Proceedings of the Twenty-Sixth RAMP Symposium
Hosei University, Tokyo, October 16-17, 2014

Solving Mixed-Integer Quadratic Programming problems with IBM-CPLEX: a progress report

Christian Bliek[1*], Pierre Bonami[2†], and Andrea Lodi[3‡]

Abstract Mixed-Integer Quadratic Programming problems have a vast impact in both theory and practice of mathematical optimization. Classical algorithmic approaches, their implementation within IBM-CPLEX and new algorithmic advances will be discussed.

Keywords Quadratic Programming, branch and bound, convex programming, bound reduction

1. Introduction

We consider the general Mixed-Integer Quadratic Programming (MIQP) problem

$$\min \quad \tfrac{1}{2}x^TQx + c^Tx \tag{1}$$
$$Ax = b \tag{2}$$
$$x_j \in \mathbb{Z} \qquad j = 1, \ldots, p \tag{3}$$
$$l \leq x \leq u \tag{4}$$

where $Q^{n\times n}$ is a symmetric matrix. Note that constraints (4) are fundamental in the complexity sense because a striking result by Jeroslow [11] proves the undecidability of unbounded problems.

The relevance of MIQP in Mathematical Optimization is twofold. On the one side, a number of applications arise in a practical setting starting from the most classical one in portfolio optimization, see, e.g., [4, 7, 17]. On the other side, MIQP has been clearly the first step for a methodological generalization of Mixed-Integer Linear Programming (MILP) to general Mixed-Integer Nonlinear Programming (MINLP).

It is well known that MIQP is NP-hard, trivially because it contains MILP as a special case. However, it is important to notice that, differently from MILP, the source of complexity of MIQP is not restricted to the integrality requirement on (some of) the variables (3). In fact, the (continuous) Quadratic Programming (QP) special case (i.e., $p = 0$) is, in general, NP-hard as well. Let $G = (N, E)$ be a graph and Q be the incidence matrix of G. The

1 CPLEX Optimization, IBM France, 1681-HB2 Route des Dolines 06560 Valbonne, France
2 CPLEX Optimization, IBM Spain, Sta. Hortensia 26-28, 28002 - Madrid (M), Spain
3 DEI, University of Bologna, and IBM-Unibo Center of Excellence in Mathematical Optimization, Viale Risorgimento 2, 40136, Bologna, Italy
∗ E-mail address: bliek@fr.ibm.com
† E-mail address: pierre.bonami@es.ibm.com
‡ E-mail address: andrea.lodi@unibo.it

optimal value of

$$\min\left\{\frac{1}{2}x^TQx : \sum_{j=1}^{n} x_j = 1, x \geq 0\right\}$$

is $\frac{1}{2}\left(1 - \frac{1}{\chi(G)}\right)$, where $\chi(G)$ is the clique number of G [14], which is well-known to be NP-hard to compute.

Therefore, we will distinguish between the case in which the relaxation obtained by dropping the integrality requirements (3) (if any) is convex (thus, solvable in polynomial time), and that in which it is nonconvex. Roughly speaking, convex MIQPs are currently solved by heavily relying on MILP techniques by either replacing in the classical branch-and-bound scheme the Linear Programming (LP) solver with a QP solver, or solving MILPs as subproblems. Instead, nonconvex MIQPs require somehow more sophisticated techniques, namely Global Optimization (GO). An intermediate case between these two is the one where Q is not convex but MIQP can easily be reformulated as a convex MIQP. This is true for example when all variables are constrained to be 0-1, or more generally when all products are between a 0-1 variable and a bounded variable.

In this short paper we review the IBM-CPLEX evolution in solving QPs and MIQPs with special emphasis on nonconvex MIQPs because the capability of solving them to proven optimality has been added into the solver very recently. Schematically, Table 1 below outlines a brief history of MIQP within CPLEX, where B&B stands for branch and bound. We remind the reader that a QP is convex if and only if the matrix Q is positive semidefinite.

class	p	Q	algorithm	V. (Year)
convex QP	0	$\succeq 0$	barrier	4.0 (1995)
–	–	–	QP simplex	8.0 (2002)
convex MIQP	> 0	$\succeq 0$	B&B	8.0 (2002)
nonconvex QP	0	$\not\succeq 0$	barrier (local)	12.3 (2011)
–	–	–	spatial B&B (global)	12.6 (2013)
nonconvex MIQP	> 0	$\not\succeq 0$	spatial B&B (global)	12.6 (2013)

Table 1 History of MIQP within CPLEX.

The paper is organized as follows. In Section 2 we briefly discuss the classical algorithms for convex MIQPs and some specific implementation choices that are peculiar of CPLEX. In Section 3 we move to the nonconvex MIQP case and we present the current status of implementation of solution techniques for them. Both Sections 2 and 3 end with some computational experiments giving a snapshot of the current performance of CPLEX on MIQPs. Since the algorithmic treatment of 0-1 nonconvex MIQPs is more similar to that of convex MIQPs than nonconvex MIQPs, we will review them as part of the section on convex MIQPs. Finally, in Section 4 we draw some conclusions.

2. Convex MIQPs

The solution of convex MINLPs has reached in the last decade a rather stable and mature algorithmic technology, see, e.g., Bonami et al. [6]. Essentially, the two corner stones of algorithms and software are Nonlinear Programming (NLP)-based branch-and-bound [10] and Outer Approximation (OA) [8] algorithms. Although the NLP-based branch and bound is the only relevant for CPLEX (see discussion at the end of the section) for the special case of (convex) MIQPs, we review the OA as well. This is because the most standard trick in MINLP is to add a variable $\alpha \in \mathbb{R}$, replace the original objective function with "$\min \alpha$" and add the constraint $f(x) \leq \alpha$. Of course, this could be done for MIQPs as well ($f(x) = \frac{1}{2}x^T Qx + c^T x$ in that case), thus transforming the problem in a convex Mixed-Integer Quadratically-Constrained Programming (MIQCP) problem and sometimes it does make sense in practice [9].

The NLP-based branch and bound is a straightforward generalization of the main enumerative algorithm for MILP. The main difference is that an NLP, a QP in our special case, is solved at every node of the branch-and-bound tree to provide a valid lower bound, instead of an LP problem. The other main components of the MILP scheme remain the same: branching on the integer constrained variables, and pruning nodes based on

- infeasibility: the node relaxation is infeasible,
- bound: the node lower bound value is not smaller than the incumbent solution value, and
- integer feasibility: the node relaxation admits an integer solution.

The main source of inefficiency of this straightforward extension is, in general convex MINLP, the difficulty of warm starting NLP solvers, i.e., reusing the information on the solution of a relaxation from a node to another. However, for the convex MIQP case this issue is relatively under control when the QP simplex is used to solve the node relaxation.

The basic idea of the OA decomposition is to take first-order approximations of constraints at different points and build an MILP equivalent to the initial MINLP. This corresponds for a nonlinear function $g(x)$ and for a set K of its points $\overline{x}^k, k \in K$ to write the constraints

$$g(\overline{x}^k) + \nabla g(\overline{x}^k)^T (x - \overline{x}^k) \leq 0. \quad k \in K \tag{5}$$

This basic idea is illustrated in Figure 1.

It is easy to see that the MILP constructed through an initial set of linearization points is a relaxation of the original convex MINLP. Then, if solved to optimality, it provides a valid lower bound value for the original problem and a solution $\hat{x}$ such that $\hat{x}_j \in \mathbb{Z}, j = 1, \ldots, p$. However, this solution might be NLP infeasible (because of the first-order approximation used) and the NLP($\hat{x}$) obtained by fixing the integer component of the original problem ($x_j = \hat{x}_j, j = 1, \ldots, p$) is solved. If NLP($\hat{x}$) is feasible, then it in turn provides an upper bound value. Otherwise, general-purpose NLP software will typically return a weighted minimization of the violation of the constraints. In both cases, a new point $\overline{x}^{|K|+1}$ can be added

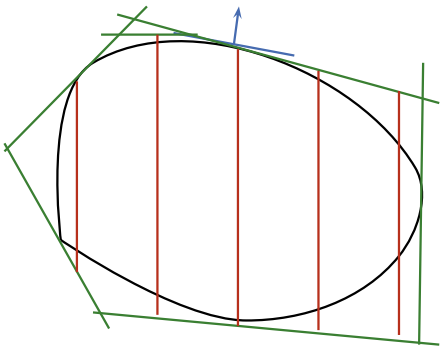


Figure 1 The Outer Approximation MILP equivalent.

to the set K, and the method is iterated by applying first-order approximation to it, thus enhancing the quality of the MILP equivalent. Duran and Grossman [8] show that the iterative scheme converges to the optimal solution value of the original problem in a finite number of steps, provided the nonlinear functions are convex and continuously differentiable, and that a constraint qualification holds for each point $\overline{x}^k, k \in K$.

The OA algorithm outlined above requires solving one MILP at each iteration. This can be rather time consuming if the initial MILP does not provide a good approximation of the original problem. However, the OA can be embedded in a single tree search [15, 6, 1]. Namely, start solving the same initial MILP by branch and bound and at each integer feasible node: (i) Solve NLP($\hat{x}$), and enrich the set of linearization points, (ii) Resolve the LP relaxation of the node with the new fist-order constraints, and (iii) Repeat as long as node is integer feasible. Of course, no pruning is allowed by integer feasibility.

Due to the efficiency of its dual QP simplex algorithm, CPLEX does not implement an OA algorithm for MIQP but it does have one for the case of problems with also convex quadratic constraints, MIQCPs. The details of this implementation are beyond our scope here and we do not discuss OA further.

2.1. 0-1 Nonconvex QPs

Here, we review the treatment of the subclass of nonconvex MIQPs that CPLEX can easily turn into convex MIQPs. The most relevant case is nonconvex QPs with only binary variables and we will first restrict our discussion to it. Precisely, we discuss two different approaches to solve nonconvex MIQPs when all variables are binary.

The first one, used in older versions of CPLEX, consists of transforming a nonconvex binary MIQP into an equivalent convex MIQP. To do this one uses the fact that when a variable x is binary $x = x^T I x$. The quadratic part of the objective $x^T Q x$ may therefore be changed in $x^T(Q + \rho I)x - \rho x$. Thus, by determining a value of $\rho > 0$ that makes $Q + \rho I$ positive semidefinite, the nonconvex MIQP is transformed into a convex MIQP that CPLEX can readily solve. A simple choice for the value of ρ is to take the absolute value of the smallest (negative) Eigenvalue of Q. Many more elaborated schemes have been proposed and have been shown to be effective (see, e.g., [5]).

The second approach, which is used in more recent CPLEX versions, is to turn a binary nonconvex MIQP into an equivalent MILP. By the argument used above, the quadratic terms $q_{ii}x_i^2$ may be replaced by linear ones $q_{ii}x_i$. For the bilinear terms $q_{ij}x_ix_j$, a new variable $y_{ij} > 0$ is introduced together with the inequalities $x_i + x_j - 1 \leq y_{ij}$ if $q_{ij} > 0$, or $y_{ij} \leq x_i$ and $y_{ij} \leq x_j$ if $q_{ij} < 0$.

In general and in theory none of these two approaches dominates the other but an extended internal testing has shown a significant advantage, on average, for the latter (see Section 2.2) and therefore it is the default in the 12.6 version(s) of CPLEX.

Note that this latter approach is actually more general and can also be applied when only one of the two variables in the bi-linear term is binary while the other one has finite bounds. Finally, the linearization approach is beneficial for convex MIQPs as well, since if everything can be linearized the problem may be solved directly using MILP techniques.

2.2. A Computational Snapshot

To conclude this section, we present in Table 2 the results of a simple experiment comparing the number of problems solved and the running time of various versions of CPLEX. We compare major releases since the first version able to solve MIQPs. The test set is CPLEX internal test of MIQP models. It is composed both of convex MIQPs and of binary MIQPs that can be convexified. All runs where executed on a cluster of identical 12 core Intel Xeon CPU E5430 machines running at 2.66 GHz and equipped with 24 GB of memory, so that hardware speed up is not included in the numbers. Table 2 reports the results on 193

		8.0	9.0		10.0		11.0		12.1		12.5		12.6	
Group	# inst.	t.o.	t.o.	speed up	t.o.	speed up	t.o.	speed up	t.o.	speed up	t.o.	speed up	t.o.	speed up
solved	193	50	49	1.01	50	1.03	48	1.30	48	1.34	39	1.69	1	5.88
$>$ 1 sec.	104	50	49	1.01	50	1.05	48	1.64	48	1.71	39	2.66	1	27.16
$>$ 10 sec.	89	50	49	1.02	50	1.03	48	1.67	48	1.72	39	2.76	1	42.94
$>$ 100 sec.	72	50	49	1.03	50	1.05	48	1.52	48	1.55	39	2.60	1	79.09
$>$ 1,000 sec.	60	50	49	1.07	50	1.06	48	1.39	48	1.41	39	2.57	1	124.02

Table 2 Comparison of results with different CPLEX versions on convex MIQP.

instances, column "t.o." indicates the number of instances where the time limit of 10,000 CPU seconds has been reached and the speed up is computed with respect to CPLEX 8.0. In addition, the 193 instances are also split into classes depending on the computing time needed: the first row shows results for all instances that were solved by at least one solver in the time limit; the subsequent rows show the results for all instances where the slowest solver took more than the prescribed computing time. The results clearly show the continuous improvement within CPLEX evolution and especially CPLEX version 12.6 shows an impressive improvement mainly due to the automatic linearization discussed in Section 2.1.

3. General Nonconvex QPs and MIQPs

As indicated by Table 1, solving general nonconvex QPs and MIQPs in CPLEX is a relatively

recent possibility. Namely, a *local* solver for nonconvex QPs has been available since version 12.3 (2011) and it is a Primal-Dual Interior Point algorithm. Indeed, interior point algorithms that are exact for convex QPs can naturally be extended to provide locally optimal solutions for the nonconvex case. This is the approach of Ipopt [20], which is however much more general because it solves general NLP problems. Thus, a number of additional steps like feasibility restoration, second order correction, filter, etc. are not needed. As a technical note, observe that the local QP solution is not computed by default. In fact, if Q is indefinite ($\not\succeq 0$) CPLEX returns a (kind of) error message, namely, `CPXERR_Q_NOT_POS_DEF`, to alert the user. The computation of the local solution is then activated by setting the option `solution target` to `2` (or `CPX_SOLUTIONTARGET_FIRSTORDER`).

Concerning solving nonconvex QPs and MIQPs to global optimality this is possible in CPLEX since version 12.6 (2013) by setting `solution target` to `3` (or `CPX_SOLUTIONTARGET_OPTIMALGLOBAL`). For solving general QPs and MIQPs, i.e., those that do not show a special structure to be exploited, CPLEX relies on GO methods and, in particular, on the so-called Spatial branch-and-bound algorithm (see, e.g., [2]). Roughly speaking, the Spatial branch and bound establishes a convex (easily solvable) relaxation of the initial QP (either the problem itself or its continuous relaxation), and performs branching on solutions of this relaxation with the twofold aim of partitioning the search space (as usual) and improving the convex relaxation as much as possible. An example of elementary convex relaxation applied to the simple nonconvex constraint $y \leq x_1^2$ depicted in Figure 2 is shown in Figure 3. Essentially, the general idea is to replace each nonconvex "piece" with a convex

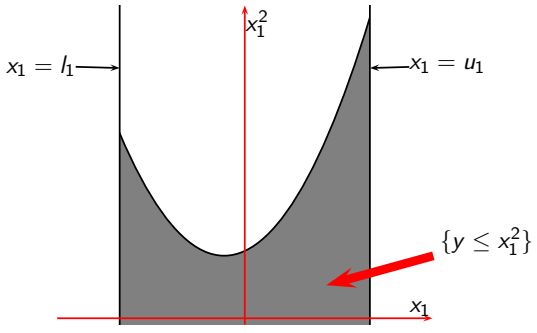
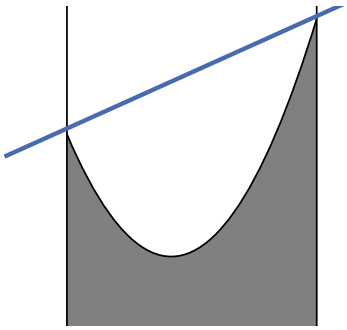


Figure 2 A nonconvex quadratic constraint.



$x_1^2 \leq y_{11}^+ := (l_1 + u_1)x_1 - l_1 u_1$

Figure 3 Elementary convex relaxation of a quadratic constraint: the secant approach.

relaxation of it, so as to produce an overall convex relaxation of the original problem that is solvable to global optimality, thus providing a valid lower bound. Of course, more complex relaxations than the one of Figure 3 can be used, thus providing a tighter approximation, an example being the convex hull relaxation of a single product, say $x_1 x_2$, given by the so-called

McCormick inequalities [13], which is of fundamental relevance for QPs

$$x_1x_2 \geq y_{12}^- := \max \left\{ \begin{array}{c} u_2x_1 + u_1x_2 - u_1u_2 \\ l_2x_1 + l_1x_2 - l_1l_2 \end{array} \right\} \tag{6}$$

$$x_1x_2 \leq y_{12}^+ := \min \left\{ \begin{array}{c} u_2x_1 + l_1x_2 - l_1u_2 \\ l_2x_1 + u_1x_2 - u_1l_2 \end{array} \right\} \tag{7}$$

CPLEX uses two different ways of constructing the initial convex relaxation of a QP.

Q-space reformulation and relaxation.
Let $Q = P + \tilde{Q}$, with P being the diagonal positive semidefinite matrix containing $q_{ii} > 0$. Add one $y_{ij} = x_ix_j$ variable for each non-zero entry q_{ij} of $\tilde{Q}$. Relax $y_{ij} = x_ix_j$ by using McCormick (6)-(7) and Secant approximations, so as to obtain the system

$$\min \ \tfrac{1}{2}x^TPx + \tfrac{1}{2}\langle \tilde{Q}, Y \rangle + c^Tx \tag{8}$$

$$Ax = b \tag{9}$$

$$x_j \in \mathbb{Z} \qquad j = 1, \ldots, p \tag{10}$$

$$y_{ij}^- \leq y_{ij} \leq y_{ij}^+ \tag{11}$$

$$y_{ii} \leq y_{ii}^+ \tag{12}$$

$$l \leq x \leq u \tag{13}$$

where $\langle Q, Y \rangle = \sum_{i,j} q_{ij}y_{ij}$.

Factorized Eigenvector space reformulation and relaxation.
Use a decomposition to get $z = Lx$ and $z^TDz = x^TQx$ and do the same steps as before (but simpler). Namely, let $D = D^+ - D^-$ with $D^\pm$ diagonal positive semidefinite matrices. Add one $y_{ii} \leq z^2$ variable for each non-zero entry of D^-. Infer finite bounds, l^z, u^z for z and relax $y_{ii} \leq z_i^2$ by using Secant approximation, so as to obtain the system

$$\min \ \tfrac{1}{2}z^TD^+z - \textstyle\sum_{i=1}^n \frac{d_{ii}}{2}y_{ii} + c^Tx \tag{14}$$

$$Ax = b, Lx = z \tag{15}$$

$$x_j \in \mathbb{Z} \qquad j = 1, \ldots, p \tag{16}$$

$$y_{ii} \leq y_{ii}^+ \tag{17}$$

$$l \leq x \leq u, l^z \leq z \leq u^z \tag{18}$$

In CPLEX, we first use a block indefinite decomposition $Q = M^TBM$ where M and B are such that M is 2-block triangular and B is 2-block diagonal, see Figure 4, and then B is diagonalized.

The two reformulations/relaxations are in general incomparable, while if Q is diagonal they are identical. If $Q \succ 0$, the Eigenvector reformulation is preferable because it preserves convexity. For this reason, CPLEX uses it if the problem at hand looks "almost" convex. Nevertheless, the Q-space reformulation provides surprising tight bounds as shown by Luedtke, Namazifar, and Linderoth [12].

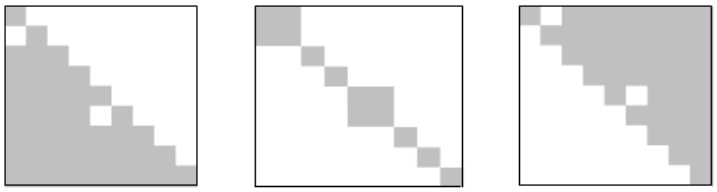


Figure 4 Q factorization in CPLEX.

Once the chosen QP relaxation has been solved integrality must still be enforced. But, as shown in the previous paragraphs, the integrality requirements (3) are not the only ones that have been relaxed. Let $(\overline{x}, \overline{y})$ be the solution of the chosen QP relaxation (after presolve/cutting) and assume $x_j \in \mathbb{Z}$, $j = 1, \ldots, p$, i.e., standard branching on the integer components has already been executed. If there exists $\overline{y}_{ij} \neq \overline{x}_i \overline{x}_j$, then $(\overline{x}, \overline{y})$ is not a solution of the problem and we need to branch. This is the so-called spatial branching that consists in picking an index i (of a continuous variable x_i), choosing a value θ between $\frac{l_i+u_i}{2}$ and $\overline{x}_i$, branching by changing the bound to θ and updating all Secant and McCormick approximations involving this bound. This is depicted in Figure 5 for the Secant approximation. It is easy to see that the effect of the spatial branching is on tightening the convex relaxation,

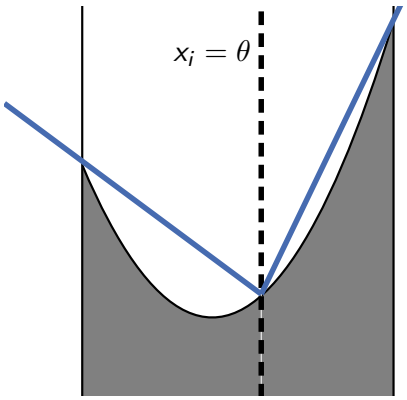


Figure 5 Spatial branching on the secant approximation.

thus leading in general to better bounds. The reader is referred to Belotti et al. [3] for more details on branching in GO.

Of course, the initial convex relaxation and the branching step are not the only fundamental components of an MIQP solver, in general, and of CPLEX, in particular. A crucial component of an efficient GO algorithm is, in particular, bound tightening. Many methods have been developed in the literature, and the reader is referred to Tawamalani and Sahinidis [16] Belotti et al. [3] and Vigerske [19] for recent reviews. In CPLEX, we mainly rely on applying bound strengthening on the KKT system to produce tighter bounds at each node (see, e.g., [18]). This is done by exploiting the fact that when all integer variables are fixed,

in an optimal solution the continuous variables need to satisfy the KKT system.

3.1. A Computational Snapshot for Nonconvex MIQPs

Because there is only one version of CPLEX that can solve general nonconvex MIQPs, we cannot present much as an illustration of the progress of CPLEX in solving this type of problems. In the development of the new solver, we compared the solution provided by those of the two academic solvers Couenne [3] and SCIP [19] and we believe that the algorithm of CPLEX compares well to those both in speed and reliability.

To illustrate here the performance of the solver we just present the results of a computation using different numbers of threads. Indeed, a distinctive feature of CPLEX compared to other GO solvers that can solve MIQPs is its ability to fully exploit modern machines with several processors. To illustrate CPLEX effectiveness, in Table 3 we show solution times on the same machine using 1 and 4 threads. The information in Table 3 is the same as in Table 2 with the addition of the ratio between the nodes explored with 1 thread and 4 threads. The table clearly shows the advantage of exploiting multiple threads. CPLEX using 4 thread

		1 thread	4 threads		
Group	# inst.	#time outs	#time outs	speed up	node ratio
solved	296	4	0	1.19	1.05
$>$ 1 sec.	107	4	0	1.58	1.03
$>$ 10 sec.	60	4	0	1.82	0.99
$>$ 100 sec.	33	4	0	2.09	1.02

Table 3 Comparison of results of CPLEX 12.6 global solver with 1 thread vs. 4 threads.

is able to solve 4 instances that are not solved using 1 thread only. The number of nodes explored with the two settings is similar.

4. Conclusions

We have briefly reviewed the main algorithmic components for solving convex and nonconvex MIQPs with special emphasis to the way IBM-CPLEX implements them. A snapshot of the computational performance of CPLEX on both classes has been reported by emphasizing the version-to-version evolution of the solver in the convex case and its scalability in terms of number of threads in the nonconvex one.

References

[1] K. Abhishek, S. Leyffer, and J.T. Linderoth: FilMINT: An outer-approximation-based solver for nonlinear mixed integer programs, *INFORMS Journal on Computing* **22** (2010), 555–567.

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

[3] P. Belotti, J. Lee, L. Liberti, F. Margot, and A. Wächter: Branching and bounds tightening techniques for non-convex MINLP, *Optimization Methods and Software* **24** (2009), 597–634.

[4] D. Bienstock: Computational study of a family of mixed-integer quadratic programming problems, *Mathematical Programming* **74** (1996), 121–140.

[5] A. Billionnet, S. Elloumi, and A. Lambert: Extending the QCR method to general mixed-integer programs, *Mathematical Programming* **131** (2012),381–401.

[6] P. Bonami, L.T. Biegler, A.R. Conn, G. Cornuéjols, I.E. Grossmann, C.D. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, and A. Wa̋chter: An algorithmic framework for convex mixed integer nonlinear programs, *Discrete Optimization* **5** (2008), 186–204.

[7] P. Bonami and M. Lejeune: An Exact Solution Approach for Integer Constrained Portfolio Optimization Problems Under Stochastic Constraints, *Operations Research* **57** (2009), 650–670.

[8] M.A. Duran and I.E. Grossmann: An outer-approximation algorithm for a class of mixed-integer nonlinear programs, *Mathematical Programming* **36** (1986), 307–339.

[9] O. Günlük and J. Linderoth: Perspective relaxation of mixed integer nonlinear programs with indicator variables, In A. Lodi, A. Panconesi and G. Rinaldi (eds.), *IPCO 2008: The Thirteenth Conference on Integer Programming and Combinatorial Optimization* (Springer, 2008), 1–16.

[10] O.K. Gupta and A. Ravindran: Branch and bound experiments in convex nonlinear integer programming, *Management Science* **31** (1985), 1533–1546.

[11] R. Jeroslow: There cannot be any algorithm for integer programming with quadratic constraints, *Operations Research* **21** (1973), 221–224.

[12] J. Luedtke, M. Namazifar, and J. Linderoth: Some results on the strength of relaxations of multilinear functions, *Mathematical Programming* **136** (2012), 325–351.

[13] G.P. McCormick: Computability of global solutions to factorable nonconvex programs: Part I - Convex underestimating problems, *Mathematical Programming* **10** (1976), 147–175.

[14] 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), 533–540.

[15] I. Quesada and I.E. Grossmann: An LP/NLP based branchâĂŞandâĂŞbound algorithm for convex MINLP optimization problems, *Computers and Chemical Engineering* **16** (1992), 937–947.

[16] 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, Dordrecht, 2002.

[17] J.P. Vielma, S. Ahmed, and G. Nemhauser: A lifted linear programming branch-and-bound algorithm for mixed integer conic quadratic programs, *INFORMS Journal on Computing* **20** (2008), 438–450.

[18] P. Van Hentenryck, L. Michel, and Y. Deville: *Numerica: a Modeling Language for Global Optimization*, The MIT Press, 1997.

[19] S. Vigerske: *Decomposition in multistage stochastic programming and a constraint integer programming approach to mixed-integer nonlinear programming*, PhD thesis, Humboldt-University Berlin, Germany, 2013.

[20] A. Wächter and L.T. Biegler: On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming, *Mathematical Programming* 106 (2006), 25–57.