CANADA EXCELLENCE RESEARCH CHAIR
DATA SCIENCE FOR REAL-TIME DECISION-MAKING

SOLVING STANDARD QUADRATIC PROGRAMMING BY CUTTING PLANES

Pierre Bonami
Andrea Lodi
Jonas Schweiger
Andrea Tramontani

June 2016

DS4DM-2016-001

POLYTECHNIQUE MONTRÉAL
DÉPARTEMENT DE MATHÉMATIQUES ET GÉNIE INDUSTRIEL
Pavillon André-Aisenstadt
Succursale Centre-Ville
C.P. 6079
Montréal - Québec
H3C 3A7 - Canada
Téléphone: 514-340-5121 # 3314

SOLVING STANDARD QUADRATIC PROGRAMMING BY CUTTING PLANES

PIERRE BONAMI*, ANDREA LODI†, JONAS SCHWEIGER‡, AND ANDREA TRAMONTANI§

Abstract. Standard quadratic programs are non-convex quadratic programs with the only constraint that variables must belong to a simplex. By a famous result of Motzkin and Straus, those problems are connected to the clique number of a graph. In this paper, we study cutting plane techniques to obtain strong bounds for standard quadratic programs. Our cuts are derived in the context of a Spatial Branch & Bound where linearization variables are introduced to represent products. Their validity is based on the result of Motzkin and Straus in that it depends on the clique number of certain graphs.

We derive in particular cuts that correspond to an underlying complete bipartite graph structure. We study the relation between these cuts and the classical ones obtained by the first level of the reformulation-linearization technique. By studying this relation, we derive a new type of valid inequalities that generalize both types of cuts and are stronger.

We present extensive computational results using the different cutting planes we propose in the context of the Spatial Branch & Bound implemented by the commercial solver CPLEX. We show that our cuts allow to obtain a significantly better bound than reformulation-linearization cuts and reduce computing times for global optimality. Finally, we show how to generalize the cuts to non-convex quadratic knapsack problems, i.e., to attack problems in which the feasible region is not restricted to be a simplex.

Key words. Standard Quadratic Programming, Non-convex Programming, Global Optimization, Cutting Planes, Reformulation-Linearization Technique

AMS subject classifications. 90C20, 90C26

1. Introduction.

In this paper, we study the problem of optimizing a quadratic function over the *standard simplex*, namely

$$\text{(StQP)} \qquad \min \left\{ x^T Q x \;\middle|\; x \in \Delta \right\},$$

where the standard simplex is defined as

$$\Delta = \left\{ x \in \mathbb{R}^d \;\middle|\; \sum_{i=1}^{d} x_i = 1, x \geq 0 \right\},$$

and $Q \in \mathbb{R}^d \times \mathbb{R}^d$ is a symmetric matrix. We do not make any further assumption on Q and the optimization problem (StQP) is a non-convex optimization problem, being generally referred to as *Standard Quadratic Program* [4]. Variants or generalizations of StQP appear in many applications where the sum of fractions has to sum up to 1 or where exactly one of several (binary) options has to be chosen. Applications in finance and the Quadratic Assignment problem are just two examples. Problem StQP also has fundamental relations with copositive programming [11]. In particular, StQP has an exact reformulation as a copositive programming problem [3] and the solution of StQP can be used to test if a matrix is copositive [7].

Although StQP is a purely continuous optimization problem it has strong connections with combinatorial optimization and in particular with the maximum clique

*CPLEX Optimization, IBM Spain, Madrid, Spain (pierre.bonami@es.ibm.com)
†École Polytechnique de Montréal, Canada (andrea.lodi@polymtl.ca)
‡CPLEX Optimization, IBM Italy, Bologna, Italy (jonas.schweiger@gmx.de)
§CPLEX Optimization, IBM Italy, Bologna, Italy (andrea.tramontani@it.ibm.com)

problem by a remarkable result of Motzkin and Straus [21]. Below, we remind the definition of the maximum clique problem and state formally this result.

A clique in a simple, undirected graph $G = (V, E)$ is a subset of nodes where every node is connected to all other nodes. The size of the largest clique in G is called *clique number of* G and denoted by $\omega(G)$. The problem of computing the clique number is one of Karp's 21 NP-hard problems [17].

The Motzkin-Straus Theorem [21] connects the clique number of a graph with StQP.

THEOREM 1 (Motzkin-Straus [21]). *Let A be the adjacency matrix of a simple, undirected graph $G = (V, E)$ and $\omega(G)$ its clique number. Then, the following relation holds:*

$$\max \left\{ x^T A x \mid x \in \Delta \right\} = 1 - \frac{1}{\omega(G)}.$$

Note the identification of each variable x_i with node $i \in G$. This can most conveniently be seen by rewriting the objective function as summation over the edges in G

$$x^T A x = \sum_{(i,j) \in E} 2 x_i x_j.$$

The factor of 2 is due to the symmetry of the adjacency matrix. For notational convenience, in the remainder, we maintain the identification of the index set of x with the set $V = \{1, \ldots, d\}$ of nodes and all considered graphs G are meant to have this node set.

It follows directly from Theorem 1 that StQP is an NP-hard problem.

Several authors have studied StQP and proposed solution methods that are exploiting the relationship with the max-clique problem. Bomze [4] coined the name StQP and proposed a reformulation that ensures the equality constraint by an appropriate objective penalty. Bomze et al. [5] reviewed and compared several bounds on the problem. Finally, Scozzari and Tardella [23] proposed a combinatorial enumeration algorithm for the problem.

In this paper our goal is to exploit Theorem 1 to obtain strong convex relaxations of StQP for general Q. We place ourselves in the context of a solution algorithm for StQP by Spatial Branch & Bound (see, e.g., [2]). We employ a classical convex relaxation of the problem using McCormick estimators [19] and strengthen it by using cutting planes that are based on solving clique problems for certain graphs. We call these inequalities *Motzkin-Straus Clique inequalities* (MSC inequalities for short). A generalization of those inequalities for the special case of bipartite graphs is then proposed and we call these inequalities *generalized MSC bipartite inequalities* (GMSC bipartite inequalities for short).

The paper is organized as follows. In section 2, we review the basics of Spatial Branch & Bound and the so-called *Reformulation-Linearization Technique* (RLT, [25]) is reviewed to an extent that is needed for the remainder of the paper. In section 3, we present valid inequalities based on the theorem of Motzkin-Straus and show connections to RLT inequalities. For complete bipartite graphs we provide an alternative RLT-based proof. Extending this result, in section 4, we propose a new type of inequalities that generalize both the RLT methodology and some of our inequalities. In Section 5 we propose separation algorithms to find violated inequalities. Section 6 provides computational results that show the effectiveness of the described inequalities. In section 7, we show how the cuts can be generalized if x is not required to be in the standard simplex, but fulfills a more general inequality. Finally, section 8 concludes the paper.

2. Q-space relaxation and RLT inequalities. The first step to solve StQP by Spatial branch-and-bound approaches is to construct a convex relaxation of the problem that is then iteratively refined by branching. Here, we place ourselves in the context of a reformulation of the problem where all the non-convex terms of the objective function are replaced with linearization variables. We say that a term $Q_{ij}x_ix_j$ is convex if and only if $i = j$ and $Q_{ij} \geq 0$ or $i \neq j$ and $Q_{ij} = 0$. Accordingly, the objective matrix Q is decomposed into $Q = S + P$, where S contains all positive diagonal entries of Q and $P = Q - S$. Linearization variables Y_{ij} are introduced for all nonzero entries of P and $Y_{ij} = x_ix_j$ has to hold in every feasible solution. By abuse of notation, we interpret the linearization variables as a matrix Y for which the equation

$$Y = xx^T,$$

holds, with the understanding that for those components for which $P_{ij} = 0$, $Y_{ij} = x_ix_j$ does not influence the optimal value and is omitted. Then, the reformulated StQP reads as

$$\min \left\{ x^T Sx + \langle P, Y\rangle \;\middle|\; Y = xx^T, x \in \Delta \right\},$$

where the function $\langle P, Y\rangle = \sum_{i=1}^{d} \sum_{j=1}^{d} P_{ij}Y_{ij}$ is the trace of the matrix product (or matrix scalar product). This formulation has a convex quadratic objective function and all the non-convexities have been moved into the constraint $Y = xx^T$.

Once this reformulation is performed a convex relaxation can be formed by taking any convex relaxation of the feasible set

$$\Gamma = \left\{ (x, Y) \in \mathbb{R}^d \times (\mathbb{R}^d \times \mathbb{R}^d) \;\middle|\; Y = xx^T, x \in \Delta \right\}. \tag{1}$$

Based on such a relaxation, a Spatial Branch & Bound can then be performed by branching on the variables x and tightening the convex relaxation of Γ with the resulting local bounds at each node see, e.g., [2, 20, 26].

McCormick estimators. The most common way to form a convex relaxation of Γ is to relax each non-convex equality $Y_{ij} = x_ix_j$ separately using its convex hull given by the McCormick inequalities [19]

$$\overline{x_j}x_i + \overline{x_i}x_j - \overline{x_i}\,\overline{x_j} \leq Y_{ij}, \tag{2}$$

$$\underline{x_j}x_i + \underline{x_i}x_j - \underline{x_i}\,\underline{x_j} \leq Y_{ij}, \tag{3}$$

$$\overline{x_j}x_i + \underline{x_i}x_j - \underline{x_i}\,\overline{x_j} \geq Y_{ij}, \tag{4}$$

$$\underline{x_j}x_i + \overline{x_i}x_j - \overline{x_i}\,\underline{x_j} \geq Y_{ij}, \tag{5}$$

where $\overline{x_i}$ and $\underline{x_i}$ are valid upper and lower bounds on x_i, respectively.

In our initial relaxation, we use the lower bounds $\underline{x_i} = 0$ and upper bounds $\overline{x_i} = 1$ and obtain the convex Quadratic Programming (QP) relaxation of StQP

$$\min \left\{ x^T Sx + \langle P, Y\rangle \;\middle|\; \begin{array}{l} \max\{0, x_i + x_j - 1\} \leq Y_{ij} \leq \min\{x_i, x_j\}, \\ x \in \Delta \end{array} \right\}. \tag{MC-StQP}$$

We refer to this relaxation as the *Q-space relaxation*.

While $Y = xx^T$ has to hold for each feasible solution, the McCormick inequalities only give a coarse approximation of the set Γ. We therefore strive to find valid inequalities that tighten this set and the Q-space relaxation.

Reformulation-Linearization Technique [24, 25]. The RLT technique consists of two steps. In the first step, valid constraints are multiplied by other constraints or by variables yielding an equation or inequality with higher order terms. In the second step, these terms are reformulated using linearization variables to obtain a linear constraint. The result is a valid constraint on the linearization variables that is often a very strong cutting plane [20].

Note that the McCormick inequalities (2)–(5) can be derived by applying this technique to the bound constraints of two variables.

By repeatedly applying this procedure to all constraints in a model, a hierarchy of valid constraints using higher order terms can be established. We restrict this exposition to the first order that involves only bilinear terms and to the case where a linear constraint is multiplied by a variable because the constraint set considered here have only one constraint that is not a simple bound constraint.

Indeed, in the context of optimization over the standard simplex Δ, the only constraint that is not a bound constraint is $\sum_{i=1}^{d} x_i = 1$. In the first step of the RLT procedure, this equation is multiplied by one of the variables x_j, which yields the equation

$$\sum_{i=1}^{d} x_i x_j = x_j. \tag{6}$$

In the second step, the quadratic and bilinear terms are replaced with the linearization variables

$$\sum_{i=1}^{d} Y_{ij} = x_j. \tag{7}$$

A second stronger relaxation, denoted by RLT-stQP, is obtained by adding each equation (7) obtained by multiplying the standard simplex $\sum_{i=1}^{d} x_i = 1$ by each $x_j,\ j = 1, \dots, d$.

Projected RLT inequalities. The RLT constraints are known to be strong, but they might use linearization variables for zero entries in P, i.e., either zero entries or convex terms (those in S) of Q. Because we build our relaxation in the space of non-zero entries of P, we need to project out those variables for which no linearization variable exists. Precisely, let the set V_j collect all indices i, for which a linearization variable Y_{ij} exists, namely

$$V_j = \{\, i \in V \mid P_{ij} \neq 0 \}\,. \tag{8}$$

Terms $x_i x_j$ for which $i \notin V_j$ are replaced by linear over- and under- estimators, i. e., linear functions $o_{ij}(x_i, x_j)$ and $u_{ij}(x_i, x_j)$ such that

$$\begin{aligned} o_{ij}(x_i, x_j) &\geq x_i x_j, \\ u_{ij}(x_i, x_j) &\leq x_i x_j. \end{aligned}$$

In our implementation, we use

$$\begin{aligned} o_{ij}(x_i, x_j) &= \overline{x_i} x_j, \\ u_{ij}(x_i, x_j) &= \underline{x_i} x_j. \end{aligned}$$

For bilinear terms the McCormick over- and under-estimators would also be natural choices. In our experiments, the objective matrix is dense and thus only the diagonal entries of P might be zero.

Then, the projected RLT-inequalities rof each j are

$$\sum_{i \in V_j} Y_{ij} + \sum_{i \notin V_j} o_{ij}(x_i, x_j) \geq x_j, \tag{9}$$

$$\sum_{i \in V_j} Y_{ij} + \sum_{i \notin V_j} u_{ij}(x_i, x_j) \leq x_j. \tag{10}$$

As most over- and under-estimators become stronger as the bounds become tighter, (9) and (10) can be separated in the nodes of the branch-and-bound tree as locally valid cuts with estimators taking into account the local variable bounds, especially after branching.

The same approach can be used to derive a valid RLT inequality from a general equation $ax = b$. There, the sign of a_i has to be considered for the estimators of $a_i x_i x_j$ such that the RLT inequalities for each j become

$$\sum_{i \in V_j} a_i Y_{ij} + \sum_{\substack{i \notin V_j \\ a_i \geq 0}} a_i o_{ij}(x_i, x_j) + \sum_{\substack{i \notin V_j \\ a_i < 0}} a_i u_{ij}(x_i, x_j) \geq b x_j,$$
$$\sum_{i \in V_j} a_i Y_{ij} + \sum_{\substack{i \notin V_j \\ a_i \geq 0}} a_i u_{ij}(x_i, x_j) + \sum_{\substack{i \notin V_j \\ a_i < 0}} a_i o_{ij}(x_i, x_j) \leq b x_j.$$

Similarly, an inequality $ax \leq b$ can be the starting point, but in this case x_j needs to be non-negative or non-positive and the relation has to be adjusted accordingly.

3. Motzkin-Straus Clique inequalities. We now come back to Theorem 1 and its use. On the one side, Theorem 1 can be seen as a method to compute the clique number of a graph. On the other side—as soon as the clique number of the graph is known—a valid inequality for Γ can be derived.

COROLLARY 2. *For any simple, undirected graph G with adjacency matrix A and clique number $\omega(G)$, the following inequality is valid for $(x, Y) \in \Gamma$:*

$$\langle A, Y \rangle \leq 1 - \frac{1}{\omega(G)}$$

Proof. The inequality $x^T A x \leq 1 - \frac{1}{\omega(G)}$ for $x \in \Delta$ follows immediately from the Motzkin-Straus theorem. Reformulation using the definition of Γ yields the result. ☐

In the remainder, we call the inequalities derived from Corollary 2 *Motzkin-Straus Clique inequalities (MSC inequalities).*

For any instance of Γ one can derive a different MSC inequality from any graph G on d nodes. However, the following Theorem establishes that the inequalities stemming from certain graphs are dominated.

THEOREM 3. *Let $G = (V, E)$ be a subgraph of $\tilde{G} = (V, \tilde{E})$ with the same clique number $\omega(G) = \omega(\tilde{G})$. Then, every point that violates the MSC inequality corresponding to G also violates the MSC inequality corresponding to $\tilde{G}$.*

Proof. Let the point (x, Y) be violated by the MSC inequality corresponding to G. Let A and $\tilde{A}$ be the adjacency matrices of G and $\tilde{G}$, respectively. Since G is a subgraph of $\tilde{G}$, it holds[1] $A \leq \tilde{A}$. Then, with $Y \geq 0$ we have

$$\left\langle \tilde{A}, Y \right\rangle \geq \langle A, Y \rangle > 1 - \frac{1}{\omega(G)}$$

Therefore, the MSC inequality corresponding to $\tilde{G}$ is also violated. $\square$

We are therefore mostly interested in graphs that are "maximal" for a certain clique number in the sense that adding any edge increases their clique number.

If the original quadratic objective Q of StQP is the adjacency matrix of a graph, then the relaxation obtained by adding the corresponding MSC inequality to the Q-space relaxation of Γ has the same objective function of StQP. Because neither relaxations MC-StQP nor RLT-stQP solve general clique problems directly, the Motzkin-Straus Clique inequalities are not dominated by McCormick inequalities and RLT inequalities. This observation is indeed formalized in the following theorem, whose proof shows that a feasible point for the RTL-stQP relaxation can violate a Motzkin-Straus Clique inequality obtained from a graph G with clique number 2. Before proceeding to the statement of the theorem, we remind the reader that a complete bipartite graph with partition $(M, \bar{M})$ (with $M \subset V$) is a graph where every node in M is connected to all nodes in $\bar{M}$, but there are no edges between any pair of nodes in M or in $\bar{M}$. An obvious property of bipartite graphs is that their clique number is 2 and the MSC inequality corresponding to a bipartite graph is given by $\sum_{j \in M} \sum_{i \in \bar{M}} Y_{ij} \leq \frac{1}{4}$.

Theorem 4. *Let $G = (V, E)$ be a complete bipartite graph, the Motzkin-Straus Clique inequality obtained from G is not implied by RLT equations.*

Proof. Let $(M, \bar{M})$ be the partition of V induced by G and define a point $(\tilde{x}, \tilde{Y})$ as

$$\tilde{x}_i = \begin{cases} \frac{1}{2m} & \text{if } i \in M \\ \frac{1}{2(d-m)} & \text{otherwise,} \end{cases}$$
$$\tilde{Y}_{ij} = \begin{cases} \frac{1}{2m(d-m)} & \text{if } (i,j) \in (M \times \bar{M}) \cup (\bar{M} \times M) \\ 0 & \text{otherwise.} \end{cases}$$

The notation $(i, j) \in (M \times \bar{M}) \cup (\bar{M} \times M)$ means that exactly one of the two indices is in M and the other in $\bar{M}$.

It is easy to verify that $\tilde{x} \in \Delta$ and $(\tilde{x}, \tilde{Y})$ fulfills the McCormick inequalities for bounds $x \in [0, 1]^d$. Furthermore, $(\tilde{x}, \tilde{Y})$ fulfills the RLT equations (7) for every $j \in V$. Indeed, for $j \in M$

$$\sum_{i \in V} \tilde{Y}_{ij} = \sum_{i \in \bar{M}} \tilde{Y}_{ij} = \frac{d-m}{2m(d-m)} = \frac{1}{2m} = \tilde{x}_j,$$

and for $j \in \bar{M}$

$$\sum_{i \in V} \tilde{Y}_{ij} = \sum_{i \in M} \tilde{Y}_{ij} = \frac{m}{2m(d-m)} = \frac{1}{2(d-m)} = \tilde{x}_j.$$

[1] Unless otherwise stated, we understand comparisons between two matrices and between a matrix and a scalar componentwise.

However, $\tilde{Y}$ violates the Motzkin-Straus Clique inequality for the bipartite graph corresponding to the partition $(M, \bar{M})$

$$\sum_{j \in M} \sum_{i \in \bar{M}} \tilde{Y}_{ij} = \sum_{j \in M} \frac{d-m}{2m(d-m)} = \frac{m(d-m)}{2m(d-m)} = \frac{1}{2} > \frac{1}{4}. \qquad \square$$

Even though the Motzkin-Straus Clique inequalities are not dominated by the RLT equations, a close relation exists. In particular, if $G = (V, E)$ is a complete graph, it is easy to see that aggregating the RLT constraints leads to an inequality that dominates the corresponding Motzkin-Straus Clique inequality. Indeed, summing up the equations (7) for all $j \in V$ yields

$$\sum_{j \in V} \sum_{i \in V} Y_{ij} = \sum_{j \in V} x_j = 1.$$

The last equation holds because x is in the standard simplex. Moving the quadratic terms to the right-hand side and observing that $\min_{x \in \Delta} \sum_{i \in V} x_i^2 = d^{-1}$, the Motzkin-Straus Clique inequality for the complete graph with d vertices is

$$\sum_{j \in V} \sum_{\substack{i \in V \\ i \neq j}} Y_{ij} = 1 - \sum_{j \in V} x_j^2 \leq 1 - \frac{1}{d}.$$

A more involved aggregation can be used to show the validity of Motzkin-Straus Clique inequalities for complete bipartite graphs. Consider a set $M \subset V$. First, sum up equations (6) obtained by multiplying the standard simplex constraint with x_j for each $j \in M$, to obtain

$$\sum_{j \in M} \sum_{i \in V} x_i x_j = \sum_{j \in M} x_j.$$

Next, regroup all the terms that have both indices in M in the right-hand side and obtain

$$\begin{aligned} \sum_{j \in M} \sum_{i \in \bar{M}} x_i x_j &= \sum_{j \in M} x_j - \sum_{j \in M} \sum_{i \in M} x_i x_j \\ &= \sum_{j \in M} x_j - \left(\sum_{j \in M} x_j \right)^2. \end{aligned}$$

Note that, so far, the linearization variables Y were not used and this last step used basic algebra to factor the right-hand side (this last step would not be satisfied by Y if only looking at the RLT inequalities). In the next step, we linearize the products on the left-hand side by using Y, namely

$$\sum_{j \in M} \sum_{i \in \bar{M}} Y_{ij} = \sum_{j \in M} x_j - \left(\sum_{j \in M} x_j \right)^2. \tag{11}$$

Now, the right-hand side is the function $g(z) = z - z^2$ applied to $\sum_{j \in M} x_j$. Basic calculus tells us that $g(z)$ attains its maximum at $g(\frac{1}{2}) = \frac{1}{4}$ (see Figure 1 for a plot of $g(z)$ on the domain of interest $[0, 1]$). Therefore, we get that the right-hand side is smaller than or equal to $\frac{1}{4}$, which is exactly the Motzkin-Straus Clique inequality for the complete bipartite graph with partition $(M, \bar{M})$.

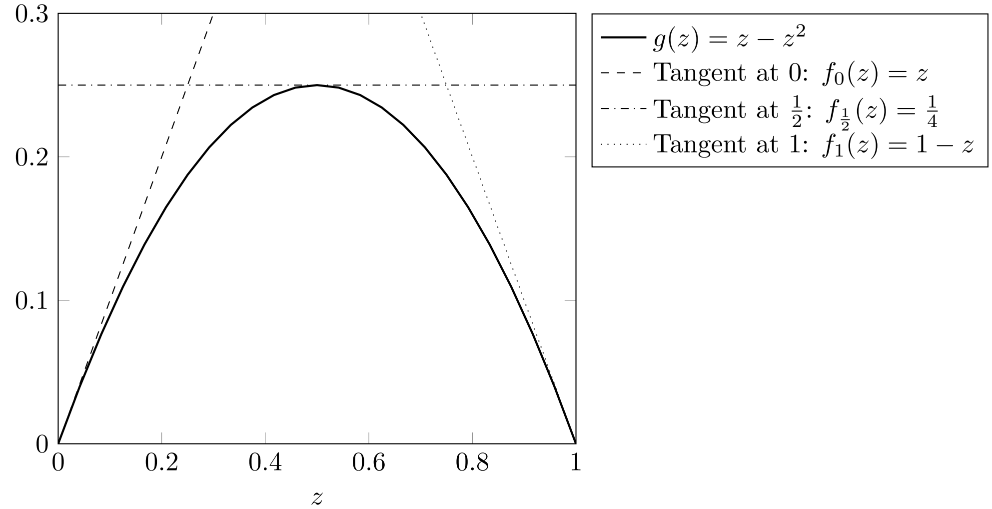


Fig. 1: The function $g(z)$ with tangents at 0, $\frac{1}{2}$, and 1.

4. Generalized MSC inequalities for bipartite graphs. In the previous section, we have introduced Motzkin-Straus Clique inequalities and showed how, if G is a complete bipartite graph, the corresponding Motzkin-Straus Clique inequality can also be deduced by performing a specific aggregation of RLT inequalities. In this section, we generalize this reasoning and deduce a new class of cutting planes that can be obtained from bipartite graphs.

Note that to go from (11) to the Motzkin-Straus Clique inequality we used a constant over-estimator of g but, due to the concavity of g, every tangent overestimates g so that for the tangent f_α taken at α, the following inequality holds:

$$\sum_{j \in M} \sum_{i \in \bar{M}} Y_{ij} \leq f_\alpha \left(\sum_{j \in M} x_j \right). \tag{12}$$

Because $f_\alpha(z)$ is an affine function, the right hand side of (12) is linear in x.

Of course, any missing linearization variable Y_{ij} can also be projected out in a similar way as in the RLT case. This way, the inequality using a tangent f_α becomes

$$\sum_{j \in M} \sum_{i \in \bar{M} \cap V_j} Y_{ij} + \sum_{j \in M} \sum_{i \in \bar{M} \cap \bar{V}_j} u_{ij}(x_i, x_j) \leq f_\alpha \left(\sum_{j \in M} x_j \right). \tag{13}$$

We denote constraint (13) as *generalized MSC bipartite inequality*, and that depends on the choice of the point α where the tangent is taken and of the subset M. It turns out that, regardless of the choice of the partition $(M, \overline{M})$, the tangent obtained from $\alpha = 0$ and $\alpha = 1$ are always implied by the RLT inequalities (10).

THEOREM 5. *If a point $(x, Y) \geq 0$ satisfies the RLT inequality (10) for all $j \in V$, then it satisfies the generalized MSC bipartite inequalities (13) for $\alpha = 0$ and $\alpha = 1$ and for all $M \subset V$.*

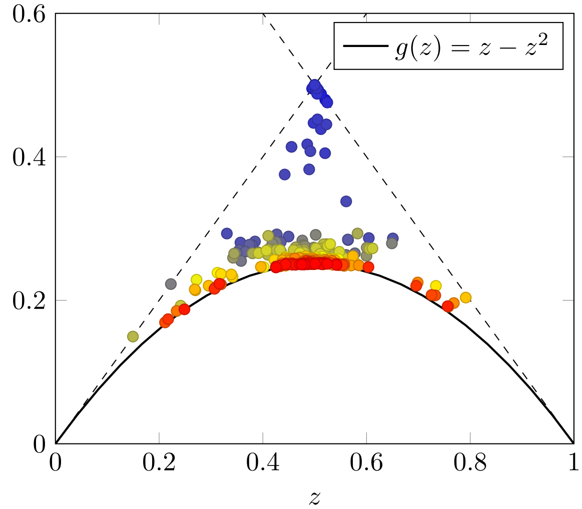


Fig. 2: Violated points separated by generalized MSC bipartite inequalities. Notice that each point corresponds to a different set M.

Proof. Take any $M \subset V$. We assume without loss of generality that the underestimators $u_{ij}(x_i, x_j)$ are chosen non-negative. Since $(x, Y) \geq 0$ and because of the validity of the RLT inequalities (10), the following chain of inequalities is valid for every $j \in V$:

$$\sum_{i \in \bar{M} \cap V_j} Y_{ij} + \sum_{i \in \bar{M} \cap \bar{V}_j} u_{ij}(x_i, x_j) \leq \sum_{i \in V_j} Y_{ij} + \sum_{i \in \bar{V}_j} u_{ij}(x_i, x_j) \leq x_j.$$

Summing over $j \in M$, we get

$$\sum_{j \in M} \sum_{i \in \bar{M} \cap V_j} Y_{ij} + \sum_{j \in M} \sum_{i \in \bar{M} \cap \bar{V}_j} u_{ij}(x_i, x_j) \leq \sum_{j \in M} x_j = f_0\left(\sum_{j \in M} x_j\right), \tag{14}$$

which is the generalized MSC bipartite inequality for M at $f_0(z) = z$.

For the generalized MSC bipartite inequality at $f_1(z) = 1 - z$, it suffices to exchange M and $\bar{M}$ in (14) and, due to $x \in \Delta$, it holds that

$$\sum_{j \in M} x_j = 1 - \sum_{j \in \bar{M}} x_j. \qquad \square$$

Figure 2 illustrates the generalized MSC bipartite inequalities than are separated in addition to RLT inequalities for one specific instance. More precisely, we separate generalized MSC bipartite inequalities as long as they are violated by using the separation algorithms that will be discussed in the next section. Each point in Figure 2 has the value $z = \sum_{i \in M} x_i^*$ on x-axis and the value $\sum_{i \in M} \sum_{j \in \bar{M}} Y_{ij}^*$ on the y-axis and corresponds to a different relaxation solution (x^*, Y^*) and a different set M. The color indicates the round in which the point was separated and warmer colors mean that it was found later in the cut loop. The plot clearly illustrates that the tangents at 0 and 1 are implied for sets M by the RLT inequalities, but many additional cutting planes can be separated.

5. Separation. To separate a violated Motzkin-Straus Clique inequality a graph has to be determined and its clique number has to be computed. On top of the fact that the latter computation is NP-hard, this boils down to a *bilevel separation problem* [18] with the determination of the graph in the first level and the computation of the clique number in the second one. This turns out to be computationally very hard because bilevel (integer-integer) optimization problems are, in general, extremely challenging both in theory and in practice (see, e.g., [8]).

We focus on finding graphs with a fixed clique number, especially bipartite graphs, and devise exact separation algorithms for these two classes. That corresponds to removing the second level above. This is by far the best way we have found to generate effective cutting planes. We have also tried a number of algorithms based on the idea of using the solution of the continuous relaxation as starting point to heuristically find graphs that yield violated inequalities. This approach corresponds to eliminating the first level. Although some neat bound improvements can be obtained in this way (e.g., roughly 1% over some of algorithms discussed in subsection 6.2), it turns out that computing the clique number of instances of relevant sizes is feasible but not computationally effective, leading to too high separation times. Detailed results on those attempts are reported in [22].

In the following, we will study mathematical programming formulations to separate MSC inequalities on specific classes of graphs striving for exact separation. This is computationally viable by restricting ourselves to graphs with known clique number so as to avoid solving bilevel programming problems (to separate a single cut). First, we will concentrate on general graphs with fixed clique number. Iterating over all possible clique sizes yields a separation algorithm for general graphs. Second, we focus on complete bipartite graphs, which clearly have clique number 2.

Graphs with fixed clique number. Consider a point (x^*, Y^*) and a fixed integer $k > 1$. The aim is finding a graph with clique size at most k whose corresponding Motzkin-Straus Clique inequality separates (x^*, Y^*) from Γ. Since $x^* \in \Delta$, without loss of generality (x^*, Y^*) can be assumed to be non-negative. Then, the following Mixed-Integer Linear Programming problem (MILP) serves the purpose:

$$\max \quad \langle A, Y^* \rangle - \left(1 - \frac{1}{k}\right) \tag{15}$$

$$\text{s.t.} \quad \sum_{\substack{i,j \in S \\ i<j}} A_{ij} \leq \frac{|S|\,(|S|-1)}{2} - 1 \qquad \text{for all } S \subseteq V, |S| = k+1 \tag{16}$$

$$A_{ij} = A_{ji} \qquad \text{for all } i,j \in V \tag{17}$$

$$A_{ij} = 0 \qquad \text{for all } i \notin V_j \tag{18}$$

$$A_{ij} \in \{0,1\} \qquad \text{for all } i,j \in V \tag{19}$$

The program maximizes the violation of the cut and (x^*, Y^*) can be separated if and only if the objective value is greater than 0. Since the clique size is fixed, only the graph (in form of its adjacency matrix A) has to be computed. Constraints (17), (18), and (19) ensure that A is indeed the adjacency matrix of a simple undirected graph. The inequalities (16) ensure that G contains no clique of size $k+1$. To this end, it requires that from every set of $S \subseteq V$ of cardinality $k+1$, at least one if the $\frac{|S|(|S|-1)}{2}$ possible edges is missing.

A posteriori, if the clique size of G is smaller than k, then all edges connecting a maximum clique with the rest of the nodes have zero weight. Adding enough of these nodes will yield a graph with clique size k and the same objective value.

The drawback of this formulation is its exponential size for fixed k, which makes it impractical for computational purposes.

Complete Bipartite Graphs. We now turn to the separation of Motzkin-Straus Clique inequality stemming from bipartite graphs with partition $(M, \bar{M})$. We will always assume that both sets in the partition are nonempty and restrict ourselves to complete bipartite graphs (bipartite graphs such that adding any edge forms a triangle), since these are maximal w.r.t. the clique number and thus yield the strongest inequalities. For our purposes, bipartite graphs have two advantages: First, the clique number is 2 and therefore the Motzkin-Straus Clique inequalities have the best right-hand side value. Second, they have a very clean structure. For a fixed subset $M \subsetneq V$ of nodes, the Motzkin-Straus Clique inequality corresponding to the complete bipartite graph with partition $(M, \bar{M})$ is

$$\sum_{i \in M} \sum_{j \in \bar{M}} 2Y_{ij} \leq 1 - \frac{1}{2}.$$

Separating a maximally violated Motzkin-Straus Clique inequality corresponding to some complete bipartite graph means finding a bipartite graph with maximum weight, where the weight for each edge (i, j) is given by Y^*_{ij}. This is equivalent to finding a maximum-weight cut and is thus NP-hard [17]. However, since both, the number of nodes and the cardinality of the support (i. e., the nonzero values) of Y^*, are typically relatively small, it is computationally feasible to separate by solving a simple binary QP. To this end, we introduce a binary variable z_i for each $i \in V$ and say that nodes whose variables take the same value are in the same set of the partition. The problem to be solved is

$$\begin{aligned} \max \quad & \sum_{i \in V} \sum_{j \in V} 2Y^*_{ij}\, z_i(1 - z_j) - \frac{1}{2} \\ \text{s.t.} \quad & z \in \{0, 1\}^{|V|}. \end{aligned} \tag{20}$$

We assume without loss of generality that Y^* is symmetric. The product $z_i(1 - z_j)$ ensures that Y^*_{ij} is counted if and only if $z_i = 1$ and $z_j = 0$, i. e., i and j are in different sets. The objective function therefore maximizes the violation of the cut. Every solution with positive objective function value corresponds to a violated cut with partition $(M, \bar{M})$, where $M = \{\, i \in V \mid z_i = 1\}$. If the optimal objective value is non-positive, no violated cut exists.

Generalized MSC bipartite inequalities. To separate a violated generalized MSC bipartite inequality a set M has to be found such that

$$\sum_{i \in M} \sum_{j \in \bar{M}} Y^*_{ij} > g\left(\sum_{i \in M} x^*_i\right).$$

The generalized MSC bipartite inequality for M and $\alpha = \sum_{i \in M} x^*_i$ will then separate this point. The separating binary QP for bipartite graphs can be generalized to separate violated generalized MSC bipartite inequalities. It maximizes the violation and adds the α as one of the decision variables. Namely,

$$\begin{aligned} \max \quad & \sum_{i\in V}\sum_{j\in V} Y^*_{ij}\, z_i(1-z_j) - (\alpha - \alpha^2) \\ \text{s.t.} \quad & \alpha = \sum_{i\in V} x^*_i z_i \\ & z \in \{0,1\}^{|V|}, \alpha \geq 0. \end{aligned} \tag{21}$$

As for bipartite graphs, nodes are partitioned according to the value of their associated variables.

6. Computational Experiments. In this section, we present the results of a large set of computational experiments. They were carried out on a cluster of Intel Xeon 5160 quadcore CPUs running at 3.00 GHz with 8 GB RAM and using RHEL5 as operating system. To avoid random noise by cache misses and alike only one process was executed on each node at a time.

The implementation is based on a slightly modified version of the IBM CPLEX Optimizer 12.6.3 [16] (CPLEX for short) where the C-API has been extended to provide callbacks the access to the linearization variables Y. The cuts are separated from the user cut callback, only at the root node, and are added with the *purgeable flag* set to `CPX_USECUT_PURGE`, in order to allow CPLEX to purge the cuts that are deemed to be ineffective according to its internal strategies.

Our computational investigation focuses on the application of the proposed cutting planes in a Spatial Branch & Bound algorithm and studies their impact on the root node and on the overall solution time. Therefore, we omit a direct comparison with alternative formulations or solution approaches as presented for example in [23]. Nevertheless, in subsection 6.4 we report computational results on the small set of publicly-available instances from [23], while in the next section we describe the large amount of randomly-generated instances we extensively based our computation on.

6.1. Instances. As anticipated, we considered a large set of randomly-generated instances, and, in particular, we considered two sizes, $d = 30$ and $d = 50$, so as only the objective matrix has to be sampled. The instances are available on http://or.dei.unibo.it/library/msc

The sign of the objective coefficients plays a major role in these instances. Assuming all terms are linearized (i. e., all diagonal entries are negative and all off-diagonal entries are not zero), the objective function only acts on the Y variables. When optimizing the value of Y over Γ, Y_{ij} with $i \neq j$ is restricted by the McCormick inequalities and whether it will take the upper or the lower bound is defined by the sign of Q_{ij}, namely

$$Y_{ij} = \begin{cases} \max\{0, x_i + x_j - 1\} & \text{if } Q_{ij} > 0, \\ \min\{x_i, x_j\} & \text{if } Q_{ij} < 0. \end{cases}$$

We therefore strive to generate instances with different fractions of positive and negative entries in Q. Since the inequalities presented in this paper can only cut points where at least some entries Y_{ij} exceed $x_i x_j$, the biggest impact is expected for instances with a lot of negative entries in Q.

We used triangular distributions, which are characterized by 3 parameters $a < c < b$. Namely, a and b are the minimum and the maximum of the value range. The mode c describes the peak of the piecewise linear density function. Of course, the sign of the coefficients is of great impact in general; on the diagonal they even decide if the respective terms are convex. We therefore use the triples $(-10, -5, 0)$ and $(0, 5, 10)$ to get instances with only negative and only positive coefficients. For instances with mixed signs, we used the triples $(-10, -3, 10)$, $(-10, 0, 10)$, and $(-10, 3, 10)$, where the second is a symmetric distribution and the other two are more likely to have positive or negative coefficients, respectively. The diagonal entries are divided by 2. In addition, 2 variants of each instance are generated by taking the positive and negative absolute values of the diagonal elements. For $(-10, -5, 0)$ only the positive and for $(0, 5, 10)$ only the negative variants are generated since the respective opposite would yield the same instance again. Furthermore, the instances with positive off-diagonal entries (distribution $(0, 5, 10)$) in the variant with negative diagonal entries are trivially solved by all approaches. Indeed, since the objective is to minimize, setting the variable x_i with lowest diagonal entry Q_{ii} in the objective to 1 gives the optimal solution. For this reason, those instances are excluded.

The 3 distributions in 3 variants, 1 distributions in 2 variants for the diagonal, and 1 distribution in 1 variant for the diagonal give 12 different instance types. For each instance type and for each size $d \in \{30, 50\}$ we generated 10 instances, yielding 120 instances with $d = 30$ and 120 instances with $d = 50$ in total. All instances are available on http://or.dei.unibo.it/library/msc. In all computational experiments we enforced a time limit of 2 hours for instances of size $d = 30$ and 6 hours for those of size $d = 50$.

Since we can only separate MSC inequalities and generalized MSC bipartite inequalities if the linearization variable Y_{ij} exceeds the respective product, i. e., $Y_{ij} > x_i x_j$, for some (i, j), one could assume that the instances $(0, 5, 10)$ will not be affected by Motzkin-Straus Clique inequalities given that the objective function drives the linearization variables towards 0. This is only true for instances with positive diagonal elements in Q. For these instances, the quadratic terms $Q_{ii}x_i^2$ are convex and thus not linearized, and the resulting projected RLT inequalities are redundant. For the variation with negative diagonal elements in Q, the quadratic terms are linearized and the RLT equations $\sum_i Y_{ij} = x_j$ for all $j \in V$ force some Y_{ij} to be positive. In all these instances it is then possible to separate MSC inequalities and generalized MSC bipartite inequalities.

6.2. Optimizing over the bipartite closures. As a first set of experiments, we want to evaluate the impact of Motzkin-Straus Clique inequalities corresponding to bipartite graphs and generalized MSC bipartite inequalities at the root node of the branch-and-cut tree. Since RLT equations and inequalities can be easily separated by enumeration and are expected to be effective, we separate our inequalities only if no violated RLT inequalities can be found.

For MSC and GMSC bipartite inequalities we have exact separation algorithms and thus we can optimize over the associated closures. The closure of a class of inequalities is the relaxation obtained by adding all possible inequalities of this class. Although the closure itself may be intractable to compute, one can optimize a linear function over it by separation. The comparison of the values gives an indication of the strength of the class of inequalities.

MSC and GMSC bipartite inequalities are separated by solving the associated mathematical models (20) and (21) in a classical cutting-plane scheme, by using CPLEX as a black box. To limit the tailing-off effect that often arises in cutting plane algorithms, we try to separate up to 5 cuts per round, i.e., at every separation round we collect the first 5 incumbent solutions returned by CPLEX that correspond to violated cuts. Specifically, we consider the following four settings:

`CPLEX` CPLEX with empty cut callback;

`RLT` Violated RLT equations and inequalities are added;

`Bipartite` At each call of the cut callback, first violated RLT equations and inequalities and then violated MSC bipartite inequalities are added. If no inequality can be separated anymore, the final dual bound gives the value of the closure over these two types of cuts;

`GMSC` Same as `Bipartite`, but GMSC bipartite inequalities are separated instead of MSC bipartite inequalities.

Each of these configurations improves the closure of the previous ones since `Bipartite` and `GMSC` also separate RLT equations and inequalities and since Motzkin-Straus Clique inequalities for bipartite graphs are generalized MSC bipartite inequality at $\alpha = 0.5$.

Tables 1 and 2 report aggregated results at the root node for these configurations on the instances of size $d = 30$ and $d = 50$, respectively. Both tables have the following structure: First, we report the average root gap to measure the strength of the separated cuts. For all considered approaches we give the %gap left at the root, computed as $(UB - LB_{root})/|UB|$, where LB_{root} is the dual bound at the root node and UB is the optimal solution value or the value of the best solution found by any of the approaches reported in subsection 6.3. Then we report the gap closed with respect to CPLEX root (resp. CPLEX with RLT inequalities), computed as $(LB_{root} - LB_{base})/(UB - LB_{base})$, where LB_{base} is the dual bound obtained by CPLEX root (resp. CPLEX with RLT). Then, the number of instances solved to proven optimality is given, along with the number of time limits hit. Next, we give the average and maximum separation time, first considering all instances and then disregarding the instances where any of the compared approaches hit the time limit. In addition, we report the average and maximum number of separated cuts, along with the number of cuts applied to the root LP at the end of root node, as reported by CPLEX. Finally, the average and maximum number of separation rounds is given, to specify how many times the callback was called (note that in the last round no cut was separated otherwise the callback would have been called again). In both tables we do not report the column for `CPLEX` because no cuts are generated and it would only have zeros.

The results reported in the tables clearly show that RLT inequalities are fundamental for StQP. Indeed, by themselves RLT inequalities already close about 85% of the root gap obtained by default CPLEX. On the other hand, MSC and GMSC bipartite inequalities are very effective to improve on the dual bound on top of RLT inequalities, and the GMSC bipartite closure appears definitely stronger than the MSC bipartite closure. `Bipartite` and `GMSC` greatly improve over `RLT` reducing the arithmetic mean of the root gap and `GMSC` gives the best dual bounds by a large amount. Concerning the separation time, `Bipartite` appears on average to be very fast, while `GMSC` is instead too time consuming. With `GMSC`, 3 instances of size $d = 30$ and 10 of size $d = 50$ do not finish the root node within the time limit. However, it is remarkable that `GMSC` is able to solve 3 instances of size $d = 30$ without resorting to branching.

To investigate the main differences between `Bipartite` and `GMSC` we analyzed closely the evolution of the root node for some specific instances. All plots reported in the following are given on one instance of size $d = 30$ with positive diagonal entries and distribution $(-10, 0, -5)$ (i.e., instance *triangular_30_-10_0_-5__04_posDiag*). The instance has been selected as the one on which the separation time of both `Bipartite`

	RLT	Bipartite	GMSC	GraphPool	Hybrid
Average root gap [%]					
Gap left	67.11	21.55	11.04	13.73	11.33
Closed wrt CPLEX root	84.95	90.02	91.22	90.94	91.20
Closed wrt RLT	–	68.81	86.19	81.93	85.69
Solved/Timeout at the root					
Solved	0	0	3	0	3
Timeout	0	0	3	0	0
Separation time in seconds					
Mean	0.00	1.24	425.39	7.23	17.07
Max	0.00	14.50	7139.10	118.30	142.00
Separation time in seconds (exclude time limit)					
Mean	0.00	1.02	255.04	5.15	15.06
Max	0.00	14.60	5651.30	98.30	147.20
Number of cuts					
Separated Mean	26.98	97.62	748.83	2456.58	433.13
Separated Max	30.00	382.00	4651.00	16491.00	782.00
Applied Mean	26.98	53.72	84.67	78.16	77.34
Applied Max	30.00	137.00	228.00	200.00	168.00
Number of separation rounds					
Mean	1.92	47.29	225.28	96.39	216.73
Max	2.00	176.00	1032.00	515.00	376.00

Table 1: Comparing the closures on StQPs of size 30.

and GMSC exceeds the respective arithmetic mean by the smallest amount, but the plots would look similar for other instances.

Figure 3 shows the evolution of the dual bound from round to round. Even if GMSC converges towards a stronger dual bound, Bipartite is superior in the first rounds and shows a very limited tailing off effect. On the contrary, GMSC stalls after about 200 rounds and after that each round of cuts increases the bound only by a very small amount.

Figure 4 plots the time used in each round of separation for Bipartite and GMSC on the same instance. For Bipartite the separation times remain almost constant at a very low value. For GMSC, in contrast, separation times are modest for the first rounds but start to increase soon, with outliers taking up to more than 20 seconds. Such a difference in the separation time between Bipartite and GMSC can be easily explained: the former separation problem is a binary QP that can be linearized and solved by MILP techniques, while the latter has a non-convex quadratic continuous variable (namely α) and requires Spatial Branch & Bound to be solved.

Finally, we analyze the diversity of the graphs that are generated by Bipartite and GMSC. To this end, we compare each graph that is returned by the separation problems to all graphs that have been separated previously. The difference between two bipartite graphs is defined in terms of the partitions: Let $\mathcal{M} = (M, \bar{M})$ and $\mathcal{N} = (N, \bar{N})$ be two partitions of the same set. Then, define the distance $d(\mathcal{M}, \mathcal{N})$

	RLT	Bipartite	GMSC	GraphPool	Hybrid
Average root gap [%]					
Gap left	61.14	20.40	10.37	13.66	11.20
Closed wrt CPLEX root	87.94	90.73	91.46	91.23	91.42
Closed wrt RLT	–	68.45	86.79	81.10	85.24
Solved/Timeout at the root					
Solved	0	0	0	0	0
Timeout	0	0	10	0	0
Separation time in seconds					
Mean	0.00	18.07	2183.95	203.22	92.98
Max	0.00	533.10	21488.30	4732.70	2097.00
Separation time in seconds (exclude time limit)					
Mean	0.00	2.23	439.81	20.02	28.17
Max	0.00	7.00	8266.00	230.20	121.40
Number of cuts					
Separated Mean	45.63	240.19	1771.27	12281.52	676.42
Separated Max	50.00	2068.00	9230.00	76722.00	2634.00
Applied Mean	45.63	97.75	159.83	150.72	136.15
Applied Max	50.00	326.00	354.00	447.00	349.00
Number of separation rounds					
Mean	1.96	102.91	477.81	285.40	285.59
Max	3.00	583.00	2175.00	1210.00	783.00

Table 2: Comparing the closures on StQPs of size 50.

between the partitions by

$$d(\mathcal{M},\mathcal{N}) = \min(|M \triangle N|, |M \triangle \bar{N}|),$$

where $\triangle$ is the symmetric difference. Note that $M \triangle N = \bar{M} \triangle \bar{N}$, so the above is well-defined.

Figure 5 plots the minimum distance of every graph to all previous graphs for `Bipartite` (5a) and `GMSC` (5b). Specifically, for each round, the picture reports the minimum distance of every graph generated at the given round. Since `Bipartite` requires only 175 rounds to converge against the 717 required by `GMSC`, the plot for `GMSC` is restricted to the first 175 rounds. The picture clearly shows that `Bipartite` tends to separate cuts associated with bipartite graphs that are more diverse with respect to `GMSC`. While for `Bipartite` the vast majority of the graphs has a distance between 5 and 10 to the previously separated graphs, we see a lot of graphs that are very similar, e. g., distance smaller or equal to 2, to one of the previous graphs for `GMSC`. Frequently we even separate from the same graph multiple times with different values of α. This is problematic since the resulting cuts in this case are very similar. Indeed, the more diversity in the observed graphs the more diverse cuts are.

In order to overcome the main drawbacks of `GMSC` discussed above while trying to approximate the GMSC bipartite closure as much as possible, we have tried a number of heuristics to combine the separation of MSC and GMSC bipartite inequalities.

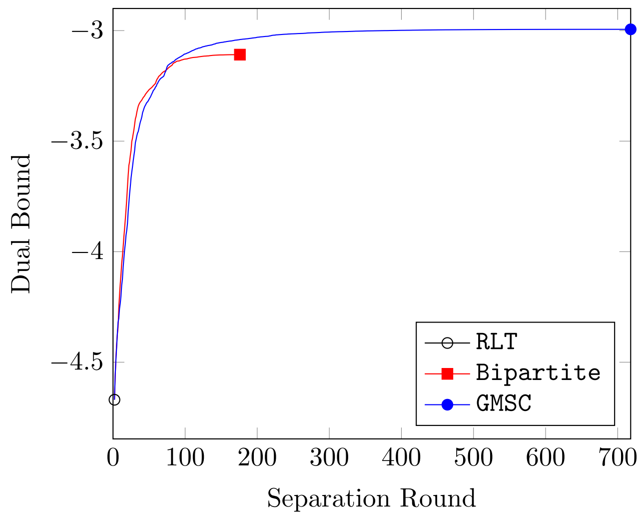


Fig. 3: Evolution of the dual bound for instance triangular_30_-10_0_-5__04_posDiag.

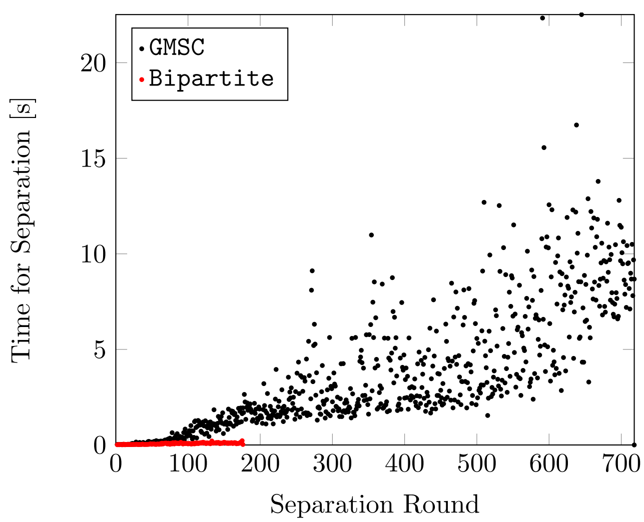
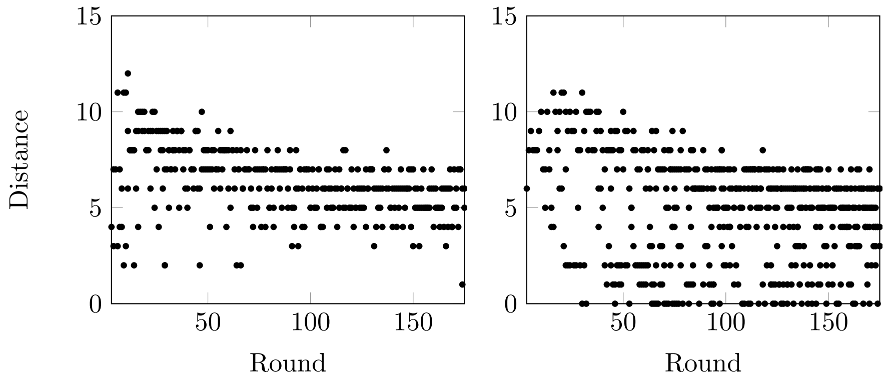


Fig. 4: Separation time per round for instance triangular_30_-10_0_-5__04_posDiag.

Detailed results on these heuristic versions are reported in [22]. In this paper, we focus on two main ideas denoted as `GraphPool` and `Hybrid`.

In `GraphPool`, we adopt a heuristic approach to find violated GMSC bipartite inequalities without solving the corresponding separation problem (21). Namely, while separating MSC bipartite inequalities, we store the new graphs (i. e., the partitions) that get separated in a *graph pool*. After adding the MSC bipartite inequalities, we compute the GMSC bipartite inequalities from all graphs in the pool with respect to the current relaxation solution (x^*, Y^*) and add the violated ones. This approach is expected to generate a lot of very similar cuts at every round, and thus we rely on



(a) `Bipartite` (b) `GMSC`

Fig. 5: Distance to know graphs for instance triangular_30_-10_0_-5__04_posDiag.

CPLEX cut purging to discard the ones that are not useful. The repeated separation of GMSC bipartite inequalities from the same graphs can be seen as a way to "update" the existing cuts based on the current relaxation solution to cut-off.

In `Hybrid`, we combine exact separation of MSC and GMSC bipartite inequalities as follows. First, we optimize over the MSC bipartite closure (i.e., perform `Bipartite`) and then we separate up to 200 rounds of GMSC bipartite inequalities (i.e., perform up to 200 rounds of `GMSC`). Further, at each separation round of `GMSC`, we enforce a deterministic work limit as follows. Let τ_0 be the deterministic time[2] needed to find the first violated cut and $\Phi > 0$ a scaling parameter. The additional deterministic time to find the ith violated cut after finding $i-1$ of them is then limited by $\tilde{\tau}_i = \tau_0 \Phi^i$. In our computations, we chose $\Phi = 0.9$, so we allow less work on every iteration.

The results obtained with `GraphPool` and `Hybrid` are reported in the last columns of Tables 1 and 2 and show that both approaches are quite effective in achieving the simultaneous goal of closing almost as much gap as generalized MSC bipartite inequalities at a quite reasonable computational price.

6.3. Branch-and-cut results. Branch-and-cut results obtained with all the approaches discussed in subsection 6.2 are given in Table 3 for the case $d = 30$ and in Table 4 for the case $d = 50$. The tables have the same structure and report aggregated results on all the instances that can be solved to optimality by at least one of the considered approaches. For the case $d = 30$, only 3 instances cannot be solved within the time limit of 2 hours, while 10 instances with $d = 50$ are not solved by any of the methods within the time limit of 6 hours. Interestingly, all the unsolved instances are generated with positive diagonal entries and distribution $(-10, -5, 0)$. This is somewhat not surprising since for those instances the Q-space relaxation (MC-StQP) is expected to be very weak. Indeed, the negative objective coefficients drive the relaxation variables Y_{ij} towards $\min(x_i, x_j)$, which is typically much further away from the correct values of $x_i x_j$ than the opposite bound that is 0. For example, taking $x_i = x_j = \frac{1}{n}$, the correct value would be $x_i x_j = \frac{1}{n^2}$, but the linearization

[2] CPLEX uses a deterministic measure of the work it performes, called *ticks*, and allows to set proper deterministic time limits accordingly. See [1, 15] for details.

variables takes the value $Y_{ij} = \frac{1}{n}$. Furthermore, on these instances the diagonal terms are not linearized and thus only the weaker projected RLT inequalities can be separated instead of RLT equations.

Each of the tables gives separate results on all solved instances and on solved "hard" instances, where an instance of size $d = 30$ (resp. $d = 50$) is considered to be hard if it takes at least 30 seconds (resp. 300 seconds) to be solved with all compared approaches. For each of the tested methods and for each class of problems, the tables report the number of solved instances, the average computing time in seconds, the shifted geometric mean of the computing times (with a shift of 10 seconds), the average number of branch-and-bound nodes, the shifted geometric mean of the number of nodes (with a shift of 100 nodes) and the average percentage gap left at the root node. Time limits are accounted in the computations on the running time, and out-of-memory errors (that only happen for `CPLEX`) are accounted as time limits. Computing average and geometric mean of the number of nodes is problematic for instances that are not solved to optimality. To make a fair comparison, the calculations for number of nodes only consider those instances that all solvers but `CPLEX` can solve within the time limit. For `CPLEX`, the number of nodes processed until running out of memory or time is used and thus the reported numbers of nodes for `CPLEX` (which is already an order of magnitude higher as for `RLT`) are underestimated.

	Solved	Time [s]		Nodes		Root Gap
		Avg.	S. Geom.	Avg.	S. Geom.	Avg.
All 117 instances solved by at least one						
`CPLEX`	88	2394.6	503.4	238799.2	64871.3	982.6 %
`RLT`	110	471.9	23.2	17581.6	1975.2	67.3 %
`Bipartite`	117	245.6	17.3	7951.0	1005.4	21.8 %
`GMSC`	115	325.4	31.3	1420.0	468.8	11.2 %
`GraphPool`	117	103.7	14.5	2245.6	533.4	13.9 %
`Hybrid`	117	92.3	23.0	2060.4	465.3	11.4 %
15 hard instances (all more than 30 seconds)						
`CPLEX`	2	6273.9	4616.2	442361.3	330591.4	1940.2 %
`RLT`	8	3561.3	1424.8	136804.2	80704.2	55.1 %
`Bipartite`	15	1850.7	630.3	62796.9	25174.6	21.5 %
`GMSC`	13	2391.6	726.5	8748.2	4481.3	14.4 %
`GraphPool`	15	757.4	333.8	15902.9	8837.0	17.0 %
`Hybrid`	15	608.9	241.7	14538.8	5613.0	15.5 %

Table 3: Branch-and-cut results on StQPs of size 30.

The branch-and-cut results given in Tables 3 and 4 are consistent with the ones reported in subsection 6.2 for the root node. On the one side, RLT inequalities appear to be fundamental for StQP, since `RLT` clearly outperforms `CPLEX`. On the other side, MSC and GMSC bipartite inequalities are also very effective. Indeed, `Bipartite` outperforms `RLT` on all the performance indicators reported in the tables (i.e., number of solved instances, computing time and number of branch-and-bound nodes), while, in turn, both `GraphPool` and `Hybrid` provide a neat improvement over `Bipartite`, especially on the hard instances. This indicates that MSC bipartite in-

	Solved	Time [s]		Nodes		Root Gap
		Avg.	S. Geom.	Avg.	S. Geom.	Avg.
All 110 instances solved by at least one						
`CPLEX`	31	17800.0	13063.7	334357.7	251486.3	1433.6 %
`RLT`	106	1697.4	167.7	22010.8	4621.9	61.6 %
`Bipartite`	110	602.2	111.2	5580.8	1668.4	21.5 %
`GMSC`	110	653.2	194.4	2641.9	675.2	10.9 %
`GraphPool`	110	248.6	85.4	2647.7	687.2	14.4 %
`Hybrid`	110	233.0	92.9	2568.6	585.4	11.6 %
16 hard instances (all more than 300 seconds)						
`CPLEX`	6	14053.3	6814.8	163380.6	103019.4	1997.0 %
`RLT`	12	10190.3	4825.9	114540.0	75263.0	64.1 %
`Bipartite`	16	3530.8	2081.9	28236.0	20190.5	42.0 %
`GMSC`	16	2857.7	2012.7	13989.6	5300.7	35.6 %
`GraphPool`	16	1260.8	988.0	14369.9	5983.9	38.1 %
`Hybrid`	16	1152.1	858.0	14482.7	6138.1	37.4 %

Table 4: Branch-and-cut results on StQPs of size 50.

equalities are important on top of RLT inequalities and that a "clever" selection of GMSC bipartite inequalities is also important to improve over MSC bipartite inequalities. More precisely, although the number of problems solved to optimality is the same for `Bipartite`, `GraphPool` and `Hybrid`, separating GMSC bipartite inequalities yields a remarkable reduction in the number of nodes, which is reflected in a significant reduction in the computing times, as mentioned, especially on the hard instances.

In order to gather more insights on the branch-and-cut results, Figures 6 and 7 show Dolan-Moré performance profiles [10] on all solved instances and on solved hard instances, respectively. This time, instances of size $d = 30$ and $d = 50$ are considered together. In such plots every approach is compared to the virtual best of all approaches according to some performance measure, in our case running time. To this end, for every $x \geq 1$, the fraction of the instances where relative performance of the approach compared to the virtual best is at most x is plotted. Consequently, higher values on the y-axis (for fixed x) and smaller values on the x-axis (for fixed y) are beneficial. We omit the results obtained with default CPLEX from the plots because it is clearly dominated by all the other methods.

The performance profiles confirm the importance of MSC and GMSC bipartite inequalities. Even from those plots one can conclude that `Bipartite` outperforms `RLT` while in turn `GraphPool` outperforms `Bipartite`. Finally, `GraphPool` appears to be the best approach if all (solved) instances are considered (Figure 6), while `Hybrid` becomes instead the best one if we restrict ourselves to hard instances only (Figure 7).

6.4. Computational results on the instances from [23]. As mentioned previously, Scozzari and Tardella [23] proposed a combinatorial algorithm for (StQP) and performed experiments on randomly generated instances from which a subset of 14 instances has been published. The website mentioned in the reference is no longer active, so we republish the instances on http://or.dei.unibo.it/library/msc. Since

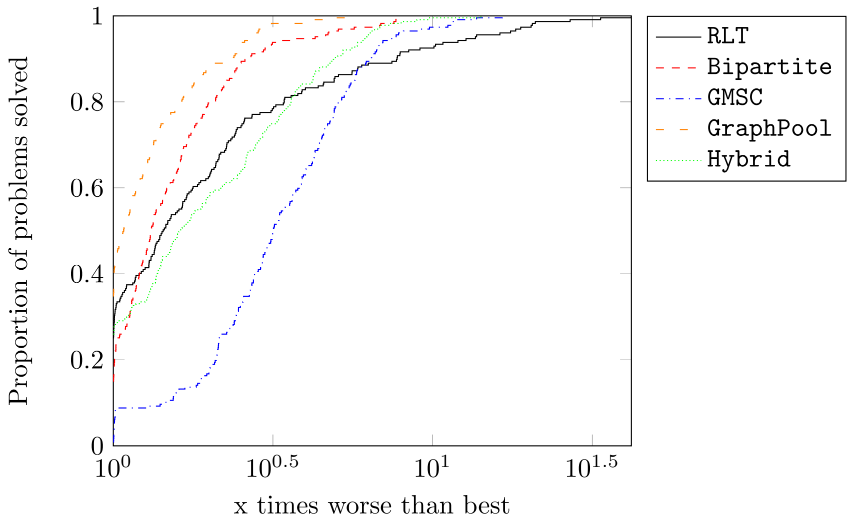


Fig. 6: Dolan-Moré performance profile for all solved instances.

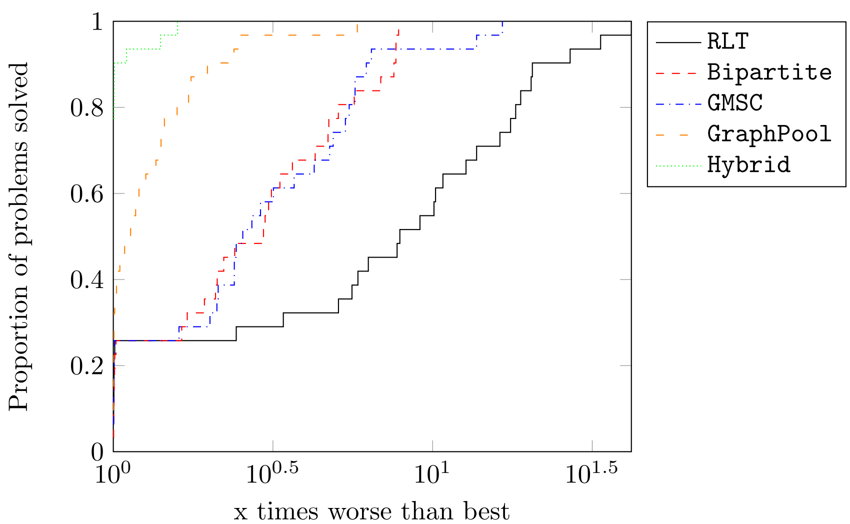


Fig. 7: Dolan-Moré performance profile for "hard" solved instances.

some of the instances have very large dimension (up to $d = 1000$), we impose a time limit of 12 hours.

Table 5 summarizes the results on this testset. Our callback is able to separate a cut at the root on 6 out of the 14 instances. Two instances (Problem_30x30_0.75 and Problem_50x50_0.75) are solved to optimality as soon as RLT inequalites are separated (i. e., in all configurations but `CPLEX`). Table 5 reports time to optimality and number of nodes for these two instances. The last column shows the average root gap of those 6 instances on which our callback separates at least one cut.

The results are consistent with those of the previous sections. Although MSC

	Problem_30x30_0.75		Problem_50x50_0.75		Root Gap
	Time [s]	Nodes	Time [s]	Nodes	Avg.
`CPLEX`	43200.0	1370290	43200.0	276080	4147.5 %
`RLT`	43.1	5130	7962.6	119634	50.6 %
`Bipartite`	12.2	1224	669.9	3239	11.2 %
`GMSC`	26.3	60	4344.6	413	13.7 %
`GraphPool`	14.9	494	513.6	440	9.2 %
`Hybrid`	31.5	63	254.0	440	8.8 %

Table 5: Results on instances from [23] affected by RLT or (G)MSC inequalities.

and GMSC bipartite inequalities do not allow to solve more instances than `RLT`, they greatly reduce the root gap, as well as the time to optimality and the number of nodes. As before, `GraphPool` and `Hybrid` show the best compromise between time needed for separation and impact on the root gap and overall solution time. Finally, note that the 14 instances are among the hardest that the combinatorial algorithm [23] can solve within 1 or 2 hours of time limit, so the overall reduced gaps are quite satisfactory.

7. Generalization. While so far we focused on the set Γ where x is in the standard simplex, we now want to generalize MSC inequalities and GMSC bipartite inequalities to more general sets. In a first step, we will show that an upper bounding inequality on the sum of the x variables suffices. In a second step, we will generalize to coefficients different than 1.

The theorem of Motzkin-Straus establishes a relation between the clique number of a graph and the optimization of a quadratic function over the standard simplex. The objective matrix is the adjacency matrix of the graph. In that QP the objective coefficients are non-negative, thus any solution with $e^T x < 1$ can be improved by a positive factor scaling, where e denotes the vector of all ones. Following this argumentation, we can relax the condition on x in the definition of the set Γ and show that MSC inequalities are valid for set

$$\Gamma_{\leq} = \left\{ (x, Y) \in \mathbb{R}^d \times (\mathbb{R}^d \times \mathbb{R}^d) \;\middle|\; Y = xx^T, e^T x \leq 1, x \geq 0 \right\},$$

where equation $e^T x = 1$ is relaxed to an inequality.

Lemma 6. *MSC inequalities are valid for all points $(x, Y) \in \Gamma_{\leq}$.*

Proof. Consider a graph G and a given $(x, Y) \in \Gamma_{\leq}$. Let

$$\beta = \frac{1}{e^T x},$$

be the scaling factor. Note that $\beta \geq 1$ is constant for fixed x. The vector $\bar{x} = \beta x$ is in the standard simplex such that

$$\langle A, Y \rangle = x^T A x \leq \beta^2 x^T A x = \bar{x}^T A \bar{x} \leq 1 - \frac{1}{\omega(G)}. \qquad \square$$

The next step is to replace the simplex inequality $e^T x \leq 1$ with a more general constraint $a^T x \leq b$ and to also relax non-negativity, i. e., to approximate the set

$$\Gamma_{a,b} = \left\{ (x, Y) \in \mathbb{R}^d \times (\mathbb{R}^d \times \mathbb{R}^d) \;\middle|\; Y = xx^T, a^T x \leq b, a_i x_i \geq 0, \forall i \in V \right\}.$$

The condition $a_i x_i \geq 0$ ensures that x_i is non-negative if a_i is positive and non-positive if a_i is negative.

A similar scaling argument as above is used to generalize the MSC inequality. This time, the coefficients a and the right hand side b also determine the coefficients of the cut.

THEOREM 7. *Let $a \in \mathbb{R}^d$, $b > 0$. Then, the following inequalities are valid for $\Gamma_{a,b}$:*

1. *Motzkin-Straus Clique inequalities*

$$\sum_{(i,j)\in E} \frac{a_i a_j}{b^2} Y_{ij} \leq 1 - \frac{1}{\omega(G)}, \tag{22}$$

where $G = (V, E)$ is a simple graph.

2. *Generalized MSC bipartite inequalities*

$$\sum_{i\in M} \sum_{j\in \bar{M}} \frac{a_i a_j}{b^2} Y_{ij} \leq f_\alpha \left(\sum_{i\in M} \frac{a_i}{b} x_i \right), \tag{23}$$

where $M \subset V$ and f_α is the tangent to the function $g(z) = z - z^2$ at $\alpha \in [0, 1]$.

Proof. Consider $(x, Y) \in \Gamma_{a,b}$ and define $(\bar{x}, \bar{Y})$ as

$$\bar{x}_i = \frac{a_i}{b} x_i,$$
$$\bar{Y}_{ij} = \frac{a_i a_j}{b^2} Y_{ij}.$$

Node that $\bar{x} \geq 0$ and, by construction, $e^T \bar{x} \leq 1$ and $\bar{Y} = \bar{x}\bar{x}^T$ and thus $(\bar{x}, \bar{Y}) \in \Gamma_{\leq}$. Therefore, the validity of (22) follows directly from the previous lemma, namely

$$\sum_{(i,j)\in E} \frac{a_i a_j}{b^2} Y_{ij} = \sum_{(i,j)\in E} \bar{Y}_{ij} \leq 1 - \frac{1}{\omega(G)}.$$

To prove (23), it suffices to realize that the same procedure used to derive the GMSC bipartite inequalities can be repeated with $(\bar{x}, \bar{Y})$. The only difference is that we start with the inequality $e^T \bar{x} \leq 1$ instead of the equation. Multiplying this inequality by $\bar{x}_j$ gives the inequality

$$\sum_{i\in V} \bar{x}_i \bar{x}_j \leq \bar{x}_j,$$

which is valid for all $\bar{x}_j \geq 0$. All remaining operations (addition of inequalities, subtraction of terms on both sides of the inequalities) preserve the sense of the inequality such that the final result is

$$\sum_{j\in M} \sum_{i\in \bar{M}} \bar{Y}_{ij} \leq f_\alpha \left(\sum_{j\in M} \bar{x}_j \right).$$

This completes the proof. ∎

Clearly, $\Gamma_{e,1} = \Gamma_{\leq}$ and it is easy to see that the scaled MSC inequality (22) is really a generalization of the MSC inequality. Also, the separation problems (20) and (21) can be easily adjusted to take the coefficients a and b into account by properly scaling the solution (x^*, Y^*) to be cut off.

We end the section by noting that Theorem 7 establishes the applicability of Motzkin-Straus theorem to a surprisingly large family of optimization problems characterized by an indefinite quadratic objective function subject to a linear inequality. Obviously, such an inequality could also be obtained by aggregation of a large(r) system of inequalities. This makes of Motzkin-Straus Clique inequalities a rather universal tool for indefinite quadratic programming.

7.1. Computational Experiments. In section 6 MSC and GMSC bipartite inequalities have been shown to be extremely effective for StQP. However, it is not obvious if this result carries over to the generalized version of such inequalities when applied to more general problems. To start investigating this question we considered the Quadratic Knapsack Problem (QKP), which is a straightforward generalization of StQP. In QKP one asks to maximize a quadratic objective function subject to a knapsack constraint $w^T x \leq c$, where x is a vector of binary variables, w are non-negative weights and c is the non-negative capacity. We considered the continuous relaxation of QKP because CPLEX reformulates the problem to a MILP if the variables x are binary. Then, the knapsack constraint is the constraint that is used as basis to formulate RLT inequalities and (generalized) Motzkin-Straus Clique inequalities.

In our experiments we considered two sets of instances. The first set, referred as QKP1, is generated by following an approach often used in the literature, see, e. g., [9, 12]. There, the instances are parametrized by their size d and density D, i. e., the fraction of nonzero elements in the objective function. After sampling the nonzero elements, the objective coefficients $Q_{ij} = Q_{ji}$ are sampled uniformly from $[1, 100]$. The weight w_i are sampled uniformly from $[1, 50]$ and the capacity c from $[50, \sum_{i=1}^{d} w_i]$. As for StQP, we use only fully dense objective matrices, i. e., $D = 1.0$, and we generated 150 instances with size $d = 30$. The second set of instances, referred as QKP2 in the following, has the same structure (again 150 instances), but the objective matrices are sampled as described in subsection 6.1. Instances from both test sets are available on http://or.dei.unibo.it/library/msc.

Computations were done for the four configurations[3] `CPLEX`, `RLT`, `Bipartite` and `Hybrid` that are defined as in subsection 6.2. In these experiments, another cutting plane technique already applied by CPLEX, namely BQP cuts [6, 14], has a substantial impact. We present results both with BQP cuts disabled, to get a fair comparison between closures, and with BQP cuts enabled, to evaluate the effect of combined cutting planes.

Tables 6 and 7 show aggregated results at the root node on instances QKP1 and QKP2, respectively. For each of the considered settings, the tables report the number of instances solved, the number of timeouts, the number of instances that are affected by each class of inequalities, and the average root gaps obtained. An instance is considered to be affected by a given class of inequalities if at least one cut from that class is separated.

As seen for StQP, applying RLT inequalities is very beneficial, both in terms of number of instances solved at the root and of root gap reduction. The effect is more pronounced on the instances of type QKP1 (Table 6) where 144 out of 150 instances

[3] In this case, `GMSC` turns out to be too expensive computationally.

	CPLEX	RLT	Bipartite	Hybrid
BQP disabled				
Average root gap [%]				
Gap left	5.65	0.29	0.11	0.10
Closed wrt CPLEX root	–	80.26	85.21	85.41
Closed wrt RLT	–	–	39.62	40.84
Solved/Timeout at the root				
Solved	6	25	39	39
Timeout	0	0	0	12
Affected				
RLT/MSC/GMSC	0/0/0	144/0/0	144/101/0	144/101/47
BQP enabled				
Average root gap [%]				
Gap left	5.65	0.02	0.00	0.00
Closed wrt CPLEX root	–	94.80	95.74	95.74
Closed wrt RLT	–	–	24.24	24.24
Solved/Timeout at the root				
Solved	6	110	144	143
Timeout	0	0	0	2
Affected				
RLT/MSC/GMSC	0/0/0	144/0/0	144/103/0	144/101/5

Table 6: Root node results on Quadratic Knapsack instances QKP1.

are affected by RLT, 19 additional instances are solved w.r.t. CPLEX when BQP cuts are disabled and 104 with BQP cuts enabled. With BQP cuts disabled, the average gap left is reduced from 5.65 to 0.29 %, Bipartite can then close an additional 39.62 % gap w.r.t. RLT leaving only 0.11 % on average and solving in total 39 instances at the root. Surprisingly, separating GMSC bipartite inequalities on top of Bipartite has almost no effect. Even tough almost one third of the instances is affected, number of instances solved and gap stay (almost) the same. When the various types of cuts are combined with BQP cuts, the overall comparison is similar but the gap closed by all techniques is even more important. Remarkably, Bipartite and Hybrid can solve almost all instances (with 6 and 7 unsolved instances at the root respectively).

Instances of type QKP2 (Table 7) have much larger average root gap, but show similar phenomena. On both test sets, RLT and Bipartite contribute to the solution of several additional instances at the root. On the seconds test set, with BQP cuts disabled, the effect of Bipartite on the average root gap is very small, but 16 % gap is closed with respect to RLT. Even if Hybrid allows to solve 3 more instances w.r.t. Bipartite, GMSC bipartite inequalities appear to be less effective w.r.t. what we observed for StQP where the impact on the root gap is much more marked. Again the combination with BQP cuts seems beneficial.

8. Conclusion. We studied cutting planes for standard quadratic programs and quadratic knapsack. Those cutting planes exploit the relation between those problems

	CPLEX	RLT	Bipartite	Hybrid
BQP disabled				
Average root gap [%]				
Gap left	124.86	118.44	116.40	115.74
Closed wrt CPLEX root	–	22.69	28.14	29.12
Closed wrt RLT	–	–	16.08	17.78
Solved/Timeout at the root				
Solved	24	32	43	46
Timeout	0	0	0	0
Affected				
RLT/MSC/GMSC	0/0/0	94/0/0	94/72/0	94/72/73
BQP enabled				
Average root gap [%]				
Gap left	48.77	45.29	44.63	44.54
Closed wrt CPLEX root	–	27.66	37.05	39.76
Closed wrt RLT	–	–	22.39	26.48
Solved/Timeout at the root				
Solved	29	40	55	56
Timeout	0	0	0	21
Affected				
RLT/MSC/GMSC	0/0/0	94/0/0	94/80/0	94/72/69

Table 7: Root node results on Quadratic Knapsack instances QKP2.

and the maximum clique problem. By analyzing the relationship between the new cutting planes and the RLT inequalities, we have shown that, interestingly, (i) MSC bipartite inequalities are not comparable with first level RLT inequalities, and (ii) the derivation of GMSC bipartite inequalities generalizes both MSC bipartite inequalities and first level RLT. Our computational experiments show that both MSC bipartite inequalities and GMSC bipartite inequalities allow to get a significantly stronger bound than first level RLT alone.

Some possible extensions of our approach would be to derive cuts corresponding to graphs with larger clique number (greater than 2) and to exploit generalized version of the Motzkin-Straus Theorem [13].

Acknowledgments. The work of the third author has been supported by the EU Initial Training Network MINO (Grant Agreement Number 316647), which is strongly acknowledged. The work of the second author was partially conducted when he was faculty member at the University of Bologna and supported by MIUR, Italy under the PRIN 2012 grant. The authors are indebted to Andrea Scozzari and Fabio Tardella for sharing their instances and their thoughts on this fascinating problem.

REFERENCES

[1] T. Achterberg, D. Junglas, and R. Wunderling, *Deterministic parallelization through*

atomic task computation, Dec. 3 2013, https://www.google.co.uk/patents/US8601486.

[2] P. Belotti, C. Kirches, S. Leyffer, J. Linderoth, J. Luedtke, and A. Mahajan, *Mixed-integer nonlinear optimization*, Acta Numerica, 22 (2013), pp. 1–131.

[3] I. Bomze, M. Dür, E. de Klerk, C. Roos, A. Quist, and T. Terlaky, *On copositive programming and standard quadratic optimization problems*, Journal of Global Optimization, 18 (2000), pp. 301–320.

[4] I. M. Bomze, *On standard quadratic optimization problems*, Journal of Global Optimization, 13 (1998), pp. 369–387.

[5] I. M. Bomze, M. Locatelli, and F. Tardella, *New and old bounds for standard quadratic optimization: dominance, equivalence and incomparability*, Mathematical Programming, 115 (2008), pp. 31–64.

[6] P. Bonami, O. Günlük, and J. Linderoth, *Solving box-constrained nonconvex quadratic programs*, Working paper, (2016).

[7] C. Brás, G. Eichfelder, and J. Júdice, *Copositivity tests based on the linear complementarity problem*, Computational Optimization and Applications, 63 (2016), pp. 461–493.

[8] A. Caprara, M. Carvalho, A. Lodi, and G. J. Woeginger, *Bilevel knapsack with interdiction constraints*, INFORMS Journal on Computing, to appear (2015).

[9] A. Caprara, D. Pisinger, and P. Toth, *Exact solution of the quadratic knapsack problem*, INFORMS Journal on Computing, 11 (1999), pp. 125–137.

[10] E. D. Dolan and J. J. Moré, *Benchmarking optimization software with performance profiles*, Mathematical Programming, 91 (2002), pp. 201–213.

[11] M. Dür, *Recent Advances in Optimization and its Applications in Engineering: The 14th Belgian-French-German Conference on Optimization*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, ch. Copositive Programming – a Survey, pp. 3–20.

[12] G. Gallo, P. Hammer, and B. Simeone, *Quadratic knapsack problems*, in Combinatorial Optimization, M. Padberg, ed., vol. 12 of Mathematical Programming Studies, Springer Berlin Heidelberg, 1980, pp. 132–149.

[13] L. Gibbons, D. Hearn, P. Pardalos, and M. Ramana, *Continuous characterizations of the maximum clique problem*, Mathematics of Operations Research, 22 (1997), pp. 754–768.

[14] IBM CPLEX 12.6.3 User's Manual, *Boolean Quadric Polytope (BQP) cuts*. http://www-01.ibm.com/support/knowledgecenter/SSSA5P_12.6.3/ilog.odms.cplex.help/CPLEX/UsrMan/topics/discr_optim/mip/cuts/28_BQPcuts.html (accessed March 18, 2016).

[15] IBM CPLEX 12.6.3 User's Manual, *Deterministic time limit*. http://www.ibm.com/support/knowledgecenter/SSSA5P_12.6.3/ilog.odms.cplex.help/CPLEX/Parameters/topics/DetTiLim.html (accessed March 18, 2016).

[16] IBM CPLEX Optimizer, *http://www.cplex.com*.

[17] R. M. Karp, *Reducibility among combinatorial problems*, in Complexity of Computer Computations, R. E. Miller, J. W. Thatcher, and J. D. Bohlinger, eds., The IBM Research Symposia Series, Springer US, 1972, pp. 85–103.

[18] A. Lodi, T. K. Ralphs, and G. J. Woeginger, *Bilevel programming and the separation problem*, Mathematical Programming, 146 (2014), pp. 437–458.

[19] G. McCormick, *Computability of global solutions to factorable nonconvex programs: Part i - convex underestimating problems*, Mathematical Programming, 10 (1976), pp. 147–175.

[20] R. Misener and C. A. Floudas, *GloMIQO: Global mixed-integer quadratic optimizer*, Journal of Global Optimization, 57 (2013), pp. 3–50.

[21] T. S. Motzkin and E. G. Straus, *Maxima for graphs and a new proof of a theorem of Turán*, Canadian Journal of Mathematics, 17 (1965), pp. 533–540.

[22] J. Schweiger, *Nonlinearity and uncertainty in energy networks*, PhD thesis, Technische Universität Berlin, to appear, 2016.

[23] A. Scozzari and F. Tardella, *A clique algorithm for standard quadratic programming*, Discrete Applied Mathematics, 156 (2008), pp. 2439 – 2448.

[24] H. D. Sherali and W. P. Adams, *A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems*, SIAM Journal on Discrete Mathematics, 3 (1990), pp. 411–430.

[25] H. D. Sherali and A. Alameddine, *A new reformulation-linearization technique for bilinear programming problems*, Journal of Global Optimization, 2 (1992), pp. 379–410.

[26] M. Tawarmalani and N. V. Sahinidis, *Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming: Theory, Algorithms, Software, and Applications*, Kluwer Academic Publishers, Boston MA, 2002.