of Global Optimization* 55.1 (2013), pp. 141–163. DOI: 10.1007/s10898-011-9817-8.
- [41] Sungsoo Park, Deokseong Kim, and Kyungsik Lee. “An integer programming approach to the path selection problems”. In: *Proceedings of the International Network Optimization Conference INOC, Evry-Paris, France*. 2003, pp. 448–453.
- [42] Artur Pessoa et al. “In-out separation and column generation stabilization by dual price smoothing”. In: *International Symposium on Experi-*

-
- mental Algorithms*. Springer. 2013, pp. 354–365. DOI: 10.1007/978-3-642-38527-8_31.
- [43] David Pisinger. “A minimal algorithm for the 0-1 knapsack problem”. In: *Operations Research* 45.5 (1997), pp. 758–767. DOI: 10.1287/opre.45.5.758.
- [44] Paul Wentges. “Weighted Dantzig-Wolfe decomposition for linear mixed-integer programming”. In: *International Transactions in Operational Research* 4.2 (1997), pp. 151–162. DOI: 10.1016/S0969-6016(97)00001-4.
- [45] Laurence A Wolsey and George L Nemhauser. *Integer and combinatorial optimization*. Vol. 55. John Wiley & Sons, 1999. DOI: 10.1002/9781118627372.
- [46] Chenxia Zhao and Xianyu Li. “Approximation algorithms on 0–1 linear knapsack problem with a single continuous variable”. In: *Journal of Combinatorial optimization* 28.4 (2014), pp. 910–916. DOI: 10.1007/s10878-012-9579-3.

A Proof of the finite convergence for the iterative resolution of the Fenchel sub-problem

In this appendix, we give proofs of the finite convergence for the method presented in Section 5 for two types of secondary normalizations. First, when the secondary normalization guarantees that a facet of Q_2 will be generated by the linear program. Second, when the secondary normalization is $\|\pi\| \leq 1$ where $\|\cdot\|$ is any norm.

A.1 Secondary normalization generating facets

In this section, we are interested in the termination of the proposed method to solve the Fenchel sub-problem when the secondary normalization guarantees that a facet of Q_2 will be generated by the separation linear program.

Theorem 2 *If the secondary normalization guarantees the generation of facets then the method generates a cut associated with a directional normalization in finite time.*

Proof We will show that in the worst case the method ends after the secondary separation has generated all the facets of Q_2 . For this, we show that each facet of Q_2 is generated at most once. As the algorithm progresses, the point x' separated during secondary separations advances along the segment (\tilde{x}, \hat{x}) in the direction of \hat{x} . Once a facet of Q_2 is generated, the point x' is projected onto that facet along the segment (\tilde{x}, \hat{x}) . The future points x' will therefore all satisfy the inequality associated with this facet which can thus no longer be generated by the secondary separation problem. Since a polyhedron has a finite number of facets, a facet intersecting the segment (\tilde{x}, \hat{x}) is generated in a finite number of steps. Once this happens, the method ends after a single call to the alternate separation problem. Indeed, the procedure places the point x' on the point of intersection between this facet and the segment (\tilde{x}, \hat{x}) . On the next iteration, the secondary separation problem indicates that the point x' belongs to Q_2 and the method stops. \square

A.2 Secondary normalization using any norm

We now present a proof of convergence when the secondary normalization is $\|\pi\| \leq 1$ where $\|\cdot\|$ is any norm. This proof uses the Lemmas 1 and 2 presented in Boyd [12]. Previously, we recall the following properties and notations. In the case of a normalization $\|\pi\| \leq 1$, the dual problem of the separation problem is: find the point of Q_2 minimizing the distance $\|x - \tilde{x}\|^*$ where $\|\cdot\|^*$ is the dual norm of $\|\cdot\|$: $\|\lambda\|^* = \max_{\|\pi\| \leq 1} \lambda x$. The solution point of the dual problem is denoted x^* and always satisfies at equality the cut generated by the separation problem. We now present the lemmas used in the proof.

Lemma 1 (Boyd [12]) *Let $\lambda^* x \leq \lambda^* x^*$ be the cut generated during the separation of a point \hat{x} from a polyhedron Q_2 with the normalization $\|\pi\| \leq 1$ where x^* is the optimal solution of the dual of the separation problem. Then $-\lambda^*$ is a sub-gradient of $x \mapsto \|x - \tilde{x}\|^*$ in x^* .*

In the second lemma $\angle(\lambda, x)$ will denote the angle between the vectors λ and x (the one lower than π radiant).

Lemma 2 (Boyd [12]) *For each norm, there exists an angle $\theta_{min} > 0$ such that at any point x and for any sub-gradient λ of this norm in x :*

$$\angle(\lambda, x) \geq \frac{\pi}{2} - \theta_{min}$$

We now present the main theorem of this section.

Theorem 3 *If the secondary normalization is $\|\pi\| \leq 1$ for any norm then the method generates a cut associated with a directional normalization in finite time.*

Proof Scheme of the proof: First we will show that once a face intersecting the segment (\check{x}, \hat{x}) has been generated the method ends after a single call to the secondary separation problem. Secondly, we will show that if another face of Q_2 is generated, the point x' advances more than ϵ along the segment (\check{x}, \hat{x}) toward \hat{x} where ϵ is a strictly non-negative distance independent of the iteration. Thus, this second case cannot happen more than $\frac{\|\hat{x} - \check{x}\|_2}{\epsilon}$ times so the procedure ends in a finite number of steps.

1) Suppose that at one iteration, a face intersecting the segment (\check{x}, \hat{x}) is generated. After generating the face, the procedure places the point x' on the intersection point. At the next iteration, the secondary separation problem indicates that the point x' belongs to Q_2 and the method stops.

2) For the rest of this proof, we will denote by $x^{(i)}$ the point x' separated during iteration i of the algorithm. We will show that if a face F of Q_2 generated does not intersect the segment (\check{x}, \hat{x}) , then the point x' advances along the segment (\check{x}, \hat{x}) a strictly non-negative distance $d_{\min} \sin(\theta_{\min})$ independent of the face F , i.e. $\|x^{(i+1)} - x^{(i)}\|_2 \geq d_{\min} \sin(\theta_{\min})$. To that end, we will use trigonometry on the triangle formed by the points associated with $x^{(i)}$, x^* , and $x^{(i+1)}$ where x^* is the dual optimal solution of the secondary separation problem. This triangle will be denoted $\triangle x^{(i)} x^* x^{(i+1)}$. We will use θ_{\min} as lower bound for the angle $\angle(x^{(i+1)} - x^*, x^{(i)} - x^*)$ and d_{\min} as lower bound for the distance $\|x^{(i)} - x^*\|_2$.

2.1) Suppose that at iteration i , the secondary separation problem of $x^{(i)}$ returns a primal-dual solution pair (λ^*, x^*) which thus correspond to the cut $\lambda^* x \leq \lambda^* x^*$. Recall that x^* is a point of Q_2 satisfying the generated cut to equality. It is thus on the generated face F . From Lemma 1, since the secondary normalization is $\|\pi\| \leq 1$, the vector $-\lambda^*$ is a sub-gradient of $\|x - x^{(i)}\|_2$ in x^* . Thus, according to Lemma 2, we have $\angle(-\lambda^*, x^* - x^{(i)}) \leq \frac{\pi}{2} - \theta_{\min}$ for a $\theta_{\min} > 0$ depending on the used norm but neither on the face F nor on the iteration. This is equivalent to $\frac{\pi}{2} - \angle(\lambda^*, x^{(i)} - x^*) \geq \theta_{\min}$. After generating the cut $\lambda^* x \leq \lambda^* x^*$, the procedure projects the point $x^{(i)}$ on the hyperplane $\lambda^* x = \lambda^* x^*$ along the segment (\check{x}, \hat{x}) . The result of this projection is the point $x^{(i+1)}$. Since both $x^{(i+1)}$ and x^* are point of the hyperplane $\lambda^* x = \lambda^* x^*$, the vector $x^{(i+1)} - x^*$ is on the hyperplane $\lambda^* x = 0$. Let us consider the smallest angle between $x^{(i)} - x^*$ and a point of the hyperplane $\lambda^* x = 0$. Firstly, it is smaller than the angle $\angle(x^{(i+1)} - x^*, x^{(i)} - x^*)$. Secondly, this smallest angle can be expressed in terms of the normal λ^* as $\frac{\pi}{2} - \angle(\lambda^*, x^{(i)} - x^*)$ which is shown above to be greater than θ_{\min} . Thus, we have shown that **the angle $\angle(x^{(i+1)} - x^*, x^{(i)} - x^*)$ is greater than θ_{\min} .**

2.2) Let d_{\min} be the distance in norm $\|\cdot\|_2$ between the segment (\check{x}, \hat{x}) and the union of the faces of Q_2 that do not intersect the segment (\check{x}, \hat{x}) . Since the generated face F does not intersect the segment (\check{x}, \hat{x}) , the distance between the segment (\check{x}, \hat{x}) and the face F is greater than d_{\min} . However, $x^{(i)}$ belongs to the segment (\check{x}, \hat{x}) and x^* to the face F so **the distance $\|x^{(i)} - x^*\|_2$ is greater than d_{\min} .**

2.3) We will now show with simple trigonometry on the triangle $\triangle x^{(i)} x^* x^{(i+1)}$ that $\|x^{(i+1)} - x^{(i)}\|_2 \geq d_{\min} \sin(\theta_{\min})$. First, the distance $\|x^{(i+1)} - x^{(i)}\|_2$ is larger than the length of the altitude of the triangle $\triangle x^{(i)} x^* x^{(i+1)}$ associated with $x^{(i)}$. However, the length of this altitude is $\|x^{(i)} - x^*\|_2 \sin(\angle(x^{(i)} - x^*, x^{(i+1)} - x^*))$ which according to paragraph 2.1) and 2.2) is greater than $d_{\min} \sin(\theta_{\min})$. Thus finally, we have: $\|x^{(i+1)} - x^{(i)}\|_2 \geq d_{\min} \sin(\theta_{\min})$.

The distance $d_{\min} \sin(\theta_{\min})$ is strictly non-negative and independent of the iteration of the algorithm which completes the proof. \square

B Solution of the knapsack oracle

All the decomposition methods presented in this work assume that there exists an efficient algorithm capable of optimizing a linear function on the polyhedron Q_2 which in our version of the unsplittable flow problem can be written as:

$$\{(f_a^k \in \{0, 1\})_{a \in A, k \in K} \mid \sum_{k \in K} f_a^k D_k \leq c_a + o_a\}.$$

Optimizing a linear function whose coefficients are $(\pi^k)_{k \in K}$ and $-\pi_o$ on this polyhedron can thus be done by following mixed integer linear program:

$$\begin{aligned} (O_a) \quad & \max_{f_a^k, o_a} \quad \sum_{k \in K} \pi^k f_a^k - \pi_o o_a \\ & \text{subject to} \quad \sum_{k \in K} f_a^k D_k \leq c_a + o_a \\ & \quad f_a^k \in \{0, 1\}, \quad o_a \in \mathbb{R}^+ \end{aligned}$$

This problem can be addressed by solving two classic 0-1 knapsack problems using a case disjunction. This method was presented by Büther and Briskorn [14] but we recall it here. Consider the following disjunction: either the flow of the commodities respects the capacity of the arc a , or the flow of the commodities exceeds the capacity of the arc a . Finding the best solution in the first case amounts to solving the following problem:

$$\begin{aligned} & \max_{f_a^k} \quad \sum_{k \in K} \pi^k f_a^k \\ & \text{subject to} \quad \sum_{k \in K} f_a^k D_k \leq c_a \\ & \quad f_a^k \in \{0, 1\} \end{aligned}$$

Indeed, we suppose that the coefficient π_o is non-negative because otherwise the problem O_a would be unbounded. Since the flow of commodities respects the capacity of the arc, the variable o_a always takes the value zero and can be removed from the problem. Note that in this first case of the disjunction, the problem to be solved is a classic knapsack problem. In the second case of disjunction, finding the best solution amounts to solving the following problem:

$$\begin{aligned} & \max_{f_a^k, o_a} \quad \sum_{k \in K} \pi^k f_a^k - \pi_o o_a \\ & \text{subject to} \quad \sum_{k \in K} f_a^k D_k \leq c_a + o_a \\ & \quad \sum_{k \in K} f_a^k d^k \geq c_a \\ & \quad f_a^k \in \{0, 1\}, \quad o_a \in \mathbb{R}^+ \end{aligned}$$

Since we assume that π_o is non-negative and that the flow of commodities does not respect the capacity of the arcs, the variable o_e is always equal to $\sum_{k \in K} f_a^k D_k - c_a$. By performing the replacement in the objective function, by replacing the variables f_a^k by their

complement $\bar{f}_a^k = 1 - f_a^k$ and by multiplying the constraint $\sum_{k \in K} f_a^k D_k \geq c_a$ by -1 we get the following reformulation:

$$\begin{aligned} & \max_{\bar{f}_a^k} \quad \sum_{k \in K} (\pi_o D_k - \pi^k) \bar{f}_a^k + C \\ & \text{subject to} \quad \sum_{k \in K} \bar{f}_a^k D_k \leq \sum_{k \in K} D_k - c_a \\ & \quad \bar{f}_a^k \in \{0, 1\} \end{aligned}$$

where the constant C is equal to $\sum_{k \in K} (\pi^k - \pi_o d^k) - \pi_o c_a$. This reformulation shows that the problem to be solved in the second case of the disjunction is also a classic knapsack problem.

Note: in the experimental study of Section 8, in order to solve these two knapsack problems, we use the MINKNAP algorithm proposed by Pisinger [43]. Since MINKNAP only accepts integer-valued weights and profits, we multiply the dual vector (π, π_o, π_o) by 10^7 and truncate before invoking MINKNAP.

C Pre/post-processing for Fenchel subproblems

Solving directly a Fenchel subproblem is sometimes too computationally expensive to be integrated into a decomposition method. However, the resolution time of this subproblem can be greatly reduced with pre/post-processing steps. Indeed, Boccia et al. [8] showed that it is possible to solve the Fenchel separation problem by focusing on a sub-polyhedron of Q_2 of far lower dimension. However, the cut generated may not be valid for Q_2 and one must use a *lifting* procedure to create a cut valid for Q_2 . These concepts are explained in the following.

Dimensionality reduction: This technique typically applies when the variables x of the initial problem (P) are binary. Instead of separating a point \tilde{x} from Q_2 , this point is separated from the sub-polyhedron $Q_2^f(\tilde{x})$ induced by the variables taking a value different from their bounds in \tilde{x} (in the binary case, induced by variables taking a fractional value in \tilde{x}). More precisely, $Q_2^f(\tilde{x}) = \{x \in Q_2 \mid x_i = \tilde{x}_i \text{ if } \tilde{x}_i \in \{0, 1\}\}$. Indeed, \tilde{x} belongs to Q_2 if and only if it belongs to $Q_2^f(\tilde{x})$ which implies that \tilde{x} is separable from Q_2 if and only if it is separable from $Q_2^f(\tilde{x})$. Replacing Q_2 with $Q_2^f(\tilde{x})$ in the separation problem has two advantages. First, the problem only considers variables taking a fractional value in \tilde{x} which reduces the number of dimensions of the problem and therefore speeds up its resolution. Forcing certain variables to take a binary value in the optimization oracle on Q_2 is also often easy to do. Second, if the separation problem (S) is solved directly through constraint generation, most of the computational time is used to generate constraints associated with vertices of Q_2 not belonging to $Q_2^f(\tilde{x})$ and which are therefore mostly unnecessary. Reducing the search space to $Q_2^f(\tilde{x})$ allows the computation to concentrate on the part of the space containing the most important vertices.

Lifting: Upon applying the dimensionality reduction described before, the cut generated will be valid for $Q_2^f(\tilde{x})$ but not necessarily for Q_2 . A procedure able to create a valid cut for Q_2 from one valid for $Q_2^f(\tilde{x})$ is the sequential lifting procedure [45] that we will now describe. In the following, we will name fixed variables, the variables eliminated from the separation problem when replacing the polyhedron Q_2 by the polyhedron $Q_2^f(\tilde{x})$. For the fixed variables, the coefficients of the generated cut are all zero. At each step of the sequential lifting procedure, a new value for the coefficient of one of the fixed variables is computed so that the cut becomes valid for the polyhedron where this variable is no longer fixed. These new coefficients are optimal in the sense that if the initial cut was a facet of the polyhedron $Q_2^f(\tilde{x})$ then the cut resulting from the sequential lifting is a facet of Q_2 . We now present an iteration of the sequential lifting procedure. Suppose therefore that $Q_2^f(\tilde{x}) = \{x \in Q_2 \mid x_1 = 1\}$; the procedure will then contain only one iteration because

only the variable x_1 is fixed. The cut generated for $Q_2^f(\check{x})$ is $\pi x \leq \pi_0$ where the coefficient π_1 associated with the variable x_1 is zero. The lifting procedure consists in creating a new cut $\gamma x \leq \gamma_0$ such that the two cuts are identical when $x_1 = 1$ and such that the new cut is valid for Q_2 . Since $\pi x \leq \pi_0$ is already valid for $Q_2^f(\check{x})$, the two previous conditions can be written:

$$\begin{aligned} \forall x \in \{x | x_1 = 1\}, \quad \gamma x - \gamma_0 &= \pi x - \pi_0 \\ \max_{x \in Q_2 \mid x_1=0} \gamma x &\leq \gamma_0 \end{aligned}$$

Let e_i be the i^{th} vector of the canonical basis. Applying the first condition for $x = e_1$, we get $\gamma_1 - \gamma_0 = \pi_1 - \pi_0$. Moreover, by applying it in $x = e_1 + e_i$ for all $i > 1$, we obtain $\gamma_1 + \gamma_i - \gamma_0 = \pi_1 + \pi_i - \pi_0$ which becomes $\gamma_i = \pi_i$ by subtracting the previous equality. We therefore know:

$$\begin{aligned} \forall i > 1, \gamma_i &= \pi_i \\ \gamma_1 &= \gamma_0 + \pi_1 - \pi_0 \end{aligned}$$

Thus, the value of γ_0 is the only element missing to know the value of all the γ_i . By replacing γ_i by π_i for all $i > 1$ in the second condition we get $\gamma_0 \geq \max_{x \in \{x \in Q_2 \mid x_1=0\}} \pi x$. The smallest value of γ_0 satisfying the second condition is, therefore, $\gamma_0 = \max_{x \in \{x \in Q_2 \mid x_1=0\}} \pi x$ which can be computed with a call to the optimization oracle on Q_2 where the variable x_1 is set to 0. It is therefore possible to compute each lifted coefficient using a single call to the oracle (O).