Lift-and-Project Inequalities

Q. Louveaux[*]

Abstract

The lift-and-project technique is a systematic way to generate valid inequalities for a mixed binary program. The technique is interesting both on the theoretical and on the practical point of view. On the theoretical side it allows you to construct the inequality description of the convex hull of all mixed-$\{0,1\}$ solutions of a binary MIP in n repeated applications of the technique, where n is the number of binary variables. On the practical side, a variant of the method allows you to derive some cutting planes from the simplex tableau rather efficiently.

Lift-and-project inequalities were proposed in the early nineties as a way to strengthen the linear programming relaxation of a mixed-integer program. The initial idea was proposed by Lovász and Schrijver [1] when they obtain a strengthened formulation in an extended space by multiplying all constraints by all variables x_i and their complement and to project back to the initial space of variables. A similar but slightly different approach was then later proposed by Sherali and Adams [2]. In this survey, we follow the approach of Balas, Ceria and Cornuéjols [3] who showed that a simplified version of the Lovász-Schrijver reformulation keeps its main theoretical property. They also showed how the approach can successfully be incorporated in a branch-and-cut solver [4].

The general idea behind lift-and-project is the following: from an initial formulation, we can create a quadratic equivalent formulation by multiplying all inequalities by a binary variable x_j of the problem and its complement $(1-x_j)$. If we linearize this formulation by introducing a variable $y_i := x_i x_j$, and replacing $x_j^2 = x_j$ (since $x_j \in \{0,1\}$), we obtain an equivalent formulation in an extended space. This is the *lifting phase*. If we project this extended formulation onto the space of the initial variables, we obtain a strengthened formulation of the initial binary MIP. This is the *projecting phase*.

This article is further subdivided into two parts. Section 1 presents the lift-and-project technique and show that it encodes a simple convexification process. We then show that this convexification process can be used sequentially in order to generate the convex hull of all feasible points. Lift-and-project inequalities can be related to other classes of inequalities. We also make this link in Section 1. Section 2 focuses on different ways to generate lift-and-project inequalities

[*]Montefiore Institute, University of Liège, Grande Traverse, 10, 4000 Liège, Belgium, e-mail: `q.louveaux@ulg.ac.be`

that cut off a fractional point, in particular starting from the optimal simplex tableau of a linear relaxation. We also present some computational tests that have addressed the strength of the lift-and-project inequalities.

Two other references on the topic are the survey on cutting planes by Cornuéjols [5] and the survey on lift-and-project by Balas and Perregaard [6].

1 General theory

Before presenting the lift-and-project on its own, we first start with some basics about the projection of polyhedra, which is a building block of the lift-and-project technique.

Definition 1 *Let $P \subseteq \mathbb{R}^{p+q}$ be a polyhedron where an element of P is denoted by $(x, y) \in P$, $x \in \mathbb{R}^p, y \in \mathbb{R}^q$. The projection of P on its first p coordinates is defined as*

$$proj_x(P) = \{x \in \mathbb{R}^p \mid \exists y \in \mathbb{R}^q, \text{ with } (x, y) \in P\}.$$

The projection of a polyhedron can be computed using the following result.

Lemma 1 *Let $P := \{(x, y) \in \mathbb{R}^{p+q} \mid Ax + Gy \geq b\}$. Let $\{u^t\}_{t \in t}$ be the list of extreme rays of $\{u \in \mathbb{R}^q \mid u^T G = 0, u \geq 0\}$. Then $proj_x(P) = \{x \in \mathbb{R}^p \mid (u^t)^T Ax \geq (u^t)^T b\}$.*

Proof: See any textbook on linear and integer programming, as for example [7, 8, 9]. $\square$

We now turn to lift-and-project. We consider the mixed-binary set $K_I = \{x \in \{0, 1\}^n \times \mathbb{R}^p_+ \mid Ax \geq b\}$ with $A \in \mathbb{R}^{m \times (n+p)}$, $b \in \mathbb{R}^m$ and its linear relaxation $K = \{x \in \mathbb{R}^{n+p}_+ \mid Ax \geq b\}$. We suppose that the constraints $x_i \geq 0, x_i \leq 1, i = 1, \ldots, n$ are included in the description $Ax \geq b$. The lift-and-project technique is summarized in the following table.

The lift-and-project technique

Step 0	Select a binary variable x_j, $j = 1, \ldots, n$.
Step 1	Multiply all constraints by x_j and $(1 - x_j)$ giving the quadratic system $Q_j(K) = \{x \in \mathbb{R}^{n+p}_+ \mid x_j(Ax - b) \geq 0, (1 - x_j)(Ax - b) \geq 0\}$
Step 2	Linearize $Q_j(K)$ by introducing a variable $y_i := x_i x_j$, $i \neq j$ and replacing x_j^2 by x_j. We obtain the polyhedron $M_j(K)$.
Step 3	Project $M_j(K)$ onto the space of the x-variables. The resulting polyhedron is $P_j(K)$.

Observation 1 $P_j(K) \subseteq K$

Proof: If we sum up all inequalities defining $Q_j(K)$, we obtain $Ax \geq b$ and therefore summing up all inequalities defining $M_j(K)$, we also obtain $Ax \geq b$. It follows that any point $(x, y) \in M_j(K)$ must satisfy $Ax \geq b$. Hence $P_j(K) \subseteq K$.□

This shows that the lift-and-project technique produces a new valid formulation for the points in K_I. We can also observe that the only operation that takes care of the integrality of the variables is when we replace x_j^2 by x_j. This operation is only valid when $x_j \in \{0, 1\}$. The first theorem of this section is that the operation actually strengthens the formulation in the best possible way considering the fact that only the integrality of x_j is considered.

Theorem 1 $P_j(K) = \text{conv}\{(K \cap \{x_j = 0\}) \cup (K \cap \{x_j = 1\})\}$

Proof: The proof is taken from [3].
(i) $\text{conv}(K \cap \{x : x_j \in \{0, 1\}\}) \subseteq P_j(K)$. Let $\bar{x} \in K \cap \{x : x_j \in \{0, 1\}\}$. Define $\bar{y}_i = \bar{x}_i\bar{x}_j$ for $i \neq j$. Then $(\bar{x}, \bar{y}) \in M_j(K)$ and hence $\bar{x} \in P_j(K)$.

(ii) $P_j(K) \subseteq \text{conv}(K \cap \{x : x_j \in \{0, 1\}\})$. Assume first $K \cap \{x : x_j = 0\} = \emptyset$. Then $x_j \geq \epsilon$ is valid for K for some $\epsilon > 0$. This implies that $(1 - x_j)(x_j - \epsilon) \geq 0$ is satisfied by any $x \in Q_j(K)$. Replacing x_j^2 by x_j, it follows that $x_j \geq 1$ is valid for $Q_j(K) \cap \{x : x_j^2 = x_j\}$ from which it follows that it is valid for $P_j(K)$. Hence $P_j(K) \subseteq K \cap \{x : x_j = 1\}$. The case $K \cap \{x : x_j = 1\} = \emptyset$ is similar.

Assume now that $K \cap \{x : x_j = l\} \neq \emptyset, l = 0, 1$. Suppose $\alpha x \geq \beta$ is valid for $\text{conv}(K \cap \{x : x_j \in \{0, 1\}\})$. Since it is valid for $K \cap \{x : x_j = 0\}$, there exists $\lambda \geq 0$ such that $\alpha x + \lambda x_j \geq \beta$ is valid for K. Similarly there exists $\mu \geq 0$ such that $\alpha x + \mu(1 - x_j) \geq \beta$. Multiplying these two inequalities by $(1 - x_j)$ and x_j respectively, we obtain that $(1 - x_j)(\alpha x + \lambda x_j - \beta) \geq 0$ and $x_j(\alpha x + \mu(1 - x_j) - \beta) \geq 0$ are valid for $Q_j(K)$. Adding these two inequalities yields $\alpha x + (\lambda + \mu)(x_j - x_j^2) - \beta \geq 0$. Hence $\alpha x \geq \beta$ is valid for $Q_j(K) \cap \{x : x_j^2 = x_j\}$ and therefore it is valid for $P_j(K)$ too. □

The nice feature about the lift-and-project technique is that it can be applied sequentially in order to generate $\text{conv}(K_I)$ in a finite number of steps.

Theorem 2 $P_n(P_{n-1}(\cdots(P_1(K))\cdots) = \text{conv}(K_I)$.

Proof: The proof is taken from [5]. We proceed by induction. Let $S_t := \{x \in \{0, 1\}^t \times \mathbb{R}^{n-t+p} : Ax \geq b\}$. We want to show that $P_t(\cdots(P_1(K)\cdots) = \text{conv}(S_t)$. This is true for $t = 1$ as was shown in Theorem 1. Thus we consider $t \geq 2$ and suppose that it is true for $t - 1$. By the induction hypothesis, we have $P_t(\cdots(P_1(K)\cdots) = P_t(\text{conv}(S_{t-1}))$ and using again Theorem 1, we have

$$P_t(\cdots(P_1(K)\cdots) = \text{conv}((\text{conv}(S_{t-1}) \cap \{x_t = 0\}) \cup (\text{conv}(S_{t-1}) \cap \{x_t = 1\})). \tag{1}$$

For any set S that lies entirely on one side of a hyperplane H, we have that $\text{conv}(S) \cap H = \text{conv}(S \cap H)$ (see for example [8]). We can therefore rewrite (1) as

$$\begin{aligned} P_t(\cdots(P_1(K)\cdots) &= \mathrm{conv}(\mathrm{conv}(S_{t-1}\cap\{x_t=0\})\cup(\mathrm{conv}(S_{t-1}\cap\{x_t=1\})) \\ &= \mathrm{conv}((S_{t-1}\cap\{x_t=0\})\cup(S_{t-1}\cap\{x_t=1\})) \\ &= \mathrm{conv}(S_t). \end{aligned}$$

□

We see that if we apply the lift-and-project technique in sequence, we will eventually generate the integer hull of the initial set. On the other hand, if we apply the lift-and-project technique in "parallel" we construct the so-called *lift-and-project closure*.

Definition 2 *The lift-and-project closure is the set* $\cap_{j=1}^n P_j(K)$.

Example We now present a little example that illustrates the technique. Consider the simple two dimensional polyhedron

$$\begin{aligned} K = \{x\in\mathbb{R}^2\ |0\le x_1,\ x_1\le 1,\ \ 0\le x_2,\ x_2\le 1 \\ 5x_1-6x_2\ge -3,\ x_1+6x_2\ge 2\quad\} \end{aligned}$$

from which we are interested in the description of the integer hull (see Fig. 1).

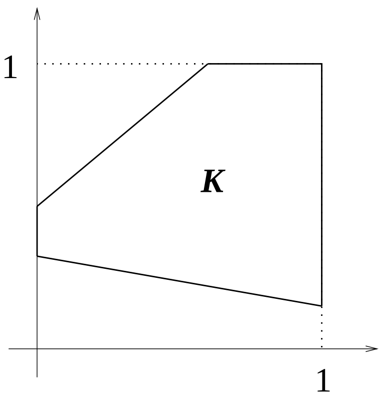

Figure 1: The initial polyhedron K

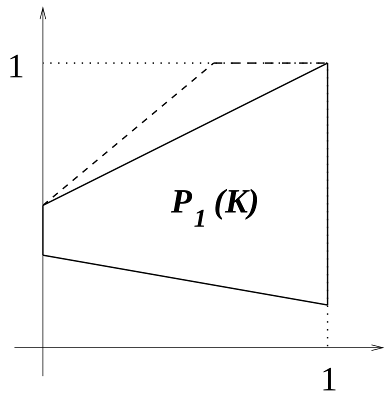

Figure 2: The polyhedron $P_1(K)$ after one iteration of the lift-and-project

Step 0 We select $j=1$.
Step 1 We multiply all inequalities by x_1 and $1-x_1$. We obtain the quadratic system

$$\begin{aligned} Q_1(K) = \{x\in\mathbb{R}^2\ |x_1^2\ge 0,\quad x_1^2\le x_1,\qquad & x_1-x_1^2\ge 0,\quad x_1-x_1^2\le 1-x_1, \\ x_1x_2\ge 0,\quad x_1x_2\le x_1,\ & x_2-x_1x_2\ge 0,\quad x_2-x_1x_2\le 1-x_1, \\ 5x_1^2-6x_1x_2\ge -3x_1,\quad & 5x_1-5x_1^2-6x_2+6x_1x_2\ge -3+3x_1, \\ x_1^2+6x_1x_2\ge 2x_1,\qquad & x_1-x_1^2+6x_2-6x_1x_2\ge 2-2x_1\quad\}. \end{aligned}$$

Step 2 We replace x_1^2 by x_1 and we linearize x_1x_2 by introducing the variable $y_2 := x_1x_2$. We obtain

$$\begin{aligned} L_1(K) = \{(x,y) \in \mathbb{R}^{2+1} \mid & x_1 \geq 0,\ y_2 \geq 0,\ x_1 - y_2 \geq 0, \\ & -x_1 \geq -1,\ x_2 - y_2 \geq 0,\ -x_1 - x_2 + y_2 \geq -1 \\ & 8x_1 - 6y_2 \geq 0, \quad -3x_1 - 6x_2 + 6y_2 \geq -3 \\ & -x_1 + 6y_2 \geq 0, \quad 2x_1 + 6x_2 - 6y_2 \geq 2 \qquad \}. \end{aligned}$$

Step 3 We now need to project out the y_2 variable. To do it, we may consider either a Fourier-Motzkin elimination or using Lemma 1. For the latter, we have to consider vectors $u \in \mathbb{R}^{10}_+$ that are extreme rays of

$$Q := \{u \in \mathbb{R}^{10} \mid u_2 - u_3 - u_5 + u_6 - 6u_7 + 6u_8 + 6u_9 - 6u_{10} = 0, u_i \geq 0\}.$$

The set Q has 18 extreme rays and considering some of them leads to redundant inequalities. If we consider $u_3^1 = 2, u_8^1 = \frac{1}{3}$, we obtain the inequality $x_1 - 2x_2 \geq -1$. Similarly if we consider $u_8^2 = 1, u_{10}^2 = 1$, we obtain the inequality $x_1 \leq 1$. We can also finally obtain the inequalities $x_1 \geq 0$ and $x_1+6x_2 \geq 2$. All the other inequalities are redundant. The resulting polyhedron $P_1(K)$ is represented in Fig. 2. If we compute $P_2(P_1(K))$, we would obtain $P_2(P_1(K)) = \{(1,1)\}$. $\square$

It is possible to make some connections between lift-and-project inequalities and some other well-known classes of inequalities. Observe indeed that Theorem 1 has several implications that we summarize below.

Observation 2 *The class of lift-and-project inequalities is a subset of*

- *disjunctive inequalities introduced by Balas [10] where the disjunction is* $(x_j \leq 0) \vee (x_j \geq 1)$,
- *Gomory mixed-integer cuts [11] or mixed-integer rounding [12],*
- *split cuts [13] where the corresponding split is* $\{0 \leq x_j \leq 1\}$,
- *intersection cuts [14].*

Lift-and-project inequalities are however only applicable in the framework of binary mixed-integer programming whereas the other classes are also valid for general mixed integer programs.

2 Generating lift-and-project cuts

In this section, we ask the question of optimizing a linear function over K_I. This is often done by considering the linear relaxation K and by finding cutting planes that cut off the optimal point of the linear relaxation. A natural question that we may ask is to find the best lift-and-project cut that cuts off an optimal vertex of the linear relaxation. This can be answered by solving a linear program as we explain below. Step 2 of the lift-and-project technique constructs the polyhedron

$$M_j(K) = \{x \in \mathbb{R}^{n+p} y \in \mathbb{R}^{n+p} \mid Ay - bx_j \geq 0, Ax + bx_j - Ay \geq b, y_j = x_j\}.$$

A simpler representation of M_j can be obtained by getting rid of the y_j variable. If we denote by $\tilde{A}$ the matrix obtained by removing from A its j^{th} column A_j, we can rewrite $M_j(K)$ as

$$M_j(K) = \{x \in \mathbb{R}^{n+p}, y \in \mathbb{R}^{n+p-1} \mid \tilde{A}y + (A_j - b)x_j \geq 0, Ax + (b - A_j)x_j - \tilde{A}y \geq b\}.$$

By renaming the matrices, we may write the set as

$$M_j(K) = \{x \in \mathbb{R}^{n+p}, y \in \mathbb{R}^{n+p-1} \mid \bar{C}x + \tilde{A}y \geq 0, \tilde{C}x - \tilde{A}y \geq b\}.$$

A valid lift-and-project inequality is a valid inequality for the set $\text{proj}_x M_j(K)$. It can be computed using Lemma 1 by considering the set $Q = \{(u, v) \in \mathbb{R}^m_+ \times \mathbb{R}^m_+ \mid u^T\tilde{A} - v^T\tilde{A} = 0, u, v \geq 0\}$. Therefore an inequality is determined as $(u\bar{C} + v\tilde{C})x \geq vb$ with $(u, v) \in Q$. If we want to cut off a specific point x^*, it can therefore be modeled as the linear program

$$\begin{aligned} \max \quad & vb - (u\bar{C} + v\tilde{C})x^* \\ \text{s.t.} \quad & u^T\tilde{A} - v\tilde{A} = 0 \\ & u, v \in \mathbb{R}^m_+. \end{aligned}$$

This is the so-called cut-generating LP (henceforth denoted (CGLP) as opposed to the initial program denoted by (LP)) which needs an additional normalization constraint in order to be bounded. The cut-generating LP (CGLP) has $2m$ variables and $n+p$ constraints and has thus twice the size of the linear relaxation (LP) of the initial problem. It is therefore fairly impractical to solve it at each step in order to obtain just one cut. Balas and Perregaard [15] have however shown that it is possible to mimick simplex pivots in (CGLP) by performing infeasible simplex pivots in (LP). Actually one simplex pivot in (LP) is not in one-to-one correspondence with a simplex pivot in (CGLP) but may correspond to several degenerate pivots in (CGLP). More specifically consider a row j in the optimal tableau (LP)

$$x_j = \bar{a}_{j0} - \sum_{l \in N} \bar{a}_{jl} x_l, \tag{2}$$

where N denotes the set of nonbasic variables. It can be shown that the Gomory mixed-integer cut generated from (2) is a lift-and-project cut from $P_j(K)$. There therefore exists a basis of (CGLP) that corresponds to it. In other words, the optimal tableau of (LP) can be put in correspondence with a feasible basis of (CGLP) that represents a Gomory mixed-integer cut generated from (2). Assume now that x_k enters the basis in row i ($i \neq j$) in (LP). In the new basis, row j will become row j to which a multiple of row i is added, yielding a row of the type

$$x_j = \tilde{a}_{j0} - \sum_{l \in N} \tilde{a}_{jl} x_l. \tag{3}$$

The Gomory mixed-integer cut generated from (3) is again a lift-and-project cut from $P_j(K)$ and corresponds to a basis of (CGLP). Performing a pivot in (LP) therefore implicitly allows us to move to another basis in (CGLP). The question is now how to choose the pivot a_{ik} in order to create a cut that is as strong as possible. There are two choices to be made : the row i and the column k. Improvement in the choice of row i is encoded through the reduced cost of u_i and v_i in (CGLP) but can be computed directly in (LP). The choice of k can also be made from the data in (LP) in order to obtain a strong cut.

The size of (CGLP) is roughly twice that of (LP) and it is no wonder that one basis of (CGLP) is not in exact one-to-one correspondence to a basis of (LP). It turns out that performing one pivot on (LP) often corresponds to several degenerate pivots on (CGLP). Hence it is much more efficient to generate lift-and-project cuts in (LP) as was also shown by computational evidence [15]. For more details on how to compute the reduced costs of u_i and v_i and how to select the variable x_k to pivot in, we refer to the paper by Balas and Perregaard [15].

We also propose a geometric interpretation of why different bases of (LP) correspond to different lift-and-project cuts. Recall that a lift-and-project inequality can equivalently be seen as a disjunctive or a split cut using the disjunction $(x_j \leq 0) \vee (x_j \geq 1)$. These cuts can in turn be interpreted as intersection cuts and are determined by the intersection of the rays of the bases with either $x_j = 0$ or $x_j = 1$. Fig. 3 shows an example of two cuts determined from two different bases. The infeasible basis actually generates the stronger cut.

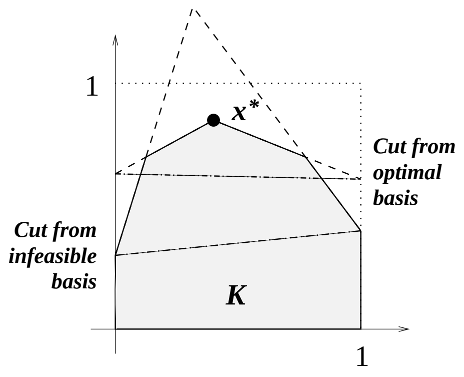

Figure 3: Geometric interpretation of lift-and-project cuts from feasible and infeasible bases

Balas and Jeroslow [16] have proposed a general method to strengthen valid inequalities of mixed-integer programs. This can also be applied in general to lift-and-project inequalities. Indeed when we generate lift-and-project cuts, we only consider the integrality of x_j. We may also want to consider the integrality of other variables $x_i, i \neq j$. This may allow us to decrease their coefficients in the cut. This strengthening has the very nice property that it is extremely easy to incorporate once the cut generating LP has been solved.

Theorem 3 *Consider the lift-and-project cut $\sum_{i=1}^{n+p} \alpha_i x_i \geq \beta$ obtained from the solution (u, v) of (CGLP) such that*

$$\alpha_k = \begin{cases} \max(u^T A_k, v^T A_k) & k \neq j \\ \max(u^T A_j - u_0, v^T A_j + v_0) & k = j, \end{cases}$$

where A_k denotes the k^{th} column of A and $\beta = \min(u^T b, v^T b + u_0)$. Define $m_k = \frac{v^T A_k - u^T A_k}{u_0 + v_0}$ and

$$\bar{\alpha}_k = \begin{cases} \max(u^T A_k + u_0 \lceil m_k \rceil, v^T A_k - v_0 \lfloor m_k \rfloor) & k = 1, \ldots, n, k \neq j \\ \max(u^T A_j - u_0, v^T A_j + v_0) & k = j, \\ \max(u^T A_k, v^T A_k) & k = n+1, \ldots, p \end{cases}$$

and $\bar{\beta} = \min(u^T b, v^T b + u_0)$. Then the inequality $\bar{\alpha} x \geq \bar{\beta}$ is valid for K_I and is called a strengthened lift-and-project cut.

Proof: See [16]. □

We close this survey article with some numerical tests that have been carried out in order to evaluate the strength of lift-and-project inequalities. Bonami and Minoux [17] experimented on 35 mixed binary problems from the MIPLIB 3 library [18]. They have generated a series of lift-and-project inequalities in order to evaluate the gap closed by this family of cuts. On the 35 MIPLIB instances, they have found that the lift-and-project closure closes 37% of the integrality gap on average. This result needs to be moderated by the very large variance of the gap closed. Indeed on 8 out of 35 instances, the gap closed is lower than 1%, whereas on 4 out of the 35 instances the gap closed is larger than 90%. By using strenghtened lift-and-project cuts, the average gap closed increases by 8% to reach 45% gap closed on average. But again on 7 instances, the gap closed is lower than 1% and on 5 instances, the gap closed is larger than 90%.

References

[1] L. Lovász, A. Schrijver, Cones of matrices and set-functions and 0-1 optimization, *SIAM Journal on Optimization* **28**, 1991, 166-190

[2] H. Sherali, W. Adams, A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems, *SIAM Journal on Discrete Mathematics*, **3**, 1990, 311-430

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

[4] E. Balas, S. Ceria, G. Cornuéjols, Mixed 0-1 programming by lift-and-project in a branch-and-cut framework, *Management Science*, **42**, 1996, 1229-1246

[5] G. Cornuéjols, Valid inequalities for mixed integer linear programs, *Mathematical Programming B*, **112**, 2008, 3-44

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

[7] A. Schrijver, Theory of Linear and Integer Programming, Wiley, New-York, 1986

[8] G. Nemhauser, L. Wolsey, Integer and Combinatorial Optimization, Wiley, New-York, 1988

[9] D. Bertsimas, R. Weismantel, Optimization over Integers, Dynamic Ideas, Belmont, 2005

[10] E. Balas, Disjunctive programming: properties of the convex hull of feasible points, GSIA Management Science Report MSRR, 348, 1974: published as invited paper in *Discrete Applied Mathematics*, **89**, 1998, 1-44

[11] R. Gomory, Outline of an algorithm for integer solutions to linear programs, *Bulletin of the American Mathematical Society*, **64**, 1958, 275-278

[12] G. Nemhauser, L. Wolsey, A recursive procedure to generate all cuts for 0-1 mixed integer programs, *Mathematical Programming*, **46**, 1990, 379-390

[13] W. Cook, R. Kannan, A. Schrijver, Chvátal closures for mixed integer programming problems, *Mathematical Programming*, **47**, 1990, 155-174

[14] E. Balas, Intersection cuts - A new type of cutting planes for integer programming, *Operations Research*, **19**, 1971, 19-39

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

[16] E. Balas, R. Jeroslow, Strengthening cuts for mixed-integer programs, *European Journal of Operational Research*, **4**, 1980, 224-234

[17] P. Bonami, M. Minoux, Using rank-1 lift-and-project closures to generate cuts for 0-1 MIPs, a computational investigation, *Discrete Optimization*, **2**, 2005, 288-307

[18] R. Bixby, S. Ceria, C. McZeal, M. Savelsbergh, An updated mixed integer programming library: MIPLIB 3.0, *Optima*, **58**, 1998, 1215