Les Cahiers du GERAD ISSN: 0711–2440

Integral simplex using decomposition with primal cuts

S. Rosat, I. Elhallaoui,
F. Soumis, A. Lodi

G–2013–79

November 2013
Revised: January 2016

CITATION ORIGINALE / ORIGINAL CITATION

Rosat, S., Elhallaoui, I., Soumis, F., Lodi, A., Integral simplex using decomposition with primal cuts. In Experimental Algorithms, Proceedings of the 13th International Symposium, SEA 2014, Copenhagen, Denmark, Lecture Notes in Computer Science Series, Vol. 8504, pp. 22-33, 2014, Springer International Publishing, http://dx.doi.org/10.1007/978-3-319-07959-2_3.

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Integral simplex using decomposition with primal cuts

Samuel Rosat [a]
Issmail Elhallaoui [a]
François Soumis [a]
Andrea Lodi [b,c]

[a] *GERAD & Polytechnique Montréal, Montréal (Québec) Canada, H3C 3A7*

[b] *DEIS, Università di Bologna, 40136, Bologna, Italy*

[c] *Now at GERAD, Canada Excellence Research Chair in Data Science for Real-Time Decision Making, & Polytechnique Montréal, Montréal (Québec) Canada, H3C 3A7*

samuel.rosat@polymtl.ca
issmail.el-hallaoui@polymtl.ca
francois.soumis@gerad.ca
andrea.lodi@polymtl.ca

November 2013
Revised: January 2016

Les Cahiers du GERAD

G–2013–79

Abstract: The integral simplex using decomposition (ISUD) algorithm [22] is a dynamic constraint reduction method that aims to solve the popular set partitioning problem (SPP). It is a special case of primal algorithms, i.e. algorithms that furnish an improving sequence of feasible solutions based on the resolution, at each iteration, of an augmentation problem that either determines an improving direction, or asserts that the current solution is optimal. To show how ISUD is related to primal algorithms, we introduce a new augmentation problem, **MRA**. We show that **MRA** canonically induces a decomposition of the augmentation problem and deepens the understanding of ISUD. We characterize cuts that adapt to this decomposition and relate them to *primal cuts*. These cuts yield a major improvement over ISUD, making the mean optimality gap drop from 33.92% to 0.21% on some aircrew scheduling problems.

Key Words: Integer programming, primal algorithms, cutting planes, primal cuts, constraint aggregation, decomposition, set partitioning.

Résumé : Le Simplexe en Nombres Entiers avec Décomposition (*Integral Simplex Using Decomposition, ISUD*) [22] est un algorithme de résolution du problème de partitionnement d'ensembles (*Set Partitioning Problem, SPP*) basé sur la réduction dynamique des contraintes. Il fait partie de la famille des algorithmes primaux : à partir d'une première solution réalisable, une suite de solutions d'objectif décroissant est déterminée. Pour passer d'une solution à la suivante, il suffit de résoudre le problème dit d'augmentation : déterminer une direction d'amélioration, ou bien certifier que la solution courante est optimale. Pour exhiber le lien entre ISUD et les algorithmes primaux, nous introduisons un nouveau problème d'augmentation, **MRA**. Nous montrons que la décomposition canoniquement associée à **MRA** est celle utilisée dans ISUD, ce qui permet d'approfondir la compréhension générale d'ISUD. Nous caractérisons les familles de coupes adaptées à cette décomposition et mettons en évidence leur lien avec les *coupes primales*. L'ajout de telles coupes permet un gain en performance par rapport à ISUD et fait notamment passer le gap d'optimalité moyen de 33.92% 0.21% sur certains groupes d'instances en planification d'horaires d'équipages aériens.

Acknowledgments: The authors thank Matthieu Delorme for his help with the implementation, Jean-Bertrand Gauthier for numerous fruitful discussions and Cem Unlubayrak who completed an internship under the supervision of Samuel Rosat.

1 Introduction

1.1 The set partitioning problem

We consider the set partitioning problem (SPP):

$$\begin{aligned} \min \quad & c{\bullet}x \\ \text{s.t.} \quad & Ax = e \\ & x \in \mathbb{B}^n \end{aligned} \tag{$\mathbb{P}$}$$

where $\mathbb{B} = \{0, 1\}$, A is an $m \times n$ binary matrix, $c \in \mathbb{R}^n$, and $e = (1, \ldots, 1) \in$. $\mathcal{M}$ and $\mathcal{N}$ respectively denote the set of indices of the rows and columns of A. $\mathbb{P}'$ is the linear program obtained from $\mathbb{P}$ by replacing the binary constraint $x \in \mathbb{B}^n$ by a nonnegativity constraint $x_j \geqslant 0, j \in \{1, \ldots, n\}$. We can assume without loss of generality that A has no zero or identical rows or columns. A_j is the j^{th} column of A, and for any set of columns T and rows U, A_T^U is the submatrix $(A_{ij})_{i \in U, j \in T}$. Moreover, for any optimization program P, $\mathcal{F}_P$ designates its feasible set.

In this paper, we present an efficient and promising primal framework for the solution of $\mathbb{P}$. It combines and extends previous work concerning *primal algorithms* on the one hand and *constraint aggregation* on the other hand.

1.2 Primal algorithms

As noted by Letchford and Lodi [10], algorithms for integer programming can be divided into three classes: *dual fractional*, *dual integral*, and *primal*. *Dual fractional* algorithms maintain optimality and linear-constraint feasibility at every iteration, and they stop when integrality is reached. They are typically standard cutting plane procedures such as Gomory's algorithm [8]. *Dual integral* methods maintain integrality and optimality, and they terminate once the primal linear constraints are satisfied. Letchford and Lodi give a single example, another algorithm of Gomory [9]. Finally, *primal methods* maintain feasibility (and integrality) throughout the process and stop when optimality is reached. These are in fact descent algorithms for which the decreasing sequence $(x_k)_{k=1\ldots K}$ satisfies:

(H0) $x^k \in \mathcal{F}_{\mathbb{P}}$ (incl. integrality constraints);
(H1) x^K is optimal;
(H2) $c{\bullet}x^{k+1} < c{\bullet}x^k$ (decreasing sequence).

In this review, we give an overview of the context of our work only (for a more extensive review, see [16]). Primal methods are sometimes classified as *augmenting algorithms* and include the so-called *integral simplex* procedures. They were first introduced in [2] and [20] and improved in [21] and [6]. In Young's method [20, 21], at iteration k, a simplex pivot is considered: if it leads to an integer solution, it is performed. Otherwise, cuts are generated and added to the problem, thereby changing the underlying structure of the constraints. Young also developed the concept of a *decreasing vector* (sometimes called an *improving direction*) at x^k, i.e., a direction $z \in \mathbb{R}^n$ s.t. $x^k + z$ is integer, feasible, and of lower cost than x^k. From this notion comes the *primal augmentation problem* (**AUG**) that involves finding such a direction if it exists or asserting that x^k is optimal. Traditionally, papers on constraint aggregation and integral simplex algorithms deal with minimization problems, whereas authors usually present generic primal algorithms for maximization problems. We therefore draw the reader's attention to the following: **to retain the usual classification, we call the improving direction problem AUG, although it supplies a decreasing direction**.

Recently, there has been a renewed interest in primal integer algorithms, inspired by Robert Weismantel (a.o.). Many recent works specifically concern 0/1-programming. However, only a few papers have addressed the practical solution of **AUG**; most of them consider it an oracle.

Most of the rare computational work since 2000 on primal algorithms concerns the *primal separation problem* (**P-SEP**): given x^k a current solution of $\mathbb{P}$ and an infeasible point $x^\star$ (typically a vertex of the linear relaxation), is there a hyperplane that separates $x^\star$ from $\mathcal{F}_{\mathbb{P}}$ and is tight at x^k? In 2003, Eisenbrand et al. [3] proved that the primal separation problem is as difficult as the integral optimization problem for 0/1-programming. It is therefore expected to be a "complicated" problem because 0/1-programming is $\mathcal{NP}$-hard. Letchford and Lodi [10, 11] and Eisenbrand et al. [3] adapt well-known algorithms for the standard separation problem to primal separation. To the best of our knowledge, very few papers present computational experiments using primal methods [13, 17, 10]. All these papers present results on relatively small instances. Only Letchford and Lodi presented results for an algorithm using primal cutting planes and, interestingly, they stated that degeneracy prevented them from solving larger instances.

Last but not least, *integral simplex methods* were first proposed for the SPP by Balas and Padberg [1]. They were the first to propose an augmenting method, specific to SPP, that yields a sequence of feasible solutions (x^k) satisfying (H0)–(H2) and also

(H3) x^{k+1} is a neighbor of x^k in $\mathcal{F}_{\mathbb{P}'}$.

This property does not prevent the algorithm from reaching optimality since the SPP is *quasi-integral* [19], i.e., every edge of $Conv(\mathcal{F}_{\mathbb{P}})$ is also an edge of its linear relaxation $\mathcal{F}_{\mathbb{P}'}$. Balas and Padberg were therefore able to base their algorithm on a sequence of well-chosen simplex pivots for the linear relaxation $\mathbb{P}'$. Other integral simplex methods have since been presented [22, 18, 14, 12].

The integral simplex using decomposition (ISUD) of Zaghrouti et al. [22] adapts Elhallaoui et al.'s work on constraint reduction for linear programming [5, 4] to the SPP. Zaghrouti et al. present promising numerical results for much larger instances than those in [10]. However, ISUD uses neither cutting planes nor exhaustive branching procedures and sometimes stops prematurely, often far from optimality. We will adapt primal cutting to ISUD to obtain an efficient primal algorithm for large (degenerate) SPPs. Note that despite the strong theoretical framework, implementation techniques prevent to reach optimality when solving problems that are too large to be solved with commercial solvers, which is precisely the ultimate goal of our study.

1.3 Objectives and contributions

This paper, which focuses on the SPP, is organized as follows. In Section 2, we introduce the *maximum reduced-mean augmentation* problem (**MRA**), which is a variant of the standard *maximum mean augmentation* problem (**MMA**); we show that **MRA** corresponds to the subproblem of ISUD, and thus we demonstrate the link between ISUD and primal algorithms. We show (Theorem 1) that **MRA** yields a canonical decomposition of the search space of **AUG**, and we underline the difference between **MRA** and **MMA**. In Section 3, we show how to add a cutting-plane procedure to ISUD and demonstrate that only primal cuts adapt to this factoring. We also give a procedure to transfer the cuts to the decomposed problems. In Section 4, we present computational results for the instances in [22] and obtain considerable improvements, particularly on those for which the best solution found was far from optimality where the mean optimality gap drops from 33.92% to 0.21%. These instances are much larger than those of [10].

2 An integral simplex for SPP

Suppose a decreasing sequence of $\mathbb{P}$-feasible solutions ending at x^k is known. We want to determine a direction $d \in \mathbb{R}^n$ and a step $r > 0$ s.t. $x^{k+1} = x^k + rd$ is $\mathbb{P}$-feasible and of lower cost than x^k or to assert that x^k is optimal.

From now on, x^k will always denote the current (binary) solution, $\mathcal{P} = \{j | x_j^k = 1\}$ the set of positive-valued variables, and $\mathcal{Z} = \{j | x_j^k = 0\}$ the set of null variables. $A_{\mathcal{P}}$ is the *reduced basis*. For the sake of readability, $\mathcal{P}$, $\mathcal{Z}$, d, and r (and later $\mathcal{C}$ and $\mathcal{I}$) will not be indexed on k although they depend on x^k.

The next solution x^{k+1} must satisfy $Ax^{k+1} = e$ and be integer. In the first part, we will require x^{k+1} to satisfy only the linear constraints of $\mathbb{P}'$ (the linear relaxation of $\mathbb{P}$). In the second part we will give conditions under which integrality is maintained when the algorithm takes a step r along d from x^k to reach x^{k+1}.

2.1 Relaxing the integrality constraints

Considering only the linear constraints, **AUG** consists in determining d s.t.

(i) d is a *feasible* direction: $\exists \rho > 0 \,|\, x^k + \rho d \in \mathcal{F}_{\mathbb{P}'}$;
(ii) d is an *improving* direction: $c \bullet d < 0$.

It is easy to see that such a d exists iff the program

$$z^\star_{\mathbf{MRA}} = \begin{array}{ll} \min & c \bullet d \\ \text{s.t.} & Ad = 0,\ e \bullet d_{\mathcal{Z}} = 1,\ d_{\mathcal{P}} \leqslant 0,\ d_{\mathcal{Z}} \geqslant 0 \end{array} \qquad (\mathbf{MRA})$$

satisfies $z^\star_{\mathbf{MRA}} < 0$. On the one hand, any optimal solution of **MRA** yields a solution to **AUG**; on the other hand, if $z^\star_{\mathbf{MRA}}$ is nonnegative, x^k is optimal.

Remark 1 *The specific augmentation problem **MRA** is close to the classical **MMA** (which is itself an extension of the* minimum mean cycle *[7] to more general linear programs; see [16]). It differs only in the normalization constraint: that of **MRA** concerns only the extra-basic variables, while that of **MMA** is $e \bullet d = 1$. This slight difference allows us to decompose the search space $\mathcal{F}_{\mathbf{MRA}}$ into two smaller subspaces and still guarantee **MRA**-optimality, as will be shown in Theorem 1. Theorem 1 does not hold for **MMA** (simple counterexamples are easy to find).*

Given x^k and $\mathcal{P}$, the corresponding reduced basis, we refer to any vector of its linear span ($span(A_{\mathcal{P}})$) as a *compatible* vector. This is extended to the columns of A: for any nonbasic index $j \in \mathcal{Z}$, A_j is a *compatible column* if it lies in $span(A_{\mathcal{P}})$; otherwise, A_j is *incompatible*. $\mathcal{C}$ (resp. $\mathcal{I}$) denotes the set of the indices of compatible (resp. incompatible) columns at the current iteration. If we partition $\mathcal{N}$ into $(\mathcal{P}, \mathcal{C}, \mathcal{I})$ and reorder the rows of A, we can write:

$$A = \begin{bmatrix} I_p & A^{\mathcal{P}}_{\mathcal{C}} & A^{\mathcal{P}}_{\mathcal{I}} \\ A^{N}_{\mathcal{P}} & A^{N}_{\mathcal{C}} & A^{N}_{\mathcal{I}} \end{bmatrix}. \qquad (1)$$

From the constraints $Ad = 0$ of **MRA**, one can easily see that the aggregation of all the increasing columns (for which $d_j \geqslant 0$) $w = A_{\mathcal{Z}} d_{\mathcal{Z}}$ is compatible. As in a reduced-gradient algorithm, we are in fact looking for an aggregate column w that can enter the reduced basis with a positive value by lowering only some variables (R) of $\mathcal{P}$. We introduce the following problems (called the *Restricted-MRA* and the *Complementary-MRA*):

$$z^\star_{\textbf{R-MRA}} = \begin{array}{lllll} \min\limits_{x \in \mathbb{R}^n} & c_{\mathcal{P}} \bullet d_{\mathcal{P}} & + & c_{\mathcal{C}} \bullet d_{\mathcal{C}} & \\ \text{s.t.} & A_{\mathcal{P}} d_{\mathcal{P}} & + & A_{\mathcal{C}} d_{\mathcal{C}} & = 0 \\ & & & e \bullet d_{\mathcal{C}} & = 1 \\ & d_{\mathcal{P}} \leqslant 0 & & d_{\mathcal{C}} \geqslant 0 & \end{array} \qquad (\textbf{R-MRA})$$

$$z^\star_{\textbf{C-MRA}} = \begin{array}{lllll} \min\limits_{x \in \mathbb{R}^n} & c_{\mathcal{P}} \bullet d_{\mathcal{P}} & + & c_{\mathcal{I}} \bullet d_{\mathcal{I}} & \\ \text{s.t.} & A_{\mathcal{P}} d_{\mathcal{P}} & + & A_{\mathcal{I}} d_{\mathcal{I}} & = 0 \\ & & & e \bullet d_{\mathcal{I}} & = 1 \\ & d_{\mathcal{P}} \leqslant 0 & & d_{\mathcal{I}} \geqslant 0 & \end{array} \qquad (\textbf{C-MRA})$$

Lemma 1 *For all $j \in \mathcal{Z}$, A_j is compatible iff $\exists! R_j \subset \mathcal{P} \,|\, A_j = \sum_{r \in R_j} A_r$.*

Theorem 1 $z^\star_{\mathbf{MRA}} = \min \{ z^\star_{\textbf{R-MRA}}, z^\star_{\textbf{C-MRA}} \}$.

Proof. Let $d = (d_{\mathcal{P}}, d_{\mathcal{C}}, d_{\mathcal{I}}) \in \mathbb{R}^n$ be an optimal solution of **MRA**. Suppose that the support of $(d_{\mathcal{C}}, d_{\mathcal{I}})$ is neither in $\mathcal{C}$ nor in $\mathcal{I}$. By Lemma 1, the surrogate column $A_{\mathcal{C}} d_{\mathcal{C}}$ can be written as a linear combination of columns of $\mathcal{P}$: $A_{\mathcal{C}} d_{\mathcal{C}} = -A_{\mathcal{P}} u'_{\mathcal{P}}$, $u'_{\mathcal{P}} \leqslant 0$. Let $u' = (u'_{\mathcal{P}}, d_{\mathcal{C}}, 0)$, and $u'' = d - u'$ and denote $d' = u'/\|d_{\mathcal{C}}\|_1$, $d'' = u''/\|d_{\mathcal{I}}\|_1$. Here d' and d'' are both feasible for **MRA** and resp. for **R-MRA** and **C-MRA**. $d = \|d_{\mathcal{C}}\|_1 d' + \|d_{\mathcal{I}}\|_1 d''$ is a convex combination since $\|d_{\mathcal{C}}\|_1 + \|d_{\mathcal{I}}\|_1 = \|d_{\mathcal{Z}}\|_1 = 1$. Therefore, d is an extreme point of $\mathcal{F}_{\mathbf{MRA}}$ iff $d = d'$ or $d = d''$. It is well known that one can always find an optimal solution of a linear program that is also an extreme point of the feasible domain, which concludes the proof. $\square$

Theorem 1 extends the results of Elhallaoui et al. [4] and Zaghrouti et al. [22], and it also justifies their procedures in a trivial way. This purely primal interpretation of their less-intuitive dual approach allows us to state a precise factoring of **MRA**. This factoring naturally leads to a sequential resolution of **R-MRA** and **C-MRA** that avoids the direct solution of **MRA**. As noted in Remark 1, it also accurately describes the differences between the classical **MMA** and the algorithms of [4] and [22].

As a consequence of Theorem 1, we will consider the pair of problems **R-MRA** and **C-MRA** instead of the more complicated **MRA**, and we will solve them sequentially. In particular, we will not solve **C-MRA** if $z^{\star}_{\textbf{R-MRA}} < 0$ because we already know an improving direction. In the next section, we discuss how to solve **R-MRA** and **C-MRA** and in particular how to ensure that x^{k+1} is an integer solution or that x^k is optimal.

2.2 Taking integrality into account

2.2.1 Integrality issues in R-MRA

An advantage of **R-MRA** is that it deals in a trivial way with the integrality constraints. For $j \in \mathcal{C}$, by Lemma 1, $\exists! R_j \subset \mathcal{P} \,|\, A_j = \sum_{r \in R_j} A_r$. Therefore, any extreme point of $\mathcal{F}_{\textbf{R-MRA}}$ is of the form $d^j_k = 1$ if $k = j$, -1 if $k \in R_j$, 0 otherwise. Thus, taking a step in this direction is strictly equivalent to entering A_j into the reduced basis by performing a single simplex pivot. The cost of such a solution is $c \bullet d^j = c_j - \sum_{r \in R} c_r = \bar{c}_j$, i.e., the reduced cost of column j as computed in the simplex algorithm ($\bar{c} = c - \pi^T A$, with π being the dual-variable vector).

The optimal value of a linear program is always at an extreme point of its domain, so we can rewrite the problem as

$$z^{\star}_{\textbf{R-MRA}} = \min_{j \in \mathcal{P}} \bar{c}_j \tag{2}$$

and given $j \in \mathcal{P}$ s.t. $\bar{c}_j$ is minimal, the corresponding optimal solution is d^j. This formulation yields a search strategy equivalent to the Dantzig criterion in the simplex algorithm. Furthermore, the restriction of the search space to $span(A_{\mathcal{P}})$ prevents degenerate pivots (step $r = 1$) and, as Proposition 1 states, preserves integrality. Note that this formulation is close to that of a reduced gradient.

Proposition 1 *If j is optimal for (2) and r is the maximum feasible step along d^j at x^k, then $x^{k+1} = x^k + rd^j$ satisfies the conditions (H0)–(H3).*

Proof. x^{k+1} is reached by performing a single simplex pivot (entering A_j into the basis), thus (H3) holds. Since $r = 1$ (trivial), (H0)–(H2) are straightforward. □

2.2.2 Integrality issues in C-MRA

AUG is as difficult as SPP [15]. Since SPP is $\mathcal{NP}$-complete and **R-MRA** is polynomial, **C-MRA** has to be *hard* and no simple formulation can be expected. However, we will show that **C-MRA** can be reduced to an $(m - p) \times |\mathcal{I}|$ linear program with specific additional constraints, and we will address the solution of this new formulation **RC-MRA**.

In **C-MRA**, the first p constraints are $d_{\mathcal{P}} = -A^{\mathcal{P}}_{\mathcal{I}} d_{\mathcal{I}}$. As in a reduced-gradient algorithm, the modification of the basic variables occurs via a linear transformation of the increasing nonbasic variables. With $d = D\delta$, the *transfer matrix* D summarizes the relationship between the reduced direction $d_{\mathcal{I}} = \delta \in \mathbb{R}^{|\mathcal{I}|}$ and the corresponding direction $d = D\delta$ in the original space $\mathbb{R}^n$. **C-MRA** becomes (*Reduced Complementary-MRA*):

$$\begin{aligned} z^{\star}_{\textbf{RC-MRA}} = \min_{\delta \in \mathbb{R}^{|\mathcal{I}|}} \quad & c \bullet (D\delta) \\ \text{s.t.} \quad & A^N_{\mathcal{P},\mathcal{I}}(D\delta) = 0,\ e \bullet \delta = 1,\ \delta \geqslant 0 \end{aligned} \tag{\textbf{RC-MRA}}$$

The set of all feasible directions at x^k is a cone whose extreme directions are those of all the edges of $\mathcal{F}_{\mathbb{P}'}$ that go through x^k. Since SPP is quasi-integral, any reduced direction δ s.t. $d = D\delta$ satisfies (H0) and (H3) is an extreme point of $\mathcal{F}_{\textbf{RC-MRA}}$. Let $\Delta = \{\delta \in \mathcal{F}_{\textbf{RC-MRA}} \,|\, d = D\delta$ satisfies (H0) and (H3)$\}$. We will now give a simple characterization of Δ. A set of indices U (or columns A_U) is called *column-disjoint* if no pair of columns $A_i, A_j \in A_U$, $i \neq j$ has a common nonzero entry, i.e., $\forall i, j \in U, i \neq j \Rightarrow A_i \bullet A_j = 0$.

Proposition 2 (Propositions 6 and 7, Zaghrouti et al. [22]) *Given δ a **vertex** of $\mathcal{F}_{\textbf{C-MRA}}$, $d = D\delta$ is an integral feasible direction iff S (the support of δ) is column-disjoint. In this case, the maximal feasible step along d is $r = |S|$.*

2.3 Algorithmic framework

From Proposition 2, $\Delta = \{\delta \,|\, \delta$ is a vertex of $\mathcal{F}_{\textbf{C-MRA}}$ and S is column-disjoint$\}$. Therefore, if we can solve

$$z^\star_{(3)} = \min_{\delta \in \Delta} c \bullet (D\delta) = \min_{\delta \in Conv(\Delta)} c \bullet (D\delta) \tag{3}$$

then we can use Algorithm 1 to solve $\mathbb{P}$.

Algorithm 1 Integral Simplex Using Decomposition for SPP

1. Find an initial solution of SPP x^0; $k \leftarrow 0$;
2. If $z^\star_{\textbf{R-MRA}} < 0$: perform a single compatible pivot to obtain x^{k+1}; $k \leftarrow k + 1$; GOTO 2;
3. If $z^\star_{(3)} < 0$: given δ (optimal solution of (3)) and r (maximum feasible step along $D\delta$ at x^k): $x^{k+1} \leftarrow x^k + rD\delta$; $k \leftarrow k + 1$; GOTO 2;
4. Return x^k, the optimal solution of $\mathbb{P}$.

3 Solving RC-MRA with cutting planes

3.1 Theory and generic cutting-plane procedure

In this section, we characterize cutting planes for **RC-MRA**. Given $\delta^\star$ an optimal solution of **RC-MRA** that is not column-disjoint, we want to determine an inequality $\bar{\Gamma}$ that separates $\delta^\star$ from $\mathcal{F}_{(3)} = \Delta$ to tighten the relaxation. We will show that $\bar{\Gamma}$ can always be obtained from a *primal cut* (in the sense of [10]).

Consider $\bar{\Gamma} : \bar{\alpha} \bullet \delta \leqslant \bar{\beta}$, a valid inequality for Δ. Since $Conv(\Delta) \subset \{\delta \,|\, e \bullet \delta = 1\}$, we can assume $\bar{\alpha} \notin Span(\{e\})$. Then $\{e \bullet \delta = 1\} \cap \{\bar{\alpha} \bullet \delta = \bar{\beta}\} = F$ is of dimension $|\mathcal{I}| - 2$ (the intersection of two nonparallel hyperplanes). Thus, $span(F \cup \{0\})$ is a hyperplane of $\mathbb{R}^{|\mathcal{I}|}$ that yields the same valid inequality as $\bar{\Gamma}$ within $\mathcal{F}_{\textbf{RC-MRA}}$ and that reads $\bar{\alpha}' \bullet \delta \leqslant 0$ for some $\bar{\alpha}'$. Without loss of generality, from now on, we consider $\bar{\alpha} = \bar{\alpha}'$ and thus $\bar{\beta} = 0$.

We will now characterize $\bar{\Gamma}$ in terms of the original SPP formulation. Let $\alpha \in \mathbb{R}^{p+|\mathcal{I}|}$ be such that

$$\bar{\alpha} = D\alpha. \tag{4}$$

Such an α always exists since $(0, \bar{\alpha})$ is a trivial solution of (4). $\bar{\Gamma}$ is valid for (3) iff for all $\delta \in \Delta$ and all $r > 0$, $\alpha \bullet (x^k + rD\delta) \leqslant \alpha \bullet x^k$. Since $\{d = D\delta \,|\, \delta \in \Delta\}$ is the cone of all feasible directions at x^k within $Conv(\mathcal{F}_{\mathbb{P}})$, this is equivalent to the inequality

$$\Gamma : \quad \alpha \bullet (x - x^k) \leqslant 0 \tag{Γ}$$

being a valid inequality for SPP. Since a *primal valid inequality* is a valid inequality that is tight at x^k, Γ is obviously a primal valid inequality.

Consider now the case where the optimal solution $\delta^\star$ of **RC-MRA** is not column-disjoint ($\delta^\star \notin \Delta$). $x^\star$ is the new fractional solution found by taking a maximal step $r^\star$ along $D\delta^\star$ at x^k: $x^\star = x^k + r^\star D\delta^\star$. We now address the problem of separating $\delta^\star$ from Δ.

Proposition 3 *$\bar{\Gamma}$ is a valid inequality for (3) that separates $\delta^\star$ from Δ iff Γ is a primal valid inequality for the SPP that separates $x^\star$ from $\mathcal{F}_{\mathbb{P}}$. In this case, $\bar{\Gamma}$ is a* cut *for (3) and Γ is a* primal cut *for $\mathbb{P}$ in the sense of [10].*

Proof. Most of the proof has been given in the previous paragraphs. We need to prove only the separation. $\bar{\Gamma}$ separates $\delta^\star$ from Δ iff $\bar{\alpha}\bullet\delta^\star > 0$. Equivalently, $\alpha\bullet(x^k + rD\delta^\star) > \alpha\bullet x^k$ or $\alpha\bullet(x^\star - x^k) > 0$, which concludes the proof. □

What we have shown must now be seen the other way round to take advantage of previous work on primal separation. Assume that we have a primal cut Γ for the SPP that separates $x^\star$ from $\mathcal{F}_{\mathbb{P}}$. Then, the inequality

$$\bar{\Gamma}: \qquad \alpha\bullet D\delta \leqslant 0 \tag{$\bar{\Gamma}$}$$

is a cut for **RC-MRA** that separates $\delta^\star$ from $Conv(\Delta)$. Moreover, we have shown that any cut for **RC-MRA** can be obtained in this way. This enables us to develop a procedure based on **P-SEP**: Given x^k and $x^\star$, is there a valid inequality for $\mathcal{F}_{\mathbb{P}}$, tight at x^k, that separates $x^\star$ from $\mathcal{F}_{\mathbb{P}}$? If it exists, it will be transferred to **RC-MRA** to tighten the relaxation of $\mathcal{F}_{(3)}$, as in Algorithm 2.

Remark 2 *$\bar{\Gamma}$ is obtained from Γ by multiplying α and the transfer matrix D, as **RC-MRA** was obtained from **C-MRA** by multiplying the objective and the constraints by D. This shows the role played by D in the transformation from the directions in $\mathbb{R}^n$ to the reduced directions in $\mathbb{R}^{|\mathcal{I}|}$.*

Algorithm 2 Cutting-plane procedure for (3)

1. $\delta^\star \leftarrow$ an optimal solution of **RC-MRA**;
2. If stopping conditions are met, return $\delta^\star$ although it may not be (3)-feasible;
3. If $\delta^\star$ is not column-disjoint, find a solution to **P-SEP** $\Gamma : \alpha \bullet (x - x^k) \leqslant 0$; transfer it to **RC-MRA** as $\bar{\Gamma} : \alpha \bullet D\delta \leqslant 0$; GOTO 1;
4. Return $\delta^\star$, the optimal solution of (3).

Algorithm 2 requires a primal separation procedure that can solve **P-SEP**. Note that any primal algorithm, by its nature, can provide cuts only in a given family $\mathcal{O}$ and actually solves $\mathcal{O}$-**P-SEP** (a primal separation but in a given cut family).

Remark 3 *There exist families of cuts for which no stopping criterion is required (e.g., Gomory–Young's cuts [20]). These families guarantee $\delta^\star \in \Delta$ after a finite number of $\mathcal{O}$-**P-SEP** problems have been solved.*

Remark 4 *For the stopping criterion, a maximal number of iterations can be fixed prior to running the algorithm. For families for which no cut may exist even if $\delta^\star \notin \Delta$, the algorithm stops whenever $\mathcal{O}$-**P-SEP** yields no new cut. In this case, $\delta^\star$ may not be column-disjoint, and either a branching procedure is used or the primal algorithm stops prematurely at x^k.*

3.2 Primal clique cut separation: $\mathcal{Q}$-P-SEP

In this section, we present a well-known cutting-plane family $\mathcal{Q}$, called the clique cuts, for which there exists a relatively simple procedure that solves $\mathcal{Q}$-**P-SEP**. Unlike Gomory–Young cuts, $\mathcal{Q}$-**P-SEP** may have no solution although $\delta^\star \notin \Delta$. However, clique inequalities are usually sufficient to reach Δ-optimality and yield deep cuts. Consider $\mathcal{G} = (\mathcal{N}, E)$, the conflict graph obtained from matrix A, i.e., $\{i,j\} \in E$ iff $A_i\bullet A_j \neq 0$. Given a clique $\mathcal{W}$ in this graph, any binary solution of SPP satisfies

$$\sum_{j\in\mathcal{W}} x_j \leqslant 1. \tag{$Q_{\mathcal{W}}$}$$

$Q_{\mathcal{W}}$ is called the *clique inequality* associated with $\mathcal{W}$, and it is valid for $\mathcal{F}_{\mathbb{P}}$. Moreover, given $x^\star$, a fractional extreme point of $\mathcal{F}_{\mathbb{P}'}$, finding a clique cut is equivalent to finding a clique of total weight greater than 1 in a weighted graph $\mathcal{G}$, with weight function $w_j = x^\star_j$, $j \in \mathcal{N}$.

In our case, $x^\star$ is typically the fractional vertex of $\mathcal{F}_{\mathbb{P}'}$ that *would* be reached *if* a step were taken in direction $D\delta^\star$, where $\delta^\star$ is an extreme optimal solution of $\mathcal{F}_{\textbf{RC-MRA}}$ that is not column-disjoint ($\delta^\star \notin \Delta$).

For $Q_{\mathcal{W}}$ to be tight at x^k, it must satisfy $\sum_{l\in\mathcal{W}\cap\mathcal{P}} x_l^k = 1$, or equivalently $|\mathcal{W}\cap\mathcal{P}| = 1$. Therefore, with $\mathcal{G}_l$ being the subgraph of $\mathcal{G}$ with the vertices l and all neighbors of l, a primal separation for the clique cuts can be found using Algorithm 3.

Algorithm 3 $\mathcal{Q}$-P-SEP

1. $\mathcal{K} \leftarrow \emptyset$ (set of all primal cliques found);
2. For all $l \in R = support(D\delta^\star) \cap \mathcal{P}$: Find a clique $\mathcal{W}_l$ of maximal weight in $\mathcal{G}_l$; $\mathcal{K} \leftarrow \mathcal{K} \cup \{Q_{\mathcal{W}_l}\}$;
3. Return $\mathcal{K}$.

4 Experimentation

4.1 Algorithm and instances

4.1.1 Algorithm.

To solve SPP, we use Algorithm 1 with Zaghrouti et al.'s multi-phase strategy. Problem (3) is solved using Algorithm 2. If the solution of **RC-MRA** is not column-disjoint, we generate primal clique inequalities using Algorithm 3 and transfer them to **RC-MRA** to tighten the relaxation of $\mathcal{F}_{(3)}$. For each solution $\delta^\star \notin \Delta$, at most 70 $\mathcal{Q}$**-P-SEP** are solved. However, note that this number is seldom reached (usually, before the 70$^{\text{th}}$ separation procedure is launched, either $\delta^\star$ is column-disjoint or $\mathcal{Q}$**-P-SEP** has no solution). When the cutting planes do not manage to ensure $\delta^\star \in \Delta$, the nonexhaustive branching procedure of [22] is used. After each branch, new cutting planes may be generated.

4.1.2 Cut pool.

As in a standard *branch-and-bound*, all the generated cuts are kept in a pool. Before generating any supplementary cuts, we transfer any cuts in the pool that can be used to eliminate $\delta^\star \notin \Delta$ at the current iteration to **RC-MRA**.

4.1.3 Instances.

Tests were run on an aircrew scheduling problem from OR-Lib of size $m = 823$, $n = 8,904$. The different instances correspond to different initial solutions x^0. These initial solutions are created to resemble typical initial solutions for aircrew scheduling problems and are far from optimality. We chose to focus on the hardest instances, i.e., those for which the solutions in [22] were furthest from optimality. See [22] for more details on these instances.

4.2 Numerical results

Table 1 shows that for the ten hardest instances, adding primal $\mathcal{Q}$-inequalities reduces the average optimality gap from 33.92% to 0.21% and decreases the maximal gap from 200.63% to 2.06%. This improvement comes with an increase in the number of steps and a small increase in the computational time.

On the nonoptimal instances (instances 3, 5, 8, and 10), `ISUD_Clique` finds a slightly worse solution for instance 5, the same solution for instance 8, and greatly improved solutions for instances 3 and 10. Note that the improvement comes with a higher number of steps: +6 (resp. +11) for instance 3 (resp. 10). This means that the cutting planes allow the algorithm to find column-disjoint combinations whereas the nonexhaustive branching did not. When the cuts worsen the solution (instance 5), the algorithm performs five additional steps but reaches a more expensive solution. This indicates that the cuts changed the *path* $(x^k)_k$ earlier in the process by taking an improving direction that *locally seemed better* than that taken by `ISUD_NoCut`.

Tables 1 and 2 indicate that solving the separation problems is not very time-consuming. The cuts add iterations (K increases). The extra primal cuts, and the corresponding reoptimizations, increase the time per iteration by 24% (on average) while reducing the average gap from 33.9% to 0.21%.

Table 1: Comparison of performance of ISUD with and without clique cuts on aircrew scheduling problems. Opt. val. $56,137$.

		`ISUD_NoCut`			`ISUD_Clique`		
Instance	Init. gap	Gap (%)	Steps	Time (s)	Gap (%)	Steps	Time (s)
1	570.22	0	42	8.75	0	42	10.27
2	560.97	0	46	13.44	0	46	16.56
3	559.92	138.08	27	12.15	**0.03**	**33**	17.20
4	557.67	0	45	9.96	0	45	11.47
5	562.5	**0.5**	54	19.88	2.06	59	24.98
6	561.03	0	35	9.85	0	39	13.84
7	573.09	0	42	7.98	0	42	9.738
8	569.64	0.02	42	9.58	0.02	43	10.98
9	569.29	0	53	11.62	0	54	13.39
10	573.74	200.63	33	6.78	**0.01**	**44**	17.75
Avg.	565.81	33.92	41.9	11.00	**0.21**	44.7	14.62

Table 2: Performance of `ISUD_Clique` on the aircrew scheduling instances.

Instance	Gap (%)	Steps	Cuts	Sep.	Fails	Tot. (s)	Sep. (s)
1	0	42	266	64	3	10.27	1.01
2	0	46	490	94	4	16.56	2.02
3	0.03	33	440	130	9	17.20	1.99
4	0	45	281	70	4	11.47	1.00
5	2.06	59	536	155	19	24.98	2.64
6	0	39	473	142	6	13.84	1.42
7	0	42	314	78	4	9.74	1.17
8	0.02	43	339	69	7	10.98	1.07
9	0	54	299	80	5	13.39	1.12
10	0.01	44	496	132	12	17.75	1.62
Avg.	0.21	44.7	393.4	101.4	7.3	14.62	1.50

Note that branching prior to cutting would have allowed `ISUD_Clique` to always find a better solution than that found by `ISUD_NoCut`. However, it seems logical to begin with cuts that do not discard any feasible solution rather than to begin with a heuristic branching. Therefore, we took the risk that the solution would deteriorate on some instances, as happened for instance 5.

5 Conclusions and future work

We have introduced **MRA** to show the link between the ISUD constraint reduction algorithm and primal algorithms. The factoring of **MRA** into **R-MRA** and **C-MRA** has been explained and justified, and the result is an efficient primal cutting-plane algorithm for the SPP. A proof of concept of this algorithm was provided by using the $\mathcal{Q}$-cuts family.

In future work, we plan to apply our algorithm to larger instances ($1,600 \times 570,000$ as in [22]) to obtain a more complete benchmark. Other families of cuts such as primal cycle cuts will be added to improve the performance of the algorithm.

References

[1] Balas, E., Padberg, M.: On the set-covering problem: 2 – an algorithm for set partitioning. Oper. Res. 23(1), 74–90 (1975)

[2] Ben-Israel, A., Charnes, A.: On some problems of diophantine programming. Cahiers du Centre d'Ét. de Rech. Opér. 4, 215–280 (1962)

[3] Eisenbrand, F., Rinaldi, G., Ventura, P.: Primal separation for 0/1 polytopes. Math. Program. 95(3), 475–491 (2003)

[4] Elhallaoui, I., Metrane, A., Desaulniers, G., Soumis, F.: An improved primal simplex algorithm for degenerate linear programs. INFORMS J. Comput. 23(4), 569–577 (2011)

[5] Elhallaoui, I., Villeneuve, D., Soumis, F., Desaulniers, G.: Dynamic aggregation of set-partitioning constraints in column generation. Oper. Res. 53(4), 632–645 (2005)

[6] Glover, F.: A new foundation for a simplified primal integer programming algorithm. Oper. Res. 16, 727–740 (1968)

[7] Goldberg, A.V., Tarjan, R.E.: Finding minimum-cost circulations by canceling negative cycles. J. ACM 36(4), 873–886 (1989)

[8] Gomory, R.E.: Outline of an algorithm for integer solutions to linear program. Bull. Amer. Math. Soc. 64(5), 275–278 (1958)

[9] Gomory, R.E.: All-integer integer programming algorithm. Ind. Sched. pp. 193–206 (1963)

[10] Letchford, A.N., Lodi, A.: Primal cutting plane algorithms revisited. Math. Methods Oper. Res. 56(1), 67–81 (2002)

[11] Letchford, A.N., Lodi, A.: Primal separation algorithms. Q. J. Belg. Fr. Ital. Oper. Res. Soc. 1(3), 209–224 (2003)

[12] Rönnberg, E., Larsson, T.: Column generation in the integral simplex method. European J. Oper. Res. 192(1), 333–342 (2009)

[13] Salkin, H.M., Koncal, R.D.: Set covering by an all-integer algorithm: Computational experience. J. ACM 20(2), 189–193 (1973)

[14] Saxena, A.: Set-partitioning via integral simplex method. Unpublished manuscript, OR Group, Carnegie-Mellon University, Pittsburgh (2003)

[15] Schulz, A.S., Weismantel, R., Ziegler, G.M.: 0/1-integer programming: Optimization and augmentation are equivalent. In: Spirakis, P. (ed.) ESA '95, LNCS, vol. 979, pp. 473–483. Springer Berlin Heidelberg (1995)

[16] Spille, B., Weismantel, R.: Primal integer programming. In: Aardal, K., Nemhauser, G., Weismantel, R. (eds.) Discrete Optimization, Handbooks in Operations Research and Management Science, vol. 12, pp. 245–276. Elsevier, Amsterdam (2005)

[17] Stallmann, M.F., Brglez, F.: High-contrast algorithm behavior: Observation, conjecture, and experimental design. In: ACM-FCRC. ACM, New York (2007), 549075

[18] Thompson, G.L.: An integral simplex algorithm for solving combinatorial optimization problems. Comput. Optim. Appl. 22(3), 351–367 (2002)

[19] Trubin, V.: On a method of solution of integer linear programming problems of a special kind. Soviet Math. Dokl. 10, 1544–1546 (1969)

[20] Young, R.D.: A primal (all-integer) integer programming algorithm. J. Res. Nat. Bureau of Standards: B. Math. and Math. Phys. pp. 213–250 (1965)

[21] Young, R.D.: A simplified primal (all-integer) integer programming algorithm. Oper. Res. 16(4), 750–782 (1968)

[22] Zaghrouti, A., Soumis, F., Elhallaoui, I.: Integral simplex using decomposition for the set partitioning problem. Oper. Res. (to appear)