On cut generation for facial disjunctive programs with two-term disjunctions

Tamás Kis

July 14, 2009

Abstract

In this paper we generalise the method of Balas and Perregaard for 0/1 mixed-integer programming to facial disjunctive programs with two-term disjunction. We apply our results to linear programs with complementarity constraints.

1 Introduction

In this paper we study facial disjunctive programs with linear constraints. A disjunctive constraint is of the form

$$d^1 x \geq d_{1,0} \vee \; d^2 x \geq d_{2,0}.$$

Here, each $d^i x \geq d_{i0}$ is a linear constraint, $d^i \in \mathbb{R}^n$, $d_{i0} \in \mathbb{R}$, and $x = (x_1, \ldots, x_n)^T$ is the vector of decision variables. It can model a *choice* between alternative options. If all the constraints are linear, cutting plane methods are likely to be applicable to find feasible or optimal solutions to disjunctive programs. In fact, such algorithms have been developed and applied for solving mixed $0-1$ programming problems quite successfully for almost two decades, see e.g., [18], [5], [16], [10].

The problem: The input data is given by matrix $A \in \mathbb{R}^{m\times n}$, vectors $b \in \mathbb{R}^{m\times 1}$, $c \in \mathbb{R}^{1\times n}$, and by an irreflexive, antisymmetric relation $D \subset \{1, \ldots, m+n\} \times \{1, \ldots, m+n\}$. Letting $(\tilde{A}, \tilde{b}) = \begin{pmatrix} A & b \\ I_n & 0 \end{pmatrix}$, where $I_n \in \mathbb{R}^{n\times n}$ is the n-dimensional identity matrix, the central object of our study is the mathematical program:

$$DP: \quad \begin{aligned} & \min_x c^T x && (1) \\ & \text{s.t.} \quad Ax \geq b, && (2) \\ & \qquad\; x \geq 0, && (3) \\ & -\tilde{A}_{k_1} x \geq -\tilde{b}_{k_1} \vee \; -\tilde{A}_{k_2} x \geq -\tilde{b}_{k_2}, \quad \forall (k_1, k_2) \in D. && (4) \end{aligned}$$

Notice that $\tilde{A}_k x \geq \tilde{b}_k$ denotes the k-th row of $\tilde{A}x \geq \tilde{b}$. Constraint (4) is a *two-term disjunction*, specifying that any feasible solution x has to satisfy at least one of the two terms for each $(k_1, k_2) \in D$. In particular, at least one of the constraints $\tilde{A}_{k_1} x \geq \tilde{b}_{k_1}$ and $\tilde{A}_{k_2} x \geq \tilde{b}_{k_2}$ has to hold with equality. Following Balas [7], a disjunctive program in *conjunctive normal form* is given by

$$\min\{c^T x \mid x \in F^S\}$$

where S is the set of *disjunctions*, and

$$F^S := \left\{x \in \mathbb{R}^n \mid Ax \geq b,\ x \geq 0,\ \wedge_{j \in S}(\vee_{i \in Q_j} d^i x \geq d_{i0})\right\},$$

where Q_j is the index set of terms in disjunction $j \in S$. It is *facial* if every $d^i x \geq d_{i0}$ induces a face of the polyhedron $P := \{x \in \mathbb{R}^n \mid Ax \geq b, x \geq 0\}$, i.e.,

$$F_i := P \cap \{x \in \mathbb{R}^n \mid d^i_x \geq d_{i0}\}$$

is a face of P (Balas [7]). It follows that the disjunctive program DP (1)-(4) is facial.

The most prominent example of facial disjunctive programs with two-term disjunctions is mixed $0-1$ linear programming, where each disjunction is of the form

$$-x_i \geq 0 \vee x_i \geq 1,$$

and the constraints $-x_i \geq -1$ are part of $Ax \geq b$. There are numerous papers dealing with cut generation for this special case, see e.g., [7], [3], [5], [12], [9], [10], [20], [11] and the surveys [8], [13].

A more general problem class is linear programming with complementarity constraints. Given $c \in \mathbb{R}^n$, $d \in \mathbb{R}^m$, $f \in \mathbb{R}^k$, $q \in \mathbb{R}^m$, $A \in \mathbb{R}^{k \times n}$, $B \in \mathbb{R}^{k \times m}$, $M \in \mathbb{R}^{m \times m}$, and $N \in \mathbb{R}^{m \times n}$. The *linear programming with linear complementarity constraints* problem aims at finding the optimal solution $(x, y) \in \mathbb{R}^n \times \mathbb{R}^m$ of the program

$$\begin{aligned} & \min_{(x,y)} c^T x + d^T y \\ LPCC: \quad & \text{s.t. } Ax + By \geq f, \\ & \qquad 0 \leq y \perp q + Nx + My \geq 0, \end{aligned}$$

where $a \perp b$ means that the two vectors are orthogonal, i.e., $a^T b = 0$. Clearly, LPCC is a facial disjunctive program with disjunctions $-y_i \geq 0 \vee -N_i x - M_i y \geq q_i$, $i = 1, \ldots, m$. A notable example for LPCC is bilevel linear programming, see e.g., [14], [2]. For solving LPCCs several mehtods have been proposed in the literature. Cutting plane methods are described in [17], [2], and [15].

Disjunctive programs with multiple-term disjunctions is the topic of [7], [17], [3], [19], and [20].

Main results: Our interest in this paper is cutting plane methods for facial disjunctive programs with two-term disjunctions. In particular, we show how to generalise the efficient cut generation method of Balas and Perregaard [10] for mixed $0-1$ programming to this more general problem class. Firtly, we show that each basis of the LP relaxation gives rise to a feasible basis solution of a cut generation linear program as well as to an equivalent disjunctive cut. This result is stronger than Theorem 4B of Balas and Perregaard [10]. Then, we demonstrate by an example that unlike in the mixed $0-1$ programming special case, the optimal solution of the cut-generation linear program cannot always be found by pivoting in the original simplex tableau. Moreover, we generalize all the formulae needed for cut generation in the original simplex tableau. Unfortunately, the cut generation method discussed in this paper can perform poorly in finding facets of the disjunctive program, which will be demonstrated by an example. We also evaluate the strength of the cuts on benchmark instances from the literature.

The structure of the paper: In the following section we recapitulate how to generate disjunctive cuts from the simplex tableau along with the concept of lift-and-project cuts. In Section 3 we prove that any basis of the linear relaxation induces a disjunctive cut and also an equivalent solution of a cut-generation linear program. In Section 4 we generalise the cut-generation procedure of [10] to facial disjunctive programming. In Section 5 we discuss the limitions of the cut-gneration method in finding facets of a disjunctive program. In Section 7, we evaluate the cut-generation procedure on benchmark problems from the literature, and conclude the paper in Section **??**.

2 Preliminaries

In this section we recapitulate known techniques for generating cuts for disjunctive programs.

2.1 Generation of disjunctive cuts from the simplex tableau

We augment the linear program LP by surplus variables s_i, $i \in S := \{1, \ldots, m\}$, and we index the structural variables as x_j, $j \in N := \{m+1, \ldots, m+n\}$. With this notation, we can rewrite the system (2)-(3) in the following augmented form:

$$\begin{aligned} &A_i x - s_i = b_i, \quad i = 1, \ldots, m \\ &x, s \geq 0 \end{aligned} \tag{5}$$

To simplify notation, similarly to Balas and Perregaard [10], we will identify each structural variable x_j with a new surplus variable s_j, $j \in N$.

The simplex tableau of LP is uniquely determined by the set of basic variables, denoted by I, or alternatively, by the set of nonbasic variables, denoted by J. Clearly, $I \cup J = N \cup S$ and $I \cap J = \emptyset$. The rows of the simplex tableau can be expressed as

$$\begin{aligned} x_i + \textstyle\sum_{j \in N \cap J} \overline{a}_{ij} x_j + \sum_{j \in S \cap J} \overline{a}_{ij} s_j = \overline{a}_{i0} \text{ for } i \in N \cap I, \\ s_i + \textstyle\sum_{j \in N \cap J} \overline{a}_{ij} x_j + \sum_{j \in S \cap J} \overline{a}_{ij} s_j = \overline{a}_{i0} \text{ for } i \in S \cap I, \end{aligned} \tag{6}$$

where $\overline{a}_{ij}$ is the coefficient of the nonbasic variable j in the row of the basic variable i, and $\overline{a}_{i0}$ is the right hand side constant (this notation is used in [10]). The basis solution corresponding to I is (x^I, s^I), where $x^I_i = \overline{a}_{i0}$ for $i \in N \cap I$ and $x^I_i = 0$ for $i \in N \cap J$, and similarly, $s^I_i = \overline{a}_{i0}$ for $i \in S \cap I$ and $s^I_i = 0$ for $i \in S \cap J$.

Now if I determines a feasible basis, i.e., all the $\overline{a}_{i0}$ are non-negative, then it is easy to verify whether x^I satisfies the disjunctive constraints (4). Namely, if $s^I_{k_1} s^I_{k_2} = 0$ for every $(k_1, k_2) \in D$, then x^I is a feasible solution of DP. Otherwise, $s_{k_1} > 0$ and $s_{k_2} > 0$, hence both s_{k_1} and s_{k_2} are basic. Therefore, the simplex tableau contains the following rows:

$$s_{k_1} + \sum_{j \in N \cap J} \overline{a}_{k_1,j} x_j + \sum_{j \in S \cap J} \overline{a}_{k_1,j} s_j = \overline{a}_{k_1,0} \tag{7}$$

$$s_{k_2} + \sum_{j \in N \cap J} \overline{a}_{k_2,j} x_j + \sum_{j \in S \cap J} \overline{a}_{k_2,j} s_j = \overline{a}_{k_2,0} \tag{8}$$

Since in a feasible solution of DP, $s_{k_1} = 0$ or $s_{k_2} = 0$ (or both), it follows that the disjunctive constraint

$$s_{k_1} \leq 0 \vee s_{k_2} \leq 0$$

is valid for the set of feasible solutions of DP. Using (7) and (8) and the substitution $x_j = s_j$ for $j \in N$, this can be rewritten as

$$\sum_{j \in J} \overline{a}_{k_1,j} s_j \geq \overline{a}_{k_1,0} \vee \sum_{j \in J} \overline{a}_{k_2,j} s_j \geq \overline{a}_{k_2,0}.$$

The disjunctive inequality, valid for DP, from the above disjunction is

$$\sum_{j \in J} \max\{\overline{a}_{k_2,0} \overline{a}_{k_1,j}, \overline{a}_{k_1,0} \overline{a}_{k_2,j}\} s_j \geq \overline{a}_{k_1,0} \overline{a}_{k_2,0}. \tag{9}$$

2.2 The mixed $0-1$ programming special case

Balas and Perregaard [10] have developed the theory of efficient cut generation for mixed 0/1 linear programming. In this section we state and prove a simple lemma which will be used later on several times to show how our results generalise those of Balas and Perregaard. Suppose that the variables $x_1, \ldots, x_p$ are binary, and the system $Ax \geq b$ contains the constraints $-x_k \geq -1$ for $k = 1, \ldots, p$. Then, the augmented form of this linear system contains for each binary variable x_k the following pair of constraints

$$\begin{aligned} x_k - s_{k_1} &= 0, \\ -x_k - s_{k_2} &= -1, \end{aligned} \tag{10}$$

where s_{k_1} and s_{k_2} are non-negative surplus variables. Suppose $\hat{x}$ is a basic feasible solution and $0 < \hat{x}_k < 1$. Then all the variables x_k, s_{k_1} and s_{k_2} are basic. The rows of the simplex tableau corresponding to s_{k_1} and s_{k_2} satisfy the following:

Lemma 1 *If $0 < \hat{x}_k < 1$ in a basic feasible solution $(\hat{x}, \hat{s})$ of the augmented linear program (5) including (10), then*

(i) $\overline{a}_{kj} = \overline{a}_{k_1,j} = -\overline{a}_{k_2,j}$ *for all* $j \in J$.

(ii) $\overline{a}_{k0} = \overline{a}_{k_1,0} = 1 - \overline{a}_{k_2,0}$.

For the sake of completeness, a proof is provided in the Appendix.

2.3 Lift-and-project cuts

To find a violated inequality for a disjunctive program, Balas [7] and Balas et al. [5] propose to solve a linear program. Given a feasible solution $\hat{x}$ of LP, and a pair $(k_1, k_2) \in D$ such that the corresponding disjunctive inequality in (4) is violated by $\hat{x}$, the *cut generation linear program* is

$$
\begin{aligned}
(CGLP)_{k_1,k_2} \qquad & \min_{(\alpha,\beta,u,v,u_0,v_0)} \alpha\hat{x} - \beta && (11)\\
\text{s.t.} \quad & \alpha - u\tilde{A} + u_0\tilde{A}_{k_1} = 0, && (12)\\
& \alpha - v\tilde{A} + v_0\tilde{A}_{k_2} = 0, && (13)\\
& \beta - u\tilde{b} + u_0\tilde{b}_{k_1} = 0, && (14)\\
& \beta - v\tilde{b} + v_0\tilde{b}_{k_2} = 0, && (15)\\
& u\mathbb{1} + v\mathbb{1} + u_0 + v_0 = 1, && (16)\\
& u, v, u_0, v_0 \geq 0.
\end{aligned}
$$

Here, $\mathbb{1}$ denotes the $m + n$ dimensional column vector of all ones. The cut sought is of the form $\alpha x \geq \beta$. The objective function prescribes the generation of a cut of maximum violation in the sense that the difference between the right and left hand sides with respect to the feasible solution $\hat{x}$ is the largest. Constraints (12) and (14) ensure that $\alpha x \geq \beta$ is valid for the system $Ax \geq b, -\tilde{A}_{k_1}x \geq -\tilde{b}_{k_1}$, whereas (13) and (15) yields that $\alpha x \geq \beta$ is valid for the system $Ax \geq b, -\tilde{A}_{k_2}x \geq -\tilde{b}_{k_2}$. Since these inequalities define a cone, we need the normalisation constraint (16), otherwise the optimum value may be unbounded. The cuts generated by $(CGLP)_{k_1,k_2}$ are called *lift-and-project cuts*. The results of Balas [7] imply that

Proposition 1 *The optimum value of $(CGLP)_{k_1,k_2}$ is negative if and only if $\hat{x}$ is not in the set* $\text{conv}\{x \in \mathbb{R}^n \mid \tilde{A}x \geq \tilde{b}, (-\tilde{A}_{k_1}x \geq -\tilde{b}_{k_1} \vee -\tilde{A}_{k_2}x \geq -\tilde{b}_{k_2})\}$.

Notice that $(CGLP)_{k_1,k_2}$ is a generalisation of $(CGLP)_k$ of Balas and Perregaard [10] for mixed 0/1 programming. By generalising the results of [10], we obtain the following:

Proposition 2 *Unless $u_0 > 0$ and $v_0 > 0$, the optimum value of $(CGLP)_{k_1,k_2}$ is non-negative.*

By substitution for α and β, we obtain an equivalent linear program

$$
\begin{aligned}
(CGLP)'_{k_1,k_2} \qquad & \min_{(u,v,u_0,v_0)} (u\tilde{A} - u_0\tilde{A}_{k_1})\hat{x} - u\tilde{b} + u_0\tilde{b}_{k_1} \\
& \text{s.t.} \quad (u,-v)\begin{pmatrix} \tilde{A} \\ \tilde{A} \end{pmatrix} - u_0\tilde{A}_{k_1} + v_0\tilde{A}_{k_2} = 0, \\
& \qquad (u,-v)\begin{pmatrix} \tilde{b} \\ \tilde{b} \end{pmatrix} - u_0\tilde{b}_{k_1} + v_0\tilde{b}_{k_2} = 0, \\
& \qquad u\mathbb{1} + v\mathbb{1} + u_0 + v_0 = 1, \\
& \qquad u, v, u_0, v_0 \geq 0.
\end{aligned}
$$

A slightly different cut generation LP is developed by Andersen et al. [1] for disjunctive programming with two term disjunctions, where each term is a conjunction of inequalities.

3 Correspondence between disjunctive cuts and lift-and-project cuts

Now we establish the connection between the disjunctive cuts (9) and the lift-and-project cuts found by $(CGLP)_{k_1,k_2}$. Consider a basis (I, J) and the corresponding simplex tableau (6). We define the index sets $B := N \cap I$, $R := N \cap J$, $P := S \cap I$ and $Q := S \cap J$ as in [10]. If $i \in Q$, then s_i is nonbasic, therefore, the corresponding row of $Ax \geq b$ holds with equality. If $i \in R$, then x_i is nonbasic, hence, $x_i \geq 0$ holds with equality. Let $\tilde{A}_J$ be the square submatrix of $\tilde{A}$ consisting of the rows indexed by J. Similarly, let T^I consist of those columns of $T := (A, -I_m)$ indexed by I. The following fact is used in [10], and we provide a proof for the sake of completeness.

Proposition 3 *T^I is non-singular if and only if $\tilde{A}_J$ is nonsingular.*

Proof We rewrite (5) with respect to I and J:

$$
\begin{array}{lllllll}
A_Q^B x_B & & & + & A_Q^R x_R & - \; s_Q & = b_Q \\
A_P^B x_B & - & s_P & + & A_P^R x_R & & = b_P.
\end{array}
$$

Recall that $T^I = \begin{pmatrix} A_Q^B & 0 \\ A_P^B & -I_{|P|} \end{pmatrix}$ and $\tilde{A}_J = \begin{pmatrix} A_Q^B & A_Q^R \\ 0 & I_{|P|} \end{pmatrix}$. We prove only sufficiency, as necessity is explicitely proved in [10]. If $\tilde{A}_J$ is nonsingular, then A_Q^B is nonsingular as well. Therefore, the matrix $\begin{pmatrix} (A_Q^B)^{-1} & 0 \\ A_P^B(A_Q^B)^{-1} & -I_{|P|} \end{pmatrix}$ is well-defined and an easy calculation shows that it is indeed the inverse of T^I. □

Before we proceed, recall the definition of matrix $\tilde{A}$, which was defined as $\begin{pmatrix} A \\ I_n \end{pmatrix}$. Therefore, for each basis variable $i \in I$, $\tilde{A}_i = e_{i-m}^T$ if $i \in N$ (i indexes a structural variable), and $\tilde{A}_i = (A_i^B, A_i^R)$ if $i \in M$ (i indexes a surplus variable). Balas and Perregaard have proved the following result:

Lemma 2 *(Balas and Perregaard [10]) In the simplex tableau (6) with basic variables I and nonbasic variables J, the coefficients $\overline{a}_{ij}$ for $i \in I$ and $j \in J$, and the right hand sides $\overline{a}_{i0}$ for $i \in I$ satisfy*

$$\overline{a}_{ij} = -(\tilde{A}_i \tilde{A}_J^{-1})_j, \tag{17}$$
$$\overline{a}_{i0} = \tilde{A}_i \tilde{A}_J^{-1} \tilde{b}_J - \tilde{b}_i. \tag{18}$$

Based on this, we prove the following:

Lemma 3 *Each basis I of the linear system (5) with $k_1, k_2 \in I$ and $\overline{a}_{k_1,0}, \overline{a}_{k_2,0} > 0$ determines a disjunctive cut $\pi s_J \geq \pi_0$ and a basic feasible solution $(\alpha, \beta, u, v, u_0, v_0)$ of $(CGLP)_{k_1,k_2}$ such that the cuts $\pi s_J \geq \pi_0$ and $\alpha x \geq \beta$ are equivalent.*

Proof $\pi_j^1 := \overline{a}_{k_1,j}\overline{a}_{k_2,0}$, $\pi_j^2 := \overline{a}_{k_2,j}\overline{a}_{k_1,0}$, $\pi_j := \max\{\pi_j^1, \pi_j^2\}$ for $j \in J$, and $\pi_0 := \overline{a}_{k_1,0}\overline{a}_{k_2,0}$, where $J \subset M \cup N$ is the set of non-basic variables. Notice that the cut $\pi s_J \geq \pi_0$ is precisely the one determined by (9) from the simplex tableau.

We define the values of $\alpha, \beta, u, v, u_0, v_0$ as follows: Let θ be a positive value to be fixed later, and

$$\begin{aligned} \theta\alpha &:= \pi \tilde{A}_J, & \theta\beta &:= \pi_0 + \pi \tilde{b}_J, \\ \theta u_J &:= \pi - \pi^1, & \theta v_J &= \pi - \pi^2, \\ \theta u_0 &:= \overline{a}_{k_2,0}, & \theta v_0 &:= \overline{a}_{k_1,0}. \end{aligned}$$

For all $j \in N \cup S \setminus J$, $u_j = v_j = 0$. We verify that this particular choice of $(\alpha, \beta, u, v, u_0, v_0)$ is a feasible solution of $(CGLP)_{k_1,k_2}$.

$$\begin{aligned} \theta(\alpha - u_J \tilde{A}_J + u_0 \tilde{A}_{k_1}) &= \pi \tilde{A}_J - (\pi - \pi^1)\tilde{A}_J + \overline{a}_{k_2,0}\tilde{A}_{k_1} \\ &= \overline{a}_{k_2,0}(\tilde{A}_{k_1} - (\tilde{A}_{k_1}\tilde{A}_J^{-1})\tilde{A}_J) = 0 \end{aligned}$$

Here, we exploited that $\overline{a}_{k_1,j} = -(\tilde{A}_{k_1}\tilde{A}_J^{-1})_j$ and then $\pi^1 = -\tilde{A}_{k_1}\tilde{A}_J^{-1}\overline{a}_{k_2,0}$. One similarly shows that $\theta(\alpha - v_J\tilde{A}_J + v_0\tilde{A}_{k_2}) = 0$. Furthermore, using (18) we obtain

$$\begin{aligned} \theta(\beta - u_J\tilde{b}_J + u_0\tilde{b}_{k_1}) &= \pi_0 + \pi\tilde{b}_J - (\pi - \pi^1)\tilde{b}_J + \overline{a}_{k_2,0}\tilde{b}_{k_1} \\ &= \pi_0 + \overline{a}_{k_2,0}(\tilde{b}_{k_1} - (\tilde{A}_{k_1}\tilde{A}_J^{-1})\tilde{b}_J) = \pi_0 + \overline{a}_{k_2,0}(-\overline{a}_{k_1,0}) = 0. \end{aligned}$$

We can prove similarly that $\theta(\beta - v_J\tilde{b}_J + v_0\tilde{b}_{k_1}) = 0$. Now we can define θ as

$$\theta := (\pi - \pi_J^1)\mathbb{1} + (\pi - \pi_J^2)\mathbb{1} + \overline{a}_{k_2 0} + \overline{a}_{k_1 0}.$$

Clearly, $\theta > 0$ and $u\mathbb{1} + v\mathbb{1} + u_0 + v_0 = 1$.

We verify that $u, v, u_0, v_0 \geq 0$. Since $\pi_j = \max\{\pi_j^1, \pi_j^2\}$ for $j \in J$, it follows that u_j and v_j are non-negative and at most one of them is greater than zero. Finally, since $\overline{a}_{k_1,0} > 0$ and $\overline{a}_{k_2,0} > 0$ by assumption, it follows that $u_0, v_0 > 0$.

To show that the solution $(\alpha, \beta, u, v, u_0, v_0)$ of $(CGLP)_{k_1,k_2}$ defined above is *basic*, firstly we define a partitioning of J into two subsets, M_1 and M_2 as follows. If $\pi_j^1 < \pi_j^2$, then $j \in M_1$; if $\pi_j^1 > \pi_j^2$, then $j \in M_2$, and if $\pi_j^1 = \pi_j^2$, break ties arbitrarily. Since $u_j = 0$ if $j \notin M_1$, and $v_j = 0$ if $j \notin M_2$, we know that $(u_{M_1}, v_{M_2}, u_0, v_0)$ satisfy the constraints

$$(u_{M_1}, -v_{M_2}) \begin{pmatrix} \tilde{A}_{M_1} \\ \tilde{A}_{M_2} \end{pmatrix} - u_0 \tilde{A}_{k_1} + v_0 \tilde{A}_{k_2} = 0, \tag{19}$$

$$(u_{M_1}, -v_{M_2}) \begin{pmatrix} \tilde{b}_{M_1} \\ \tilde{b}_{M_2} \end{pmatrix} - u_0 \tilde{b}_{k_1} + v_0 \tilde{b}_{k_2} = 0, \tag{20}$$

$$u_{M_1} \mathbb{1}_{M_1} + v_{M_2} \mathbb{1}_{M_2} + u_0 + v_0 = 1. \tag{21}$$

Since $\tilde{A}_J = \begin{pmatrix} \tilde{A}_{M_1} \\ \tilde{A}_{M_2} \end{pmatrix}$, and $\tilde{A}_J$ is invertible by Proposition 3, we have from (19):

$$(u_{M_1}, -v_{M_2}) = u_0 \tilde{A}_{k_1} \tilde{A}_J^{-1} - v_0 \tilde{A}_{k_2} \tilde{A}_J^{-1}.$$

Using (17) we obtain

$$\begin{aligned} u_j &= -u_0 \overline{a}_{k_1,j} + v_0 \overline{a}_{k_2,j}, \quad j \in M_1, \\ v_j &= u_0 \overline{a}_{k_1,j} - v_0 \overline{a}_{k_2,j}, \quad j \in M_2. \end{aligned} \tag{22}$$

Moreover, from (20) it follows that

$$u_0 \tilde{A}_{k_1} \tilde{A}_J^{-1} \tilde{b}_J - v_0 \tilde{A}_{k_2} \tilde{A}_J^{-1} \tilde{b}_J - u_0 \tilde{b}_{k_1} + v_0 \tilde{b}_{k_2} = 0.$$

Using (18) we get

$$u_0 \overline{a}_{k_1,0} - v_0 \overline{a}_{k_2,0} = 0. \tag{23}$$

On the other hand, substituting (22) into (21) gives

$$u_0 (1 - \sum_{j \in M_1} \overline{a}_{k_1,j} + \sum_{j \in M_2} \overline{a}_{k_1,j}) + v_0 (1 + \sum_{j \in M_1} \overline{a}_{k_2,j} - \sum_{j \in M_2} \overline{a}_{k_2,j}) = 1. \tag{24}$$

The determinant of the linear system (23) and (24) over the variables u_0 and v_0 is

$$D := \overline{a}_{k_2 0} (1 - \sum_{j \in M_1} \overline{a}_{k_1,j} + \sum_{j \in M_2} \overline{a}_{k_1,j}) + \overline{a}_{k_1 0} (1 + \sum_{j \in M_1} \overline{a}_{k_2,j} - \sum_{j \in M_2} \overline{a}_{k_2,j}).$$

Observe that this quantity is precisely the value θ, defined above. To see this, using the definitions of π^1, π^2, M_1 and M_2 we obtain

$$\begin{aligned} \theta &= (\pi - \pi_J^1) \mathbb{1} + (\pi - \pi_J^2) \mathbb{1} + \overline{a}_{k_2 0} + \overline{a}_{k_1 0} \\ &= \sum_{j \in M_1} (\overline{a}_{k_2,j} \overline{a}_{k_1,0} - \overline{a}_{k_1,j} \overline{a}_{k_2,0}) + \sum_{j \in M_2} (\overline{a}_{k_1,j} \overline{a}_{k_2,0} - \overline{a}_{k_2,j} \overline{a}_{k_1,0}) + \overline{a}_{k_1,0} + \overline{a}_{k_2,0}. \end{aligned}$$

Rearranging terms gives the determinant D. Since the determinant equals θ, a positive number, u_0 and v_0 are uniquely determined. Since w.l.o.g. α and β, are

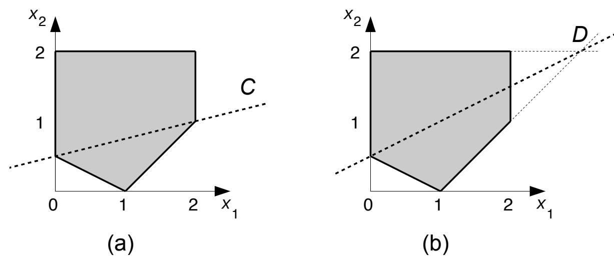

Figure 1: The disjunctive cuts C and D of Example 1.

basic, u_{M_1} and v_{M_2} are uniquely determined by u_0 and v_0, and the rest of the variables are non-basic, $(\alpha, \beta, u, v, u_0, v_0)$ is a basic solution.

Finally, the two cuts are equivalent, since

$$\theta(\alpha x - \beta) = \pi \tilde{A}_J x - (\pi_0 + \pi \tilde{b}_J) = \pi(\tilde{A}_J x - \tilde{b}_J) - \pi_0 = \pi s_J - \pi_0.$$

□

Note that the content of Lemma 3 is different from that of Theorem 4A of [10] for mixed 0/1 linear programming. Namely, we start out from a (not necessarily feasible) basis I of LP, and show that $(CGLP)_{k_1,k_2}$ do has a *feasible basis* which determines a cut equivalent to the disjunctive cut (9). In contrast, by Theorem 4A of [10], given any basic feasible solution of the cut generation linear program, there exists an equivalent disjunctive cut with respect to the corresponding set of non-basic variables of LP. Moreover, Lemma 3 is more general than Theorem 4B of [10], which states that given an $n \times n$ non-singular submatrix $\hat{A}$ of $\tilde{A}$ with corresponding row indexes J, it gives rise to a disjunctive cut from the simplex tableau with non-basic variables J, and also to an equivalent basic feasible solution of the cut-generation linear program. On the one hand, we assume only that J is the set of non-basic variables in the original simplex tableau from which the disjunctive cut is derived. On the other hand, we derive our result for a more general cut-generation linear program corresponding to general two-term disjunctions.

Unfortunately, the proof of Theorem 4A of [10] does not carry over to the more general facial disjunctive programming problem. In fact, this is not a coincidence as shown by the following example.

Example 1 *Consider the facial disjunctive program*

$$\min_{x_1,x_2} -x_2$$

$$s.t.\quad x_1 + 2x_2 \geq 1 \tag{25}$$

$$-x_1 + x_2 \geq -1 \tag{26}$$

$$-x_2 \geq -2$$

$$x_1 \geq 0$$

$$-x_1 \geq -2$$

$$-x_1 - 2x_2 \geq -1 \text{ or } x_1 - x_2 \geq 1. \tag{27}$$

The disjunctive constraint (27) consists of the disjunction of the negation of constraints (25) and (26). The corresponding polytope is shown in Figure 1. The feasible points of DP lie on the line segments between $(0, \frac{1}{2})$ *and* $(1, 0)$*, and between* $(1, 0)$ *and* $(2, 1)$*. The disjunctive cut* $\frac{1}{4}x_1 - x_2 \geq -\frac{1}{2}$ *(indicated by line C in Figure 1 (a)) is determined by the first term of the disjunction and the defining inequality* $x_1 \geq 0$*. On the other hand, it is also determined by the second term of the disjunction and by the defining inequality* $-x_1 \geq -2$*. However, the only way to derive this disjunctive cut is that the slack variables of* $x_1 \geq 0$ *and* $-x_1 \geq -2$ *are both non-basic, which is impossible in any basis of LP. Stated differently, the submatrix* $\begin{pmatrix} 1 & 0 & 0 & 1 \\ -1 & 0 & -2 & 1 \\ -1 & -2 & -1 & 1 \\ 1 & -1 & 1 & 1 \end{pmatrix}$ *constitutes a basis of* $(CGLP)'_{k_1,k_2}$*, where the first two rows correspond to* $x_1 \geq 0$ *and* $-x_1 \geq -2$*, respectively, and the last two rows to the two terms of the disjunction. But then* $\tilde{A}_J = \begin{pmatrix} 1 & 0 \\ -1 & 0 \end{pmatrix}$*, which is a singular matrix, therefore, there is no corresponding basis of the simplex tableau by Proposition 3. This shows that Theorem 4A of [10] is not valid in general for facial disjunctive programming.*

Nevertheless, there are two disjunctive cuts that may be obtained, one of them (line D) is shown in Figure 1 (b), and the other is symmetric to this.

3.1 The $0-1$ special case

Now we derive known results for the mixed 0/1 linear programming special case. Firstly, using (10) and Lemma 1, the disjunctive cut (9) becomes

$$\sum_{j \in J} \max\{\overline{a}_{kj}(1 - \overline{a}_{k0}), -\overline{a}_{kj}\overline{a}_{k0}\} s_j \geq \overline{a}_{k0}(1 - \overline{a}_{k0}),$$

which is precisely the *simple cut* in [10]. Thanks to Lemma 1 we do not need the inequalities (10) to derive this cut.

Corollary 1 *Let* x_k *be a 0/1 variable. Each basis I of the augmented linear program (5) with* $0 < \overline{a}_{k0} < 1$ *determines a basic feasible solution* $(\alpha, \beta, u, v, u_0, v_0)$ *of* $(CGLP)_{k_1,k_2}$ *such that the disjunctive cut* $\pi s_J \geq \pi_0$ *and* $\alpha x \geq \beta$ *are equivalent.*

It is worth mentioning that in the 0/1 case, θ becomes

$$\theta = 1 - \sum_{j \in M_1} \overline{a}_{kj} + \sum_{j \in M_2} \overline{a}_{kj}.$$

Since $u_0 = (1 - \overline{a}_{k0})\theta^{-1}$ and $v_0 = \overline{a}_{k0}\theta^{-1}$, we can easily compute $u_0 + v_0$:

$$u_0 + v_0 = (1 - \overline{a}_{k0} + \overline{a}_{k0})\theta^{-1} = \theta^{-1}.$$

Consequently, for any partitioning $M_1 \cup M_2$ of J such that $u, v \geq 0$,

$$\begin{aligned} u_j &= -(u_0 + v_0)\overline{a}_{kj} = -\overline{a}_{kj}\theta^{-1}, \quad \text{for } j \in M_1, \\ v_j &= (u_0 + v_0)\overline{a}_{kj} = \overline{a}_{kj}\theta^{-1}, \quad \text{for } j \in M_2. \end{aligned}$$

These are precisely the formulae obtained in [10].

4 Computations in the small tableau

The results of this section are generalisations of those of [10] for mixed 0/1 integer linear programming. The formulae become more intricate, since we allow more general disjunctions than $x_k \leq 0$ or $x_k \geq 1$.

The procedure for finding the most violated disjunctive cut (9) is the same as in [10]. Let $(\hat{x}, \hat{s})$ be a basic feasible solution of LP corresponding to the basic variables I and non-basic variables J, such that at least one disjunctive constraint is violated. Suppose $(k_1, k_2) \in D$ identifies a violated disjunctive constraint, i.e., $\hat{s}_{k_1} > 0$ and $\hat{s}_{k_2} > 0$. This gives a violated disjunctive cut (9), which will be improved subsequently. Recall that $I \cup J = N \cup S$, $I \cap J = \emptyset$. For each $i \in I$ we determine the reduced cost values r_{u_i} and r_{v_i} in $(CGLP)_{k_1,k_2}$. If $r_{u_i} < 0$ or $r_{v_i} < 0$, then we determine a pivot column $j \in J$, such that after exchanging i and j, the disjunctive cut in the new basis I' has a larger violation. Repeating this while the violation of the cut increases, we obtain a more violated cut. The output of the procedure does not necessarily correspond to an optimal solution of $(CGLP)_{k_1,k_2}$. On the one hand, in case of degeneracy, the search stops as there is no immediate improvement. On the other hand, by Example 1, the optimal solution of $(CGLP)_{k_1,k_2}$ may not be represented by a disjunctive cut. Therefore, upon termination of the above cut-generation procedure, the output is an improved disjunctive cut, which may not correspond to an optimal solution of $(CGLP)_{k_1,k_2}$.

For the sake of completeness, we summarise the cut-generation procedure in Algorithm 1. The loop is repeated as long as a more violated cut is found than the actual one. Upon termination, the cut is computed in the space of structural variables, i.e., all surplus variables with $\pi_j \neq 0$, $j \in J$, are replaced by the corresponding inequalities of LP.

In the next two subsections we explain how to select the basic variable $i \in I \setminus \{k_1, k_2\}$ to leave the basis and then how to choose the non-basic variable $\ell \in J$ to enter the basis.

Algorithm 1 cut-generation

Require: simplex tableau of LP, basis I, feasible basis solution $(\hat{x}, \hat{s})$, disjunctive constraint $(k_1, k_2) \in D$ such that $s_{k_1} > 0$ or $s_{k_2} > 0$.

Ensure: disjunctive cut $\alpha x \geq \beta$ violated by $(\hat{x}, \hat{s})$.

1: Compute the disjunctive cut $\pi s_J \geq \pi_0$ with respect to J.
2: $cutviol := \pi \hat{s}_J - \pi_0$, $optviol := 0$.
3: **while** $cutviol < optviol$ **do**
4:     $optviol := cutviol$.
5:     Determine $i \in I \setminus \{k_1, k_2\}$ with negative reduced cost r_{u_i} or r_{v_i} in $(CGLP)_{k_1,k_2}$.
6:     **if** no $i \in I$ exists with negative reduced cost r_{u_i} or r_{v_i} **then**
7:         **goto** line 14.
8:     **end if**
9:     Determine $\ell \in J$ such that after pivoting on (i, ℓ), the disjunctive cut in the new basis has the largest violation, and $\overline{a}_{k_1} > 0$, $\overline{a}_{k_2} > 0$.
10:     Pivot on (i, ℓ) in the simplex tableau of LP. $I := I \setminus \{i\} \cup \{\ell\}$, $J := J \setminus \{\ell\} \cup \{i\}$.
11:     Compute the disjunctive cut $\pi s_J \geq \pi_0$ with respect to J.
12:     $cutviol := \pi \hat{s}_J - \pi_0$.
13: **end while**
14: Determine $\alpha x \geq \beta$ from $\pi \hat{s}_J \geq \pi_0$ by substitutions into surplus variables.

4.1 Computation of reduced costs

Given the set of non-basic variables in the current simplex tableau, by Lemma 3 there is at least one feasible basis (M_1, M_2, k_1, k_2) of $(CGLP)'_{k_1,k_2}$ such that $M_1 \cup M_2 = J$ (the variables α and β can be assumed basic). Namely, for $j \in J$, let $\Delta_j = \overline{a}_{k_2 j}\overline{a}_{k_1 0} - \overline{a}_{k_1 j}\overline{a}_{k_2 0}$. To induce a feasible basis solution, M_1 has to contain all the $j \in J$ with $\Delta_j > 0$, and M_2 has to contain all the $j \in J$ with $\Delta_j < 0$. However, those $j \in J$ with $\Delta_j = 0$ can be distributed arbitrarily between M_1 and M_2.

Lemma 4 *Let J be the set of non-basic variables in the current simplex tableau such that $\overline{a}_{k_1}, \overline{a}_{k_2} > 0$, and M_1, M_2 a partitioning of J. Moreover, let $\hat{s}$ be the value of the surplus variables with respect to a basic feasible solution $\hat{x}$. The reduced costs of the variables u_i and v_i with $i \in I$ can be computed as*

$$r_{u_i} = \sum_{j \in M_1} \overline{a}_{ij}\hat{s}_j - \sigma(1 + \xi_i) - \frac{\overline{a}_{i0}\omega}{\theta} + \hat{s}_i \tag{28}$$

$$r_{v_i} = \sum_{j \in M_1} -\overline{a}_{ij}\hat{s}_j - \sigma(1 - \xi_i) + \frac{\overline{a}_{i0}\omega}{\theta} \tag{29}$$

where

$$\xi_i = \sum_{j\in M_1} \overline{a}_{ij} - \sum_{j\in M_2} \overline{a}_{ij},$$
$$\tau_1 = \sum_{j\in M_1} \overline{a}_{k_1 j} - \sum_{j\in M_2} \overline{a}_{k_1 j},$$
$$\tau_2 = \sum_{j\in M_1} \overline{a}_{k_2 j} - \sum_{j\in M_2} \overline{a}_{k_2 j},$$
$$\theta = \overline{a}_{k_1,0}(1+\tau_2) + \overline{a}_{k_2,0}(1-\tau_1),$$
$$\sigma = \sum_{j\in M_1} \frac{(\overline{a}_{k_2 j}\overline{a}_{k_1 0} - \overline{a}_{k_1 j}\overline{a}_{k_2 0})\hat{s}_j}{\theta} - \frac{\overline{a}_{k_2 0}\hat{s}_{k_1}}{\theta},$$
$$\omega = (1+\tau_2)\sum_{j\in M_1} \overline{a}_{k_1 j}\hat{s}_j + (1-\tau_1)\sum_{j\in M_1} \overline{a}_{k_2 j}\hat{s}_j + (1+\tau_2)\hat{s}_{k_1}.$$

Proof The idea of the proof is that we express the variables u_j, $j \in M_1$, in terms of u_i, v_i, u_0 and v_0, and then we rewrite the objective function of $(CGLP)'_{k_1,k_2}$ using these formulae. We restrict the constraints of $(CGLP)'_{k_1,k_2}$ to the variables u_j for $j \in M_1$, v_j for $j \in M_2$, u_i, v_i, u_0 and v_0.

$$(u_{M_1}, -v_{M_2}) \begin{pmatrix} \tilde{A}_{M_1} \\ \tilde{A}_{M_2} \end{pmatrix} + u_i\tilde{A}_i - v_i\tilde{A}_i - u_0\tilde{A}_{k_1} + v_0\tilde{A}_{k_2} = 0 \qquad (30)$$
$$(u_{M_1}, -v_{M_2}) \begin{pmatrix} \tilde{b}_{M_1} \\ \tilde{b}_{M_2} \end{pmatrix} + u_i\tilde{b}_i - v_i\tilde{b}_i - u_0\tilde{b}_{k_1} + v_0\tilde{b}_{k_2} = 0 \qquad (31)$$
$$u_{M_1}\mathbb{1}_{M_1} + v_{M_2}\mathbb{1}_{M_2} + u_i + v_i + u_0 + v_0 = 1 \qquad (32)$$

Since $\tilde{A}_J = \begin{pmatrix} \tilde{A}_{M_1} \\ \tilde{A}_{M_2} \end{pmatrix}$, we have from (30):

$$(u_{M_1}, -v_{M_2}) = -u_i\tilde{A}_i\tilde{A}_J^{-1} + v_i\tilde{A}_i\tilde{A}_J^{-1} + u_0\tilde{A}_{k_1}\tilde{A}_J^{-1} - v_0\tilde{A}_{k_2}\tilde{A}_J^{-1} \qquad (33)$$

Substituting into (31) gives

$$-(u_i - v_i)(\tilde{A}_i\tilde{A}_J^{-1}\tilde{b}_J - \tilde{b}_i) + u_0(\tilde{A}_{k_1}\tilde{A}_J^{-1}\tilde{b}_J - \tilde{b}_{k_1}) - v_0(\tilde{A}_{k_2}\tilde{A}_J^{-1}\tilde{b}_J - \tilde{b}_{k_2}) = 0,$$

which is equivalent to

$$u_0\overline{a}_{k_1,0} - v_0\overline{a}_{k_2,0} = (u_i - v_i)\overline{a}_{i0}. \qquad (34)$$

On the other hand, substituting into (32) gives

$$-(u_i - v_i)(\tilde{A}_i\tilde{A}_J^{-1}\mathbb{1}_{M_1} - \tilde{A}_i\tilde{A}_J^{-1}\mathbb{1}_{M_2}) + u_i + v_i +$$
$$u_0(1 + \tilde{A}_{k_1}\tilde{A}_J^{-1}\mathbb{1}_{M_1} - \tilde{A}_{k_1}\tilde{A}_J^{-1}\mathbb{1}_{M_2}) + v_0(1 - \tilde{A}_{k_2}\tilde{A}_J^{-1}\mathbb{1}_{M_1} + \tilde{A}_{k_2}\tilde{A}_J^{-1}\mathbb{1}_{M_2}) = 1$$

which is equivalent to

$$u_0(1 - \sum_{j\in M_1} \overline{a}_{k_1,j} + \sum_{j\in M_2} \overline{a}_{k_1,j}) + v_0(1 + \sum_{j\in M_1} \overline{a}_{k_2,j} - \sum_{j\in M_2} \overline{a}_{k_2,j}) \\ = 1 - u_i - v_i - (u_i - v_i)(\sum_{j\in M_1} \overline{a}_{ij} - \sum_{j\in M_2} \overline{a}_{ij}). \tag{35}$$

Consequently, we can express u_0 and v_0 from the equations (34) and (35):

$$\frac{\overline{a}_{k_1,0}}{\overline{a}_{k_2,0}} u_0 - \frac{\overline{a}_{i0}}{\overline{a}_{k_2,0}}(u_i - v_i) = v_0$$

from which

$$u_0 = \frac{\overline{a}_{k_2,0} - \eta_i u_i - \lambda_i v_i}{\theta},$$

where

$$\eta_i = \overline{a}_{k_2 0}(1 + \xi_i) - \overline{a}_{i0}(1 + \tau_2),$$
$$\lambda_i = \overline{a}_{k_2 0}(1 - \xi_i) + \overline{a}_{i0}(1 + \tau_2),$$

with τ_2 and ξ_i defined in the statement of the lemma. On the other hand, from

$$\frac{\overline{a}_{k_2,0}}{\overline{a}_{k_1,0}} v_0 + \frac{\overline{a}_{i0}}{\overline{a}_{k_1,0}}(u_i - v_i) = u_0$$

we obtain

$$v_0 = \frac{\overline{a}_{k_1,0} - \mu_i u_i - \nu_i v_i}{\theta},$$

where

$$\mu_i = \overline{a}_{k_1 0}(1 + \xi_i) + \overline{a}_{i0}(1 - \tau_1),$$
$$\nu_i = \overline{a}_{k_1 0}(1 - \xi_i) - \overline{a}_{i0}(1 - \tau_1),$$

with τ_1 defined in the statement of the lemma. From (33) we also get expressions for u_j and v_j:

$$u_j = (u_i - v_i)\overline{a}_{ij} - u_0\overline{a}_{k_1,j} + v_0\overline{a}_{k_2,j} \text{ for } j \in M_1,$$
$$v_j = -(u_i - v_i)\overline{a}_{ij} + u_0\overline{a}_{k_1,j} - v_0\overline{a}_{k_2,j} \text{ for } j \in M_2.$$

Combining all these, the objective function of $(CGLP)'_{k_1,k_2}$ can be rewritten as

$$\sum_{j\in M_1} (\tilde{A}_j\hat{x} - \tilde{b}_j)u_j + (\tilde{A}_i\hat{x} - \tilde{b}_i)u_i - (\tilde{A}_{k_1}\hat{x} - \tilde{b}_{k_1})u_0 = \sum_{j\in M_1} \hat{s}_j u_j + \hat{s}_i u_i - \hat{s}_{k_1} u_0$$
$$= \sum_{j\in M_1} \left((u_i - v_i)\overline{a}_{ij} - \frac{\overline{a}_{k_1,j}(\overline{a}_{k_2,0} - \eta_i u_i - \lambda_i v_i)}{\theta} + \frac{\overline{a}_{k_2,j}(\overline{a}_{k_1,0} - \mu_i u_i - \nu_i v_i)}{\theta} \right) \hat{s}_j$$
$$+ \hat{s}_i u_i - \frac{\hat{s}_{k_1}(\overline{a}_{k_2,0} - \eta_i u_i - \lambda_i v_i)}{\theta}.$$

The constant term in this expression is σ as defined in the statement of the lemma. The reduced costs of u_i and v_i are the coefficients of u_i and v_i, respectively, in the objective function. Therefore,

$$r_{u_i} = \sum_{j \in M_1} \overline{a}_{ij}\hat{s}_j + \sum_{j \in M_1} \frac{(\overline{a}_{k_1,j}\eta_i - \overline{a}_{k_2,j}\mu_i)\hat{s}_j}{\theta} + \frac{\hat{s}_{k_1}\eta_i}{\theta} + \hat{s}_i,$$
$$r_{v_i} = \sum_{j \in M_1} -\overline{a}_{ij}\hat{s}_j + \sum_{j \in M_1} \frac{(\overline{a}_{k_1,j}\lambda_i - \overline{a}_{k_2,j}\nu_i)\hat{s}_j}{\theta} + \frac{\hat{s}_{k_1}\lambda_i}{\theta}.$$

With further algebraic manipulations we obtain the desired results, the details are omitted. □

For computing efficiently all the reduced costs, we fix a partitioning $M_1 \cup M_2 = J$, and compute the terms ω, σ and θ, which are independent of row i. Let $M_3 = \{j \in J \mid \overline{a}_{k_1 0}\overline{a}_{k_2 j} - \overline{a}_{k_2 0}\overline{a}_{k_1 j} = 0\}$.

Proposition 4 *Let $M_1' \cup M_2'$ be any partitioning of J such that $M_1 \setminus M_3 = M_1' \setminus M_3$ and $M_2 \setminus M_3 = M_2' \setminus M_3$. Then the values of σ and θ are the same with respect to $M_1 \cup M_2$, and $M_1' \cup M_2'$.*

Proof Concerning θ:

$$\begin{aligned}
\theta &= \overline{a}_{k_1,0}(1+\tau_2) + \overline{a}_{k_2,0}(1-\tau_1) \\
&= \overline{a}_{k_1 0} + \overline{a}_{k_2 0} + \sum_{j \in M_1} (\overline{a}_{k_1 0}\overline{a}_{k_2 j} - \overline{a}_{k_2 0}\overline{a}_{k_1 j}) - \sum_{j \in M_2} (\overline{a}_{k_1 0}\overline{a}_{k_2 j} - \overline{a}_{k_2 0}\overline{a}_{k_1 j}) \\
&= \overline{a}_{k_1 0} + \overline{a}_{k_2 0} + \sum_{j \in M_1 \setminus M_3} (\overline{a}_{k_1 0}\overline{a}_{k_2 j} - \overline{a}_{k_2 0}\overline{a}_{k_1 j}) - \sum_{j \in M_2 \setminus M_3} (\overline{a}_{k_1 0}\overline{a}_{k_2 j} - \overline{a}_{k_2 0}\overline{a}_{k_1 j}) \\
&= \overline{a}_{k_1 0} + \overline{a}_{k_2 0} + \sum_{j \in M_1' \setminus M_3} (\overline{a}_{k_1 0}\overline{a}_{k_2 j} - \overline{a}_{k_2 0}\overline{a}_{k_1 j}) - \sum_{j \in M_2' \setminus M_3} (\overline{a}_{k_1 0}\overline{a}_{k_2 j} - \overline{a}_{k_2 0}\overline{a}_{k_1 j}) \\
&= \overline{a}_{k_1 0} + \overline{a}_{k_2 0} + \sum_{j \in M_1'} (\overline{a}_{k_1 0}\overline{a}_{k_2 j} - \overline{a}_{k_2 0}\overline{a}_{k_1 j}) - \sum_{j \in M_2'} (\overline{a}_{k_1 0}\overline{a}_{k_2 j} - \overline{a}_{k_2 0}\overline{a}_{k_1 j}).
\end{aligned}$$

The proof for σ is similar. □.

Unfortunately, the above statement does not hold for ω, i.e., it matters how the elements of M_3 are distributed between M_1 and M_2.

With respect to a fixed partitioning $M_1 \cup M_2 = J$, for each row i it suffices to determine $\overline{a}_{i0}$, and compute ξ_i and $\sum_{j \in M_1} \overline{a}_{ij}\hat{s}_j$ to obtain r_{u_i} and r_{v_i}.

4.1.1 The mixed $0-1$ programming special case

Concerning the mixed $0-1$ programming case, it can be shown that Lemma 4 reduces to Theorem 9 of [10] and Theorem 2.9 of [20]. Namely, the first two terms in the definition of r_{u_i} and r_{v_i} are easily seen equivalent to that of the

$0-1$ case. Concerning the term $\overline{a}_{i0}\omega/\theta$, we exploit that $\overline{a}_{k_1 j} = -\overline{a}_{k_2 j}$ and $\overline{a}_{k_1 0} + \overline{a}_{k_2 0} = 1$ by Lemma 1, which implies $\tau_1 = -\tau_2$, and

$$\begin{aligned}
\frac{\overline{a}_{i0}\omega}{\theta} &= \frac{\overline{a}_{i0}((1+\tau_2)\sum_{j\in M_1}\overline{a}_{k_1 j}\hat{s}_j + (1-\tau_1)\sum_{j\in M_1}\overline{a}_{k_2 j}\hat{s}_j + (1+\tau_2)\hat{s}_{k_1})}{\overline{a}_{k_1 0}(1+\tau_2) + \overline{a}_{k_2 0}(1-\tau_1)} \\
&= \frac{\overline{a}_{i0}((1+\tau_2)\sum_{j\in M_1}\overline{a}_{k_1 j}\hat{s}_j + (1+\tau_2)\sum_{j\in M_1} -\overline{a}_{k_1 j}\hat{s}_j + (1+\tau_2)\hat{s}_{k_1})}{1+\tau_2} \\
&= \overline{a}_{i0}\hat{s}_{k_1}.
\end{aligned}$$

Consequently,

$$\begin{aligned}
r_{u_i} &= \sum_{j\in M_1} \overline{a}_{ij}\hat{s}_j - \sigma(1+\xi_i) - \overline{a}_{i0}\hat{s}_{k_1} + \hat{s}_i, \\
r_{v_i} &= \sum_{j\in M_1} -\overline{a}_{ij}\hat{s}_j - \sigma(1-\xi_i) + \overline{a}_{i0}\hat{s}_{k_1},
\end{aligned}$$

which is equivalent to formulae (2.22) of [20].

4.2 Selection of pivot element

After selecting the pivot row based on reduced costs, an element has to be chosen to pivot on. Suppose row $i \in I \setminus \{k_1, k_2\}$ is the pivot row and column $\ell \in J$ is chosen as pivot column. Pivoting on $\overline{a}_{i\ell}$ in the simplex tableau (6) consists of adding $-a_{k\ell}/a_{i\ell}$ times row i to row k, for each row $k \neq i$, and multiplying row i by $1/a_{i\ell}$. Since $i \in I$ and $\ell \in J$, this operation transforms row k as follows:

$$s_k + \gamma s_i + \sum_{j\in J}(\overline{a}_{kj} + \gamma\overline{a}_{ij})s_j = \overline{a}_{k0} + \gamma\overline{a}_{i0},$$

where $\gamma = -\overline{a}_{k\ell}/\overline{a}_{i\ell}$. Notice that in this formula, s_j may be replaced by x_j if $j \in N \cap (J \cup \{i,k\})$. For $k = k_1$ and $k = k_2$, we obtain the rows k_1 and k_2 after the pivot operation, i.e.,

$$\begin{aligned}
s_{k_1} + \gamma_1 s_i + \sum_{j\in J}(\overline{a}_{k_1,j} + \gamma_1\overline{a}_{ij})s_j &= \overline{a}_{k_1,0} + \gamma_1\overline{a}_{i0}, \\
s_{k_2} + \gamma_2 s_i + \sum_{j\in J}(\overline{a}_{k_2,j} + \gamma_2\overline{a}_{ij})s_j &= \overline{a}_{k_2,0} + \gamma_2\overline{a}_{i0},
\end{aligned} \tag{36}$$

where $\gamma_1 = -\overline{a}_{k_1,\ell}/\overline{a}_{i\ell}$ and $\gamma_2 = -\overline{a}_{k_2,\ell}/\overline{a}_{i\ell}$. The disjunctive cut from the updated tableau rows (36) is

$$\pi_i s_i + \sum_{j\in J}\pi_j s_j \geq \pi_0, \tag{37}$$

where

$$\pi_0 := (\overline{a}_{k_1,0} + \gamma_1 \overline{a}_{i0})(\overline{a}_{k_2,0} + \gamma_2 \overline{a}_{i0})/\theta,$$
$$\pi_i := \max\{\pi_i^1(\gamma_1,\gamma_2), \pi_i^2(\gamma_1,\gamma_2)\}/\theta,$$
$$\pi_j := \max\{\pi_j^1(\gamma_1,\gamma_2), \pi_j^2(\gamma_1,\gamma_2)\}/\theta \text{ for all } j \in J,$$
$$\pi_i^1 := \gamma_1(\overline{a}_{k_2,0} + \gamma_2 \overline{a}_{i0}), \quad \pi_j^1 := (\overline{a}_{k_1,j} + \gamma_1 \overline{a}_{ij})(\overline{a}_{k_2,0} + \gamma_2 \overline{a}_{i0}),$$
$$\pi_i^2 := \gamma_2(\overline{a}_{k_1,0} + \gamma_1 \overline{a}_{i0}), \quad \pi_j^2 := (\overline{a}_{k_2,j} + \gamma_2 \overline{a}_{ij})(\overline{a}_{k_1,0} + \gamma_1 \overline{a}_{i0}).$$

The normalisation constant θ is needed to ensure that the cut is equivalent to a feasible solution of $(CGLP)_{k_1,k_2}$. Recall from the proof of Lemma 3 that

$$\begin{aligned}
\theta &= (\pi - \pi^1) + (\pi - \pi^2) + (\overline{a}_{k_2 0} + \gamma_2 \overline{a}_{i0}) + (\overline{a}_{k_1 0} + \gamma_1 \overline{a}_{i0}) \\
&= \sum_{j \in J} |\pi_j^1 - \pi_j^2| + |\pi_i^1 - \pi_i^2| + (\overline{a}_{k_2 0} + \gamma_2 \overline{a}_{i0}) + (\overline{a}_{k_1 0} + \gamma_1 \overline{a}_{i0}) \\
&= \sum_{j \in J} |\overline{a}_{k_1,j}(\overline{a}_{k_2 0} + \gamma_2 \overline{a}_{i0}) - \overline{a}_{k_2,j}(\overline{a}_{k_1 0} + \gamma_1 \overline{a}_{i0}) + \overline{a}_{ij}(\gamma_1 \overline{a}_{k_2 0} - \gamma_2 \overline{a}_{k_1 0})| \\
&\quad + |\gamma_1 \overline{a}_{k_2 0} - \gamma_2 \overline{a}_{k_1 0}| + (\overline{a}_{k_2 0} + \gamma_2 \overline{a}_{i0}) + (\overline{a}_{k_1 0} + \gamma_1 \overline{a}_{i0}).
\end{aligned}$$

Clearly, we choose that column $\ell \in J$ for which

$$\pi_i \hat{s}_i + \sum_{j \in J} \pi_j \hat{s}_j - \pi_0,$$

is minimal, and

$$\overline{a}_{k_1,0} + \gamma_1 \overline{a}_{i0} > 0 \text{ and } \overline{a}_{k_2,0} + \gamma_2 \overline{a}_{i0} > 0.$$

If $\overline{a}_{k_1,0} + \gamma_1 \overline{a}_{i0} \le 0$ or $\overline{a}_{k_2,0} + \gamma_2 \overline{a}_{i0} \le 0$ for all columns $\ell \in J$, then no pivot column can be chosen with respect to row i. Finally, by Lemma 3, the basis $(I \setminus \{i\}) \cup \{\ell\}$ induces a feasible solution of $(CGLP)_{k_1,k_2}$.

Numerically the formulae of π_0, π_i and those of the π_j-s are unattractive as they contain terms with 4 numbers. We can eliminate those terms by adding $-\pi_i$ times row i of the simplex tableau to (π, π_0).

Lemma 5 *If $\pi_i^1 \ge \pi_i^2$ then adding $-\pi_i(s_i + \overline{a}_i^J s_J - \overline{a}_{i0})$ to the cut (37) yields*

$$\pi_0' = \overline{a}_{k_1 0}(\overline{a}_{k_2 0} + \gamma_2 \overline{a}_{i0})/\theta,$$
$$\pi_i' = 0,$$
$$\pi_j' = \frac{\max\{\overline{a}_{k_1,j}(\overline{a}_{k_2 0} + \gamma_2 \overline{a}_{i0}), \overline{a}_{k_2 j}(\overline{a}_{k_1 0} + \gamma_1 \overline{a}_{i0}) + \overline{a}_{ij}(\gamma_2 \overline{a}_{k_1 0} - \gamma_1 \overline{a}_{k_2 0})\}}{\theta}.$$

while if $\pi_i^1 \le \pi_i^2$, then

$$\pi_0' = \overline{a}_{k_2 0}(\overline{a}_{k_1 0} + \gamma_1 \overline{a}_{i0})/\theta,$$
$$\pi_i' = 0,$$
$$\pi_j' = \frac{\max\{\overline{a}_{k_2,j}(\overline{a}_{k_1 0} + \gamma_1 \overline{a}_{i0}), \overline{a}_{k_1 j}(\overline{a}_{k_2 0} + \gamma_2 \overline{a}_{i0}) + \overline{a}_{ij}(\gamma_1 \overline{a}_{k_2 0} - \gamma_2 \overline{a}_{k_1 0})\}}{\theta}.$$

In either case, $\pi_i \hat{s}_i + \sum_{j \in J} \pi_j \hat{s}_j - \pi_0 = \pi_i' \hat{s}_i + \sum_{j \in J} \pi_j' \hat{s}_j - \pi_0'$.

Proof First suppose $\pi_i^1 \geq \pi_i^2$, i.e., $\pi_i = \pi_i^1 = \gamma_1(\overline{a}_{k_2 0} + \gamma_2 \overline{a}_{i0})$. Since

$$\theta\pi_0 = (\overline{a}_{k_2 0} + \gamma_2\overline{a}_{i0})(\overline{a}_{k_1 0} + \gamma_1\overline{a}_{i0}) = \overline{a}_{k_2 0}\overline{a}_{k_1 0} + \gamma_2\overline{a}_{i0}\overline{a}_{k_1 0} + \gamma_1\overline{a}_{i0}\overline{a}_{k_2 0} + \gamma_1\gamma_2\overline{a}_{i0}^2,$$

we have

$$\theta\pi_0 - \gamma_1(\overline{a}_{k_2 0} + \gamma_2\overline{a}_{i0})\overline{a}_{i0} = \overline{a}_{k_1 0}(\overline{a}_{k_2 0} + \gamma_2\overline{a}_{i0}).$$

To verify the formula for π_j, we compute the modified π_j^1 and π_j^2 values:

$$\theta\pi_j^1 - \theta\pi_i\overline{a}_{ij} = \theta\pi_j^1 - \gamma_1(\overline{a}_{k_2 0} + \gamma_2\overline{a}_{i0})\overline{a}_{ij} = \overline{a}_{k_1 j}(\overline{a}_{k_2 0} + \gamma_2\overline{a}_{i0}),$$

and

$$\begin{aligned}\theta\pi_j^2 - \theta\pi_i\overline{a}_{ij} &= \theta\pi_j^2 - \gamma_1(\overline{a}_{k_2 0} + \gamma_2\overline{a}_{i0})\overline{a}_{ij} \\ &= \overline{a}_{k_2 j}\overline{a}_{k_1 0} + \gamma_2\overline{a}_{ij}\overline{a}_{k_1 0} + \gamma_1\overline{a}_{k_2 j}\overline{a}_{i0} + \gamma_1\gamma_2\overline{a}_{ij}\overline{a}_{i0} - \gamma_1(\overline{a}_{k_2 0} + \gamma_2\overline{a}_{i0})\overline{a}_{ij} \\ &= \overline{a}_{k_2 j}(\overline{a}_{k_1 0} + \gamma_1\overline{a}_{i0}) + \overline{a}_{ij}(\gamma_2\overline{a}_{k_1 0} - \gamma_1\overline{a}_{k_2 0}).\end{aligned}$$

The case of $\pi_i^1 \leq \pi_i^2$ can be verified similarly. Finally, since variable s_i is in the basis, $\hat{s}_i + \overline{a}_i^J \hat{s}_J = \overline{a}_{i0}$, and therefore, we have modified the value of (37) by 0.
□

5 The pitfalls of determining facets of DP

The linear program $(CGLP)_{k_1,k_2}$ with the normalization constraint (16) may yield cuts far from being facets of the disjunctive program. Consider the facial disjunctive program

$$\min_{x_1,x_2} -x_2 \tag{38}$$
$$\text{s.t. } -x_1 \geq -4.5 \tag{39}$$
$$2x_1 - x_2 \geq -1 \tag{40}$$
$$x_1 - x_2 \geq -2 \tag{41}$$
$$-x_1 - x_2 \geq -6 \tag{42}$$
$$-2x_1 - x_2 \geq -10 \tag{43}$$
$$x_1 \geq 0 \tag{44}$$
$$x_2 \geq 0 \tag{45}$$
$$-x_1 \geq 0 \text{ or } x_1 \geq 4.5 \tag{46}$$

The disjunctive constraint consists of the disjunction of the negation of inequalities (44) and (39). The optimal basis solution of the LP relaxation is $\hat{x}(B_1) = (\hat{x}_1, \hat{x}_2) = (2, 4)$, indicated on the left of Figure 2. The cut-generation-

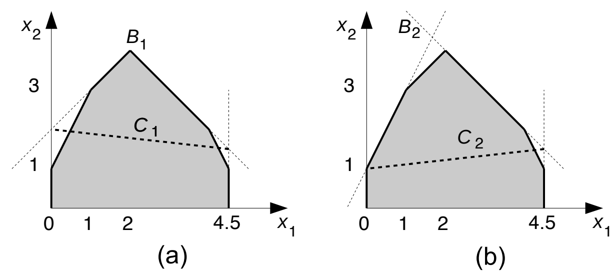

Figure 2: Disjunctive cuts for DP (38)-(46).

linear-program with respect to disjunction (46) is

$$\begin{aligned}
&\min 2.5u_1 + u_2 + 2u_5 + 2u_6 + 4u_7 - 2.5u_0 \\
&\text{s.t. } -(u_1 - v_1) + 2(u_2 - v_2) + (u_3 - v_3) \\
&\qquad\qquad -(u_4 - v_4) - 2(u_5 - v_5) + (u_6 - v_6) - u_0 - v_0 = 0 \\
&\quad -(u_2 - v_2) - (u_3 - v_3) - (u_4 - v_4) - (u_5 - v_5) - (u_7 - v_7) = 0 \\
&\quad -4.5(u_1 - v_1) - (u_2 - v_2) - 2(u_3 - v_3) \\
&\qquad\qquad -6(u_4 - v_4) - 10(u_5 - v_5) - 4.5v_0 = 0 \\
&\quad \sum_{i=1}^{7}(u_i + v_i) + u_0 + v_0 = 1 \\
&\quad u_i, v_i \geq 0, \quad \forall i = 0, \ldots, 7.
\end{aligned}$$

The disjunctive cut derived from basis B_1 is

$$-\frac{1}{36}x_1 - \frac{1}{4}x_2 \geq -\frac{1}{2}, \tag{47}$$

which is depicted by line C_1 in the figure. It corresponds to the basis solution of $(CGLP)_{(44)(39)}$ with $u_0 = \frac{10}{36}$, $u_3 = \frac{1}{4}$, $v_0 = \frac{8}{36}$ and $v_4 = \frac{1}{4}$. The difference between the left and the right hand sides in the point $(2, 4)$ is $-\frac{38}{36} + \frac{18}{36} = -\frac{5}{9}$. Notice that the slack variables of (41) and (42) are both non-basic, since $(\hat{x}_1, \hat{x}_2)$ satisfies both (41) and (42) at equality.

Now, if we perform a basis-exchange of the surplus variable of inequality (40) (which is in the basis), with that of (41) (which is non-basic), we obtain basis B_2, indicated on the right of Figure 2. The corresponding disjunctive cut is

$$\frac{1}{45}x_1 - \frac{1}{5}x_2 \geq -\frac{1}{5}, \tag{48}$$

depicted by line C_2 in Figure 2(b). The basis solution of $(CGLP)_{(44)(39)}$ is $u_0 = \frac{17}{45}$, $u_2 = \frac{9}{45}$, $v_0 = \frac{10}{45}$, $v_4 = \frac{9}{45}$. The violation of the latter inequality in the point $\hat{x}(B_1) = (2, 4)$ is $-\frac{5}{9}$ again. In fact, both of these inequalities are optimal solutions of $(CGLP)_{(44)(39)}$.

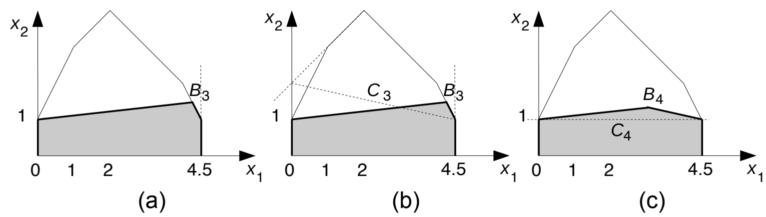

Figure 3: The steps of deriving the cut $-x_2 \geq -1$ (line C_4).

Unfortunately, the facet $-x_2 \geq -1$ of DP can be obtained only in three rounds of cut generation. Namely, suppose we choose cut (48), then we obtain a smaller polytope, shown in Figure 3(a). Its defining inequalities are (39)-(45), and (48). The basis solution w.r.t. B_3 is $\hat{x}(B_3) = (\frac{81}{19}, \frac{28}{19})$. The unique optimal solution is

$$-\frac{2}{45}x_1 - \frac{1}{5}x_2 \geq -\frac{2}{5} \tag{49}$$

shown by line C_3 in Figure 3(b). This cut is violated by $-\frac{8}{95}$ in the basis solution $(\frac{81}{19}, \frac{28}{19})$. There are various ways to derive a cut equivalent to $-x_2 \geq -1$. For instance, using (48), (43) and the two terms of (46) we obtain the cut

$$-\frac{45}{365}x_2 \geq -\frac{45}{365}$$

normalised to satisfy the last equation of $(CGLP)_{(44)(39)}$. This cut is violated by 0.058399 in point $(\frac{81}{19}, \frac{28}{19})$, which is less than the violation of (49). Another option is to use (40), (43), and the two terms of (46), giving the normalized cut

$$-\frac{1}{6}x_2 \geq -\frac{1}{6}.$$

This cut is violated by $-\frac{9}{114}$, which is still less than the violation of (49). By adding (49) to the polytope of Figure 3(b), we obtain the polytope in Figure 3(c). In this polytope, the optimal normalized cut w.r.t. the basis B_4 is

$$-\frac{1}{6}x_2 \geq -\frac{1}{6}.$$

Notice that had we dropped the inequalities (40)-(42) from the description of the polytope in Figure 3(a), we would have obtained the above facet of DP immediately.

The above example indicates the two main reasons of the possible slow convergence of the cut-generation process in deriving facets of conv(DP):

(i) The normalisation constraint prohibits the immediate generation of facets of DP.

(ii) After adding cuts, some of the defining inequalities may become superfluous, but lead to further cuts which are not facets of DP.

The role of normalisation is discussed e.g., in [5], [19] and [20]. Concerning (ii), in general it is hard to single out superfluous constraints.

6 Lifting of cuts

In this section we discuss how to lift those inequalities generated in a search-tree node. Firstly, we note that a straightforward branching rule for creating two descendants of a node is to select a violated disjunction $(k_1, k_2) \in D$ and then add each ineqalty in turn to the linear system of the current node. More precisely, if there exists $(k_1, k_2) \in D$ with

$$\tilde{A}_{k_1} x_0 > \tilde{b}_{k_1} \text{ and } \tilde{A}_{k_2} x_0 > \tilde{b}_{k_2},$$

for the LP solution x_0 of the current search-tree node, then two siblings are obtained by adding each one of $-\tilde{A}_{k_1} x \geq -\tilde{b}_{k_1}$ and $-\tilde{A}_{k_2} x \geq -\tilde{b}_{k_2}$ in turn to the linear system of the current node. Moreover, the union of the sets of all feasible solutions of the two siblings equals that of the current node. However, branching on inequalities raises the problem of how to make a cut, generated in a descendant node, to be globally valid.

Another use of cut lifting is that we can generate cuts only in a subspace of the columns. This technique has been applied in [5] the first time.

Let F be the set of those indices $k \in \{1, \ldots, m+n\}$ with $-\tilde{A}_k x \geq -\tilde{b}_k$ added to the initial linear program along the path from the root to the current search-tree node. Let $\alpha x \geq \beta$ be a cut derived in the current search-tree node. By *lifting* of $\alpha x \geq \beta$ we mean the determination of coefficients λ_k such that the inequality

$$(\alpha + \sum_{k \in F} \lambda_k \tilde{A}_k) x \geq \beta + \sum_{k \in F} \lambda_k \tilde{b}_k$$

is globally valid, i.e., valid for the convex hull of DP. We determine the coefficients λ_k one-by-one. There are two cases to consider:

- $k \in \{m+1, \ldots, m+n\}$, i.e., $\tilde{A}_k x \geq \tilde{b}_k$ is equivalent to $x_k \geq 0$, and the variable x_k is fixed to 0, or
- $k \in \{1, \ldots, m\}$, i.e., the slack variable s_k of $\tilde{A}_k x \geq \tilde{b}_k$ is fixed to 0.

The first case is well-studied in the literature, see e.g., [5], [6], [20]. It boils down to solving the cut generation problem in a subspace of variables (those not fixed to 0) and then lifting is merely the determination of the missing coefficients which is particularly easy for the procedure of Balas and Perregaard.

Concerning the second case, as in the simplex tableau there is no difference between the slack and the decision variables, the lifting of slack variables can be done similarly.

7 Computational evaluation

7.1 The branch-and-cut procedure

The cut-strengthening procedure has been embedded in a branch-and-cut algorithm for solving facial disjunctive programs with two-term disjunctions. One round of the cut-generation procedure consists of generating a disjunctive cut for each violated disjunctive constraint, but only the first 50 most violated disjunctions are considered, where the violation of a disjunction $(k_1, k_2) \in D$ is measured as $s_{k_1} \cdot s_{k_2}$. All the cuts are generated with respect to the same basic feasible solution of the linear program. After adding simultaneously all the cuts found to the linear program, it is reoptimised. After the generation of cuts, if the node cannot be fathomed by the standard rules of branch-and-cut, branching occurs. Each unfathomed node has two descendant nodes, which are obtained by adding the first and the second term, respectively, of a violated disjunction to the linear program. The disjunction $(k_1, k_2) \in D$ with largest $s_{k_1} \cdot s_{k_2}$ value is selected for branching. In another variant of the method, CGLP cuts were generated in the root node by solving $(CGLP)_{k_1,k_2}$ for the violated disjunctions $(k_1, k_2) \in D$.

In our experiments, in the root node of the search tree disjunctive cuts were generated in at most 3 rounds. The L&P cut generation method performed at most 50 pivots. The method always selected the first row with negative reduced cost r_{u_i} or r_{v_i}, and the column giving the most negative objective function value (largest violation in the row). In the search tree, disjunctive cuts were generated only in one round in the nodes of depth not greater than 4, and with one pivot only. The L&P cuts were generated in the subspace of variables containing all the basis variables and all the non-basic slack variables.

7.2 The test environment

The algorithm has been implemented in C++ programming language using the Xpress-MP mathematical programming package. All tests have been conducted on a PC with Pentium IV processor, 2 GHz clock speed, 512 MB RAM and Windows XP operating system.

7.3 Evaluation on LPCC instances

We have evaluated our method on test instances for linear programs with linear complementarity constraints (LPCCs). We have compared our results to those of Hu et al. [15] using the test instances [21]. We tested our method on datasets with 50, 300 complementarity constraints. In the datasets with 300 complementarity constraints, $B = 0$. We summarize our findings in Tables 1, 2, 3 and 4. In Tables 1 and 2, columns *lb* and *ub* provide the value of the LP relaxation and that of the optimal solution, respectively. The columns LPs and IPs are taken from [15] and indicate the number of linear and integer programs solved, respectively. Finally, the last four columns provide information about

Table 1: Comparion of Hu et al. [15] and branch-and-cut with L&P cuts on general LPCCs with $B \neq 0$, $n = m = 50$, $k = 55$.

			Hu et al.		L&P			
#	*lb*	*ub*	LPs	IPs	nodes	cuts	rounds	time
1	28.7739	29.0501	21	2	8	19	6	0.2
2	36.1885	37.5509	229	9	25	36	6	0.34
3	33.8630	37.0022	4842	696	107	87	10	0.75
4	33.7618	34.2228	102	7	29	20	6	0.25
5	21.4187	22.2835	209	24	35	39	8	0.32
6	29.8919	30.0829	108	13	31	26	6	0.34
7	37.6712	38.0405	92	7	17	15	7	0.13
8	20.8210	22.3969	187	21	35	35	6	0.27
9	39.0227	40.3380	321	14	31	58	6	0.43
10	40.0135	41.3957	190	19	43	44	8	0.42

our method: the number of nodes explored, the number of L&P cuts generated, the total number of cut-generation rounds, and the run-time of the computation in seconds. Notice that the linear program is reoptimized once for each node and after each round of the cut-generarion method. We can observe that our method solves considerably fewer linear programs than that of Hu et al. In Tables 3 and 4 we compare three variants of the method presented in this paper: pure branch-and-bound, branch-and-cut with L&P cuts, and branch-and-cut with CGLP cuts.

As can be seen, on these instances, pure branch-and-bound is the fastest method, albeit it does not rule out that a different branching stratefy might give even better results when combined with cut generation. On the other hand, even on small instances, L&P cut generation is favourable to CGLP cuts.

Unfortunately, cuts do not decrease drastically the number of nodes, and this shows that stronger cuts are needed. In the 0/1 case, the modularization technique of Balas [3] gives excellent results, see e.g. [5],[8].

Acknowledgements

This work has been supported by the Hungarian National Research Fund, grant no. K76810.

References

[1] Andersen, K, Cornuéjols, G., Li, Y., Split closure and intersection cuts, Mathematical Programming, Ser A, 102 (2005) 457-493.

Table 2: Comparion of Hu et al. [15] and branch-and-cut with L&P cuts on general LPCCs with $B = 0$, $n = m = 300$, $k = 300$.

			Hu et al.		L&P			
#	*lb*	*ub*	LPs	IPs	nodes	cuts	rounds	time
1	2469.4402	2478.2256	125	1	15	26	6	7.32
2	3213.7179	3270.1844	4071	62	31	59	6	5.65
3	3639.4496	3660.5412	350	2	8	24	6	3.63
4	3127.3706	3176.4108	1249	15	25	70	6	6.91
5	2958.9144	2959.9495	5	1	2	7	1	1.45
6	2630.3286	2672.5709	4511	70	29	47	6	6.66
7	2616.985	2617.2638	0	0	2	2	1	0.69
8	2766.9542	2771.2372	26	1	7	17	6	2.37
9	2842.4483	2847.6926	319	2	7	11	4	2.75
10	3207.6865	3230.9896	1569	16	10	26	6	3.54

Table 3: Comparion of pure B&B, branch-and-cut with L&P cuts, and branch-and-cut with CGLP cuts on general LPCCs with $B \neq 0$, $n = m = 50$, $k = 55$.

	B&B		L&P			CGLP		
#	nodes	time	nodes	cuts	time	nodes	cuts	time
1	8	0.05	8	19	0.2	8	13	0.34
2	33	0.08	25	36	0.34	25	26	0.58
3	111	0.2	107	87	0.75	91	63	1.15
4	29	0.05	29	20	0.25	35	27	0.58
5	31	0.06	35	39	0.32	19	26	0.62
6	27	0.06	31	26	0.34	31	23	0.5
7	17	0.06	17	15	0.13	13	8	0.27
8	45	0.07	35	35	0.27	27	19	0.45
9	35	0.09	31	58	0.43	31	50	0.82
10	41	0.09	43	44	0.42	41	23	0.55

Table 4: Comparion of pure B&B, branch-and-cut with L&P cuts, and branch-and-cut with CGLP cuts on general LPCCs with $B = 0$, $n = m = 300$, $k = 300$.

	B&B		L&P			CGLP		
#	nodes	time	nodes	cuts	time	nodes	cuts	time
1	15	0.81	15	26	7.32	11	24	33.89
2	39	1.49	31	59	5.65	31	58	64.98
3	15	0.87	8	24	3.63	8	22	31.48
4	25	1.3	25	70	6.91	23	77	66.66
5	7	0.43	2	7	1.45	1	4	7.91
6	23	0.99	29	47	6.66	15	51	51.52
7	3	0.41	2	2	0.69	2	3	7.74
8	13	0.59	7	17	2.37	3	14	20.96
9	4	0.52	7	11	2.75	7	12	23.17
10	13	0.78	10	26	3.54	11	24	27.29

[2] Audet, C., Savard, G., Zghall, W., New branch-and-cut algorithm for bilevel linear programming, Journal of Optimization Theory and Applications, 134 (2007) 353-370.

[3] Balas, E., Disjunctive Programming, Annals of Discrete Mathematics, 5 (1979) 3-51.

[4] Balas, E., Tama, J.M., Tind, J., Sequential convexification in reverse convex and disjunctive programming, Mathematical Programming, 44 (1989) 337-350.

[5] Balas, E., Ceria, S., Cornuéjols, G., A lift-and-project cutting plane algorithm for mixed 0-1 programs, Mathematical Programming, 58 (1993) 295-324.

[6] Balas, E., Ceria, S., Cornuéjols, G., Mixed 0-1 programming by lift-and-project in a branch-and-cut framework, Management Science, (1996).

[7] Balas, E., Disjunctive programming: Properties of the convex hull of feasible points, Discrete Applied Mathematics, 89 (1998) 3-44.

[8] Balas, E., Projection and lifting in combinatorial optimization, In: M. Jünger, D. Naddef (eds.), Computational Combinatorial Optimization, LNCS 2241 (2001) 26-56.

[9] Balas, E., Perregaard, M., Lift-and-project for mixed $0-1$ programming: recent progress. Discrete Applied Mathematics 123 (2002) 129-154.

[10] Balas, E., Perregaard, M., A precise correspondence between lift-and-project cuts, simple disjunctive cuts, and mixed integer Gomory cuts for 0-1 programming, Mathematical Programming, Ser B, 94 (2003) 221-245.

[11] Balas, E., Bonami, P., New variants of lift-and-project cut generation from the LP tableau: open source implementation and testing, IPCO 2007, LNCS 4513 (2007) 89-103.

[12] Ceria, S., Pataki, G., Solving integer and disjunctive programs by lift and project, IPCO VI, LNCS 1412 (1998) 271-283.

[13] Cornuejols, G., Li, Y., Elementary closures for integer programs, Operations Research Letters 28 (2001) 1-8.

[14] Dempe, S., Foundations of Bilevel Programming, Kluwer Academic, Dordrecht, 2002.

[15] Hu, J., Mitchell, J.E., Pang, J-S., Bennett, K.P., Kunapuli, G., On the Global Solution of Linear Programs with Linear Complementarity Constraints, SIAM J. Optimization, 19 (2008) 445-471.

[16] M. Jünger, G. Reinelt and S. Thienel, Practical problem solving with cutting plane algorithms in combinatorial optimization, DIMACS Ser. in Discr. Math. and Theor. Comput. Sci., 20 (1995), 111-152.

[17] Jeroslow, R.G., Cutting-planes for complementarity constraints, SIAM J. Control and Optimization, 16 (1978) 56-62.

[18] M. W. Padberg, T. J. Van Roy and L. A. Wolsey, Valid linear inequalities for fixed charge problems, Operations Research, 33/4 (1985) 842-861.

[19] Perregaard, M., Balas, E., Generating cuts from multiple-term disjunctions, In: K. Aaardal, B. Gerards (eds.), IPCO 2001, LNCS 2081 (2001) 348-360.

[20] Perregaard, M., Generating Disjunctive Cuts for Mixed Integer Programs, PhD Disseration, Carnegie Mellon University, Graduate School of Industrial Administration, 2003.

[21] http://www.rpi.edu/∼mitchj/generators/lpcc/.

Appendix

The proof of Lemma 1

Since s_{k_1} and s_{k_2} are basic, they belong to the set of basic variables I. Let $T = (A, -I_m)$ and T^I the set of columns corresponding to the basic variables. W.l.o.g. the last three columns of T^I correspond to x_k, s_{k_1} and s_{k_2}. Since I is a basis, matrix T^I is invertible, and let $(T^I)^{-1}$ denote its unique inverse. The two matrices have the following form

$$T^I := \left(\begin{array}{ccc|ccc} * & * & * & * & & \\ * & * & * & * & 0 & 0 \\ * & * & * & * & & \\ \hline & 0 & & 1 & -1 & 0 \\ & 0 & & -1 & 0 & -1 \end{array}\right) \qquad (T^I)^{-1} := \left(\begin{array}{ccc|cc} * & * & * & * & * \\ * & * & * & * & * \\ \hline & a & & t_1 & t_4 \\ & c & & t_2 & t_5 \\ & d & & t_3 & t_6 \end{array}\right)$$

Here, the row vectors a, c, d belong to $\mathbb{R}^{m-2}$, and the values of the elements denoted by "$*$" are unimportant for our derivation. Since $T^I \times (T^I)^{-1} = I_m$, we have

$$\left.\begin{aligned} a_j - c_j &= 0 \\ -a_j - d_j &= 0 \end{aligned}\right\} \quad \text{for all } j \in I \setminus \{k_1, k_2\}$$

It follows that $a = c = -d$. Let the matrix T^J consist of those columns of T indexed by the nonbasic variables J. Clearly, the last two rows of T^J contain only zero entries. Therefore,

$$\begin{aligned} \overline{a}_{kj} &= \left((T^I)^{-1}_k \times T^J)\right)_j = ((a, t_1, t_4) \cdot T^J)_j \\ \overline{a}_{k_1,j} &= \left((T^I)^{-1}_{k_1} \times T^J)\right)_j = ((c, t_2, t_5) \cdot T^J)_j = ((a, t_2, t_5) \cdot T^J)_j = \overline{a}_{kj} \\ \overline{a}_{k_2,j} &= \left((T^I)^{-1}_{k_2} \times T^J)\right)_j = ((d, t_3, t_6) \cdot T^J)_j = ((-a, t_3, t_6) \cdot T^J)_j = -\overline{a}_{kj} \end{aligned}$$

This proves the first part of the statement.

Concerning the second part, the scalar product of the last but one row of T^I and the last column of $(T^I)^{-1}$ gives the constraint

$$t_4 - t_5 = 0.$$

On the other hand, the scalar product of the last row of T^I and the last column of $(T^I)^{-1}$ yields

$$-t_4 - t_6 = 1.$$

The right hand side of (5) is $b' = (b'_{I \setminus \{k,k_1,k_2\}}, -1, 0, -1)$. Since $(T^I)^{-1} \cdot b$ gives the values of the basis variables, we have

$$\begin{aligned} &\overline{a}_{k0} = a \cdot \begin{pmatrix} b_{I \setminus \{k,k_1,k_2\}} \\ -1 \end{pmatrix} - t_4, \\ &\overline{a}_{k_1,0} = a \cdot \begin{pmatrix} b_{I \setminus \{k,k_1,k_2\}} \\ -1 \end{pmatrix} - t_5 = \overline{a}_{k0}, \text{ since } t_4 = t_5, \\ &\overline{a}_{k_2,0} = -a \cdot \begin{pmatrix} b_{I \setminus \{k,k_1,k_2\}} \\ -1 \end{pmatrix} - t_6 = 1 - a \cdot \begin{pmatrix} b_{I \setminus \{k,k_1,k_2\}} \\ -1 \end{pmatrix} + t_4 = 1 - \overline{a}_{k0}, \end{aligned}$$

since $-t_6 = 1 + t_4$. $\square$