Exact solutions of polynomial programs through quadratic convex reformulations: theory and applications

Amélie Lambert

HABILITATION À DIRIGER DES RECHERCHES

du

CONSERVATOIRE NATIONAL DES ARTS ET MÉTIERS

Spécialité Informatique

École doctorale Sciences des Métiers de l'Ingénieur (SMI)

Exact solutions of polynomial programs through quadratic convex reformulations: theory and applications

---

Mémoire d'habilitation à diriger des recherches présenté et soutenu publiquement par

Amélie LAMBERT

le 19 novembre 2021

devant le jury composé de

Présidente du jury
Ruth MISENER - Professeure, Imperial College London.

Rapporteurs
Christoph BUCHHEIM - Professeur, Dortmund University.
Claudia D'AMBROSIO - Directrice de Recherche CNRS, Lix.
Andrea LODI - Professeur, Cornell Tech.

Examinateurs
Sourour ELLOUMI - Professeure, ENSTA-ParisTech
Safia KEDDAD-SIDHOUM - Professeure, Cnam.
Jon LEE - Professeur, University of Michigan.

Abstract

Optimization is a mathematical tool that allows to model and solve problems, in which a criterion is to be optimized. When the function to be minimized and the constraints that characterize the problem are multivariate polynomial functions, it is called polynomial optimization. It allows to model many real problems where the links between the variables are naturally non-linear, as for example in the fields of physics or electricity. Two main difficulties appear when solving a polynomial optimization problem: the non-convexity of its functions and the integrality of some of its variables. In the presence of convex functions, we can distinguish two cases. If the variables are continuous, the problem is polynomial. If this is not the case, the standard resolution algorithm is the branch-and-bound based on continuous relaxation. Under these assumptions, there exist tools that efficiently implement these algorithms.

Our research focuses on the case where the functions are quadratic and non convex. We have proposed several exact algorithms, each depending on the class of the problem under study. More precisely, starting from the most specific class, we have extended our approaches so that they can efficiently handle increasingly complex problems. From this incremental work comes a hierarchy of methods, where each new algorithm is a generalization of the previous one. These algorithms work in two phases. The first one consists in computing a convex quadratic relaxation with a strong bound. To do this, we first introduce a family of formulations equivalent to the problem in an extended space of variables. In these formulations, all initial functions are convex, and the only non-convexity lies in an additional constraint and in the integrality of the variables. Then, within this family, we calculate a quadratic convex relaxation whose optimal value is maximized by solving a semidefinite programming problem. An advantage of our method is that the optimal value of the obtained relaxation has the same optimal value as the semidefinite relaxation used to compute it. In the second phase, we solve the initial problem by a branch-and-bound algorithm based on our tight relaxation. In this way, we take advantage of the efficiency of convex quadratic programming algorithms, as well as the strength of the semidefinite programming bounds.

We develop the solver `SMIQP (Solver of Mixed-Integer Quadratic Programs)` that implements the proposed algorithms for the quadratic case and is available online [1]. This software has allowed us to compare our algorithms on many instances of the literature, and it is largely competitive with the best existing solvers.

Furthermore, we study several applications. In particular in the field of electricity, where we have specialized our method to the problem of optimal power flows in an electrical network. We have also improved the solution of classical combinatorial optimization problems, such as the quadratic assignment problem.

Finally, after considering problems where the functions are quadratic, our work now focuses on problems where the degree of the function to be minimized can be arbitrary. Our first results concern the unconstrained binary case. We have proposed an exact algorithm, which after rewriting the objective function into a quadratic function, convexifies it by a dedicated approach. This algorithm allows us to address instances of an application in the field of physics that were not yet solved.

Keywords Non-linear Optimization, Global Optimization, Discrete Optimization, Semidefinite Programming, Experiments, Solver.

[1] https://github.com/amelie-lambert/SMIQP

Contents

Chapter 1

Introduction

Mixed Integer Optimization is a very active area of research. It provides a powerful framework for modeling and solving many problems in different application areas such as telecommunications, finance and engineering. Although tremendous progress has been made to efficiently solve optimization problems during the last decades, there are still some challenging theoretical and practical questions arising from more and more difficult problems. One such challenging problem is Mixed Integer Nonlinear Programming (MINLP) which consists in minimizing a nonlinear function over a feasible region described by nonlinear functions and where some of the variables are constrained to only take integer values. In recent years MINLP has received a sustained attention from the research community with the development of new theoretical results, algorithms and solvers.

In this manuscript, our first purpose is to solve exactly a rich subclass of MINLPs called Mixed-Integer Quadratically Constrained Programs (MIQCP). These are the class of MINLPs where the objective function to minimize and the constraints are all quadratic. Although, somewhat more simple than the general MINLP, MIQCP still constitutes a very broad and challenging class of problems. By itself, it has applications in a wide range of areas: chemical process design, optimal control, finance, telecommunications, and combinatorial optimization. Examples of applications in those areas are among others: the famous unit commitment problem or the pooling problem introduced by Haverly [4, 13, 78], the Markowitz portfolio-optimization problem [62] or the market prices computation [108]. Moreover, it designs graph theory problems [74], and in particular the Quadratic Assignment Problem (QAP), that consists in allocating n facilities to n locations while minimizing a quadratic cost [47]. In the pure continuous case, it also arises in several applications, including facility layout [150], package planning [46], chemical process planning [106], circle packing problems [107, 138], euclidean distance geometry [104], or triangulation problems [8]. Several applications of the box-constrained case, where the only constraints are the upper and lower bounds on the variables, were mentioned by Moré and Toraldo in [121]. It also models the Optimal Power Flow (OPF) [119], a fundamental problem in the domain of electricity networks, that consists in the determination of the power production at different points of an electricity network with a minimized production cost.

Moreover, MIQCP generalizes several difficult problems such as binary programming, fractional programming, or polynomial programming. This is due to the fact that these problems can be reformulated into a MIQCP by introducing additional variables, thus making MIQCP one of the most versatile optimization models. Methods are available for solving particular cases of MIQCP. If all functions are linear, we revert to Mixed Integer Linear Programming, which is already $\mathcal{NP}$-hard, but for which a large body of methods are very well developed. If we assume all

constraints to be linear the problem reduces to Mixed Integer Quadratic Programming for which a few number of methods have also been proposed. Finally, if the integer variables are restricted to be binary, we come back to Binary Quadratic Programming (BQP) that has itself a large set of applications.

If we assume all quadratic functions to be convex and to describe a convex feasible region, MIQCP is qualified as convex [123]. If moreover there are no integer variables, we have a convex problem that can be solved in polynomial time. Thanks to the later property, convex problems with integer variables can be solved by a branch-and-bound based on continuous relaxation. In these cases there exist fairly efficient solvers [30, 82, 69]. The main challenge of MIQCP is the combination of the combinatorial nature of the integer variables and the non-convexity of the objective function and constraints. If we assume all integer variables to be bounded MIQCP belongs to the class of $\mathcal{NP}$-hard problems (if integer variables are unbounded it is undecidable). Without convexity assumptions of its functions, problem MIQCP, even with only continuous variables, is significantly harder to solve. Without convexity the best solvers, with a fairly efficient implementation, are typically able to solve BQPs with a few dozens of variables and constraints (e.g. `Gurobi` [69] or `Cplex` [82]). If the integer variables have general bounds, it requires the introduction of additional variables making the problems even more difficult to solve. In this case, these solvers are able to solve smaller sized problems even when the objective function and the constraints are already convex functions.

Traditional approaches for solving MIQCP are based on branch-and-bound algorithms. They consist in a tree search based on two operations: bounding and branching. Branching strategies depend on the nature of the variables. For integer variables, branching is commonly done by recursively dividing the solution set into two subsets in such a way that a current fractional solution is discarded. For continuous variables, branching is done by considering a variable x_i whose current interval is $[\ell_i, u_i]$, choosing some value $\bar{x}_i$ in $]\ell_i, u_i[$, and dividing the solution set into two subsets according to $\bar{x}_i$. Doing this does not directly discard undesired points. However, when appropriate inequalities involving the bounds ℓ_i and u_i are added (e.g the McCormicks inequalities [111]), it may change the structure of the relaxed problem and improve the bound. This branching is called spatial branch-and-bound in the global optimization literature and is due to Falk and Soland [55]. There exist several bounding strategies. The main common is the *linearization* that rewrites MIQCP as a MILP [111, 136]. To do so, all products are removed from the problem usually at the expense of the addition of an important number of variables and constraints. The method has at least two major drawbacks. First, the resulting MILP is often very difficult to solve because the continuous relaxation bound is usually very poor. Second, if any of the initial functions are convex, the information is lost. A second approach is to build a *semidefinite relaxation* of MIQCP [10, 135]. The basic idea is the same as for the linearization, but the relations between the auxiliary variables and the initial ones are relaxed by use of one semidefinite constraint instead of linear inequalities. As a result, the quality of the relaxation is much improved. However, solving such a problem is slow and is thus not suitable for branch-and-bound algorithms where the relaxation has to be computed a large number of times. Another approach is the *convexification* that rewrites MIQCP into a problem that has convex quadratic objective function and constraints [75, 22, 29]. Generally, this is done by the definition of a parameterized equivalent convex formulation of MIQCP. Then, thanks to semidefinite programming, the parameters can be computed such that they optimize the bound. The main advantage is that the associated bound captures the strength of

semidefinite programming. The drawbacks are that convex MIQCP and semidefinite programming solvers are usually less developed than their counterparts for MILP.

Our second aim concerns a broader class of problems where the objective function can be any polynomial of binary variables. We thus consider Unconstrained Binary Polynomial (UBP) optimization that allows to formulate many important problems. For instance, the large class of satisfiability problems known as 3-SAT, can be formulated as a cubic optimization problem [90]. For higher degrees, there also exists many applications. For example, the construction of binary sequences with low aperiodic correlation [19] that is one of the most challenging problems in signal design theory. Problem (UBP) is $\mathcal{NP}$-hard in general [65]. As in the quadratic case, practical difficulties come from the non-convexity of the polynomial and of the binarity of its variables. In the case where the objective function is a polynomial, but the variables are continuous, Lasserre proposes in [99] an algorithm based on a hierarchy of semidefinite relaxations. The idea is, at each rank of the hierarchy, to successively tighten semidefinite relaxations in order to reach its optimal solution value. It is also proven in [99] that this hierarchy converges in a finite number of iterations to the optimal solution of the considered problem. Further, this work has been extended to hierarchies of second order conic programs [7, 67, 93], and of sparse doubly non-negative relaxation [83]. Although these algorithms were not originally tailored for binary programming, they can handle (UBP) by considering the quadratic constraint $x_i^2 = x_i$. Methods devoted to the binary polynomial case were also proposed. In [35, 100], the authors use separable or convex underestimators to approximate a given polynomial. Other methods based on linear reformulations can be found in [42, 61, 137, 36], in which linear equivalent formulations to (UBP) are proposed and then improved. Finally, some approaches are based on *quadratization*. We mention the works in [12, 33, 48] that focus on rewriting the polynomial into an equivalent quadratic function with a minimal number of auxiliary variables.

Obviously, solution methods of the more general class of MINLP are able to solve MIQCP or UBP. Conventional approaches are based on convex relaxations of the feasible solution set [17, 139, 140, 141, 154, 155], and some softwares are available, see for instance `Couenne` [18], `Baron` [133], or `Scip` [144]. We detail here the well known αbb method [5, 6, 9, 34, 58, 59, 60, 146, 145] that is based on the construction of a non-linear and convex underestimator of each initial function. For this purpose, each Hessian matrix is perturbed by an α parameter on its diagonal terms which allows to make it positive semidefinite. This leads to an increase in the value of the function which is linearly compensated in order to ensure the underestimation. The difficulty is then to calculate the value of the α parameter which allows to obtain a good underestimator. To this end, several methods have been proposed. They are implemented into the softwares `GloMIQO` [113, 114, 117] for the quadratic case and `ANTIGONE` [115] for the more general cases.

The first contribution of this thesis is an algorithm for solving MIQCP at ϵ-optimality. This method is called `Mixed Integer Quadratic Convex Reformulation (MIQCR)`, and is based on the concept of `Quadratic Convex Reformulation` (`QCR`) introduced in [22, 29]. It is based on a reformulation. By this, we mean an algorithm that works in two phases. The first phase consists in designing an equivalent formulation to the initial problem. The key idea of the method is that we can derive from

this equivalent formulation a tight convex relaxation that has a quadratic objective function and linear constraints. We show that using semidefinite programming to compute this relaxation provides a bound as good as the one of the semidefinite relaxation used to calculate it. In the second phase, the equivalent formulation can be solved to ϵ-global optimality by a branch-and-bound procedure based on this tight convex relaxation.

In the specific case where each non-convex relation between two variables involved at least one integer variable, MIQCR can be specialized to be more efficient. In that case, we are able to compute an equivalent formulation of the initial problem where the non-convexity only remains in the integrality constraints. Consequently, the second phase of MIQCR can be delegated to a standard MIQP solver as illustrated in Figure 1. In the general case, we are not yet able to compute an equivalent formulation where all the functions are convex. We thus compute an equivalent formulation where all the non-convexity is moved into a family of additional constraints, but also remains into the integrality constraints. Then, we design our own branch-and-bound process based on the sharp relaxation obtained by relaxing the non-convex constraints, as illustrated in Figure 2. In both cases, a key advantage of this approach over a classical branch-and-bound based on a semidefinite programming, is that we solve once a semidefinite program at the root node of the branch-and-bound algorithm.

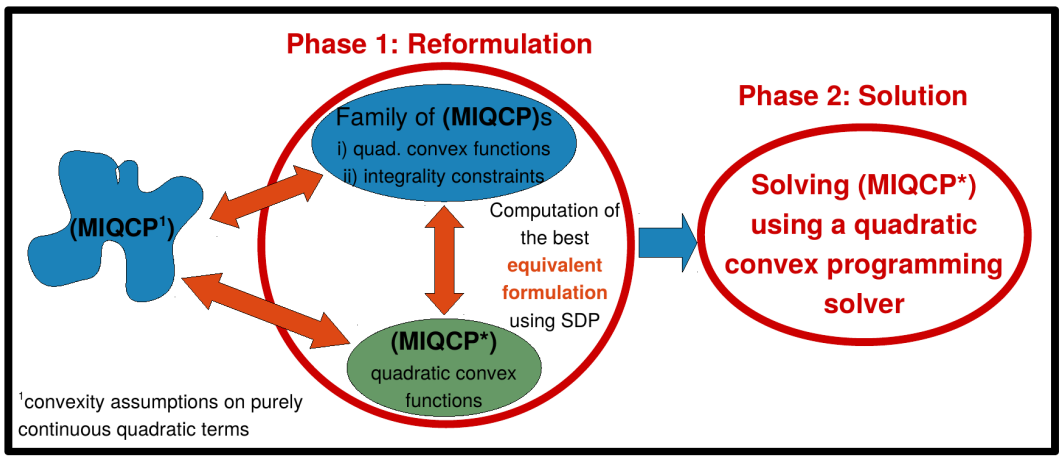


FIGURE 1: Method MIQCR in the mixed-pure integer case with convexity assumptions on purely continuous quadratic terms

The idea of convexifying MIQCPs in order to solve it to global optimality is not new. Indeed, the first convex reformulation method was introduced in the 70's by Hammer and Rubin [75] for the case where the variables are binary. This approach was then refined in method QCR [22, 29] where the authors took advantage of semidefinite programming to calculate optimal convex reformulations. The common feature between these methods, and somehow what limits them, is that they compute the coefficients of convex functions by perturbing only the diagonal terms of the Hessian matrix. The innovative idea of MIQCR is to allow any perturbation of the Hessian matrix by increasing the size of the reformulation. This contribution has unlocked two main challenges for this family of approaches. It first allowed a theoretical comparison with the literature. Indeed, while the original QCR method

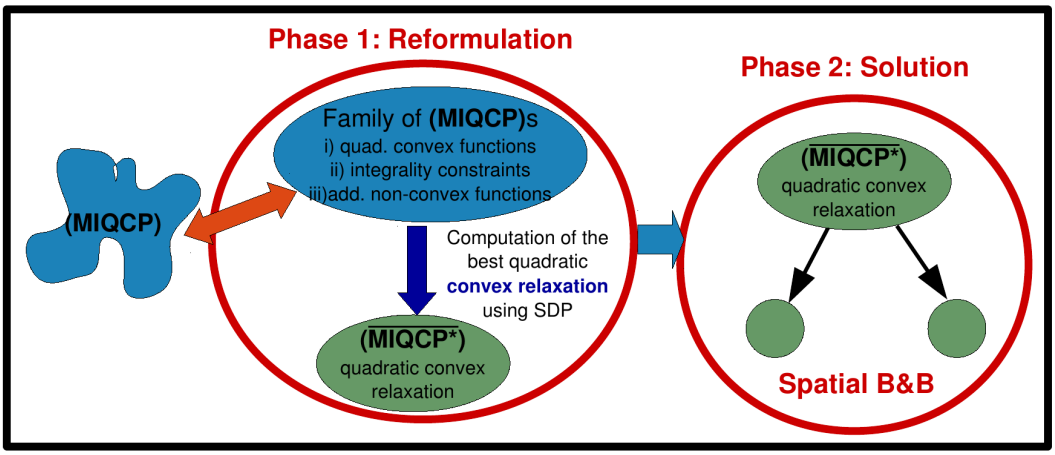


FIGURE 2: Method MIQCR in the general case

and the standard linearization have uncomparable bound values, a main result is that they can both be derived from MIQCR framework. This naturally induces a hierarchy in terms of relaxation bound values, where method MIQCR is the top of it. The other consequence is the scope of the problems handled by this family of methods. It is indeed the use of auxiliary variables that has allowed MIQCR to handle integer variables with general bounds, continuous variables and quadratic constraints.

In the rest of this manuscript, we detail the MIQCR method for the most general class of quadratic problems. We have chosen to present our different contributions in a unified way, since their combination gave rise to the final algorithm. In order to clarify the chronology of our results, we briefly describe in this introduction each significant contribution.

We start with a new linearization that can handle general integer variables, i.e. when $\ell \leq x \leq u$. A very well-known linearization for quadratic programs with binary variables is due to Fortet [61] and consists in replacing any product of two binary variables x_i and x_j by an additional variable Y_{ij}, together with a set of linear constraints enforcing the equality $Y_{ij} = x_i x_j$. The idea of linearization can be extended to quadratic programs with general bounded integer variables with a tricky binary expansion of the general integers. This is what we propose in the Binary Integer Linearization (BIL) [23, 77] method. We start by replacing each integer variable by its binary decomposition. Then, our main contribution is to replace in each product of two different integer variables only one of them by its binary decomposition. Thus, each integer product becomes an expression of products of a binary variable by an integer one. Finally, we linearize these new products by the standard mixed-integer linearization [111].

An alternative to linearization is convexification. The basic idea is to reformulate a non-convex quadratic program with binary variables into a convex quadratic program within the same space of variables. It appeared in Hammer and Rubin [75] where the authors use the equality $x_i^2 = x_i$ which holds for any binary variable x_i, and the smallest eigenvalue in order to shift the diagonal terms of the Hessian matrix of the objective function and obtain an equivalent convex problem. An improvement of this approach was considered in [22] where the authors perturb the diagonal terms of the Hessian matrix with non-uniform parameters. In particular, they prove

that the parameters that lead to the sharper continuous relaxation bound can be deduced from the optimal dual solution of the classical semidefinite relaxation. They also show that the continuous relaxation value of the equivalent formulation has the same optimal value as the classical semidefinite relaxation. The solution method was then extended in [29] to the case of binary quadratic programs with linear equalities. Here, the semidefinite relaxation whose solution gives the best reformulation also contains the Reformulation-Linearization Technique (RLT) constraints [136]. This well known algorithm is called `Quadratic Convex Reformulation (QCR)`. When variables are general integers, the equality $x_i^2 = x_i$ is no longer true. However, we can generalize the `QCR` method to this case by adding n new variables z_i that model x_i^2. Then, adding linear constraints coming from the linearization `BIL` we can enforce equalities $z_i = x_i^2$ by convex relations. Here again, other null quadratic functions can be built by squaring the linear equalities in order to get stronger relaxations. This method that we called `Compact Quadratic Convex Reformulation (CQCR)` [25] handles MIQCP restricted to general integer variables. An interesting result is that we come back to method `QCR` [29] when fixing the bounds ℓ to 0 and u to 1. Hence, method `CQCR` is a generalization of `QCR` to the general integer case.

The next extension we build in [24] was for the linearly constrained case where the variables can be integer or continuous, but with the restriction that in the objective function, all quadratic sub-functions of purely continuous variables are already convex. As in method `BIL`, we add new variables Y_{ij} that represent the product of two general integer variables x_i and x_j. These additional variables allow to widen the family of potential reformulations since any perturbation of each element of the Hessian matrix involving at least one integer variable is now considered. In this extension, the reformulation phase is based on an even stronger semidefinite relaxation, called "Shor plus RLT" in the literature [10]. The reformulated problem is a convex quadratic problem with continuous and binary variables, which can be solved by a quadratic convex programming solver. This extension was called `MIQCR`.

In order to improve the efficiency of method `MIQCR` in the pure integer case, we introduced in [26], a specific branch-and-bound algorithm to solve the equivalent formulation. Constraints $Y_{ij} = x_i x_j$ are not linearized but are rather enforced within a branch-and-bound procedure based on the relaxation of these quadratic constraints only, i.e. we keep the integrality constraints into our relaxation. The bound is thus tightened, but at the price of the solution of an integer program at each node of the branch-and-bound.

Then, we present in [27] `MIQCR-Quad` an extension to the case of problems with quadratic inequalities, with the same restriction on the quadratic sub-functions of purely continuous variables. The quadratic convex reformulation is presented in a new setting which includes linearization as a particular case. More precisely, the equivalent problem has additional variables Y_{ij}, additional quadratic constraints $Y_{ij} = x_i x_j$, a convex objective function and a set of valid inequalities. The latter quadratic equalities are linearized with method `BIL`, and the best equivalent formulation is computed with semidefinite programming. The obtained equivalent problem is a convex mixed-integer quadratic program that can be solved by a MIQP solver. In addition, we prove two interesting results. First, the initial quadratic inequalities can be linearized in an optimal reformulation. Second, for any potential convexification of initial quadratic equalities, the associated reformulation reaches the "best" bound.

Next, in [51], we designed the latest algorithm `MIQCR-BB` that handles any MIQCPs. Contrarily to the reformulation proposed in [24, 27], and following the ideas of [26], we keep in our reformulation the non-convex quadratic constraints $Y_{ij} = x_i x_j$. Then, to solve the reformulated problem, we design a spatial branch-and-bound algorithm based on the relaxation of constraints $Y_{ij} = x_i x_j$ and of the integrality constraints. This extension solves general MIQCPs to $\epsilon-$global optimality, with a quadratic convex programming bound value equals to the "Shor plus RLT" semidefinite relaxation.

When performing a spatial branch-and-bound, equalities $Y_{ij} = x_i x_j$ are often relaxed by use of the McCormick's envelopes that involved the lower and upper bounds (ℓ and u) on the original variables x. Then, since the branching rules consist in updating ℓ or u on a well chosen variable, these inequalities become tighter in the course of the branch-and-bound, and thus allow to improve the value of the relaxation at sub-nodes of the tree. In order to improve the behavior `MIQCR-BB`, we aim to design quadratic inequalities that are valid at the root node and depend on ℓ and u. In [31], the authors give a procedure to generate linear cuts involving general bounds, from the inequalities that describe the boolean quadratic polytope [37, 124]. In [95], we focus on the extension of the regular triangle inequalities to variables that belong to a generic interval, and propose a new methodology for this case. Then, we prove that our inequalities are as tight as the ones generated by the procedure described in [31]. Finally, in `MIQCR-T` we show how we can integrate them into our approach. From a general outlook, these inequalities can be used in any branch-and-bound process based on the relaxation of the Constraints $Y = xx^T$.

In each incremental version of method `MIQCR`, the equivalent formulation is computed thanks to the solution of a semidefinite relaxation. In practice, due to its size, the solution of this semidefinite problem often constitutes the bottleneck of `MIQCR`. However, once the equivalent formulation is computed, solving the obtained reformulated program is practical, since the continuous relaxation bound of the reformulation is tight. Hence, to handle larger instances we designed in [28, 95] a subgradient algorithm within a Lagrangian duality framework for solving approximately this large semidefinite program. Then, we parameterize our algorithm obtaining a dual heuristic that controls the size of the semidefinite relaxation used in the first phase of the algorithm, and in a sense the tightness of the associated relaxation. Finally, this algorithm can be viewed as a separation algorithm for selecting the most violated inequalities of the semidefinite relaxation used.

In addition to our theoretical contributions, we have implemented our algorithms in the software `SMIQP (Solution of Mixed-Integer Quadratic Programs)`, which is available online [94]. Thanks to `SMIQP`, we were able to quantify the significant improvement of the root node gap over other approaches from the literature. Finally, our experiments on several state-of-the-art benchmarks show that our software solves more challenging instances compared to the best existing solvers.

In Figure 3, we recall the different algorithms for exact solution of quadratic optimization problems described in this introduction. In particular, we present their hierarchy where for each arrow the method pointed indicates that it is a generalization of the method referred to its starting point.This relationship also holds with respect to the quality of the relaxation bound value. We also illustrate in this figure the scope of each method. Finally, the methods that we have developed allow us to

treat increasingly complex problems while keeping their generalization property in the hierarchy.

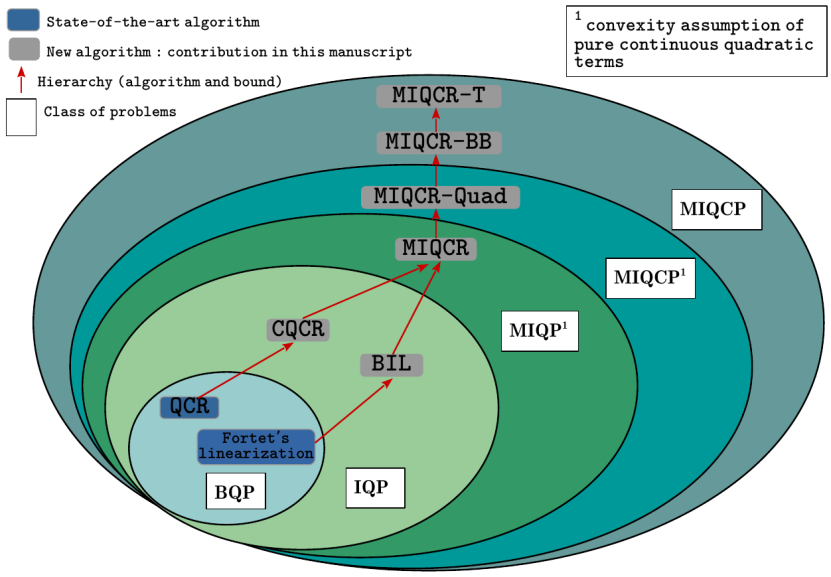


FIGURE 3: Hierarchy and scope of the algorithms of this thesis

Our second contribution is the specialization of MIQCR to classical combinatorial optimization problems or to applications from industry. We have thus specialized it to the Quadratic Assignment Problem (QAP) [47]. The results obtained in [50] show that by taking into account the structure of this problem, our method can be significantly faster. Moreover, one of the most competitive methods for solving these problems is an efficient implementation of MIQCR [127]. For real-world problems, our methods can gain in efficiency and scale by specializing to each problem addressed. This concerns in particular the results obtained for the Optimal Power Flow (OPF) [119]. This problem deals with the determination of the power production at different nodes of an electric network that minimizes a production cost. It can be formulated as a QCQP for which we have introduced two specializations of our method. In [49], we construct a convex quadratic relaxation as tight as the one obtained with MIQCR, but by solving the compact rank relaxation. Then in [96], again using the rank relaxation, we compute a relaxation that is equally tight, but much more compact than that of the previous approach.

The last contribution of this thesis is the extension of quadratic convex reformulation approaches to the case of Unconstrained Binary Polynomial Problems (UBP). As mentioned above, any polynomial problem can be reformulated as a MIQCP by adding new variables in a *quadratization* process. A direct solution approach is thus a third-phase algorithm: a quadratization phase where we reformulate the polynomial program into a MIQCP, and then the application of method MIQCR. Unfortunately, the large size of the MIQCP obtained after quadratization makes the whole method impracticable even for quite small instances. In [53], we thus design a more tricky convexification based algorithm that can handle unconstrained binary polynomial programs. In fact, we use the structure of the quadratization to compute a convex reformulation. This method is called Polynomial Quadratic Convex Reformulation (PQCR).

The manuscript is organized as follows. In Chapter 2, we presents an overview of method `MIQCR`. Then, Chapter 3 refers to the applications on which we have specialized our method. Next, we present in Chapter 4 algorithm `PQCR`, a quadratic convex reformulation method that handles unconstrained binary polynomial problems. Finally, Chapter 5 sums up our more significant results and gives further research directions.

Chapter 2

MIQCR, a global solution algorithm for quadratic programs

In this chapter, we present `Mixed Integer Quadratic Convex Reformulation (MIQCR)`, an algorithm that solves to $\epsilon-$global optimality MIQCPs. Such a problem can be formulated as (P):

$$(P)\begin{cases} \min f_0(x) \equiv \langle Q_0, xx^T\rangle + c_0^T x & \\ \text{subject to} & \\ \quad f_r(x) \equiv \langle Q_r, xx^T\rangle + c_r^T x \leq b_r & r \in \mathcal{R} \\ \quad \ell_i \leq x_i \leq u_i & i \in \mathcal{I} \\ \quad x_i \in \mathbb{N} & i \in \mathcal{J} \\ \quad x_i \in \mathbb{R} & i \in \mathcal{I}\backslash\mathcal{J} \end{cases}$$

with $\langle A, B\rangle = \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}b_{ij}$, and where $\mathcal{I} = \{1,\ldots,n\}$, $\mathcal{J} \subset \mathcal{I}$, $\mathcal{R} = \{1,\ldots,m\}$, $\forall\, r \in \{0\} \cup \mathcal{R}$, $(Q_r, c_r) \in \mathcal{S}_n \times \mathbb{R}^n$, $b \in \mathbb{R}^m$, and $u \in \mathbb{R}^n$. Without loss of generality we suppose that $\ell \in \mathbb{R}^n_+$. We assume the feasible domain of (P) to be non-empty. Problem (P) trivially contains the case where there are quadratic equalities, since an equality can be replaced by two inequalities. It also contains the case of linear constraints since a linear equality is a quadratic constraint with a zero quadratic part.

The method `MIQCR` works in two stages. The aim of the first phase is to construct a strong quadratic convex relaxation of (P). For this purpose, we introduce a parametrized family of equivalent formulations to (P) where the functions $f_r(x)$ are rewritten as convex functions. This rewriting leads to the addition of auxiliary variables and non-convex constraints. We then determine, within this family, the equivalent formulation whose convex relaxation, i.e. obtained by relaxing the additional non-convex constraints, is the tightest. We compute this optimal relaxation using semidefinite programming.

The second phase consists in solving (P) with a branch-and-bound algorithm based on this strong quadratic convex relaxation. We separate two cases. When (P) has only integer variables, we show that we can delegate the branch-and-bound to any miqp solver. For this, we rewrite (P) as an equivalent problem whose non-convexity remains only into the integrality constraints. In the general case, we describe a tailored branch-and-bound algorithm that exploits the integrality of the variables, but that amounts to a classical spatial branch-and-bound in the purely continuous case.

Defining a family of convex relaxation is the key point of method MIQCR. Obviously, the more valid inequalities are considered in this family, the more the associated bound will be sharp. In that context, we introduced quadratic inequalities that are valid for (P). We show that these inequalities are a generalization of the well known triangle inequalities to the case of general bounds ℓ and u. Moreover, by construction these inequalities involved the lower and upper bounds ℓ and u on the original variables x. Thus, the behavior of the branch-and-bound algorithm performed in the second phase may be improved since the branching strategies consist in moving ℓ and u.

From a theoretical point of view, one wants to consider in the first phase of method MIQCR a relaxation as sharp as possible. However, from the practical point of view, computing this relaxation implies the solution of a large semidefinite program. As expected, the more the relaxation will be sharp, the more the size of the semidefinite program will be large. In order to keep our whole algorithm practicable, we designed a tailored sub-gradient algorithm to solve this large semidefinite program. An important fact is that a feasible solution of this semidefinite relaxation is sufficient to convexify the functions. We thus parameterized our sub-gradient algorithm and obtained a dual heuristic. This heuristic has a double interest. First it solves the semidefinite program significantly faster to "nearly global optimality". Moreover, by construction, we bound the number of valid inequalities considered in the semidefinite program, and thus of the associated quadratic convex relaxation. As a consequence, the branch-and-bound algorithm of phase 2 solves at each node a quadratic convex problem with a reasonable size. In a sense, this heuristic parameterizes the tightness of the computed relaxation.

This chapter is organized as follows. In Section 2.1, we present the first phase of method MIQCR where we compute the best quadratic convex relaxation using semidefinite programming. Then, in Section 2.2, we describe, depending if we consider problem (P) in the pure integer or in the general case, how we can solve (P) to $\epsilon-$global optimality using this best quadratic convex relaxation. Further, in Section 2.3, we introduce valid quadratic inequalities. Finally, in Section 2.4, we present the sub-gradient algorithm for solving the semidefinite relaxation of Phase 1.

2.1 Building a tight quadratic convex relaxation [24, 25, 27, 51]

In this section, we describe the first phase of method MIQCR. First, we present a family of quadratic equivalent formulations to (P) that is parameterized by a set of positive semidefinite matrices. By construction, for each equivalent formulation of this family, it is easy to derive a convex quadratic relaxation. Then, we focus on finding the tightest quadratic convex relaxation within this family. We show that this best relaxation can be deduced from the optimal dual variables of a semidefinite relaxation of (P). In particular, we show that, among all the considered quadratic reformulations of the objective function and of the quadratic inequality constraints, in an optimal quadratic convex relaxation, the constraints can just be linearized and only the objective function remains quadratic. Further, we characterize other convex reformulations of the equality constraints that can be considered in an optimal quadratic convex relaxation. We prove another interesting result: for equality constraints, any linear or quadratic convex reformulation of these constraints can be used to build the best reformulation. Finally, we show that the general framework of MIQCR method can be viewed as a generalization of the standard linearization.

Designing a family of quadratic convex relaxations to (P)

We start by introducing a family of equivalent formulations to (P) from which we will derive a family of convex relaxations. To build these equivalent formulations, we introduce up to n^2 new variables Y to model the products $x_i x_j$. More formally, the n^2 new variables Y will satisfy :

$$Y_{ij} = x_i x_j \quad \forall (i,j) \in \mathcal{I}^2, \text{ or, equivalently } Y = xx^T$$

Then, for each $r = \{0\} \cup \mathcal{R}$, we consider a positive semidefinite matrix $S_r \in \mathcal{S}_n^+$ and we formulate $f_r(x)$ as a sum of a quadratic function of the x variables and a linear function of the Y variables $f_{r,S_r}(x,Y)$. We define function $f_{r,S_r}(x,Y)$ as follows:

$$f_{r,S_r}(x,Y) = \langle S_r, xx^T \rangle + c_r^T x + \langle Q_r - S_r, Y \rangle \qquad \forall\ r\{0\} \cup \mathcal{R}$$

It is easy to see that each new function $f_{r,S_r}(x,Y)$ is equal to $f_r(x)$ under the condition $Y = xx^T$. We can now replace each function $f_r(x)$ in problem (P) by its associated reformulated function $f_{r,S_r}(x,Y)$, and we obtain a family of equivalent problems to (P). We call $(P_{S_0,\ldots,S_m})$ this family that is parameterized by the set of positive semidefinite matrices $S_0,\ldots,S_m$. Hence, problem (P) can be equivalently stated as:

$$(P_{S_0,\ldots,S_m}) \begin{cases} \min \quad f_{0,S_0}(x,Y) \equiv \langle S_0, xx^T \rangle + c_0^T x + \langle Q_0 - S_0, Y \rangle & \\ \text{s.t.} & \\ \quad f_{r,S_r}(x,Y) \equiv \langle S_r, xx^T \rangle + c_r^T x + \langle Q_r - S_r, Y \rangle \le b_r & r \in \mathcal{R} \\ \quad Y_{ij} = x_i x_j & (i,j) \in \mathcal{I}^2 \\ \quad \ell_i \le x_i \le u_i & i \in \mathcal{I} \\ \quad x_i \in \mathbb{N} & i \in \mathcal{J} \\ \quad x_i \in \mathbb{R} & i \in \mathcal{I} \backslash \mathcal{J} \end{cases}$$

Because, by construction, matrices $S_0,\ldots S_m$ are positive semidefinite the new objective function $f_{0,S_0}(x,Y)$ and the reformulated quadratic constraints $f_{r,S_r}(x,Y)$ are convex functions. Thus, each equivalent formulation $(P_{S_0,\ldots,S_m})$ has the property that, when Constraints $x_i \in \mathbb{N}$ and $Y = xx^T$ are relaxed, it is a convex problem. Then, it is now easy to build a family of quadratic convex relaxations to (P), by relaxing the latter constraints into the well known McCormick envelopes [111] that are described in set $\mathcal{MC}$:

$$\mathcal{MC} = (x,Y) \begin{cases} Y_{ii} \ge x_i & i \in \mathcal{J} \\ Y_{ij} \le u_j x_i + \ell_i x_j - u_j \ell_i & (i,j) \in \mathcal{U} \\ Y_{ij} \le u_i x_j + \ell_j x_i - u_i \ell_j & (i,j) \in \mathcal{U} \\ Y_{ij} \ge u_j x_i + u_i x_j - u_i u_j & (i,j) \in \mathcal{U} \\ Y_{ij} \ge \ell_j x_i + \ell_i x_j - \ell_i \ell_j & (i,j) \in \mathcal{U} \\ Y_{ji} = Y_{ij} & (i,j) \in \overline{\mathcal{U}} \\ x_i \in \mathbb{R} & i \in \mathcal{I} \\ Y_{ij} \in \mathbb{R} & (i,j) \in \mathcal{I}^2 \end{cases}$$

where $\mathcal{U} = \{(i,j) \in \mathcal{I}^2 : i \le j\}$, where $\overline{\mathcal{U}} = \{(i,j) \in \mathcal{I}^2 : i < j\}$, and Constraints $Y_{ii} \ge x_i$ come from $x_i^2 \ge x_i$, a valid inequality for any integer variable.

Finally, we obtain $(\overline{P}_{S_0,\dots,S_m})$ a family of quadratic convex relaxations to (P):

$$(\overline{P}_{S_0,\dots,S_m}) \begin{cases} \min \quad \langle S_0, xx^T \rangle + c_0^T x + \langle Q_0 - S_0, Y \rangle \\ \text{s.t.} \\ \quad \langle S_r, xx^T \rangle + c_r^T x + \langle Q_r - S_r, Y \rangle \leq b_r \qquad r \in \mathcal{R} \\ \quad (x, Y) \in \mathcal{MC} \end{cases}$$

$(\overline{P}_{S_0,\dots,S_m})$ includes two extreme cases. The first one is $S_r = Q_r$, when all Q_r matrices are already positive semidefinite. In that case, functions $f_r(x)$ are left unchanged. The second one is when all S_r are zero-matrices. In this case, the reformulation consists in replacing each product of two x variables by a Y variable. This amounts to a complete linearization.

Computing the best quadratic and convex relaxation $(\overline{P}^*)$

We then consider the problem of finding a best set of positive semidefinite matrices $S_0^*, \dots, S_m^*$, in the sense that the optimal solution value of $(\overline{P}_{S_0^*,\dots,S_m^*})$ is as large as possible. By denoting $v(P)$ the optimal value of problem (P), this problem amounts to solving the following problem (OPT_S):

$$(OPT_S) \begin{cases} \max \limits_{S_0,\dots,S_m \succeq 0} \quad v(\overline{P}_{S_0,\dots,S_m}) \end{cases}$$

Theorem 1 states that $v(OPT_S)$ is equal to the optimal value of a semidefinite program which is known as the "Shor plus RLT" semidefinite relaxation of (P). Moreover, it characterizes an optimal set of semidefinite matrices. We called (SDP) the "Shor plus RLT" semidefinite relaxation of (P):

$$(SDP) \begin{cases} \min f(X,x) \equiv \langle Q_0, X \rangle + c_0^T x & & \\ \text{s.t.} & & \\ \langle Q_r, X \rangle + c_r^T x \leq b_r & r \in \mathcal{R} \leftarrow \alpha_r & (1) \\ X_{ii} \geq x_i & i \in \mathcal{J} \leftarrow \varphi_i & (2) \\ X_{ij} \leq u_j x_i + \ell_i x_j - u_j \ell_i & (i,j) \in \mathcal{U} \leftarrow \phi_{ij}^1 & (3) \\ X_{ij} \leq u_i x_j + \ell_j x_i - u_i \ell_j & (i,j) \in \mathcal{U} \leftarrow \phi_{ij}^2 & (4) \\ X_{ij} \geq u_j x_i + u_i x_j - u_i u_j & (i,j) \in \mathcal{U} \leftarrow \phi_{ij}^3 & (5) \\ X_{ij} \geq \ell_j x_i + \ell_i x_j - \ell_i \ell_j & (i,j) \in \mathcal{U} \leftarrow \phi_{ij}^4 & (6) \\ \begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix} \succeq 0 & \leftarrow \rho & \\ x \in \mathbb{R}^n \quad X \in \mathcal{S}_n & & \end{cases}$$

Theorem 1 *It holds that $v(OPT_S) = v(SDP)$. Moreover, an optimal solution $(S_0^*, \dots, S_m^*)$ of (OPT_S) can be built as follows:*

i) $\forall r \in \mathcal{R}$, $S_r^* = \mathbf{0}_n$ *(i.e. we linearize the initial quadratic constraints)*

ii) $S_0^* = Q_0 + \sum_{r=1}^{m} \alpha_r^* Q_r + \Phi^*$ *where:*

$\diamond$ α^* *is the vector of optimal dual variables associated with Constraints (1),*

⋄ *matrix* $\Phi^* = \Phi^{1*} + \Phi^{2*} - \Phi^{3*} - \Phi^{4*} - \text{diag}(\varphi^*)$, *where* φ^* *is the vectors of dual variables associated with Constraints (2), and* Φ^{1*}, Φ^{2*}, Φ^{3*}, *and* Φ^{4*} *are the symmetric matrices built from the optimal dual variables* ϕ^t_{ij}, $t = 1, \ldots, 4$ *associated with Constraints (3)–(6). More precisely, if* ϕ^1_{ij} *is the dual variable associated to constraint (3), then* $\Phi^1_{ij} = \Phi^1_{ji} = \frac{\phi^1_{ij}}{2}$ *for* $i < j$, *and* $\Phi^1_{ii} = \phi^1_{ii}$.

Proof.
⋄ To prove that $v(OPT_S) \leq v(SDP)$, we show that for any $\bar{S}_0, \ldots, \bar{S}_m \in \mathcal{S}^+_n$, we have $v(\overline{P}_{\bar{S}_0,\ldots,\bar{S}_m}) \leq v(SDP)$, which in turn implies that $v(OPT_S) \leq v(SDP)$ since the right hand side is constant. For this, we show that, if $(\bar{x}, \bar{X})$ is feasible for (SDP), then $(x, Y) := (\bar{x}, \bar{X})$ is $i)$ feasible for $(\overline{P}_{\bar{S}_0,\ldots,\bar{S}_m})$ and $ii)$ its objective value is less or equal than $v(SDP)$. Since $(\overline{P}_{\bar{S}_0,\ldots,\bar{S}_m})$ is a minimization problem, $v(\overline{P}_{\bar{S}_0,\ldots,\bar{S}_m}) \leq v(SDP)$ follows.

i) We prove that (x, Y) is feasible to $(\overline{P}_{\bar{S}_0,\ldots,\bar{S}_m})$. Obviously $(x, Y) \in \mathcal{MC}$, we have to prove that $f_{r,\bar{S}_r}(x, Y) \leq b_r$:

$$\begin{aligned}\langle \bar{S}_r, xx^T\rangle + c_r^T x + \langle Q_r - \bar{S}_r, Y\rangle &= \langle \bar{S}_r, \bar{x}\bar{x}^T\rangle + c_r^T \bar{x} + \langle Q_r - \bar{S}_r, \bar{X}\rangle \\ &= \langle \bar{S}_r, \bar{x}\bar{x}^T - \bar{X}\rangle + c_r^T \bar{x} + \langle Q_r, \bar{X}\rangle \leq b_r\end{aligned}$$

from Constraints (1) and (7), and since $\bar{S}_r \succeq 0$.

ii) Let us compare the objective values. For this, we prove that $\langle \bar{S}_0, \bar{x}\bar{x}^T\rangle + c_0^T \bar{x} + \langle Q_0 - \bar{S}_0, \bar{X}\rangle - \langle Q_0, \bar{X}\rangle - c_0^T \bar{x} \leq 0$ or that $\langle \bar{S}_0, \bar{x}\bar{x}^T - \bar{X}\rangle \leq 0$. This last inequality follows from $\bar{S}_0 \succeq 0$ and Constraint (7).

⋄ Let us secondly prove that $v(OPT_S) \geq v(SDP)$ or equivalently $v(OPT_S) \geq v(D)$ where (D) is the dual of (SDP):

$$(D)\begin{cases}
\max\ g(\alpha, \Phi, \rho) = -\sum_{r=1}^m \alpha_r b_r + \sum_{i=1}^n \sum_{j=i}^n \phi^1_{ij} \ell_i u_j + \sum_{i=1}^n \sum_{j=i}^n \phi^2_{ij} u_i \ell_j - \sum_{i=1}^n \sum_{j=i}^n \phi^3_{ij} u_i u_j - \sum_{i=1}^n \sum_{j=i}^n \phi^4_{ij} \ell_i \ell_j - \rho & \\
\text{s.t.} & \\
\quad S = Q_0 + \sum_{r=1}^m \alpha_r Q_r + \Phi - \text{diag}(\varphi) & (7)\\
\quad d = c_0 + \sum_{r=1}^m \alpha_r c_r - (U(\phi^1) + L(\phi^2) - (U(\phi^3) + L(\phi^3)))^T u - (L(\phi^1) + U(\phi^2) - (U(\phi^4) + L(\phi^4)))^T \ell + \varphi & (8)\\
\quad \Phi^t_{ij} = \Phi^t_{ji} = \frac{\phi^t_{ij}}{2} \qquad (i,j) \in \overline{U}, t = 1, \ldots, 4 & (9)\\
\quad \Phi^t_{ii} = \phi^t_{ii} \qquad i \in \mathcal{J} & (10)\\
\quad \Phi = \Phi^1 + \Phi^2 - \Phi^3 - \Phi^4 & (11)\\
\quad \begin{pmatrix} \rho & \frac{1}{2} d^T \\ \frac{1}{2} d & S \end{pmatrix} \succeq 0 & (12)\\
\quad \alpha \in \mathbb{R}^m_+,\ \Phi \in \mathcal{S}_n,\ \varphi \in \mathbb{R}^n_+,\ \phi^t \in \mathbb{R}^{n(n+1)/2}_+, t = 1, \ldots, 4.
\end{cases}$$

where $\rho \in \mathbb{R}_+$ is the dual variable associated to constraint (7), $\alpha \in \mathbb{R}^m_+$ are the dual variables associated to constraints (1), $\varphi \in \mathbb{R}^n_+$ are the dual variables associated to constraints (2) and $\phi^t \in \mathbb{R}^{n(n+1)/2}_+$, $t = 1, \ldots, 4$ are the dual variable associated with constraints (3)–(6), respectively. We denote by $U(M)$ ($L(M)$ resp.) the upper (lower, resp.) triangle matrix build from the coefficient of M.

Let $(\alpha^*, \Phi^*, \rho^*)$ be an optimal solution to (D), we build the following positive semidefinite matrices: $\forall r \in \mathcal{R}$, $\bar{S}_r = \mathbf{0}_n$, and $\bar{S}_0 = S^* = Q_0 + \sum_{r=1}^{m} \alpha_r^* Q_r + \Phi^*$. By Constraint (12), $\bar{S}_0 \succeq 0$, and $(\bar{S}_0, \ldots, \bar{S}_m)$ forms a feasible solution to (OPT_S). The objective value of this solution is equal to $v(\overline{P}_{\bar{S}_0,\ldots,\bar{S}_m})$.

We now prove that $v(\overline{P}_{\bar{S}_0,\ldots,\bar{S}_m}) \geq v(D)$. For this, we prove that for any feasible solution $(\bar{x}, \bar{Y})$ to $(\overline{P}_{\bar{S}_0,\ldots,\bar{S}_m})$, the associated objective value is not smaller than $g(\alpha^*, \Phi^*, \rho^*)$. Denote by Δ the difference between the objective values, i.e., $\Delta = \langle \bar{S}_0, \bar{x}\bar{x}^T \rangle + c_0^T \bar{x} + \langle Q_0 - \bar{S}_0, \bar{Y} \rangle - g(\alpha^*, \Phi^*, \rho^*)$. We below prove that $\Delta \geq 0$.

$$
\begin{aligned}
\Delta &= \langle \bar{S}_0, \bar{x}\bar{x}^T \rangle + c_0^T \bar{x} + \langle Q_0 - \bar{S}_0, \bar{Y} \rangle + \sum_{r=1}^{m} \alpha_r^* b_r - \sum_{i=1}^{n}\sum_{j=i}^{n} \phi_{ij}^1 \ell_i u_j - \sum_{i=1}^{n}\sum_{j=i}^{n} \phi_{ij}^2 u_i \ell_j \\
&\quad + \sum_{i=1}^{n}\sum_{j=i}^{n} \phi_{ij}^3 u_i u_j + \sum_{i=1}^{n}\sum_{j=i}^{n} \phi_{ij}^4 \ell_i \ell_j + \rho^* \\
&= \langle \bar{S}_0, \bar{x}\bar{x}^T \rangle + c_0^T \bar{x} - \langle \sum_{r=1}^{m} \alpha_r^* Q_r + \Phi^*, \bar{Y} \rangle + \sum_{r=1}^{m} \alpha_r^* b_r - \sum_{i=1}^{n}\sum_{j=i}^{n} \phi_{ij}^1 \ell_i u_j - \sum_{i=1}^{n}\sum_{j=i}^{n} \phi_{ij}^2 u_i \ell_j \\
&\quad + \sum_{i=1}^{n}\sum_{j=i}^{n} \phi_{ij}^3 u_i u_j + \sum_{i=1}^{n}\sum_{j=i}^{n} \phi_{ij}^4 \ell_i \ell_j + \rho^* \\
&\text{since } Q_0 - \bar{S}_0 = -(\sum_{r=1}^{m} \alpha_r^* Q_r + \Phi^*) \\
&= \langle \bar{S}_0, \bar{x}\bar{x}^T \rangle + c_0^T \bar{x} + \sum_{r=1}^{m} \alpha_r^* (b_r - \langle Q_r, \bar{Y} \rangle) - \langle \Phi^*, \bar{Y} \rangle - \sum_{i=1}^{n}\sum_{j=i}^{n} \phi_{ij}^1 \ell_i u_j - \sum_{i=1}^{n}\sum_{j=i}^{n} \phi_{ij}^2 u_i \ell_j \\
&\quad + \sum_{i=1}^{n}\sum_{j=i}^{n} \phi_{ij}^3 u_i u_j + \sum_{i=1}^{n}\sum_{j=i}^{n} \phi_{ij}^4 \ell_i \ell_j + \rho^* \\
&\geq \langle \bar{S}_0, \bar{x}\bar{x}^T \rangle + c_0^T \bar{x} + \sum_{r=1}^{m} \alpha_r^* c_r^T \bar{x} - \langle \Phi^*, \bar{Y} \rangle - \sum_{i=1}^{n}\sum_{j=i}^{n} \phi_{ij}^1 \ell_i u_j - \sum_{i=1}^{n}\sum_{j=i}^{n} \phi_{ij}^2 u_i \ell_j \\
&\quad + \sum_{i=1}^{n}\sum_{j=i}^{n} \phi_{ij}^3 u_i u_j + \sum_{i=1}^{n}\sum_{j=i}^{n} \phi_{ij}^4 \ell_i \ell_j + \rho^*
\end{aligned}
$$

as $c_r^T \bar{x} + \langle Q_r, \bar{Y} \rangle \leq b_r$ and $\alpha_r^* \geq 0$. Moreover, by Constraint (11) we get:

$$
\begin{aligned}
\Delta &\geq \langle \bar{S}_0, \bar{x}\bar{x}^T \rangle + c_0^T \bar{x} + \sum_{r=1}^{m} \alpha_r^* c_r^T \bar{x} - \langle \phi^1 + \phi^2 - \phi^3 - \phi^4 - diag(\varphi), \bar{Y} \rangle - \sum_{i=1}^{n}\sum_{j=i}^{n} \phi_{ij}^1 \ell_i u_j \\
&\quad - \sum_{i=1}^{n}\sum_{j=i}^{n} \phi_{ij}^2 u_i \ell_j + \sum_{i=1}^{n}\sum_{j=i}^{n} \phi_{ij}^3 u_i u_j + \sum_{i=1}^{n}\sum_{j=i}^{n} \phi_{ij}^4 \ell_i \ell_j + \rho^*
\end{aligned}
$$

By constraints of set $\mathcal{MC}$, and since all coefficients ϕ_{ij}^t are non-negative, we get:

$$\begin{aligned}\Delta &\geq \langle \bar{S}_0, \bar{x}\bar{x}^T\rangle + \Big(c_0 + \sum_{r=1}^{m} \alpha_r c_r - (U(\phi^1) + L(\phi^2) - (U(\phi^3) + L(\phi^3)))^T u \\ &\qquad - (L(\phi^1) + U(\phi^2) - (U(\phi^4) + L(\phi^4)))^T \ell + \varphi\Big)^T \bar{x} + \rho^* \\ &= \langle \bar{S}_0, \bar{x}\bar{x}^T\rangle + d^{*T}\bar{x} + \rho^*\end{aligned}$$

We end the proof by showing that $\langle \bar{S}_0, \bar{x}\bar{x}^T\rangle + d^{*T}\bar{x} + \rho^* \geq 0$. From Constraint (12), we know that for all $x \in \mathbb{R}^n$, $\begin{pmatrix} 1 \\ x \end{pmatrix}^T \begin{pmatrix} \rho^* & \frac{1}{2}d^{*T} \\ \frac{1}{2}d^* & \bar{S}_0 \end{pmatrix} \begin{pmatrix} 1 \\ x \end{pmatrix} \geq 0$, which prove that $\Delta \geq 0$.

□

We call $(\overline{P}^*)$ our optimal quadratic convex relaxation of (P) that has linear constraints:

$$(\overline{P}^*) \begin{cases} \min & f_{0,S_0^*}(x,Y) \equiv \langle Q_0 + \sum_{r=1}^{m} \alpha_r^* Q_r + \Phi^*, xx^T\rangle + c_0^T x - \langle \sum_{r=1}^{m} \alpha_r^* Q_r + \Phi^*, Y\rangle \\ \text{s.t.} & \\ & f_{r,S_r^*}(x,Y) \equiv \langle Q_r, Y\rangle + c_r^T x + \leq b_r \qquad \forall r \in \mathcal{R} \\ & (x,Y) \in \mathcal{MC} \end{cases}$$

We describe here how we deal with the presence of a linear inequalities $a_r^T x \leq b_r$ in (P). Indeed, in Theorem 1 these inequalities are not taken into account in the computation of $(\overline{P}^*)$. A simple way to consider these inequalities into the convexification process is to add the valid RLT quadratic inequality $\sum_{r\in\mathcal{R}} x^T a_r a_r^T x \leq b_r^2$ to (P) in a pre-processing phase. They then will be handled as any initial quadratic inequality of problem (P).

Two advantages of method `MIQCR` are in order:

i) the optimal value of $(\overline{P}^*)$ is equal to the optimal value of (SDP). Hence, in a branch-and-bound algorithm where the bounding step is the solution of $(\overline{P}^*)$, the root-node gap is the same as the SDP-relaxation gap which is known to be strong. In [10, 17] it is recalled that (SDP) dominates six other considered semidefinite relaxations. It is further shown that it provides the same bound as the doubly non-negative relaxation.

ii) $(\overline{P}^*)$ is a quadratic convex program with linear constraints that can be solved efficiently by standard miqp solvers for medium size instances.

In Theorem 1, we state that we can linearize the inequality constraints to reach the "Shor plus RLT" semidefinite bound. In the next section, we prove a stronger result : any convexification of the equality constraints leads to a relaxation that reaches this best bound.

Which convexifications work for the equality constraints ?

In method MIQCR the fact that we linearize the constraints is an advantage from the computational point of view since efficient mixed-integer quadratic convex solvers are available. However, if we consider the size of the relaxation, the complete linearization of all constraints requires to introduce up to n^2 new variables Y_{ij}. Then, a natural question arises: is-it possible to reach the "Shor plus RLT" semidefinite bound value by convexifying differently the constraints ? In this section, we answer to this question for the equality case.

Here, we study the case where the initial problem contains some quadratic equality constraints. Without loss of generality, we consider the case of just one equality, $\langle Q_1, xx^T\rangle + c_1^T x = b_1$, which corresponds to the first two inequalities. More formally, the initial quadratic constraints in (P) are precisely:

$$\begin{cases} \langle Q_1, xx^T\rangle + c_1^T x \leq b_1 \\ \langle -Q_1, xx^T\rangle - c_1^T x \leq -b_1 \\ \langle Q_r, xx^T\rangle + c_r^T x \leq b_r \qquad r \in \mathcal{R}\backslash\{1,2\} \end{cases}$$

We start by the following observation. In an optimal quadratic convex relaxation described by Theorem 1, we have:

$$S_0^* = Q_0 + (\alpha_1^* - \alpha_2^*)Q_1 + \sum_{r=3}^{m} \alpha_r^* Q_r + \Phi^*$$

and thus we get the following objective function of $(\overline{P}^*)$:

$$\begin{aligned} f_{0,S_0^*}(x,Y) &= \langle Q_0 + (\alpha_1^* - \alpha_2^*)Q_1 + \sum_{r=3}^{m} \alpha_r^* Q_r + \Phi^*, xx^T\rangle + c_0^T x \\ &\quad - \langle (\alpha_1^* - \alpha_2^*)Q_1 + \sum_{r=3}^{m} \alpha_r^* Q_r + \Phi^*, Y\rangle \\ &= \langle Q_0, xx^T\rangle + c_0^T x + \langle \sum_{r=3}^{m} \alpha_r^* Q_r + \Phi^*, xx^T - Y\rangle + (\alpha_1^* - \alpha_2^*)\Big(\langle Q_1, xx^T - Y\Big) \end{aligned}$$

and since $\langle Q_1, Y\rangle + c_1^T x = b_1$

$$f_{0,S_0^*}(x,Y) = \langle Q_0, xx^T\rangle + c_0^T x + \langle \sum_{r=3}^{m} \alpha_r^* Q_r + \Phi^*, xx^T - Y\rangle + (\alpha_1^* - \alpha_2^*)\Big(\langle Q_1, xx^T\rangle + c_1^T x - b_1\Big)$$

We recognize the initial equality constraint multiplied by an unsigned scalar parameter $\nu = \alpha_1^* - \alpha_2^*$. Hence, if equality constraints are considered in the initial formulation and when we build the optimal solution $(S_0^*, \ldots, S_m^*)$ as in Theorem 1, it amounts to explicitly integrating equality constraints into the objective function multiplied by a scalar parameter.

We now consider a slightly different reformulation scheme where the equality constraint is lifted in the objective function weighted by a scalar ν. We thus build the

following convex relaxation $(\overline{P}_{S_0,S_1,S_1',S_3,\ldots,S_m,\nu})$:

$$\begin{cases} \min f_{0,S_0,\nu}(x,Y) \equiv \langle S_0, xx^T\rangle + c_0^T x + \langle Q_0 - S_0, Y\rangle + \nu\left(\langle Q_1, xx^T\rangle + c_1^T x - b_1\right) \\ \text{s.t.} \\ \quad \langle S_1, xx^T\rangle + c_1^T x + \langle Q_1 - S_1, Y\rangle \leq b_1 & (13) \\ \quad \langle S_1', xx^T\rangle - c_1^T x - \langle Q_1 + S_1', Y\rangle \leq -b_1 & (14) \\ \quad \langle S_r, xx^T\rangle + c_r^T x + \langle Q_r - S_r, Y\rangle \leq b_r \qquad \forall r \in \mathcal{R} \\ \quad (x,Y) \in \mathcal{MC} \end{cases}$$

Finding the best relaxation within this new scheme amounts to solving problem $(OPT_{S,\nu})$:

$$(OPT_{S,\nu}) \begin{cases} \max\limits_{\nu\in\mathbb{R},\ S_0+\nu Q_1\succeq 0} \\ S_1,S_1',S_3\ldots,S_m\succeq 0 \end{cases} v(\overline{P}_{S_0,S_1,S_1',S_3,\ldots,S_m,\nu})$$

We state in Theorem 2 that when quadratic equalities are explicitly integrated into the objective function, in an optimal solution to $(OPT_{S,\nu})$, matrices S_1 and S_1' associated to the equality constraint can be any positive semidefinite matrices.

Theorem 2 *Let $(\nu^*, S_0^*, S_1^*, S_1'^*, S_3^*, \ldots, S_m^*)$ be an optimal solution to $(OPT_{S,\nu})$, then for any semi-definite matrices $\bar{S}_1$, $\bar{S}_1'$, the solution $(\nu^*, S_0^*, \bar{S}_1, \bar{S}_1', S_3^*, \ldots, S_m^*)$ is also an optimal solution to $(OPT_{S,\nu})$.*

Proof sketch. The proof is based on the equivalence between $(OPT_{S,\nu})$ and the following problem:

$$(OPT'_{S,\nu}) \begin{cases} \max\limits_{\nu\in\mathbb{R},\ S_0+\nu Q_1\succeq 0} \\ S_3\ldots,S_m\succeq 0 \end{cases} v(\overline{P}'_{S_0,S_3,\ldots,S_m,\nu})$$

where $(\overline{P}'_{S_0,S_3,\ldots,S_m,\nu})$ is $(\overline{P}_{S_0,S_1,S_1',S_3,\ldots,S_m,\nu})$ without Constraints (13) and (14). We have:

$\diamond$ $v(OPT_{S,\nu}) \geq v(OPT'_{S,\nu})$ since $v(\overline{P}_{S_0,S_1,S_1',S_3,\ldots,S_m,\nu}) \geq v(\overline{P}'_{S_0,S_3,\ldots,S_m,\nu})$.

$\diamond$ $v(OPT_{S,\nu}) \leq v(OPT'_{S,\nu})$ can be proved by showing that $v(OPT_{S,\nu}) = v(DOPT_{S,\nu}) \leq v(OPT'_{S,\nu})$, where $(DOPT_{S,\nu})$ is obtained by dualizing Constraints (13) and (14), and the equality comes from convexity of both problems.

□

Let us now state Corollary 1 that shows that for any equivalent reformulated equality constraints the value of the best reformulation is reached.

Corollary 1 $v(OPT_{S,\nu}) = v(SDP) = v(OPT_S)$

Proof. We know by Theorem 1 that $v(SDP) = v(OPT_S)$. Moreover, by convexity and by dualizing the first two inequalities of (SDP) which are equivalent to the equality $\langle Q_1, X\rangle + c_1^T x = b_1$, we obtain $v(SDP) = v(OPT'_{S,\nu})$. Thus, as $v(OPT_{S,\nu}) = v(OPT'_{S,\nu})$ by Theorem 2, we obtain $v(OPT_{S,\nu}) = v(SDP)$. □

Thus, in some specific cases, when for instance the matrix $Q_0 - S_0^*$ is sparse, convexify the objective function can require to introduce a few variables Y_{ij}. In that

case, it could be interesting from the computational point of view to first replace each inequality constraint by an equality using a slack variable. Then, by explicitly integrating these equalities into the objective function, one can convexify them by perturbing, for instance, only their diagonal terms, and thus by the addition of much less variables in comparison to a complete linearization.

A generalization of the classical linearization

One can note the generality of our first phase that works with any set of positive semidefinite matrices $S_0, \ldots, S_m$. As mentioned in the introduction, branch-and-bound algorithms developed to solve (P) are classically based on complete linearization which corresponds to reformulation $(\overline{P}_{S_0,\ldots,S_m})$ where we set all matrices to $\mathbf{0}_n$. Hence, a valuable contribution is that this general scheme includes linearization of the objective function and of the constraints as a particular reformulation. From this remark, we can deduce Corollary 2.

Corollary 2 *Take the following feasible solution to (OPT_S) that amounts to the complete linearization of (P):*

$$\bar{S}_r = \mathbf{0}_n \qquad r \in \{0\} \cup \mathcal{R}$$

Obviously, we have $v(\overline{P}_{\bar{S}_0,\ldots,\bar{S}_m}) \leq v(\overline{P}^*)$. In other words, the bound obtained by the complete linearization is weaker than the bound we get with the solution of $(\overline{P}^*)$.

Method MIQCR also generalizes other methods that we design, and that we shortly described in the introduction. In particular, the Binary Integer Linearization (BIL) [23], and the Compact Quadratic Convex Reformulation (CQCR) [26].

2.2 Spatial branch-and-bounds based on our relaxation [23, 24, 26, 27, 51]

In this section, we present the second phase of method MIQCR, where we distinguish two cases. We start in Section 2.2 by considering problem (P) with only integer variables. In this case, it is possible to rewrite the non convex constraints $Y = xx^T$ as a set of linear inequalities. We then obtain an equivalent formulation to (P) with a quadratic and convex objective function, linear constraints, and where all the non-convexity remains in the integrality constraints. Such a formulation can then be solved by a standard mixed-integer quadratic convex solver. Then, we present in Section 2.2 the general case, where problem (P) contains continuous and/or integer variables. In this case, we cannot build an equivalent convex formulation of constraints $Y = xx^T$. We then design a specialized spatial branch-and-bound algorithm that is based on our best quadratic convex relaxation $(\overline{P}^*)$, and that exploits the integrality of some of the variables.

Global solution in the pure integer case: solution by an equivalent quadratic convex formulation

We start with the case where (P) has only integer variables, i.e. when $\mathcal{I} = \mathcal{J}$. To simplify the presentation, we consider that for all $i \in \mathcal{I}$, the lower bounds ℓ_i on variables x_i are set to 0 with the appropriate change of variable. In this case, it is

possible to replace the following non-convex set of constraints:

$$\begin{cases} Y = xx^T \\ 0 \leq x \leq u \\ x \in \mathbb{N}^{|\mathcal{J}|} \end{cases}$$

by the set of linear inequalities we introduced with method `BIL (Binary Integer Linearization)` [23]. The idea is that each variable x_i is replaced by its unique binary expansion $x_i = \sum_{k=0}^{\lfloor \log(u_i) \rfloor} 2^k t_{ik}$ by introducing new binary variables t_{ik}. We can then express the product Y_{ij} of two variables x_i and x_j as a linear function of the products of a variable x by a variable t: $Y_{ij} = \sum_{k=0}^{\lfloor \log(u_i) \rfloor} x_j 2^k t_{ik}$. These products are then linearized by replacing them with another auxiliary variable z and an appropriate set of linear constraints. To get closer to the convex hull, we furthermore add the McCormick inequalities on variables Y that are not redundant. We obtain the following set $\mathcal{L}$:

$$\mathcal{L} = (x, Y, z, t) \begin{cases} x_i = \sum_{k=0}^{\lfloor \log(u_i) \rfloor} 2^k t_{ik} & i \in \mathcal{J} \\ Y_{ij} = \sum_{k=0}^{\lfloor \log(u_i) \rfloor} 2^k z_{ijk} & (i,j) \in \mathcal{J}^2 \\ z_{ijk} \leq u_j t_{ik} & (i,k) \in \mathcal{W},\ j \in \mathcal{J} \\ z_{ijk} \leq x_j & (i,k) \in \mathcal{W},\ j \in \mathcal{J} \\ z_{ijk} \geq x_j - u_j(1 - t_{ik}) & (i,k) \in \mathcal{W},\ j \in \mathcal{J} \\ z_{ijk} \geq 0 & (i,k) \in \mathcal{W},\ j \in \mathcal{J} \\ Y_{ii} \geq x_i & i \in \mathcal{J} \\ Y_{ij} \geq u_j x_i + u_i x_j - u_i u_j & (i,j) \in \mathcal{J}^2 \\ Y_{ji} = Y_{ij} & (i,j) \in \mathcal{J}^2 : i \neq j \\ t_{ik} \in \{0,1\} & (i,k) \in \mathcal{W} \end{cases}$$

where $\mathcal{W} = \{(i,k) : i \in \mathcal{J},\ k = 0, \ldots, \lfloor \log(u_i) \rfloor\}$. The number of binary variables is $|\mathcal{W}| = \sum_{i=1}^{n} (1 + \lfloor \log(u_i) \rfloor)$ and the number of real variables is $n + n^2 + n|\mathcal{W}|$, so that the set $\mathcal{L}$ has $\mathcal{O}(n|\mathcal{W}|)$ variables and constraints. We obtain the following best equivalent convex formulation to (P):

$$(P_I^*) \begin{cases} \min \quad f_{0,S_0^*}(x,Y) = \langle Q_0 + \sum_{r=1}^{m} \alpha_r^* Q_r + \Phi^*, xx^T \rangle + c_0^T x - \langle \sum_{r=1}^{m} \alpha_r^* Q_r + \Phi^*, Y \rangle \\ \text{s.t.} \\ \quad f_{r,S_r}(x,Y) \equiv \langle Q_r, Y \rangle + c_r^T x \leq b_r \qquad r \in \mathcal{R} \\ \quad (x,Y,z,t) \in \mathcal{L} \end{cases}$$

An important remark is that problem (P_I^*) has the same value as problem (P) at

each integer point, thus it is not anymore a relaxation, but an equivalent formulation. Moreover, in (P_I^*) only the binarity of variables t_{ik} is non-convex. Relaxing these constraints leads to a quadratic convex optimization problem which optimal value can be computed in polynomial time. Hence, problem (P_I^*) can for example be handled by a mixed-integer quadratic programming solver which performs a branch-and-cut algorithm to solve it.

An interesting result is stated in Theorem 3. Indeed, it implies that the continuous relaxation of (P_I^*) obtained by relaxing Constraints $t_{ik} \in \{0,1\}$ has the same optimal value as $(\overline{P}^*)$. As a consequence, the bound at the root node gap of the quadratic convex programming solver equals to the optimal value of the "Shor plus RLT" semidefinite relaxation (SDP).

Theorem 3 *Let $\overline{\mathcal{L}}$ the polyhedron obtained from set $\mathcal{L}$ by relaxing Constraints $t_{ik} \in \{0,1\}$ into $0 \leq t_{ik} \leq 1$. The projection of $\overline{\mathcal{L}}$ on variables x and Y is the polyhedron $\mathcal{MC}$. We thus have $v(\overline{P}_I^*) = v(\overline{P}^*) = v(SDP)$.*

This linearization set $\mathcal{L}$ of equalities $Y_{ij} = x_i x_j$ can also be used when x_i is an integer variable and x_j a continuous one. In particular, in [24, 27] we present an extension of this method to the mixed-integer case, with the restriction that in each function $f_r(x)$, all the quadratic convex terms of real variables describe a convex function. This extension is also based on this linearization.

Global solution in the general case: solution with a branch-and-bound based on $(\overline{P}^*)$

We now propose a branch-and-bound algorithm where we keep the integrality constraints into the relaxation $(\overline{P}^*)$. Consequently, we solve at each node of the branch-and-bound process a mixed-integer quadratic convex program that we use to get a lower bound over the original problem (P). Then, we propose different branching rules according to the nature of a product of two variables, i.e. integer by integer, integer by continuous, or continuous by continuous. More formally, at each node of the branch-and-bound we solve the following mixed-integer quadratic program:

$$(\overline{P}^*_{MI}) \begin{cases} \min \quad f_{0,S_0^*}(x,Y) = \langle Q_0 + \sum_{r=1}^{m} \alpha_r^* Q_r + \Phi^*, xx^T \rangle + c_0^T x - \langle \sum_{r=1}^{m} \alpha_r^* Q_r + \Phi^*, Y \rangle \\ \text{s.t.} \\ \quad f_{r,S_r}(x,Y) \equiv \langle Q_r, Y \rangle + c_r^T x + \leq b_r \qquad r \in \mathcal{R} \\ \quad (x,Y) \in \mathcal{MC} \\ \quad x_i \in \mathbb{N} \qquad\qquad\qquad\qquad i \in \mathcal{J} \end{cases}$$

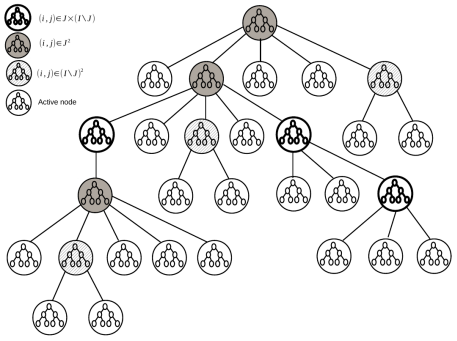
Obviously, $v(\overline{P}^*_{MI}) \geq v(\overline{P}^*)$, and consequently, the branch-and-bound based on $(\overline{P}^*_{MI})$ starts with an even better bound than the "Shor plus RLT" semidefinite relaxation.

We now present our branching rules. For $(\bar{x}, \bar{Y})$ an optimal solution to $(\overline{P}^*_{MI})$, we select a pair of indices (i,j) such that $\bar{x}_i \bar{x}_j \neq \bar{Y}_{ij}$, and depending on the nature of the variables x_i and x_j, we branch as follows:

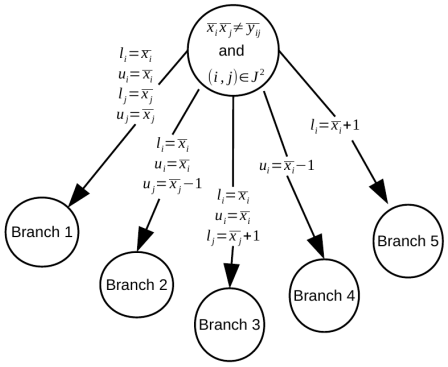
- if x_i and x_j are integers, we have 3 cases that amounts in a separation in 5 branches (cf. Figure 4b): we set x_i to $\bar{x}_i$ and x_j to $\bar{x}_j$ (branch 1), we set x_i to

$\bar{x}_i$ and separate on x_j (branches 2 and 3), and we separate on x_i (branches 4 and 5).

- if x_i is integer and x_j is continuous, we have 2 cases that amounts in a separation in 3 branches (cf. Figure 5a): we set x_i to $\bar{x}_i$ (branch 1), and we separate on x_i (branches 2 and 3).
- if x_i and x_j are continuous, we classically separate on x_i with any value v_i such that $\ell_i < v_i < u_i$ (cf. Figure 5b).



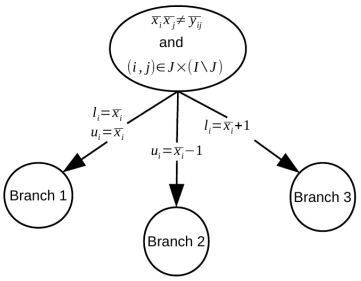
(A) Generic scheme of our branch-and-bound



Branch 1	$\iff$	$x_i = \bar{x}_i$ and $x_j = \bar{x}_j$
Branches 2 and 3	$\iff$	$x_i = \bar{x}_i$ and $x_j \neq \bar{x}_j$
Branches 4 and 5	$\iff$	$x_i \neq \bar{x}_i$

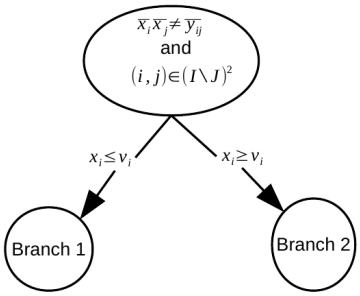
(B) Branching rule when $(i,j) \in \mathcal{I}^2$

FIGURE 4: Generic Scheme of the branch-and-bound, and branching rule when for integer variables



Branch 1	$\iff$	$x_i = \bar{x}_i$
Branches 2 and 3	$\iff$	$x_i \neq \bar{x}_i$

(A) Branching rule when $(i,j) \in \mathcal{J} \times (\mathcal{I} \setminus \mathcal{J})$



(B) Branching rule when $(i,j) \in (\mathcal{I} \setminus \mathcal{J})^2$

FIGURE 5: Branching rule when at least on variable is continuous

One originality of our algorithm lies in the fact that, at each node of the search tree, we choose to compute a bound that comes from the solution of the mixed-integer program $(\overline{P}^*_{MI})$. Of course, to solve $(\overline{P}^*_{MI})$ the solver develops its own Branch and Bound algorithm. An illustration of the generic scheme of our Branch and Bound is presented in Figure 4a.

Obviously, it is also possible to relax the integrality constraints to come back to a classical spatial branch-and-bound, where the separation for integer variables is handled as in branches 4 and 5 of Figure 4b, while for the mixed and pure continuous cases the separation is handled as in Figure 5a and 5b, respectively. In the context of pure continuous variables, we used this approach in [51, 95].

A last remark concerns the specific case of binary quadratic programming. It is well know that the McCormick inequalities [111] combined with binary constraints are equivalent to equalities $Y_{ij} = x_i x_j$. We can trivially deduce that in $(\overline{P}^*_{MI})$ Constraints $Y_{ij} = x_i x_j$ are redundant, and thus that our branch-and-bound is not relevant for binary quadratic programming, since the whole process amounts to solve the root node of Figure 4a.

2.3 Generation of valid quadratic inequalities to strengthen MIQCR [95]

Whether they are based on linearization or convexification, spatial branch-and-bound algorithms for MIQCPs have in common the relaxation of equalities $Y_{ij} = x_i x_j$ by use of the McCormick's envelopes. These inequalities involve the lower and upper bounds (ℓ and u) on the original variables x. Then, since the branching rules consist in updating ℓ or u on a well chosen variable, they become tighter in the course of the branch-and-bound, and allow to improve the value of the relaxation at sub-nodes of the tree. In order to improve the behavior of the second phase of MIQCR, we aim to design inequalities that improve the value of the root node relaxation, but also dynamically tighten the sub-node relaxations by involving ℓ and u. In [31] a procedure was introduced to generate inequalities involving general bounds $[\ell, u]$, from the inequalities that describe the boolean quadratic polytope [124, 151, 37]. In this section, we focus on the generalisation of the regular triangle inequalities[1]. We start by the presentation of our methodology to extend these inequalities to general bounds, obtaining a total of 12 valid inequalities. Then, we prove that our inequalities are as tight as the 4 inequalities generated by the procedure described in [31].

We now describe how we generate the general triangle inequalities that strengthen $(\overline{P}^*)$. As the McCormick's envelopes, they are derived from the ranges $[\ell_i, u_i]$ of each variable x_i. The idea is to consider $\forall (i,j,k) \in \mathcal{V} = \{(i,j,k) \in I^3 \ : \ i < j < k\}$, three variables x_i, x_j and x_k, and we have $(u_i - x_i)(u_j - x_j)(u_k - x_k) \geq 0$, or equivalently:

$$u_k x_i x_j + u_j x_i x_k + u_i x_j x_k - u_i u_k x_j - u_j u_k x_i - u_i u_j x_k + u_i u_j u_k \geq x_i x_j x_k$$

using the McCormick inequality $x_j x_k \geq \ell_j x_k + \ell_k x_j - \ell_j \ell_k$, we get:

$$u_k x_i x_j + u_j x_i x_k + u_i x_j x_k - u_i u_k x_j - u_j u_k x_i - u_i u_j x_k + u_i u_j u_k \geq x_i(\ell_j x_k + \ell_k x_j - \ell_j \ell_k)$$

or equivalently the new quadratic inequality:

$$\boxed{(\ell_k - u_k) x_i x_j + (\ell_j - u_j) x_i x_k - u_i x_j x_k + u_i u_k x_j + (u_j u_k - \ell_j \ell_k) x_i + u_i u_j x_k - u_i u_j u_k \leq 0}$$

In the example above, we also could have chosen to use the other McCormick envelope, i.e. $-x_j x_k + u_j x_k + u_k x_j - u_j u_k \leq 0$, to substitute the product $x_j x_k$, or, to

[1] $\{-Y_{ij} - Y_{ik} - Y_{jk} + x_i + x_j + x_k \leq 1, Y_{ik} - Y_{ij} + Y_{jk} - x_k \leq 0, Y_{ij} - Y_{ik} + Y_{jk} - x_j \leq 0, Y_{ik} - Y_{jk} + Y_{ij} - x_i \leq 0\}$

substitute either the product $x_i x_j$ or $x_i x_k$ by one of its two McCormick over estimators leading to 6 different inequalities. Hence, by considering all possible combinations of the the products of degree 3 of Constraints $\ell_i \leq x_i \leq u_i$, we obtain 8 families of 6 inequalities with a total of 48 inequalities.

These inequalities are obviously valid by construction. The question is now to determine which families of inequalities, when they are linearized using the auxiliary variables Y, are non redundant in $(\overline{P}^*)$. We proved in [95] that 12 out of 48 of them cut feasible solutions of $(\overline{P}^*)$.

Proposition 1 *The inequalities of set $\mathcal{T}$ cut feasible solutions of $(\overline{P}^*)$:*

$$\mathcal{T} = (x, Y) \begin{cases} (\ell_k - u_k)y_{ij} + (\ell_j - u_j)y_{ik} - u_i y_{jk} + u_i u_k x_j + (u_j u_k - \ell_j \ell_k)x_i + u_i u_j x_k - u_i u_j u_k \leq 0 & (i,j,k) \in \mathcal{V} \\ (\ell_k - u_k)y_{ij} - u_j y_{ik} + (\ell_i - u_i)y_{jk} + (u_i u_k - \ell_i \ell_k)x_j + u_j u_k x_i + u_i u_j x_k - u_i u_j u_k \leq 0 & (i,j,k) \in \mathcal{V} \\ -u_k y_{ij} + (\ell_j - u_j)y_{ik} + (\ell_i - u_i)y_{jk} + u_i u_k x_j + u_j u_k x_i + (u_i u_j - \ell_i \ell_j)x_k - u_i u_j u_k \leq 0 & (i,j,k) \in \mathcal{V} \\ (u_j - \ell_j)y_{ik} + (\ell_k - u_k)y_{ij} + u_i y_{jk} + (\ell_j u_k - u_j \ell_k)x_i - u_i \ell_k x_j - u_i u_j x_k + u_i u_j \ell_k \leq 0 & (i,j,k) \in \mathcal{V} \\ u_j y_{ik} + (\ell_k - u_k)y_{ij} + (u_i - \ell_i)y_{jk} - u_j \ell_k x_i + (\ell_i u_k - u_i \ell_k)x_j - u_i u_j x_k + u_i u_j \ell_k \leq 0 & (i,j,k) \in \mathcal{V} \\ (u_k - \ell_k)y_{ij} + u_i y_{jk} + (\ell_j - u_j)y_{ik} - u_i u_k x_j + (u_j \ell_k - \ell_j u_k)x_i - u_i \ell_j x_k + u_i \ell_j u_k \leq 0 & (i,j,k) \in \mathcal{V} \\ u_k y_{ij} + (u_i - \ell_i)y_{jk} + (\ell_j - u_j)y_{ik} - u_i u_k x_j - \ell_j u_k x_i + (\ell_i u_j - u_i \ell_j)x_k + u_i \ell_j u_k \leq 0 & (i,j,k) \in \mathcal{V} \\ (u_k - \ell_k)y_{ij} + (\ell_i - u_i)y_{jk} + u_j y_{ik} + (u_i \ell_k - \ell_i u_k)x_j - u_j u_k x_i - \ell_i u_j x_k + \ell_i u_j u_k \leq & (i,j,k) \in \mathcal{V} \\ u_k y_{ij} + (\ell_i - u_i)y_{jk} + (u_j - \ell_j)y_{ik} - \ell_i u_k x_j - u_j u_k x_i + (u_i \ell_j - \ell_i u_j)x_k + \ell_i u_j u_k \leq 0 & (i,j,k) \in \mathcal{V} \\ -u_i y_{jk} + (u_j - \ell_j)y_{ik} + (u_k - \ell_k)y_{ij} + u_i \ell_k x_j + (\ell_j \ell_k - u_j u_k)x_i + u_i \ell_j x_k - u_i \ell_j \ell_k \leq 0 & (i,j,k) \in \mathcal{V} \\ (u_i - \ell_i)y_{jk} - u_j y_{ik} + (u_k - \ell_k)y_{ij} + (\ell_i \ell_k - u_i u_k)x_j + u_j \ell_k x_i + \ell_i u_j x_k - \ell_i u_j \ell_k \leq 0 & (i,j,k) \in \mathcal{V} \\ -u_k y_{ij} + (u_i - \ell_i)y_{jk} + (u_j - \ell_j)y_{ik} + \ell_i u_k x_j + \ell_j u_k x_i + (\ell_i \ell_j - u_i u_j)x_k - \ell_i \ell_j u_k \leq 0 & (i,j,k) \in \mathcal{V} \end{cases}$$

Proposition 2 states that these inequalities amounts to the regular triangle inequalities in the specific case where $\ell_i = 0$, and $u_i = 1$, for all $i \in I$.

Proposition 2 *The set of inequalities $\mathcal{T}$ is an extension of the regular triangle inequalities, introduced in [124] for the box constrained case (i.e. $x_i \in [0,1]$), to the general case (i.e. $x_i \in [\ell_i, u_i]$).*

We now compare the set of inequalities $\mathcal{T}$ with the inequalities generated by the method of [31]. This generic methodology allows to transform each inequality valid for variables belonging to $[0,1]$ into a new inequality valid for variables with general bounds. By Applying their transformation to regular triangle inequalities, we get for all $(i,j,k) \in \mathcal{V}$, the following set $\mathcal{B}$ of constraints:

$$\begin{cases} (\ell_k - u_k)Y_{ij} + (\ell_j - u_j)Y_{ik} + (\ell_i - u_i)Y_{jk} + (u_i u_k - \ell_i \ell_k)x_j + (u_j u_k - \ell_j \ell_k)x_i + (u_i u_j - \ell_i \ell_j)x_k - u_i u_j u_k + \ell_i \ell_j \ell_k \leq 0 \\ (u_j - \ell_j)Y_{ik} + (\ell_k - u_k)Y_{ij} + (u_i - \ell_i)Y_{jk} + (\ell_j u_k - u_j \ell_k)x_i + (\ell_i u_k - u_i \ell_k)x_j + (\ell_i \ell_j - u_i u_j)x_k - \ell_i \ell_j u_k + u_i u_j \ell_k \leq 0 \\ (u_k - \ell_k)Y_{ij} + (\ell_j - u_j)Y_{ik} + (u_i - \ell_i)Y_{jk} + (u_j \ell_k - \ell_j u_k)x_i + (\ell_i u_j - u_i \ell_j)x_k + (\ell_i \ell_k - u_i u_k)x_j - \ell_i u_j \ell_k + u_i \ell_j u_k \leq 0 \\ (u_j - \ell_j)Y_{ik} + (\ell_i - u_i)Y_{jk} + (u_k - \ell_k)Y_{ij} + (u_i \ell_j - \ell_i u_j)x_k + (u_i \ell_k - \ell_i u_k)x_j + (\ell_j \ell_k - u_j u_k)x_i - u_i \ell_j \ell_k + \ell_i u_j u_k \leq 0 \end{cases}$$

We start by observing that Inequalities of set $\mathcal{B}$ can be generated with the same methodology. Indeed, starting again with the valid inequality:

$$u_k x_i x_j + u_j x_i x_k + u_i x_j x_k - u_i u_k x_j - u_j u_k x_i - u_i u_j x_k + u_i u_j u_k \geq x_i x_j x_k \qquad (i)$$

and using inequalities $-x_j x_k + \ell_j x_k + \ell_k x_j - \ell_j \ell_k \leq 0$, and $x_i - \ell_i \geq 0$, we get:

$$0 \geq (x_i - \ell_i)(-x_j x_k + \ell_j x_k + \ell_k x_j - \ell_j \ell_k)$$

or equivalently

$$x_i x_j x_k \geq (x_i - \ell_i)(\ell_j x_k + \ell_k x_j - \ell_j \ell_k) + \ell_i x_j x_k \qquad (ii)$$

Combining inequalities (i) and (ii), and using variables Y for linearizing the obtained constraints, we get the first inequalities of set $\mathcal{B}$. The other inequalities can be generated symmetrically. We observe that the difference between sets $\mathcal{T}$ and $\mathcal{B}$ comes from a change of variables associated to a translation of the bounds ℓ to 0. Proposition 3 states that set $\mathcal{T}$ is as strong as set $\mathcal{B}$.

Proposition 3 *The set $\mathcal{T}$ is as strong as the set $\mathcal{B}$.*

In [31], the set $\mathcal{B}$ and many other valid inequalities are used into a branch-and-cut algorithm that is based on a linear relaxation. Here we aim to use the inequalities of the set $\mathcal{T}$ into the convexification process of MIQCR. For this purpose, we handle the new cuts as classical quadratic inequalities, and we consider the following "Shor plus RLT plus Triangle" semidefinite relaxation:

$$
(SDP')\ \begin{cases}
\min f(X,x) = \langle Q_0, X\rangle + c_0^T x & & \\
\text{s.t.} \langle Q_r, X\rangle + c_r^T x \le b_r & r \in C \leftarrow \alpha_r & (1)\\
X_{ii} \ge x_i & i \in J \leftarrow \varphi_i & (2)\\
X_{ij} \le u_j x_i + \ell_i x_j - u_j \ell_i & (i,j) \in \mathcal{U} \leftarrow \phi^1_{ij} & (3)\\
X_{ij} \le u_i x_j + \ell_j x_i - u_i \ell_j & (i,j) \in \mathcal{U} \leftarrow \phi^2_{ij} & (4)\\
X_{ij} \ge u_j x_i + u_i x_j - u_i u_j & (i,j) \in \mathcal{U} \leftarrow \phi^3_{ij} & (5)\\
X_{ij} \ge \ell_j x_i + \ell_i x_j - \ell_i \ell_j & (i,j) \in \mathcal{U} \leftarrow \phi^4_{ij} & (6)\\
(\ell_k - u_k)X_{ij} + (\ell_j - u_j)X_{ik} - u_i X_{jk} + u_i u_k x_j + (u_j u_k - \ell_j \ell_k)x_i + u_i u_j x_k - u_i u_j u_k \le 0 & (i,j,k) \in \mathcal{V} \leftarrow \delta^1_{ijk} & (15)\\
(\ell_k - u_k)X_{ij} - u_j X_{ik} + (\ell_i - u_i)X_{jk} + (u_i u_k - \ell_i \ell_k)x_j + u_j u_k x_i + u_i u_j x_k - u_i u_j u_k \le 0 & (i,j,k) \in \mathcal{V} \leftarrow \delta^2_{ijk} & (16)\\
-u_k X_{ij} + (\ell_j - u_j)X_{ik} + (\ell_i - u_i)X_{jk} + u_i u_k x_j + u_j u_k x_i + (u_i u_j - \ell_i \ell_j)x_k - u_i u_j u_k \le 0 & (i,j,k) \in \mathcal{V} \leftarrow \delta^3_{ijk} & (17)\\
(u_j - \ell_j)X_{ik} + (\ell_k - u_k)X_{ij} + u_i X_{jk} + (\ell_j u_k - u_j \ell_k)x_i - u_i \ell_k x_j - u_i u_j x_k + u_i u_j \ell_k \le 0 & (i,j,k) \in \mathcal{V} \leftarrow \delta^4_{ijk} & (18)\\
u_j X_{ik} + (\ell_k - u_k)X_{ij} + (u_i - \ell_i)X_{jk} - u_j \ell_k x_i + (\ell_i u_k - u_i \ell_k)x_j - u_i u_j x_k + u_i u_j \ell_k \le 0 & (i,j,k) \in \mathcal{V} \leftarrow \delta^5_{ijk} & (19)\\
(u_k - \ell_k)X_{ij} + u_i X_{jk} + (\ell_j - u_j)X_{ik} - u_i u_k x_j + (u_j \ell_k - \ell_j u_k)x_i - u_i \ell_j x_k + u_i \ell_j u_k \le 0 & (i,j,k) \in \mathcal{V} \leftarrow \delta^6_{ijk} & (20)\\
u_k X_{ij} + (u_i - \ell_i)X_{jk} + (\ell_j - u_j)X_{ik} - u_i u_k x_j - \ell_j u_k x_i + (\ell_i u_j - u_i \ell_j)x_k + u_i \ell_j u_k \le 0 & (i,j,k) \in \mathcal{V} \leftarrow \delta^7_{ijk} & (21)\\
(u_k - \ell_k)X_{ij} + (\ell_i - u_i)X_{jk} + u_j X_{ik} + (u_i \ell_k - \ell_i u_k)x_j - u_j u_k x_i - \ell_i u_j x_k + \ell_i u_j u_k \le & (i,j,k) \in \mathcal{V} \leftarrow \delta^8_{ijk} & (22)\\
u_k X_{ij} + (\ell_i - u_i)X_{jk} + (u_j - \ell_j)X_{ik} - \ell_i u_k x_j - u_j u_k x_i + (u_i \ell_j - \ell_i u_j)x_k + \ell_i u_j u_k \le 0 & (i,j,k) \in \mathcal{V} \leftarrow \delta^9_{ijk} & (23)\\
-u_i X_{jk} + (u_j - \ell_j)X_{ik} + (u_k - \ell_k)X_{ij} + u_i \ell_k x_j + (\ell_j \ell_k - u_j u_k)x_i + u_i \ell_j x_k - u_i \ell_j \ell_k \le 0 & (i,j,k) \in \mathcal{V} \leftarrow \delta^{10}_{ijk} & (24)\\
(u_i - \ell_i)X_{jk} - u_j X_{ik} + (u_k - \ell_k)X_{ij} + (\ell_i \ell_k - u_i u_k)x_j + u_j \ell_k x_i + \ell_i u_j x_k - \ell_i u_j \ell_k \le 0 & (i,j,k) \in \mathcal{V} \leftarrow \delta^{11}_{ijk} & (25)\\
-u_k X_{ij} + (u_i - \ell_i)X_{jk} + (u_j - \ell_j)X_{ik} + \ell_i u_k x_j + \ell_j u_k x_i + (\ell_i \ell_j - u_i u_j)x_k - \ell_i \ell_j u_k \le 0 & (i,j,k) \in \mathcal{V} \leftarrow \delta^{12}_{ijk} & (26)\\
\begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix} \succeq 0 & & \\
x \in \mathbb{R}^n \quad X \in \mathcal{S}_n & &
\end{cases}
$$

By applying Theorem 1, we calculate $S_0^* = Q_0 + \sum_{r=1}^{m} \alpha_r^* Q_r + \Phi^* + \Delta^*$ where:

- α^* and Φ^* are computed as described in Theorem 1
- $\Delta^* = \sum_{(i,j,k)\in\mathcal{V}} \sum_{t=1}^{12} \delta_{ijk}^{t*} Q_{ijk}^t$, with δ_{ijk}^{t*}, $t = 1,\dots,12$ the optimal dual variables associated with Constraints (15)–(26), and Q_{ijk}^t the Hessian matrices of the associated constraints.

We thus obtain the following quadratic convex relaxation of (P):

$$(\overline{P}'^{*}) \begin{cases} \min \quad \langle Q_0 + \sum_{r=1}^{m} \alpha_r^* Q_r + \Phi^* + \Delta^*, xx^T \rangle + c_0^T x - \langle \sum_{r=1}^{m} \alpha_r^* Q_r + \Phi^* + \Delta^*, Y \rangle \\ \text{s.t.} \\ \quad f_{r,S_r}(x,Y) \equiv \langle Q_r, Y \rangle + c_r^T x + \leq b_r & r \in C \\ \quad (x,y) \in \mathcal{MC} \\ \quad (x,y) \in \mathcal{T} \end{cases}$$

Corollary 3 *It holds that* $v(\overline{P}'^{*}) = v(SDP')$.

Observe that inequalities of set $\mathcal{T}$ can be used in any convex relaxation of (P) that relax the non-convex constraints $Y = xx^T$.

In order to manage the huge size of (SDP'), we propose a trick that can be applied when problem (P) contains only continuous variables. We know by Proposition 2 that when for all $i \in I$, $u_i = 1$ and $\ell_i = 0$, the 12 general Triangle inequalities of $\mathcal{T}$ can be equivalently reduced to the 4 standard triangle inequalities. Then, in this case, one can proceed as follows:

i) Apply a change of variables on (P) such that the new variables $\bar{x}_i$ belongs to $[0,1]$, via the equations $\bar{x}_i = \frac{x_i - \ell_i}{u_i - \ell_i}$.

ii) Solve (SDP') only with the 4 families of regular triangle inequalities and deduce the optimal matrix S_0^*.

iii) Add to the constraint set of the relaxation $(\overline{P}'^{*})$ the appropriate "duplication" of each active inequality of an optimal solution of (SDP').

However, in the general integer case, this change of variable is not possible, and in that case there exists a huge number of general Triangle ($\mathcal{O}(n^3)$). We present in Section 2.4 how we separate these inequalities within the solution of the huge semidefinite program (SDP') that we need to solve.

2.4 A dual heuristic for computing our convex relaxation [28, 95]

In practice, due to its size, the solution of the semi-definite problem of Phase 1 often constitutes the bottleneck of `MIQCR` method. However, once the equivalent formulation is computed, solving the obtained reformulated program is practical, since the continuous relaxation bound of the reformulation is tight. Hence, to handle larger instances method `MIQCR` needs an appropriate algorithm to solve Phase 1. In this section, we present a sub-gradient algorithm within a Lagrangian duality framework for solving (SDP') approximately following the procedure introduced in [57]. Then, we parameterize our algorithm obtaining a dual heuristic for solving (SDP'). In fact, we use this dual heuristic for selecting the p most violated inequalities of the "Shor plus RLT plus Triangle" semi-definite relaxation. It can thus be viewed as a separation method of sets $\mathcal{MC}$ and $\mathcal{T}$. With this suitable algorithm, the time for

computing Phase 1 significantly decreases and allows us to handle large-scale instances. Moreover, we obtain also a speed up of Phase 2, since by construction the equivalent formulation computed with the new algorithm can be smaller than the reformulation obtained with a standard semi-definite solver. Indeed, the density of the matrix $Q_0 - S_0^*$ influences the size of the reformulated problem. One thus can ask which structure for this matrix would best fit with MIQCR. We do not yet theoretically answer to this question, but with the bundle method proposed in this section, we are able to control the size of problem $(\overline{P}'^*)$.

Before a brief description of the method, we start by three observations:

i) any set of positive semidefinite matrices $\bar{S}_0, \ldots, \bar{S}_m$ can be used to make the relaxation $(\overline{P}'_{\bar{S}_0,\ldots,\bar{S}_m})$ convex.

ii) from any feasible dual solution (α, Φ, Δ) of (SDP'), the associated relaxation $(\overline{P}'_{S_0,\ldots,S_m})$ is convex. Hence, we do not need to solve (SDP') to global optimality to compute a convex relaxation.

iii) from any nearly feasible dual solution (α, Φ, Δ) of (SDP'), if $f_{0,S_0}(x, Y)$ is not convex, we can always make it convex by taking $(\alpha, \Phi + diag(-\lambda_{min}), \Delta)$ where λ_{min} is the smallest eigenvalue of the Hessian matrix of function $f_{S_0,\ldots,S_m}(x, Y)$.

We now present a bundle method to obtain a reasonable solution of (SDP'). We start by rewriting (SDP') as a maximization problem:

$$(SDP'_{H,H'}) \begin{cases} \max f(X,x) = -\langle Q_0, X\rangle - c_0^T x \\ \text{s.t.} \\ \quad (X,x) \in S \\ \quad h_{ij}^t(X,x) \le 0 \ (i,j,t) \in H \\ \quad h'^t_{ijk}(X,x) \le 0 \ (i,j,k,t) \in H' \end{cases}$$

where:

$$S = \begin{cases} \langle Q_r, X\rangle + c_r^T x \le b_r & r \in \mathcal{R} \\ X_{ii} - u_i x_i - l_i x_i + u_i l_i \le 0 & i \in \mathcal{I} \\ -X_{ii} + u_i x_i + u_i x_i - u_i u_i \le 0 & i \in \mathcal{I} \\ -X_{ii} + l_i x_i + l_i x_i - l_i l_i \le 0 & i \in \mathcal{I} \\ -X_{ii} + x_i \le 0 & i \in \mathcal{J} \\ \begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix} \succeq 0 & \\ x \in \mathbb{R}^n \quad X \in \mathcal{S}_n & \end{cases}$$

and for all $(i,j,t) \in H = \{(i,j,t) := \{(i,j) \in \mathcal{I}^2 i < j,\ t = 1,\ldots,4\}$, $h_{ij}^t(X,x)$ are the non-diagonal McCormick envelopes:

$$h_{ij}^t(X,x) = \begin{cases} X_{ij} - u_j x_i - \ell_i x_j + u_j \ell_i & t = 1 \quad (3) \\ X_{ij} - u_i x_j - \ell_j x_i + u_i \ell_j & t = 2 \quad (4) \\ -X_{ij} + u_j x_i + u_i x_j - u_i u_j & t = 3 \quad (5) \\ -X_{ij} + \ell_j x_i + \ell_i x_j - \ell_i \ell_j & t = 4 \quad (6) \end{cases}$$

and for all $(i,j,k,t) \in H' = \{(i,j,k,t) : (i,j,k) \in \mathcal{V},\ t = 1,\dots,12\}$, $h'^{t}_{ijk}(X,x)$ are the general triangle inequalities:

$$h'^{t}_{ijk}(X,x) = \begin{cases}
(\ell_k - u_k)X_{ij} + (\ell_j - u_j)X_{ik} - u_iX_{jk} + u_iu_kx_j + (u_ju_k - \ell_j\ell_k)x_i + u_iu_jx_k - u_iu_ju_k & t = 1 \quad (15)\\
(\ell_k - u_k)X_{ij} - u_jX_{ik} + (\ell_i - u_i)X_{jk} + (u_iu_k - \ell_i\ell_k)x_j + u_ju_kx_i + u_iu_jx_k - u_iu_ju_k & t = 2 \quad (16)\\
-u_kX_{ij} + (\ell_j - u_j)X_{ik} + (\ell_i - u_i)X_{jk} + u_iu_kx_j + u_ju_kx_i + (u_iu_j - \ell_i\ell_j)x_k - u_iu_ju_k & t = 3 \quad (17)\\
(u_j - \ell_j)X_{ik} + (\ell_k - u_k)X_{ij} + u_iX_{jk} + (\ell_ju_k - u_j\ell_k)x_i - u_i\ell_kx_j - u_iu_jx_k + u_iu_j\ell_k & t = 4 \quad (18)\\
u_jX_{ik} + (\ell_k - u_k)X_{ij} + (u_i - \ell_i)X_{jk} - u_j\ell_kx_i + (\ell_iu_k - u_i\ell_k)x_j - u_iu_jx_k + u_iu_j\ell_k & t = 5 \quad (19)\\
(u_k - \ell_k)X_{ij} + u_iX_{jk} + (\ell_j - u_j)X_{ik} - u_iu_kx_j + (u_j\ell_k - \ell_ju_k)x_i - u_i\ell_jx_k + u_i\ell_ju_k & t = 6 \quad (20)\\
u_kX_{ij} + (u_i - \ell_i)X_{jk} + (\ell_j - u_j)X_{ik} - u_iu_kx_j - \ell_ju_kx_i + (\ell_iu_j - u_i\ell_j)x_k + u_i\ell_ju_k & t = 7 \quad (21)\\
(u_k - \ell_k)X_{ij} + (\ell_i - u_i)X_{jk} + u_jX_{ik} + (u_i\ell_k - \ell_iu_k)x_j - u_ju_kx_i - \ell_iu_jx_k + \ell_iu_ju_k & t = 8 \quad (22)\\
u_kX_{ij} + (\ell_i - u_i)X_{jk} + (u_j - \ell_j)X_{ik} - \ell_iu_kx_j - u_ju_kx_i + (u_i\ell_j - \ell_iu_j)x_k + \ell_iu_ju_k & t = 9 \quad (23)\\
-u_iX_{jk} + (u_j - \ell_j)X_{ik} + (u_k - \ell_k)X_{ij} + u_i\ell_kx_j + (\ell_j\ell_k - u_ju_k)x_i + u_i\ell_jx_k - u_i\ell_j\ell_k & t = 10 \quad (24)\\
(u_i - \ell_i)X_{jk} - u_jX_{ik} + (u_k - \ell_k)X_{ij} + (\ell_i\ell_k - u_iu_k)x_j + u_j\ell_kx_i + \ell_iu_jx_k - \ell_iu_j\ell_k & t = 11 \quad (25)\\
-u_kX_{ij} + (u_i - \ell_i)X_{jk} + (u_j - \ell_j)X_{ik} + \ell_iu_kx_j + \ell_ju_kx_i + (\ell_i\ell_j - u_iu_j)x_k - \ell_i\ell_ju_k & t = 12 \quad (26)
\end{cases}$$

We now consider a partial Lagrangian dual of (SDP') where we dualize the set of constraints $\mathcal{MC} \cup \mathcal{T}$, i.e. constraints (3)–(6) and (15)–(26). For this, with each constraint $h^t_{ij}(X,x) \le 0$ of $(SDP'_{H,H'})$ we associate a non-negative Lagrange multiplier ϕ^t_{ij}, and with each constraints $h'^{t}_{ijk}(X,x) \le 0$ a non-negative Lagrange multiplier δ^t_{ijk}. We get the following partial Lagrangian:

$$\mathcal{L}_{H,H'}(X,x,\phi,\delta) = -\langle Q_0, X\rangle - c_0^Tx - \sum_{(i,j,t)\in H}\phi^t_{ij}h^t_{ij}(X,x) - \sum_{(i,j,k,t)\in H'}\delta^t_{ijk}h'^{t}_{ijk}(X,x)$$

and we obtain the dual functional

$$g_{H,H'}(\phi,\delta) = \max_{(X,x)\in S}\ \mathcal{L}_{H,H'}(X,x,\phi,\delta)$$

By minimizing this dual functional we obtain the partial Lagrangian dual problem $(LD_{H,H'})$ associated with $(SDP'_{H,H'})$,

$$(LD_{H,H'})\begin{cases}\min\limits_{\substack{\phi^t_{ij}\ge 0,\ (i,j,t)\in H\\ \delta^t_{ijk}\ge 0,\ (i,j,k,t)\in H'}} & g_{H,H'}(\phi,\delta)\end{cases}$$

We recall here the outline of the bundle method. For a given $\bar\phi^t_{ij} \ge 0$ and $\bar\delta^t_{ijk} \ge 0$, we evaluate $g_{H,H'}(\bar\phi,\bar\delta)$ and determine the associate primal solution $(\bar X,\bar x)$, such that $g_{H,H'}(\bar\phi,\bar\delta) = \mathcal{L}_{H,H'}(\bar X,\bar x,\bar\phi,\bar\delta)$. We call a pair $((\bar\phi,\bar\delta),(\bar X,\bar x))$ a matching pair for $g_{H,H'}$. Evaluating function $g_{H,H'}$ for given $(\bar\phi,\bar\delta)$ amounts to maximize a linear function in (X,x) over the set S. This is a semi-definite program that has much less constraints than $(SDP'_{H,H'})$ and can be solved efficiently by interior-point methods. From the solution $(\bar X,\bar x)$, we compute a sub-gradient $(h^t_{ij}(\bar X,\bar x), h'^{t}_{ijk}(\bar X,\bar x)) \in \partial g_{H,H'}(\bar\phi,\bar\delta)$. The bundle method is an iterative algorithm that maintains at each iteration k a "best" approximation $(\hat\phi_k,\hat\delta_k)$ and a sequence $\mathcal{X} = \Big((\bar X_1,\bar x_1),\dots,(\bar X_k,\bar x_k)\Big)$ where $((\hat\phi_k,\hat\delta_k),(\bar X_i,\bar x_i))$ is a matching pair. Then, from the sequence $\mathcal{X}$, the best approximation $(\hat\phi_k,\hat\delta_k)$, and the new sub-gradient, the bundle method computes a new value $(\hat\phi_{k+1},\hat\delta_{k+1})$ that will be used at the next iteration.

The number of elements in $H \cup H'$ is $4\binom{n}{2} + 12\binom{n}{3}$. However, we are interested only in the subset of $H \cup H'$ for which the constraints $h_{ij}^t(X,x) \leq 0$ and $h_{ijk}^{'t}(X,x) \leq 0$ are tight at the optimum. The idea is to dynamically add and remove constraints in the course of the algorithm. Then, we now consider $\mathcal{H} \subseteq H$ and $\mathcal{H}' \subseteq H'$, and work with the function:

$$g_{\mathcal{H},\mathcal{H}'}(\phi,\delta) = \max_{(X,x)\in S} \quad \mathcal{L}_{\mathcal{H},\mathcal{H}'}(X,x,\phi,\delta).$$

Initially we set $\mathcal{H} \cup \mathcal{H}' = \emptyset$ and after a first function evaluation we separate violated inequalities and add the elements to set $\mathcal{H} \cup \mathcal{H}'$ accordingly. We keep on updating this set in course of the bundle iterations by removing elements with associated multiplier close to zero and separate newly violated constraints.

In our context, we know that any feasible dual solution to $(SDP'_{H,H'})$ allows us to build a quadratic convex relaxation of (P). The more this solution is close to the optimum, the more the associated bound at the root node of the branch-and-bound process is sharp. A possible way to get such a solution is to drop some constraints from $\mathcal{MC} \cup \mathcal{T}$ of $(SDP'_{H,H'})$ and compute a dual "nearly" optimal solution $(\bar{\alpha}, \bar{\Phi}, \bar{\Delta})$ of the reduced problem. Then, a feasible dual solution to $(SDP'_{H,H'})$ can be obtained by completing $(\bar{\alpha}, \bar{\Phi}, \bar{\Delta})$ with zeros for those dual variables ϕ_{ij}^t and δ_{ijk}^t corresponding to the dropped constraints. To carry out this idea, we consider a parameter p that is an upper bound on the cardinality of $\mathcal{H} \cup \mathcal{H}'$, i. e. $|\mathcal{H} \cup \mathcal{H}'| \leq p$. In other words, p is the maximum number of constraints considered in the reduced problem. Introducing this parameter p leads to a dual heuristic that has two extreme cases:

- if $p = 4\binom{n}{2} + 12\binom{n}{3}$, we solve $(SDP'_{H,H'})$ and get the associated dual solution as described in Theorem 1.
- if $p = 0$, we make a single iteration: we get the optimal solution of the reduced problem obtained from $(SDP'_{H,H'})$ where we drop all constraints of $\mathcal{MC} \cup \mathcal{T}$ (this amounts to solving the "Shor plus diagonal RLT" semi-definite relaxation).

The algorithm returns a solution (Φ^*, Δ^*) having at most p positive components. Thus, the number of variables Y_{ij} of problem (P'^*) is also at most p only and Phase 2 can also be solved much faster. This parameter p controls the size, and in a sense the tightness, of the semi-definite relaxation used for computing the equivalent formulation of method MIQCR. Finally, we use the dynamic bundle method to separate the set of valid inequalities $\mathcal{MC} \cup \mathcal{T}$ during the solution of (SDP').

2.5 Conclusion and experiments

Our algorithm MIQCR is summed up in Algorithm 1. It is implemented in the open source software `Solution of Mixed Integer Quadratic Programs (SMIQP)` [94].

Algorithm 1 `MIQCR`(P)

Phase 1: Building the quadratic convex relaxation $(\overline{P}^*)$
Solve (SDP) (or (SDP')) by the Bundle algorithm described in Section 2.4, and determine matrix S_0^* from the optimal dual solutions as described in Theorem 1 (or Corollary 3). For the constraints take $S_r = \mathbf{0}_n$ for all $r \in \mathcal{R}$.
Phase 2: Solving (P) **using** $(\overline{P}^*)$
Develop a spatial branch-and-bound where the branching rules follows the description of Section 2.2, depending of the nature of the variables x_i and x_j involved in the violation of the constraint $Y_{ij} = x_i x_j$.

We now give an illustration of the experimental behavior our methods thought two sets of instances. We compare our algorithms `MIQCR` and `MIQCR-T` with the solvers `Baron` [133] and `Gurobi` [69]. We used 2 sets of instances. The first one consists in 135 instances of pure-continuous quadratically constrained quadratic programs from [16] called *unitbox*. The second one consists in inequality constrained pure-integer instances of class $(IQCP_5)$ from [27]. Our experiments were carried out on a server with 2 CPU Intel Xeon each of them having 12 cores and 2 threads of 2.5 GHz and $4 * 16$ GB of RAM using a Linux operating system.

Results for the *unitbox* instances

Each *unitbox* instance from [16] consists in minimizing a quadratic function of n continuous variables in the interval $[0, 1]$, subject to m quadratic inequalities. For the considered instances, n varies from 8 to 50, and m from 8 to 100. We set the time limit to 2 hours. For the solver `Baron`, we use the multi-threading version of `Cplex 12.9` ([82]) with up to 64 threads. For methods `MIQCR` and `MIQCR-T`, we used the solver `csdp` ([32]) together with the Conic Bundle library ([79]) for solving semi-definite programs, as described in Section 2.4. We used the C interface of the solver `Cplex` for solving the quadratic convex relaxations at each node of the search tree. For computing feasible local solutions, we use the local solver `Ipopt` ([147]). Parameter p is set to $0.4 \cdot |\mathcal{MC}|$ for `MIQCR`, and to $0.04 \cdot |\mathcal{MC} \cup \mathcal{T}|$ for `MIQCR-T`.

In Figure 6, we present the performance profile of the CPU times for methods `MIQCR-T`, `MIQCR`, and the solvers `Baron 19.3.24` and `Gurobi 9.1.1` for the *unitbox* instances. We observe that `MIQCR-T` and `MIQCR` outperform the compared solvers in terms of CPU time and number of instances solved. More precisely, `Baron` solve 109 instances, `Gurobi` solves 114 instances, `MIQCR` solves 119 instances, and `MIQCR-T` solves 128 instances out of 135 within the time limit.

We end with the following observations for methods `MIQCR` and `MIQCR-T`. First, the initial gap is smaller for `MIQCR-T` than for `MIQCR`: we pass from 2.47% to 1.98% on average on the 135 instances. Surprisingly, the CPU time for solving the semi-definite relaxation is divided by 2.7 on average for `MIQCR-T` in comparison to `MIQCR`. This is due to the sub-gradients considered in the course of the Conic Bundle algorithms that can be different for the two methods. Another consequence of the use of the new inequalities within the branch-and-bound process is the significant reduction of the number of nodes (by a factor 3.3). Hence, we can also see a reduction of CPU time for this phase, and the total time is divided by a factor 2 for `MIQCR-T`.

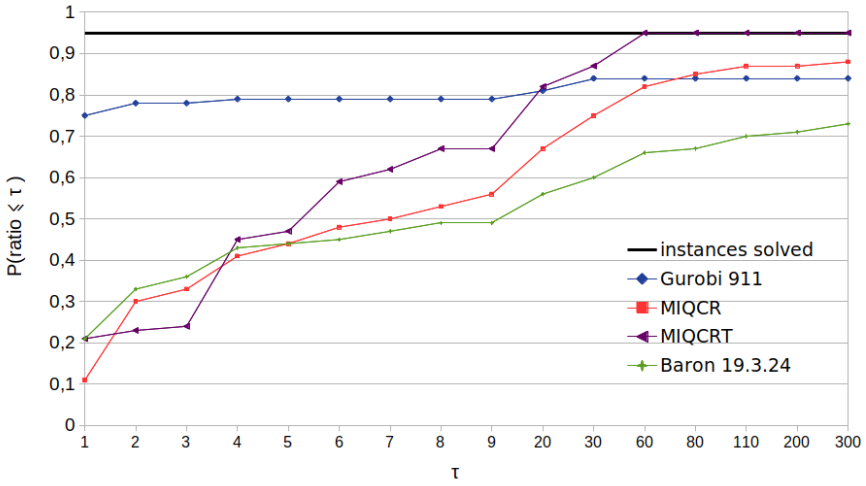


FIGURE 6: Performance profile of the CPU time for the *unitbox* instances with $n = 8$ to 50 with a time limit of 2 hours.

Results for the $IQCP_5$ instances

Each $IQCP_5$ instance consists of minimizing a quadratic function of n general integer variable subject to 5 quadratic inequality constraints. For the considered instances, n varies from 10 to 50, and the each variable belongs to the interval $[0, 20]$. We set the time limit to 2 hours. For methods `MIQCR` and `MIQCR-T`, we used the solver `Mosek` [122] together with the Conic Bundle library [79] for solving semi-definite programs. For solving these instances, we used the linearization `BIL`. Hence, the impact of the general triangle inequalities lies in the root node relaxation value. For solving the equivalent integer convex quadratic formulation, we used the C interface of the solver `Gurobi` [69]. Parameter p is set to $0.4 \cdot |\mathcal{MC}|$ for `MIQCR`, and to $0.015 \cdot |\mathcal{MC} \cup \mathcal{T}|$ for `MIQCR-T`.

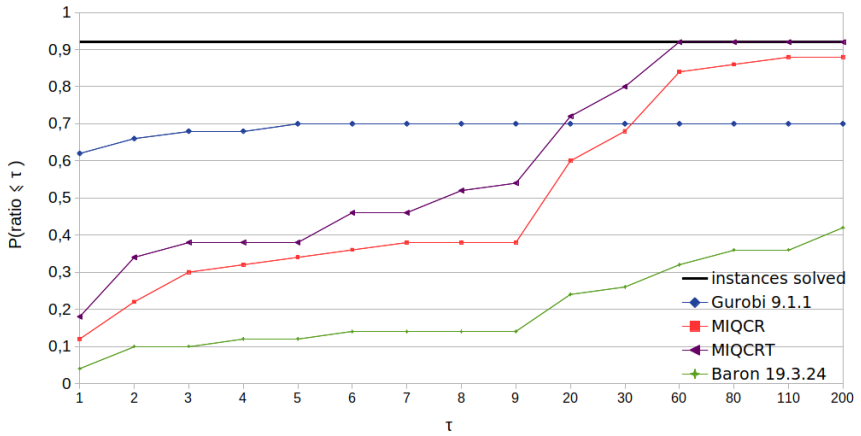


FIGURE 7: Performance profile of the total time for the $IQCP_5$ instances with $n = 10$ to 50 with a time limit of 2 hours.

In Figure 7, we present the performance profile of the CPU times for methods `MIQCR-T`, `MIQCR`, and for the solvers `Baron`and `Gurobi`. We observe that algorithm `MIQCR-T` outperforms the compared methods. In particular, `Baron` solve 21 instances, `Gurobi` solves 35 instances, `MIQCR` solves 44 instances, and `MIQCR-T` solves 46 instances out of 50 within the time limit. These results show the strong impact of the new valid inequalities on the sharpness of the root bound, since the initial gap is divided by a factor of 2.5 for `MIQCR-T` compared to `MIQCR`.

Conclusion

In this chapter, we consider the general problem (P) of minimizing a quadratic function subject to quadratic constraints where the variables can be integer or continuous. To solve (P), we start by designing an equivalent formulation to (P), then we solve a semi-definite program in order to build a strong quadratic convex relaxation of (P) that captures its tightness. A significant result is that our framework generalizes the standard linearization. We then develop an appropriate spatial branch-and-bound to solve the reformulated problem based on this relaxation. We also introduce quadratic inequalities, valid for our convex relaxation, that generalize the regular triangle inequalities. In addition to tighten the root relaxation, they also improve the behavior of the spatial branch-and-bound, since they involve the upper and lower bounds on the variables that move at each node of the search tree. Finally, we present a sub-gradient algorithm to solve our semi-definite relaxation. This last algorithm has two roles. First, it is used to separate the inequalities, and moreover it serves as a dual heuristic.

Chapter 3

Specialization of method MIQCR to two applications

In this chapter, we specialize method MIQCR on two problems. The first one is a classical combinatorial optimization problem: the quadratic Assignment Problem (QAP), for which the exact resolution is still a challenge even for the best algorithms of the literature. The second is a purely continuous problem: the Optimal Power Flow (OPF), for which only a few exact solution methods are available. For these two problems, we show that by taking advantage of their structure, we can improve the efficiency of method MIQCR and therefore deal with larger instances.

3.1 The Quadratic Assignment Problem (QAP) [50]

The Quadratic Assignment Problem (QAP) was introduced by Koopmans and Beckmann in 1957 as a mathematical model for the location of indivisible economical activities [92]. Consider the problem of allocating n facilities to n locations, with the cost being a function of the distance and flow between the facilities plus the costs associated with placing a facility at a certain location. The objective is to assign each facility to a location such that the total cost is minimized. Specifically, let n be the number of facilities and locations and denote by $\mathcal{I}$ the set $\mathcal{I} = \{1, 2, \ldots, n\}$. We are given three $n \times n$ input real matrices $A = (a_{ij})$, $B = (b_{kl})$, and $C = (c_{ik})$, where a_{ij} is the flow between facility i and facility j , b_{kl} is the distance between location k and location l, and c_{ik} is the cost of placing facility i at location k. A general formulation was introduced by Lawler [102]. In this version, we are given a four-dimensional array $Q = (q_{ijkl})$ of coefficients instead of the three matrices A, B and C, where the coefficients of C are shifted to the diagonal terms of matrix Q. The problem can be stated as:

$$
(QAP)\begin{cases}
\min f(x) = \displaystyle\sum_{(i,j,k,l)\in\mathcal{I}^4} q_{ijkl}x_{ij}x_{kl} & & \\
\text{s.t.} \displaystyle\sum_{i=1}^{n} x_{ij} = 1 & j \in \mathcal{I} & (27)\\
\displaystyle\sum_{j=1}^{n} x_{ij} = 1 & i \in \mathcal{I} & (28)\\
x_{ij} \in \{0,1\} & (i,j) \in \mathcal{I}^2 & (29)
\end{cases}
$$

where, in a facility location application, the decision variable x_{ij} corresponds to facility i being assigned to location j, and q_{ijkl} is the cost incurred by assigning facility i to location j and facility k to location l.

Sahni and Gonzalez [134] have shown that the (QAP) is $\mathcal{NP}$-hard. Although extensive research has been done for more than four decades, the (QAP) remains one of the hardest optimization problems. In the more general case, where the Hessian matrix of the objective function is fully dense, no exact algorithm can solve problems of size $n > 35$ in reasonable computational time nowadays. This problem appears in many applications such as chip design [76], scheduling, process communications, or turbine balancing. Another application discusses a campus planning problem with the objective to minimize the total walking distance between the buildings [45]. A related hospital lay-out model was also considered in [54].

Several algorithms have been introduced to solve (QAP) mainly based on branch-and-bound techniques. To compute polynomial bounds, many linearizations of the quadratic objective function have been proposed [3, 2, 63, 89, 102]. Other families of underestimators have also been introduced, and in particular those based on the eigenvalues of the formulation of [92]. It was first proposed by Finke et al. [56], and then improved and generalized in [70, 71, 72, 73, 128, 130]. Anstreicher and Brixius [11] developed a powerful bounding technique which is based on convexity arguments for quadratic programs. Finally, many semidefinite relaxations have been considered for this problem, see [88, 152, 153, 131, 129, 132]. In the following, we propose to construct convex quadratic reformulations of (QAP) based on two of these semidefinite relaxations: the R2 [153] and R4 [131] relaxations.

Equivalent quadratic convex formulations to (QAP)

We consider the problem of reformulating (QAP) by an equivalent quadratic 0-1 program. For this purpose, we propose to perturb the objective function $f(x)$ with quadratic functions that vanish on the feasible domain of (QAP). Thus, we consider the following three sets of functions:

(i) equalities $(x_{ij}^2 - x_{ij}) = 0$ for all $(i,j) \in \mathcal{I}^2$, that are satisfied for any binary variable.

(ii) equalities $x_{ij}x_{il} = 0$ for all $(i,j,l) \in \mathcal{I}^3 : j < l$ coming from the fact that a facility cannot be affected to more than one location.

(iii) equalities $x_{ij}x_{kj} = 0$ for all $(i,j,k) \in \mathcal{I}^3 : i < k$ coming from the fact that a location cannot be affected to more than one facility.

We now introduce real scalar parameters : $\beta_{ij}\ \forall \in (i,j) \in \mathcal{I}^2$, $\lambda_{ijl}\ \forall (i,j,l) \in \mathcal{I}^3 : j < l$, and $\lambda'_{ijk}\ \forall (i,j,k) \in \mathcal{I}^3 : i < k$, and the following quadratic function :

$$f_{\beta,\lambda,\lambda'}(x) = f(x) + \sum_{(i,j)\in\mathcal{I}^2} \beta_{ij}(x_{ij}^2 - x_{ij}) + \sum_{\substack{(i,j,l)\in\mathcal{I}^3 \\ j<l}} \lambda_{ijl}x_{ij}x_{il} + \sum_{\substack{(i,j,k)\in\mathcal{I}^3 \\ i<k}} \lambda'_{ijk}x_{ij}x_{kj}$$

It is clear that if x is feasible for (QAP), then $f(x) = f_{\beta,\lambda,\lambda'}(x)$. Hence, problem (QAP) is equivalently stated as:

$$(QAP_{\beta,\lambda,\lambda'}) \begin{cases} \min & f_{\beta,\lambda,\lambda'}(x) \\ \text{s.t.} & (27) - (29) \end{cases}$$

We now calculate parameters $(\beta^*, \lambda^*, \lambda'^*)$ that make convex function $f_{\beta,\lambda,\lambda}(x)$, and that maximize the value of $(\overline{QAP}_{\beta,\lambda,\lambda'})$ the continuous relaxation of $(QAP_{\beta,\lambda,\lambda'})$. For this, we consider the semidefinite relaxation (SDP_{R2}) introduced in [153]:

$$(SDP_{R2})\begin{cases}\min f(X,x) = \langle Q, X\rangle & & & \\ \text{s.t.} \quad (27)(28) & & & \\ \quad X_{ijij} - x_{ij} = 0 & (i,j) \in \mathcal{I}^2 & \leftarrow \beta_{ij} & (30)\\ \quad X_{ijil} = 0 & (i,j,l) \in \mathcal{I}^3,\ j<l & \leftarrow \lambda_{ijl} & (31)\\ \quad X_{ijkj} = 0 & (i,j,k) \in \mathcal{I}^3,\ i<k & \leftarrow \lambda'_{ijk} & (32)\\ \quad \begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix} \succeq 0 & & & (33)\\ \quad x \in \mathbb{R}^{n^2} \quad X \in \mathcal{S}_{n^2} & & & (34)\end{cases}$$

Theorem 4 *We have $v(SDP_{R2}) = v(\overline{QAP}_{\beta^*,\lambda^*,\lambda'^*})$. Moreover, the optimal $\beta^*, \lambda^*, \lambda'^*$ are the vectors of optimal dual variables associated to Constraints (30)-(32) respectively.*

A significant advantage of this formulation is that we remain in the space of the initial variables of (QAP). This formulation is efficient for solving exactly instances of (QAP) of medium size. However, for larger instances, it is limited by the quality of the bound it provides. This is why we propose to improve this reformulation, by adding auxiliary variables Y that model the products xx^T. More precisely, let $\phi_{ijkl} \geq 0\ \forall (i,j,k,l) \in \mathcal{I}^4\ i \neq k$ and $j \neq l$ be real scalar parameters, and the following function:

$$f_{\beta,\lambda,\lambda',\phi}(x,Y) = f_{\beta,\lambda,\lambda'}(x) - \sum_{\substack{(i,j,k,l)\in\mathcal{I}^4\\ i\neq k\\ j\neq l}} \phi_{ijkl}(x_{ij}x_{kl} - Y_{ijkl})$$

It is clear that if x is feasible for (QAP) and $Y_{ijkl} = x_{ij}x_{kl}$, then $f(x) = f_{\beta,\lambda,\lambda',\phi}(x,Y)$. To get the equivalence, we then have to enforce equalities $Y_{ijkl} = x_{ij}x_{kl}$. As $\phi_{ijkl} \geq 0$, and since additional variables Y_{ijkl} only appear in the objective function, it is easy to see that in an optimal solution Y_{ijkl} will be set to its smallest possible value. Hence, we get the following equivalent formulation to (QAP):

$$(QAP_{\beta,\lambda,\lambda',\phi})\begin{cases}\min \quad f_{\beta,\lambda,\lambda',\phi}(x,Y) & \\ \text{s.t.} \quad (27)-(29) & \\ \quad y_{ijkl} \geq x_{ij} + x_{kl} - 1 & (i,j,k,l)\in\mathcal{I}^4,\ i\neq k,\ j\neq l\\ \quad y_{ijkl} \geq 0 & (i,j,k,l)\in\mathcal{I}^4,\ i\neq k,\ j\neq l\end{cases}$$

To calculate the optimal parameters $(\beta^*, \lambda^*, \lambda'^*, \phi^*)$, we show that its continuous relaxation, $(\overline{QAP}_{\beta,\lambda,\lambda',\phi})$, reaches the value of the "Shor plus RLT" semidefinite relaxation by solving the semidefinite relaxation (SDP_{R4}) [131] which is of smaller size:

$$(SDP_{R4})\begin{cases}\min f(X,x) = \langle Q, X\rangle & & & \\ \text{s.t.} \quad (27)(28) & & & \\ \quad (30)-(34) & & & (35)\\ \quad -X_{ijkl} \leq 0 & (i,j,k,l)\in\mathcal{I}^4,\ k\neq i,\ j\neq l & \leftarrow \phi_{ijkl} & (36)\end{cases}$$

Theorem 5 *We have* $v(SDP_{R4}) = v(\overline{QAP}_{\beta^*,\lambda^*,\lambda'^*,\phi^*})$. *Moreover, the optimal* $\beta^*, \lambda^*, \lambda'^*$ *are deduced as in Theorem 4, and the optimal* ϕ^* *is the vector of optimal dual variables associated to Constraint (36).*

A proof of Theorem 5 comes from Theorem 1 and from the fact that constraints : $Y_{ijkl} - 1 - x_{ij} - x_{kl} \geq 0$ are redundant in (SDP_{R4}) [21].

Short computational results

We present some experiments on the Nugent instances from QAPLIB [38]. We compare our two algorithms, denoted by `Algorithm R2` and `Algorithm R4`, with the solver `Cplex`. The results are presented in table 1. Each line corresponds to one instance which *Name* is `nug_n`, where `n` is the number of facilities and of locations. Column *Gap* is $\left|\frac{Opt - Cont}{Opt}\right| * 100$, where *Cont* is the optimal value of the continuous relaxation, and *Opt* is the optimal solution value of the instance. Column *Time* is the total CPU time including the reformulation time. It is limited to 3 hours, and if the optimum is not found within this time, we present the final gap $(g\%)$, $g = \left|\frac{Opt - b}{Opt}\right| * 100$ where b is the best bound obtained within the time limit. We observe that `Algorithm R2` and `Algorithm R4` outperform `Cplex` in terms of initial gap and Total CPU time. Indeed, `Cplex` has always an initial gap of 100% since it starts its branch-and-bound procedure with a bound equals to 0. Then, we notice that the initial gap obtained with `Algorithm R4` is always smaller than the gap obtained by `Algorithm R2`. Finally, `Cplex` is unable to solve instances with more than 12 initial variables while `Algorithm R2` and `Algorithm R4` solve instances with up to 18 variables within the 3 hours of CPU time.

	`Algorithm R2`		`Algorithm R4`		`Cplex`	
Name	*Gap*	*Time*	*Gap*	*Time*	*Gap*	*Time*
nug03	0.0	1	0.0	0	100	2
nug06	5.5	4	0.9	1	100	3
nug12	9.3	33	3.5	69	100	1273
nug14	6.7	87	1.7	213	100	(49.0%)
nug15	8.4	153	2.4	380	100	(58.2%)
nug16a	6.2	309	1.8	658	100	(71.3%)
nug16b	12.9	1847	4.2	779	100	(56.6%)
nug17	7.5	1119	2.5	1281	100	(75.6%)
nug18	7.8	6436	3.0	3499	100	(76.9%)
nug20	9.1	(5.0%)	3.6	(2.6%)	100	(86.8%)
Average	**7.3**	**1110 (9)**	**2.4**	**764 (9)**	**100**	**426 (3)**

TABLE 1: Results for `Algorithm R2`, `Algorithm R4`, and `Cplex` for the Nugent instances ($n = 3$ to 20, time limit 3 hours).

3.2 The Optimal Power Flow (OPF) [49, 96]

The Optimal Power Flow (OPF) problem consists in the determination of the power production at different points of an electric network that minimizes a production cost. The electrical transmission network is modeled by a graph $G = (\mathcal{N}, \mathcal{E})$. Each

network point belongs to the set $\mathcal{N}$ of nodes (i.e the set of buses), and their connections (i.e. the set of transmission lines) are modeled by the set of edges $\mathcal{E}$. We assume that there is an electric demand at each node also called load. We distinguish two classes of nodes: $\mathcal{N} = \mathcal{N}_g \cup \mathcal{N}_d$, where $\mathcal{N}_g$ is the set of nodes that generates and flows the power (the generator nodes), and $\mathcal{N}_d$ is the set of nodes that only flows the power (the consuming nodes). The aim of the OPF problem is to satisfy demand of all buses while minimizing the total production costs of the generators such that the solution obeys Ohm's law and and Kirchhoff's law.

This problem is naturally formulated with complex variables. Let $\mathbf{Y} \in \mathbb{C}^{n\times n}$, where $|\mathcal{N}| = n$ be the admittance matrix, which has component $\mathbf{Y}_{i,k} = \mathbf{R}_{i,k} + \mathrm{j}\mathbf{I}_{i,k}$ for each line (i,k) of the network, and $\mathbf{R}_{i,i} = \mathbf{r}_{i,i} - \sum_{i\neq k}\mathbf{R}_{i,k}$, $\mathbf{I}_{i,i} = \mathbf{i}_{i,i} - \sum_{i\neq k}\mathbf{I}_{i,k}$, where $\mathbf{r}_{i,i}$ (resp. $\mathbf{i}_{i,i}$) is the shunt conductance (resp. susceptance) at bus i, and $\mathrm{j}^2 = -1$. Let p_i, q_i be the real and reactive power output of the generator node i, and $\mathbf{p}_i, \mathbf{q}_i$ the given real and reactive power output of the load node i. We consider here the complex voltage in the rectangular form: $V_i = e_i + \mathrm{j}f_i$ where $|V_i|^2 = e_i^2 + f_i^2$ is the voltage magnitude, and we denote by $\delta(i)$ the set of adjacent nodes of bus i. With the above notation, the OPF problem can be modeled by the well known rectangular formulation of [142]:

$$
(OPF)\ \begin{cases}
\min h(p) = \sum\limits_{i\in\mathcal{N}_g}\left(\mathbf{C}_{i,i}p_i^2 + \mathbf{c}_i p_i\right) & \\
\text{s.t.} & \\
\quad p_i - \mathbf{p}_i = \mathbf{R}_{i,i}(e_i^2+f_i^2) + \sum\limits_{k\in\delta(i)}\left[\mathbf{R}_{i,k}(e_ie_k + f_if_k) - \mathbf{I}_{i,k}(e_if_k - e_kf_i)\right] & i\in\mathcal{N}_g \quad (37)\\
\quad -\mathbf{p}_i = \mathbf{R}_{i,i}(e_i^2+f_i^2) + \sum\limits_{k\in\delta(i)}\left[\mathbf{R}_{i,k}(e_ie_k + f_if_k) - \mathbf{I}_{i,k}(e_if_k - e_kf_i)\right] & i\in\mathcal{N}_d \quad (38)\\
\quad q_i - \mathbf{q}_i = -\mathbf{I}_{i,i}(e_i^2+f_i^2) + \sum\limits_{k\in\delta(i)}\left[-\mathbf{I}_{i,k}(e_ie_k + f_if_k) - \mathbf{R}_{i,k}(e_if_k - e_kf_i)\right] & i\in\mathcal{N}_g \quad (39)\\
\quad -\mathbf{q}_i = -\mathbf{I}_{i,i}(e_i^2+f_i^2) + \sum\limits_{k\in\delta(i)}\left[-\mathbf{I}_{i,k}(e_ie_k + f_if_k) - \mathbf{R}_{i,k}(e_if_k - e_kf_i)\right] & i\in\mathcal{N}_d \quad (40)\\
\quad \underline{\mathbf{v}}_i \le e_i^2 + f_i^2 \le \overline{\mathbf{v}}_i & i\in\mathcal{N} \quad (41)\\
\quad \underline{\mathbf{p}}_i \le p_i \le \overline{\mathbf{p}}_i & i\in\mathcal{N}_g \quad (42)\\
\quad \underline{\mathbf{q}}_i \le q_i \le \overline{\mathbf{q}}_i & i\in\mathcal{N}_g \quad (43)\\
\quad (e,f)\in(\mathbb{R}^n,\mathbb{R}^n),\ (p,q)\in(\mathbb{R}^{|\mathcal{N}_g|},\mathbb{R}^{|\mathcal{N}_g|}) & \quad (44)
\end{cases}
$$

where $\mathbf{C} \in \mathbf{S}^+_{|\mathcal{N}_g|}$ is a diagonal and semi-definite matrix, $\mathbf{c} \in \mathbb{R}^{|\mathcal{N}_g|}$ is the vector of linear costs of the power injection at each generator node, $(\underline{\mathbf{v}}, \overline{\mathbf{v}}) \in (\mathbb{R}^n, \mathbb{R}^n)$ are the bounds on the voltage magnitude, and $(\underline{\mathbf{p}}, \overline{\mathbf{p}}, \underline{\mathbf{q}}, \overline{\mathbf{q}}) \in (\mathbb{R}^{|\mathcal{N}_g|}, \mathbb{R}^{|\mathcal{N}_g|}, \mathbb{R}^{|\mathcal{N}_g|}, \mathbb{R}^{|\mathcal{N}_g|})$.

(OPF) has $2(n+2|\mathcal{N}_g|)$ variables, $2n$ quadratic equalities (37)-(40) that enforces the active and reactive power balances at each node, $2n$ quadratic inequalities (41) that models the voltage magnitude, and $4|\mathcal{N}_g|$ box constraints (42)-(43). We observe that the structure of this formulation is specific. First, only variables p are involved into the objective function. Moreover, the matrix $\mathbf{C}$ is diagonal and positive semi-definite, hence function $h(p)$ is convex and separable. It follows that the non convexities only come from the quadratic constraints (37)-(41) where variables e and f are only involved into quadratic forms, while variables p and q only in linear forms.

The first results of the literature for solving the OPF problem were focused on optimal local solutions, mostly by adapting interior point methods, see, e.g., [149,

142, 86, 148]. In the context of global optimization, one requires furthermore to determine lower bounds on the OPF problem. For this, the second-order cone programming (SOCP) and the semidefinite programming relaxations were first used (see [84, 15, 14, 101]). The most used semidefinite relaxation, also named rank relaxation, leads to very tight lower bounds on the OPF problem. In particular, it was proven in [64] that this relaxation is exact for a restricted class of problems and under some assumptions. In other cases, where it is not exact, it can be strengthen until the optimal solution value, following the ideas of the hierarchy of moment relaxation problems ([97, 125]). This approach was specialized in the context of the OPF problem in [87], and showed its efficiency to solve small-sized problems. It was also used in [118] to strengthen the lower bounds for larger problems. Unfortunately, in practice, using interior point methods for solving several large SDP relaxations, which sizes increase at each rank of the hierarchy, is intractable for large networks. Several specialized algorithms that exploit the sparsity of power networks were thus proposed as in [84, 85, 120, 118, 109]. More recently, several cheaper computable convex relaxations were introduced for the OPF problem. For instance, linear and quadratic envelopes for trigonometric functions in the polar formulation of the OPF problem are constructed in [39, 40, 41], and strong SOCP relaxations were introduced in [91]. These polynomial bounds can then be used within a spatial branch-and-bound framework to solve the problem to global optimality.

In this section we present two specializations of our approaches to (OPF). In the first method RC-OPF [68, 49], we focus on the semidefinite relaxation to be solved to compute the best convex quadratic relaxation. We show that it is possible to reach the bound provided by the "shor plus RLT" semidefinite relaxation by solving the rank relaxation. Thus, the time to compute the best relaxation is significantly reduced. In RC-OPF, the equivalent problem has a quadratic convex objective function and linear constraints. Moreover, it has a size of $\mathcal{O}((2n)^2)$, since it relies on the introduction of $(2n)^2$ additional variables that model all the possible products of the original variables. Unfortunately, in practice enforcing these $(2n)^2$ equalities can be very time consuming. In the second method COPF [96], we propose a compact convex quadratic relaxation which also reaches the rank relaxation value. This relaxation has only $\mathcal{O}(n)$ auxiliary variables and constraints, and quadratic convex objective function and constraints. We thus get a significantly smaller convex relaxation than is method RC-OPF, that is as tight as the rank relaxation. A key advantage of our compact relaxation is that we only have to enforce the equality between $2n$ additional variables and their corresponding products to prove global optimality.

The algorithm RC-OPF [68, 49]

The equivalent quadratic formulation of RC-OPF directly follows from method MIQCR. For simplicity, we start by rewriting the initial equality constraints (37)-(40) by using the notation $y = (p, q) \in \mathbb{R}^{2|\mathcal{N}_g|}$ and $x = (e, f) \in \mathbb{R}^{2n}$ as follows:

$$\langle A_r, xx^T \rangle + a_r^T y = b_r \qquad r \in \mathcal{C} \tag{45}$$

where $\mathcal{C} = (\mathcal{N}_g, \mathcal{N}_g, \mathcal{N}_d, \mathcal{N}_d)$, with $|\mathcal{C}| = 2n$, and $\forall\, r \in \mathcal{C}$, $A_r \in \mathbf{S}_{2n}$ is the Hessian sub-matrix of the r^{th} constraints within (37)-(40), that corresponds to the quadratic terms involving variables e and f only, $a_r \in \mathbb{R}^{2|\mathcal{N}_g|}$ is the vector of linear coefficients of constraint r, and $b \in \mathbb{R}^{2n}$ where b_r is the the right-hand side of constraint r.

Now, we consider a semi-definite matrix $S \in \mathbf{S}^+_{2n}$, and the matrix variable $X \in \mathbf{S}_{2n}$ that modeled the products xx^T, and we obtain the following family of equivalent formulations to (OPF):

$$(OPF_S)\begin{cases} \min f_S(p,x,X) = \sum_{i\in\mathcal{N}_g}\left(\mathbf{C}_{i,i}p_i^2 + \mathbf{c}_i p_i\right) + \langle S, xx^T - X\rangle & \\ \text{s.t.} & \\ \quad (42)(43) & \\ \quad \langle A_r, X\rangle + a_r^T y = b_r & r \in \mathcal{C} \quad (46)\\ \quad \underline{\mathbf{v}}_i \le X_{i,i} + X_{i+n,i+n} \le \overline{\mathbf{v}}_i & i \in \mathcal{N} \quad (47)\\ \quad X = xx^T & (48)\\ \quad x = (e,f) \in \mathbb{R}^{2n},\ y = (p,q) \in \mathbb{R}^{2|\mathcal{N}_g|}, X \in \mathbb{S}_{2n} & (49) \end{cases}$$

We now build $(\overline{OPF}_S)$, the convex relaxation of (OPF_S), obtained by replacing the non-convex equalities (48) by the McCormick envelopes. For this, we need upper and lower bounds on each variable x_i, we use trivial bounds that can be deduced from Constraints (41), i.e. $\ell_i = -\sqrt{\overline{\mathbf{v}}_i}$ and $u_i = \sqrt{\overline{\mathbf{v}}_i}$.

We know by Theorem 1, that we can compute the best matrix S^* that maximizes the value of $(\overline{OPF}_S)$ using the dual optimal solutions of the "Shor plus RLT" semi-definite relaxation of (OPF). We state in Theorem 6 that the best matrix S^* can also be derived from an optimal dual solution of (SDP_{OPF}), the rank relaxation of (OPF):

$$(SDP_{OPF})\begin{cases} \min h(Y,p) = \sum_{i\in\mathcal{N}_g}\left(\mathbf{C}_{i,i}Y_{i,i} + \mathbf{c}_i p_i\right) & \\ \text{s.t.} & \\ \quad \langle A_r, X\rangle + a_r^T y = b_r & \forall r \in \mathcal{C} \quad \leftarrow \beta_r \quad (50)\\ \quad X_{i,i} + X_{i+n,i+n} \le \overline{\mathbf{v}}_i & i \in \mathcal{N} \quad \leftarrow \overline{\alpha}_i \quad (51)\\ \quad -X_{i,i} - X_{i+n,i+n} \le -\underline{\mathbf{v}}_i & i \in \mathcal{N} \quad \leftarrow \underline{\alpha}_i \quad (52)\\ \quad \underline{\mathbf{p}}_i \le p_i \le \overline{\mathbf{p}}_i & i \in \mathcal{N}_g \quad (53)\\ \quad \underline{\mathbf{q}}_i \le q_i \le \overline{\mathbf{q}}_i & i \in \mathcal{N}_g \quad (54)\\ \quad \begin{bmatrix} 1 & y^T \\ y & Y \end{bmatrix} \succeq 0 & (55)\\ \quad X \succeq 0 & (56)\\ \quad y \in \mathbb{R}^{2|\mathcal{N}_g|}, (Y,X) \in (\mathbf{S}_{2|\mathcal{N}_g|}, \mathbf{S}_{2n}) & \end{cases}$$

Theorem 6 *We have $v(SDP_{OPF}) = v(\overline{OPF}_{S^*})$. Moreover, the optimal matrix S^* can be derived from the dual optimal values of (SDP_{OPF}) as follows:*

$$S^* = \sum_{r\in\mathcal{C}} \beta_r^* A_r + \text{diag}(\alpha^*)$$

where β^* is the vector of optimal dual variables associated to Constraints (50), vectors $\overline{\alpha}^*$ and $\underline{\alpha}^*$ are the vectors of optimal dual variables associated to Constraints (51) and (52) respectively, $\alpha^* = \left((\overline{\alpha}^* - \underline{\alpha}^*), (\overline{\alpha}^* - \underline{\alpha}^*)\right)$, and $\text{diag}(\alpha^*)$ is the diagonal matrix where each diagonal term equals α_i^*.

To prove Theorem 6, it is sufficient to show that the inequalities of set $\mathcal{MC}$ are redundant in (SDP_{OPF}).

Hence, for computing the best convex relaxation of the OPF problem, we need to solve a semi-definite relaxation with a significant fewer number of constraints than the "Shor plus RLT" classically used in method MIQCR. Moreover, doing this ensure us to reach the best possible bound within our family of equivalent convex relaxation.

The algorithm COPF [96]

We now present method COPF. The idea is to build an equivalent problem to (OPF), where the original constraints are convexified using only $2n$ auxiliary variables $z = (z^e, z^f) \in \mathbb{R}^{2n}$ that model the squares of the initial variables e and f:

$$\begin{cases} z_i^e = e_i^2 & i \in \mathcal{N} \quad (57) \\ z_i^f = f_i^2 & i \in \mathcal{N} \quad (58) \end{cases}$$

Using these new variables it is easy to rewrite Constraints (41) into a convex form by simply linearizing them, and we get:

$$\underline{\mathbf{v}}_i \leq z_i^e + z_i^f \leq \overline{\mathbf{v}}_i \quad i \in \mathcal{N} \quad (59)$$

We now handle the equality constraints. For this, the first step is to transform each equality (45) into two inequalities:

$$\begin{cases} \langle A_r, xx^T \rangle + a_r^T y \leq b_r & r \in \mathcal{C} \\ \langle -A_r, xx^T \rangle - a_r^T y \leq -b_r & r \in \mathcal{C} \end{cases}$$

Then, to make them convex, we apply the smallest eigenvalue method introduced in [75]. Let $\lambda_r = \lambda_{min}(A_r)$ ($\lambda'_r = \lambda_{min}(-A_r)$, resp.) be the smallest eigenvalue of matrix A_r ($-A_r$, resp.), and the following inequalities:

$$\begin{cases} \langle A_r, xx^T \rangle + a_r^T y - \lambda_r \sum_{i=1}^{2n} (x_i^2 - z_i) \leq b_r & r \in \mathcal{C} \quad (60) \\ \langle -A_r, xx^T \rangle - a_r^T y - \lambda'_r \sum_{i=1}^{2n} (x_i^2 - z_i) \leq -b_r & r \in \mathcal{C} \quad (61) \end{cases}$$

We recall that for all $i \in \mathcal{N}$, $x_i = e_i$, $x_{i+n} = f_i$, $z_i = z_i^e$, and $z_{i+n} = z_i^f$. It is easy to prove that Constraint (60)-(61) are convex. Moreover, if $\forall i \; x_i^2 = z_i$, they are equivalent to equalities (45). Hence, by replacing Constraints (37)-(41) by Constraints (57)-(61), we obtain an equivalent problem to (OPF) where the only non-convexity remains into the equalities (57)-(58). Then, we get a quadratic convex relaxation of (OPF) by relaxing the latter equalities with the following set of convex inequalities:

$$\mathcal{D} = (x, z) \begin{cases} z_i \leq (u_i + \ell_i) x_i - u_i \ell_i \\ z_i \geq x_i^2 \end{cases}$$

We now rewrite the objective function of (OPF). Let $(\phi, \alpha) \in (\mathbb{R}^{2n}, \mathbb{R}^{2n})$ be two vector parameters, we build the following parameterized function:

$$h_{\phi,\alpha}(x,y,z) = h(y) + \sum_{r\in\mathcal{C}}\Big(\phi_r(\langle A_r, xx^T\rangle + a_r^T y - b_r)\Big) + \sum_{i\in\mathcal{N}}\alpha_i\Big(x_i^2 + x_{i+n}^2 - z_i - z_{i+n}\Big)$$

where $h(y)$ is the initial objective function. Observe that there exist parameters (ϕ, α) such that $h_{\phi,\alpha}$ is a convex function. Indeed, as mentioned above, function $h(y)$ is convex and separable. Now, the two additional terms are linear in y and z, and pure quadratic in x. By taking $\forall\, r \in \mathcal{C}$, $\phi_r = 0$, and $\forall i \in \mathcal{N}$, α_i any non negative value, the associated function $h_{\phi,\alpha}(x,y,z)$ is obviously convex.

We thus obtain the following family of quadratic convex relaxations to (OPF):

$$(\overline{OPF}_{\phi,\alpha}) \begin{cases} \min h_{\phi,\alpha}(x,y,z) \\ \text{s.t.} \\ \quad (42)(43)(57) - (61) \\ \quad (x,z) \in \mathcal{D} & (62) \\ \quad x = (e,f) \in \mathbb{R}^{2n},\ y = (p,q) \in \mathbb{R}^{2|\mathcal{N}_g|} & (63) \end{cases}$$

Problem $(\overline{OPF}_{\phi,\alpha})$ is a compact quadratic convex relaxation to (OPF), since we only add $\mathcal{O}(n)$ variables and constraints to the original formulation

We are now interested in the best parameters (ϕ^*, α^*) that maximize the optimal value of $(\overline{OPF}_{\phi,\alpha})$ while making convex the parameterized function $h_{\phi,\alpha}$. We state in Theorem 7 that these best parameters can be deduced from the dual optimal solution of the rank relaxation of (OPF).

Theorem 7 *We have $v(SDP_{OPF}) = v(\overline{OPF}_{\phi^*,\alpha^*})$. Moreover, the best parameters ϕ^*, α^* can be deduced as in Theorem 4.*

Some computational results

We illustrate on some experiments the behavior of the algorithms `RC-OPF` and `COPF` for exact solution of instances of the *PG-lib* library [126]. We compare `COPF` with the non-linear solver `Baron` [133]. Our experiments were carried out on a server with 2 CPU Intel Xeon each of them having 12 cores and 2 threads of 2.5 GHz and $4 * 16$ GB of RAM using a Linux operating system. We set the time limit to 3 hours for all methods. For the solver `Baron`, we use the multi-threading version of `Cplex 12.9` [82] with up to 64 threads. For methods `COPF` and `RC-OPF`, we used the semidefinite solver `Mosek` [122] for solving semidefinite programs. At each node of the spatial branch-and-bound, we used the solver `Mosek` for solving the QCQP of method `COPF`, and the solver `Cplex 12.9` for solving QP of method `RC-OPF`. For computing feasible local solutions, we use the local solver `Ipopt` [147].

For our experiences, we considered medium-sized data of power networks having 3 to 300 buses, and we took the formulation of the OPF described by Constraints (37)-(43). We report in Table 1 the characteristics of each instance: its *Name*, and the number of *Buses*, *Generators*, and *Lines* of the considered power network. We also indicate in the column *Opt* the best solution found by `COPF` and `RC-OPF`, within 3

Name	*Buses*	*Generators*	*Lines*	*Opt*	$\lvert(x,y)\rvert$
caseWB2	2	1	1	9.0567	6
caseWB3	3	1	2	417.2453	8
pglib_opf_case3_lmbd	3	3	3	5 694.5249	12
caseWB5	5	2	6	13.7797	14
pglib_opf_case5_pjm	5	5	5	14 997.0431	20
case6ww	6	3	11	3 126.3145	18
pglib_opf_case14_ieee	14	5	20	2 178.0893	38
pglib_opf_case24_ieee_rts	24	33	38	63 344.6382	114
pglib_opf_case30_as	30	6	41	801.5451	72
pglib_opf_case30_ieee	30	6	41	6 592.9534	72
pglib_opf_case39_epri	39	10	46	133 801.7063	98
pglib_opf_case57_ieee	57	7	80	37 589.3248	128
pglib_opf_case73_ieee_rts	73	99	120	189 741.3755	344
pglib_opf_case89_pegase	89	12	210	106 696.9325	202
pglib_opf_case118_ieee	118	54	186	96 881.5257	344
pglib_opf_case162_ieee_dtc	162	12	284	84 785.2377	348
pglib_opf_case179_goc	179	29	263	750 158.5809	416
pglib_opf_case200_activ	200	38	245	27 557.5673	476
pglib_opf_case240_pserc	240	143	448	-	766
pglib_opf_case300_ieee	300	69	411	-	738

TABLE 2: Characteristics of the considered instances of *PG-lib* library.

hours of computing time. The column $\lvert(y,x)\rvert$ specifies the dimension of the variable vector (y,x) in (OPF).

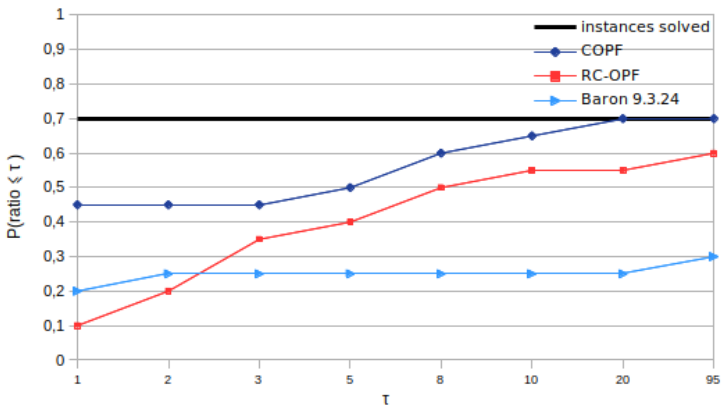


FIGURE 8: Performance profile of the total time for networks with 2 to 300 buses (time limit 3 hours).

In Figure 8, we present the performance profile of the CPU times for methods COPF, RC-OPF, and the solver Baron 19.3.24. We observe that COPF and RC-OPF significantly outperform the solver Baron. In fact Baron solves to optimality only 6 instances out of the 20 considered, the largest of which is pglib_opf_case14_ieee. The two other approaches are more efficient, since they solve within 3 hours of CPU time, 12 instances for RC-OPF and 14 for COPF. Moreover, this profile shows that COPF is faster than RC-OPF for these instances.

Instance		COPF					RC-OPF				
Name	*Gap*	$\|z\|$	*UB*	*SDP*	*CPU*	*Nodes*	$\|z\|$	*UB*	*SDP*	*CPU*	*Nodes*
caseWB2	1.947	4	0	1	2	25	8	0	1	-	44265
caseWB3	0.000	6	0	1	1	1	14	0	1	1	0
case3_lmbd	0.000	6	0	1	1	1	18	0	1	1	0
caseWB5	28.093	10	0	1	357	8 267	34	0	1	-	292231
case5_pjm	0.000	10	0	1	1	0	34	0	1	1	0
case6ww	0.000	12	3	1	4	0	56	0	1	1	0
case14_ieee	0.000	28	0	1	1	0	108	0	1	2	0
case24_ieee_rts	0.000	48	0	1	1	0	184	42	1	43	0
case30_as	0.000	60	8	0	8	0	224	0	1	1	0
case30_ieee	0.000	60	0	1	1	0	224	5	1	6	0
case39_epri	0.000	78	1	1	2	0	262	5	1	6	0
case57_ieee	0.003	114	167	1	-	2 659	426	31	1	-	2193
case73_ieee_rts	0.000	146	67	2	70	0	578	191	3	196	0
case89_pegase	0.000	178	31	3	34	0	1002	72	3	76	0
case118_ieee	0.005	236	933	2	-	973	952	2864	5	-	1023
case162_ieee_dtc	1.890	324	342	4	-	353	1444	518	7	-	1033
case179_goc	0.034	358	1517	5	-	343	1246	3045	10	-	1127
case200_activ	0.000	400	420	5	426	0	1380	2156	10	2172	0
case240_pserc	-	480	3624	27	-	219	1872	3328	27	-	619
case300_ieee	-	600	3295	42	-	29	2236	3124	41	-	1171

TABLE 3: Initial gap, Sizes, CPU times and Nodes for methods COPF and RC-OPF.

Finally, we present in Table 2, a detailed comparison between methods COPF and RC-OPF. Column *Gap*: $= \left|\frac{Opt - Cont}{Opt}\right| * 100$, is the initial gap at the root node of the branch-and-bound, where *Cont* is the optimal value of the rank relaxation, *Opt* is defined as in Table 1, and $-$ indicates that no feasible solution has been found by the algorithm. Column $|z|$ specifies the number auxiliary variables in the relaxation, that is also the number of non-convex equalities to force during the spatial branch-and-bound. Columns *UB* and *SDP* report the CPU times in seconds for finding an initial upper bound and for solving the rank relaxation. Column *CPU* is the total CPU time, where $-$ means that the instance is unsolved within the time limit of 3 hours. Finally, Column *Nodes* is the number of nodes visited by the branch-and-bound. A first observation concerns the reformulation time which is significantly shorter than the global resolution time (always less than 42 seconds), while the rank relaxation is solved directly with a standard semidefinite solver. The zero gap at the root of the branch-and-bound for 12 instances out of the 20 considered confirms the strength of the rank relaxation for the OPF problem. On the other hand, these experiments clearly show that the exact resolution of the instances where the gap is non-zero remains very hard. Indeed, the method RC-OPF does not solve instances having only 2 or 5 buses in the considered power network. In fact, during its branch-and-bound, RC-OPF does not increase the lower bound, despite a large number of explored nodes. This is not the case for COPF, which, with a much smaller number of nodes, slightly increases the lower bound over the course of the branch-and-bound, even for the largest instances. This is because the number of auxiliary variables and relaxed equalities $z_i = x_i^2$ is strongly reduced in COPF. Let us finally note that determining a feasible solution is also difficult in practice, at least with a generic interior point algorithm. Moreover, the CPU time for this step is not constant since for example it goes from 400 to 1380 seconds for the instance pglib_opf_case200_activ.

Chapter 4

Global solution of binary unconstrained polynomial problems

In this chapter, we focus on finding equivalent convex formulations of another subclass of MINLP where the objective function can be any polynomial of binary variables. Such a problem can be stated as follows:

$$(UBP)\begin{cases} \min F(x) = \sum_{p=1}^{m} a_p \prod_{i \in \mathcal{M}_p} x_i \\ \text{s.t.} \\ \quad x_i \in \{0,1\} \qquad\qquad i \in \mathcal{I} \end{cases}$$

where $\mathcal{I} = \{1, .., n\}$, function $F(x)$ is an $n-$variable polynomial of degree d having m monomials. For a monomial p, we denote by $\mathcal{M}_p$ the subset of $\mathcal{I}$ containing the indexes of the variables involved in p. It follows that $d = \max_p |\mathcal{M}_p|$.

We start by the presentation of a direct convexification that we relate and compare to the αbb [6]. With this first solution algorithm, we compute an equivalent convex formulation to (UBP) in an extended space of variables, but with the same degree as the original problem. Following the spirit of Quadratic Convex Reformulation approaches, the convex formulation is calculated thanks to the solution of a semidefinite relaxation involving an approximation of the non constant Hessian matrix of (UBP). The performance of this approach relies on this approximation than can be weak for many problems.

Another way to address this problem is to reformulate it first into an equivalent quadratic problem in a step of *quadratization*. The main advantage is that the Hessian matrix of the obtained problem is constant. Motivated by the fact that the application of methods `QCR` and `MIQCR` is inefficient after a quadratization step, we present a convexification algorithm suitable for (UBP). We call this method `PQCR` (`Polynomial Quadratic Convex Reformulation`). Our contribution is that we use the features of the quadratization to optimize the convexification phase. As several quadratizations can be applied to (UBP), it is interesting to evaluate the impact of this choice on the sharpness of the bound obtained after convexification. We give some first answers to this question by characterizing families of quadratizations that provide the same bounds. Finally, the last phase of `PQCR` consists in solving the convexified problem using a MIQP solver.

4.1 A convex reformulation obtained by direct convexification [103]

In this section, we present a direct convexification of problem (UBP). We begin by recalling the main features of the αbb algorithm [6] that applies to MINLPs. It is a branch-and-bound based on a convex underestimator:

$$g_\alpha(x) = F(x) + \sum_{i=1}^{n} \alpha_i (x_i - u_i)(x_i - \ell_i)$$

A key point of this algorithm is to determine a vector $\alpha \in \mathbb{R}^n$ that gives a tight underestimation. Several ways for computing a good α were proposed [6, 5, 9, 110, 116]. The most efficient is derived from the Scaled Gerschgorin's Theorem [66] and is computed as follows. The authors first approximate the Hessian matrix H of $F(x)$ by use of the interval matrix $[H] = [\underline{H}, \overline{H}] = \{H \in \mathcal{S}_n : \underline{H} \leq H \leq \overline{H}\}$ that is a family of matrices where $\underline{H}, \overline{H} \in \mathcal{S}_n$ are given matrices, and the inequality is considered element-wise. Then, a vector $\alpha \in \mathbb{R}^n$ that makes any matrix $H \in [H]$ positive semi-definite can be computed as:

$$\alpha_i = \max\left\{0, -\left(\underline{h}_{ii} - \sum_{j \neq i} \max\{|\underline{h}_{ij}|, |\overline{h}_{ij}|\} \frac{d_j}{d_i}\right)\right\} \quad \text{for any } d > 0$$

where $\underline{h}_{ij}$ ($\overline{h}_{ij}$ resp.) is the element (i, j) of matrix $\underline{H}$ ($\overline{H}$ resp.)

In [6], the authors also proposes to compute the vector α that minimizes the separation distance between $F(x)$ and its underestimator $g_\alpha(x)$ using semi-definite programming. In fact they first build a matrix M such that, for any $H \in [H]$, $\lambda_{min}(M) \leq \lambda_{min}(H)$ as follows:

$$(M)_{ij} = \begin{cases} \underline{h}_{ii} + \dfrac{1}{2}\displaystyle\sum_{k \neq i} (\underline{h}_{ik} - \overline{h}_{ik}) & \text{if } i = j \\ \dfrac{1}{2}(\underline{h}_{ij} + \overline{h}_{ij}) & \text{if } i \neq j \end{cases}$$

Then, they look for a diagonal matrix $\Delta \in \mathcal{S}_n$ that is the optimal solution of the following semi-definite program:

$$SDP_{\alpha bb} \begin{cases} \min \langle \Delta, (u - \ell)(u - \ell)^T \rangle \\ M + \Delta \succeq 0 \end{cases}$$

In practice, the solver `ANTIGONE` [115] that implements the αbb, does handle polynomial functions only with this α underestimator. Indeed, in this case, each non-linear function is decomposed into a sum of terms belonging to one of several categories : linear, bi-linear, tri-linear, convex or general non-convex. Then the treatment of each terms is as follows: linear and convex are not modified, and bi-linear, tri-linear are linearized thanks to the McCormick [111] envelopes and extensions [6, 5, 9, 110, 116]. Then, the remaining general non convex terms are underestimated with a the α underestimator.

We now present an equivalent convex formulation of (UBP) that is also based on the perturbation of the Hessian matrix of $F(x)$. We introduce $\frac{n(n-1)}{2}$ new variables

Y_{ij} that represent the products $x_i x_j$, and we build the following perturbed function:

$$F_{\varphi,\Phi}(x,Y) = \sum_{p=1}^{m} a_p \prod_{i\in\mathcal{M}_p} x_i + \sum_{i=1}^{n} \varphi_i(x_i^2 - x_i) + \sum_{i=1}^{n}\sum_{j>i}^{n} \Phi_{ij}(x_i x_j - Y_{ij})$$

We have the equality $F_{\varphi,\Phi}(x,Y) = F(x)$ if $x \in \{0,1\}^n$, and $Y_{ij} = x_i x_j$, for all $(i,j) \in \mathcal{I}^2 : i < j$, and problem (UBP) is thus equivalent to the following linearly polynomial program $(UBP_{\varphi,\Phi})$:

$$(UBP_{\varphi,\Phi}) \begin{cases} \min F_{\varphi,\Phi}(x,Y) \\ \text{s.t.} \\ \quad (x,Y) \in \mathcal{F} \\ \quad x_i \in \{0,1\} \qquad i \in \mathcal{I} \end{cases}$$

where the set $\mathcal{F}$ of Fortet [61] inequalities is:

$$\mathcal{F} = (x,Y) \begin{cases} Y_{ij} \leq x_i & (i,j) \in \mathcal{I}^2 : i < j \\ Y_{ij} \leq x_j & (i,j) \in \mathcal{I}^2 : i < j \\ Y_{ij} \geq x_i + x_j - 1 & (i,j) \in \mathcal{I}^2 : i < j \\ Y_{ij} \geq 0 & (i,j) \in \mathcal{I}^2 : i < j \end{cases}$$

Let us first notice that in the binary case, the underestimator $g_\alpha(x)$ corresponds to the function $F_{\varphi,\Phi}(x,Y)$, when Φ_{ij} is fixed at 0 for all (i,j). We can observe that even in this case, the two approaches are conceptually different. Indeed, our method constructs a function equivalent to $F(x)$ for each feasible point of (UBP), and the perturbation φ is computed only once before the exploration of the search tree. Then, a branch-and-bound process whose lower bound is the continuous relaxation value of the equivalent and convex formulation is performed. The αbb is based on a convex underestimator computed at each node of the search tree which serves as a lower bound. In this context, α must be recalculated at each node to ensure the convergence of the whole algorithm.

We are interested in the best parameters φ^* and Φ^* that maximize the value of $(\overline{UBP}_{\varphi,\Phi})$, the continuous relaxation of $(UBP_{\varphi,\Phi})$, and make convex $F_{\varphi,\Phi}(x,Y)$. More formally, we want to solve the following problem:

$$(OPT_{\varphi,\Phi}) \begin{cases} \max\limits_{H+\text{diag}(\varphi)+\Phi+\succeq 0} & v(\overline{UBP}_{\varphi,\Phi}) \end{cases}$$

where H is the Hessian matrix of $F(x)$. Problem $(OPT_{\varphi,\Phi})$ is clearly difficult to solve since H is not constant, but it is possible to calculate feasible solutions. The idea is to determine a matrix M, such that $M \preceq H$, for all $H \in [H]$, $[H]$ being the interval matrix associated to the Hessian H of $F(x)$. Then, by computing a matrix perturbation such that $M + diag(\varphi) + \Phi \succeq 0$, we can deduce that the matrix $H + diag(\varphi) + \Phi$ is positive semidefinite. Indeed, since for all $H \in [H]$, we have $H + diag(\varphi) + \Phi \succeq M + diag(\varphi) + \Phi \succeq 0$. Given a matrix M, we can therefore deduce from Theorem 1, that feasible $(\varphi, \Phi = \phi^1 + \phi^2 - \phi^3 - \phi^4)$ for $(OPT_{\varphi,\Phi})$ can be

derived from the optimal dual variables of (SDP_M):

$$(SDP_M)\begin{cases} \min f(X,x) = \langle M, X\rangle & \\ \text{s.t.} & \\ X_{ii} = x_i & \forall i \in \mathcal{I} \leftarrow \varphi_i \\ X_{ij} \leq x_i & (i,j) \in \mathcal{I}^2 : i < j \leftarrow \phi^1_{ij} \\ X_{ij} \leq x_j & (i,j) \in \mathcal{I}^2 : i < j \leftarrow \phi^2_{ij} \\ X_{ij} \geq x_i + x_j - 1 & (i,j) \in \mathcal{I}^2 : i < j \leftarrow \phi^3_{ij} \\ X_{ij} \geq 0 & (i,j) \in \mathcal{I}^2 : i < j \leftarrow \phi^4_{ij} \\ \begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix} \succeq 0 & \\ x \in \mathbb{R}^n \quad X \in \mathcal{S}_n & \end{cases}$$

The performance of the method depends on the computation of matrix M, since it impacts the value of (SDP_M) and by equivalence that of $(\overline{UBP}_{\varphi,\Phi})$. We can of course use the value of M described above in the context of the *αbb*, but characterizing a good matrix M constitutes a future work.

4.2 PQCR, a global solution algorithm for polynomials programs [52, 53]

In this section, we describe method PQCR. It can be decomposed into three phases. We start by a quadratization of (UBP), obtaining a MIQP with linear constraints. This equivalent quadratic formulation depends on the quadratization used. Then, in a second phase, we propose a convexification based on the quadratization. The third phase of PQCR consists in submitting the convex quadratic reformulation to a MIQP solver.

Quadratizations of (UBP)

We start by a description of how we build equivalent quadratic formulations to (UBP). For this, in each monomial of degree 3 or greater, we iteratively replace each product of two variables by an additional variable. Thus, each auxiliary variable models a product of 2 variables which can be initial or auxiliary variables. We start with the formal definition of a quadratization.

Definition 1 *Quadratization* $\mathcal{Z} = (\mathcal{A}, \mathcal{E}, f, s)$
Let $\mathcal{A} = \{n+1, .., N\}$ *be the set of indexes of the additional variables, we define* $\mathcal{E}_i$ $\forall i \in I \cup \mathcal{A}$ *and mappings* $f : \mathcal{A} \to I \cup \mathcal{A}$ *and* $s : \mathcal{A} \to I \cup \mathcal{A}$ *as follows:*

- *If* $i \in I$, *i.e.* x_i *is an initial variable, then we set* $\mathcal{E}_i = \{i\}$
- *If* $i \in \mathcal{A}$, *i.e.* x_i *is an additional variable, then there exist two indexes* $(f(i), s(i)) \in (I \cup \mathcal{A})^2$ *such that* $x_i = x_{f(i)}x_{s(i)}$ *and we set* $\mathcal{E}_i = \mathcal{E}_{f(i)} \cup \mathcal{E}_{s(i)}$

It follows from Definition 1 that, $\forall i \in I \cup \mathcal{A}$, $\mathcal{E}_i$ is the subset of indexes from I whose product is equal to x_i, precisely: $x_i = \prod_{i' \in \mathcal{E}_i} x_{i'}$

We now define a *valid* quadratization as a reformulation with N variables where any monomial of degree greater than or equal to 3 is replaced by the product of two variables.

Definition 2 *Valid quadratization*
A quadratization $\mathcal{Z} = (\mathcal{A}, \mathcal{E}, f, s)$ with $N = |I \cup \mathcal{A}|$ variables is valid if $\forall p : |\mathcal{M}_p| \geq 3$, there exist $(j,k) \in (I \cup \mathcal{A})^2$ such that $\mathcal{M}_p = \mathcal{E}_j \cup \mathcal{E}_k$ and $\prod_{i \in \mathcal{M}_p} x_i = x_j x_k$.

Several valid quadratizations exist and can be easily computed. Given a valid quadratization $\mathcal{Z}$ and by considering that $x \in \mathbb{R}^N$, we derive from Definition 2 the following reformulation of F into a quadratic form $g^{\mathcal{Z}}$:

$$g^{\mathcal{Z}}(x) = \sum_{\substack{p:|\mathcal{M}_p|\geq 3 \\ \mathcal{M}_p = \mathcal{E}_j \cup \mathcal{E}_k}} a_p x_j x_k + \sum_{p:|\mathcal{M}_p|\leq 2} a_p \prod_{i \in \mathcal{M}_p} x_i$$

Then, by splitting $g^{\mathcal{Z}}$ into its quadratic and linear parts, and by introducing $Q \in S_N$ and $c \in \mathbb{R}^N$, we will below adopt the following notation for function $g^{\mathcal{Z}}$:

$$g^{\mathcal{Z}}(x) \equiv x^T Q x + c^T x$$

From Definitions 1 and 2, it holds that $\forall i \in \mathcal{A}$ where $x_i = x_{f(i)} x_{s(i)}$, the value of $g^{\mathcal{Z}}(x)$ equals the value of $F(\tilde{x})$, where $\tilde{x}$ is the vector of initial variables. Our reformulation of (UBP) into a non-convex mixed-integer quadratic program $(QP^{\mathcal{Z}})$ follows from all this.

$$(QP^{\mathcal{Z}}) \begin{cases} \min g^{\mathcal{Z}}(x) = x^T Q x + c^T x \\ \text{s.t.} \\ \quad x \in \mathcal{F}^{\mathcal{Z}} \end{cases} \tag{64}$$

with $\mathcal{F}^{\mathcal{Z}}$ the set of Fortet inequalities [61] that enforces the identity $x_i = x_{f(i)} x_{s(i)}$:

$$\mathcal{F}^{\mathcal{Z}} \begin{cases} x_i \leq x_{f(i)} & i \in \mathcal{A} & (65) \\ x_i \leq x_{s(i)} & i \in \mathcal{A} & (66) \\ x_i \geq x_{f(i)} + x_{s(i)} - 1 & i \in \mathcal{A} & (67) \\ x_i \geq 0 & i \in \mathcal{A} & (68) \\ x_i \in \{0,1\} & i \in I \cup \mathcal{A} & (69) \end{cases}$$

A tailored quadratic convex reformulation

We now consider the problem of reformulating $(QP^{\mathcal{Z}})$ by an equivalent quadratic program with a convex objective function. To do this, we start by adding to $g^{\mathcal{Z}}(x)$ a combination of four sets of functions defined in set $\mathcal{S}^{\mathcal{Z}}$ that vanish on the feasible domain $\mathcal{F}^{\mathcal{Z}}$:

$$\mathcal{S}^{\mathcal{Z}} \begin{cases} x_i^2 - x_i = 0 & i \in I \cup \mathcal{A} & (70) \\ x_i - x_{f(i)} x_{s(i)} = 0 & i \in \mathcal{A} & (71) \\ x_i - x_i x_j = 0 & (i,j) \in \mathcal{A} \times (I \cup \mathcal{A}) : \ \mathcal{E}_j \subset \mathcal{E}_i & (72) \\ x_i x_j - x_k x_l = 0 & (i,j,k,l) \in (I \cup \mathcal{A})^4 : \ \mathcal{E}_i \cup \mathcal{E}_j = \mathcal{E}_k \cup \mathcal{E}_l & (73) \end{cases}$$

Constraint (70) comes from the binarity of the variables, and Constraint (71) from the definition of the quadratisation $\mathcal{Z}$. Constraints (72) and (73) arise from the symmetries induced by $\mathcal{Z}$.

We now associate a real scalar parameter to each constraint of $\mathcal{S}^{\mathcal{Z}}$: α_i for Constraint (70), δ_i for Constraint (71), β_{ij} for Constraint (72), and λ_{ijkl} for Constraint (73). Then, we introduce the following quadratic function :

$$\begin{aligned} g^{\mathcal{Z}}_{\alpha,\delta,\beta,\lambda}(x) &= g^{\mathcal{Z}}(x) + \sum_{i\in I\cup\mathcal{A}} \alpha_i(x_i^2 - x_i) + \sum_{i\in\mathcal{A}} \delta_i(x_i - x_{f(i)}x_{s(i)}) \\ &\quad + \sum_{\substack{(i,j)\in\mathcal{A}\times(I\cup\mathcal{A}) \\ \mathcal{E}_j\subset\mathcal{E}_i}} \beta_{ij}(x_i - x_i x_j) + \sum_{\substack{(i,j,k,l)\in(I\cup\mathcal{A})^4 \\ \mathcal{E}_i\cup\mathcal{E}_j=\mathcal{E}_k\cup\mathcal{E}_l}} \lambda_{ijkl}(x_i x_j - x_k x_l) \end{aligned}$$

Function $g^{\mathcal{Z}}_{\alpha,\delta,\beta,\lambda}(x)$ has the same value as $g^{\mathcal{Z}}(x)$ for any $x \in \mathcal{F}^{\mathcal{Z}}$. Moreover, there exist vector parameters α, β, δ and λ such that $g^{\mathcal{Z}}_{\alpha,\delta,\beta,\lambda}$ is a convex function. Take for instance, α equals to the opposite of the smallest eigenvalue of Q, and $\beta = \delta = \lambda = 0$.

By replacing $g^{\mathcal{Z}}$ by the new function, we obtain the following family of convex equivalent formulations to $(QP^{\mathcal{Z}})$, and thus to (UBP):

$$(QP^{\mathcal{Z}}_{\alpha,\delta,\beta,\lambda}) \begin{cases} \min g_{\alpha,\delta,\beta,\lambda}(x) = x^T Q_{\alpha,\delta,\beta,\lambda} x + c^T_{\alpha,\delta,\beta,\lambda} x \\ \text{s.t.} \\ \quad x \in \mathcal{F}^{\mathcal{Z}} \end{cases}$$

where $Q_{\alpha,\delta,\beta,\lambda} \in S_N$ is the Hessian matrix of $g^{\mathcal{Z}}_{\alpha,\delta,\beta,\lambda}(x)$, and $c_{\alpha,\delta,\beta,\lambda} \in \mathbb{R}^N$ its vector of linear coefficients.

We are now interested in parameters (α, δ, β, λ) such that $g^{\mathcal{Z}}_{\alpha,\delta,\beta,\lambda}$ is a convex function, and the continuous relaxation $(\overline{QP}^{\mathcal{Z}}_{\alpha,\delta,\beta,\lambda})$ of $(QP^{\mathcal{Z}}_{\alpha,\delta,\beta,\lambda})$ give the tightest bound. Theorem 8 states that these best parameters can be deduced from the optimal dual solution of $(SDP^{\mathcal{Z}})$:

$$(SDP^{\mathcal{Z}}) \begin{cases} \min < Q, X > + c^T x & & \\ \text{s.t.} & & \\ \quad X_{ii} - x_i = 0 & i \in I\cup\mathcal{A} \leftarrow \alpha_i & (74) \\ \quad x_i - X_{f(i)s(i)} = 0 & i \in \mathcal{A} \leftarrow \delta_i & (75) \\ \quad x_i - X_{ij} = 0 & (i,j) \in \mathcal{A}\times(I\cup\mathcal{A}) : \ \mathcal{E}_j \subset \mathcal{E}_i \leftarrow \beta_{ij} & (76) \\ \quad X_{ij} - X_{kl} = 0 & (i,j,k,l) \in (I\cup\mathcal{A})^4 : \ \mathcal{E}_i\cup\mathcal{E}_j = \mathcal{E}_k\cup\mathcal{E}_l \leftarrow \lambda_{ijkl} & (77) \\ \quad \begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix} \succeq 0 & & (78) \\ \quad x \in \mathbb{R}^N, \ X \in S_N & & (79) \end{cases}$$

Theorem 8 *We have $v(SDP^{\mathcal{Z}}) = v(\overline{QP}^{\mathcal{Z}}_{\alpha^*,\delta^*,\beta^*,\lambda^*})$. Moreover, the optimal α^*, δ^*, β^*, λ^* are the vectors of optimal dual variables associated to Constraints (74)–(77), respectively.*

To prove Theorem 8, we must first show that the inequalities of the set $\mathcal{F}^{\mathcal{Z}}$ are redundant in $(SDP_{\mathcal{Z}})$, and then use Lagrangian duality. Our three-phase algorithm `PQCR` is summed up in Algorithm 2.

Algorithm 2 PQCR an exact solution method for (UBP)

Step 1: Apply a quadratization $\mathcal{Z} = (\mathcal{A}, \mathcal{E}, f, s)$ to (UBP).
Step 2: Solve $(SDP^{\mathcal{Z}})$, deduce optimal values α^*, δ^*,β^*, λ^* as described in Theorem 8, and build (QP^*).
Step 3: Solve (QP^*) by a branch-and-bound based on continuous relaxation using a standard miqp solver.

Discussion on the impact of the chosen quadratization [103]

The algorithm PQCR is working for any quadratization $\mathcal{Z}$ applied in Step 1, and the choice of $\mathcal{Z}$ impacts the value of the continuous relaxation of the equivalent convex formulation. Moreover, for a given quadratization $\mathcal{Z}$, there can be several ways to rewrite the objective function into a quadratic function, depending on the meaning of the auxiliary variables. This implies that a quadratization $\mathcal{Z}$, is in fact a family of quadratizations, where each element differs by the variables involved in the quadratic objective function $g^{\mathcal{Z}}$. We want therefore caracterize families of quadratizations that are *stable* by convexification, i.e. that gives the same bound after convexification. A first result is stated in Proposition 1

Proposition 1 *Stability of a family of quadratizations.*
Given a quadratization $\mathcal{Z} = (\mathcal{A}, \mathcal{E}, f, s)$, suppose there is $\mathcal{B} = \{1, \ldots, s\}$ different ways to express $F(x)$ as a quadratic function leading to $|\mathcal{B}|$ quadratic functions $g_i^{\mathcal{Z}}$ and associated optimal reformulations $(QP_i^{\mathcal{Z}*})$. We have $\forall (i,j) \in \mathcal{B} \times \mathcal{B}$, $v(\overline{QP}_i^{\mathcal{Z}*}) = v(\overline{QP}_j^{\mathcal{Z}*})$.

Two quadratizations that give the same bound after convexification can be qualified as "equivalent". Then we can choose the one with the smallest size to speed up the algorithm. We now give a few rules in order to compare these families among themselves.

Proposition 2 *Inclusion of two families of quadratizations.*
We consider two quadratization $\mathcal{Z}^1 = (\mathcal{A}^1, \mathcal{E}^1, f^1, s^1)$ and $\mathcal{Z}^2 = (\mathcal{A}^2, \mathcal{E}^2, f^2, s^2)$ involving N^1 and N^2 variables respectively, with $N^1 \leq N^2$. We say that the quadratization $\mathcal{Z}^1 \subset \mathcal{Z}^2$, if for any $i \in \{1, \ldots, N^1\}$, there exists $j \in \{1, \ldots, N^2\}$ such that $\mathcal{E}_i^1 = \mathcal{E}_j^2$. We have $v(\overline{QP}^{\mathcal{Z}^1*}) \leq v(\overline{QP}^{\mathcal{Z}^2*})$.

We end with the description of two particular quadratizations. The first is called *full quadratization* and consists in the introduction of an additional variable for each possible product of variables up to degree 2. This is the smallest quadratization that can be used to quadratically reformulate any polynomial of degree d. This quadratization is not sensitive to the structure of the initial problem since the number of additional variables only depends on the number of initial variables n. It has been introduced and used in different frameworks as in [97, 98]. The other one is the *partial quadratization* that consists in the introduction of an additional variable for each possible product of variables up to degree 2, but that is present in the objective function. By Proposition 2, we know that the bound obtained by the *partial quadratization* will be always weaker that the one obtained by the *full quadratization*. However, the size of the first one is more practicable from a computational point of view.

A picture of computational results

We consider the *low auto-correlation binary sequence (LABS)* problem [19]. This problem has numerous practical applications in communication engineering, or theoretical physics. It consists in finding binary sequences with low off-peak auto-correlations. More formally, given a sequence $S = (s_1, \ldots, s_n)$ with $s_i \in \{-1, 1\}$, and an integer $k = 1, \ldots, n-1$, we consider the auto-correlation of S:

$$C_k(S) = \sum_{i=1}^{n-k} s_i s_{i+k}$$

The problem is to find a sequence S of length n up to a certain distance $n_0 \leq n$ that minimizes the polynomial:

$$E_{n_0}(S) = \sum_{k=1}^{n_0-1} C_k^2(S)$$

We use the instances introduced by [105] and available on the `MINLPLib` website [112]. We convert the variables from $\{-1, 1\}$ to $\{0, 1\}$ using the standard transformation $x = \frac{s+1}{2}$. These instances are dense and very hard to solve, and for most of them, the optimal solution value is not known. To illustrate the computational behavior of `PQCR`, we compare it to the solvers `Baron` [133] and `Scip` [1]. We present in Figure 9, the performance profile of the CPU times for `PQCR`, `Baron` and `Scip` over the 19 *LABS* instances solved within the time limit of 3 hours by at least one method. We can see that `PQCR` outperforms the two solvers both in terms of the total CPU time and of the number of instances solved. We mention that the number of valid equalities generated by `PQCR` in $(SDP^{\mathcal{Z}})$ can be more than one million for the largest/densest instances, what required here again an adequate implementation of the semidefinite solver based on the bundle method described in Section 2.4.

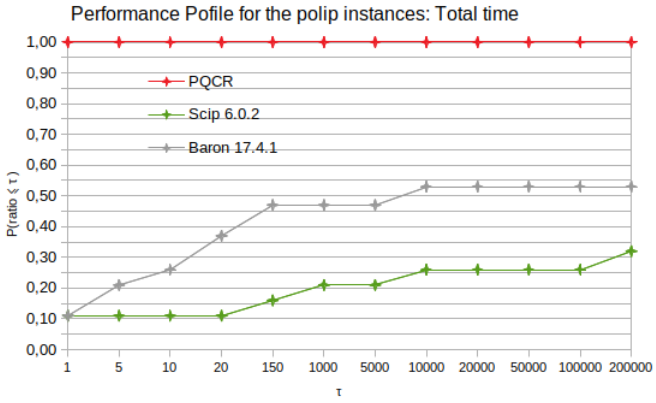


FIGURE 9: Performance profile of the CPU times between `PQCR`, `Baron 17.4.1` and `Scip 6.0.2` for the *LABS* instances - Time limit 3 hours

Last, we increase the time limit for `PQCR` and compare our results to the best lower bounds and the best known solution values reported in `MINLPLib`. We present in Table 4 the values of the best solutions (*BKN*) and of the final lower bounds (*BestLB*) obtained by `PQCR`, and those available on the `MINLPLib` website. Each line corresponds to one instance stated as unsolved on `MINLPLib` that is labeled `b.n.`$\mathtt{n}_0$. For `MINLPLib`, Columns *BKN* and *BestLB* are the best upper and lower bounds, respectively, among the results of the solvers `ANTIGONE`, `Baron`, `Couenne`, `Lindo`, and `Scip`.

We also report the final gaps in Column $Gap_f = \left|\frac{BKN - BestLB}{BKN}\right| * 100$. We observe that PQCR solves to optimality 10 unsolved instances (labeled as **). It also improves the best known solution values of 13 instances (labeled as $^{\#}$), and improves the final dual bounds of all the unsolved instances (labeled as *). Column *Imp* is the improvement factor of PQCR.

Instance	MINLPLib[112]			PQCR			
Name	*BKN*	*BestLB*	Gap_f	*BKN*	*BestLB*	Gap_f	*Imp.*
b.25.19**	-14644	-16108	10.00	**-14644**	**-14644.00**		
b.25.25**	-10664	-12494	17.16	**-10664**	**-10664.00**		
b.30.15**	-15744	-19780	25.64	**-15744**	**-15744.00**		
b.30.23**	-30420	-72030	136.79	**-30460**	**-30460.00**		
b.30.30**	-22888	-54014	135.99	**-22888**	**-22888.00**		
b.35.09**	-5108	-6312	23.57	**-5108**	**-5108.00**		
b.35.18**	-31160	-74586	139.36	**-31168**	**-31168.00**		
b.35.26#*	-55184	-191466	246.96	-55288	-55484.64	0.36	**694**
b.35.35#*	-41068	-290424	607.18	-41068	-41730.96	1.61	**376**
b.40.10**	-8240	-14618	77.40	**-8248**	**-8248.00**		
b.40.20#*	-50516	-162365	221.41	-50576	-51248.66	1.33	**166**
b.40.30#*	-94768	-398617	320.62	-94952	-102375.97	7.82	**41**
b.40.40*	-67964	-302028	344.39	-67928	-78364.82	15.36	**22**
b.45.11**	-12740	-30771	141.53	**-12748**	**-12748.00**		
b.45.23#*	-85248	-320397	275.84	-85424	-88547.05	3.66	**75**
b.45.34*	-152368	-752427	393.82	-152248	-164316.84	7.93	**50**
b.45.45*	-112764	-685911	508.27	-112568	-142414.81	26.51	**19**
b.50.06#*	-2160	-2921	35.23	-2160	-2199.87	1.85	**19**
b.50.13**	-23772	-74768	214.52	**-23792**	**-23792.00**		
b.50.25#*	-124748	-562446	350.87	-124948	-138994.51	11.24	**31**
b.50.38#*	-232496	-1318325	467.03	-232664	-270363.98	16.20	**29**
b.50.50*	-168216	-1173058	597.35	-167824	-246094.07	46.64	**13**
b.55.06#*	-2400	-3439	43.29	-2400	-2460.02	2.50	**17**
b.55.14#*	-33168	-116748	251.99	-33272	-33717.52	1.34	**188**
b.55.28#*	-190472	-989145	419.31	-190696	-214840.20	12.66	**33**
b.55.41*	-337388	-2494477	639.35	-335840	-474910.72	41.41	**15**
b.55.55*	-241912	-1947633	705.10	-241780	-302301.26	25.03	**28**
b.60.08#*	-6792	-13915	104.87	-6792	-7008.60	3.19	**33**
b.60.15#*	-44896	-169767	278.13	-45232	-46160.31	2.05	**136**
b.60.30*	-261048	-1491016	471.17	-260304	-324244.90	24.56	**19**
b.60.45*	-478528	-3687344	670.56	-476664	-951320.31	99.58	**7**
b.60.60*	-350312	-3021077	762.40	-349560	-496399.25	42.01	**18**

TABLE 4: Comparison of the best known solution and best lower bound values of PQCR and of MINLPLib for the unsolved *LABS* instances. $**$: solved for the first time, #: best known solution improved, and $*$: best known lower bound improved

Chapter 5

Conclusion and future research

In this thesis, we have summarized our main contributions to the exact solution of mixed-integer polynomial optimization problems. In particular, we started by considering the case were the functions are quadratic, and we have presented an overview of our method MIQCR which is dedicated to this class of problem. This algorithm starts with a reformulation phase in which we construct a parameterized family of equivalent problems whose functions are convex. The strength of the approach is that we then compute, within this family, the best quadratic convex relaxation of the initial problem. Using the powerful tool of semidefinite programming, we obtain a quadratic convex relaxation that is as strong as the semidefinite relaxation used to compute it. The second phase of our approach is dedicated to the resolution of the equivalent formulation by a branch-and-bound algorithm. We distinguish two cases. When the variables are all integer, we can delegate this resolution to standard quadratic programming solvers. This is not the case in presence of continuous variables. We thus develop an appropriate spatial branch-and-bound to solve the reformulated problem where the bounding step solves our strong quadratic convex relaxation. A significant contribution is that the framework of our approach allows the generalization of classical methods from the literature, such as the QCR method or the standard linearization.

We have also proposed a methodology to extend the regular triangle inequalities defined for binary variables to the case of general bounds. These inequalities bring two improvements to MIQCR. First, it strengthens the semidefinite relaxation used to compute the convex quadratic relaxation, and thus improves the bound at the root node of the branch-and-bound. Secondly, as by construction these inequalities involve bounds on the variables, they allow the relaxations to be dynamically tightened along the tree. From a general outlook, these inequalities can be used in any branch-and-bound procedure based on the relaxation of the Constraints $Y = xx^T$.

Our approach requires solving once a semidefinite relaxation with a large number of constraints. To address this difficulty, we have introduced a heuristic algorithm dedicated to solving our relaxation. It is a subgradient algorithm based on the dynamic Lagrangian relaxation of the constraints. It is also parameterized, which allows us to control the number of constraints considered and, at the same time, the size of the convex relaxation calculated. Since only a few considered constraints are active at the root node of branch-and-bound, it also serves as a separation algorithm during which only the most violated constraints are selected.

Then, we have also shown how our generic method can be specialized to two applications that have particular structures. We first presented our results on the Quadratic Assignment Problem (QAP). Indeed, using the structural properties of the QAP, we showed that, while keeping the quality of the bounds, we can construct compact quadratic convex relaxations. This specialization allowed us to solve

instances of much larger sizes than the original method. Then, we showed that our methods could be applied to industrial issues. In particular, we worked on the problem of optimizing the power flow in an electrical network (OPF). For this problem, we proved that a specialization of our approach allowed us to use a compact semidefinite relaxation, but also to construct convex relaxations of significantly smaller size than in the original method. This allowed us to handle medium-sized real instances of this problem.

The last part of this manuscript focuses on the exact solution of a wider class of problems than MIQCP. These are unconstrained polynomial optimization problems with binary variables. Our solution algorithm, `PQCR`, is based on a quadratization of the polynomial objective function which implies the addition of variables and constraints. Then, we have introduced a convexification, which is based on the links between the initial variables and the auxiliary ones. The solution of the obtained equivalent convex problem is then delegated to a standard MIQP solver. This approach has proven to be efficient, since it has been able to solve to optimality still unsolved instances of an application: the low auto-correlation binary sequence.

Our scientific contributions are twofold. The first one is theoretical and the second is practical. Indeed, following the incremental theoretical improvement of method `MIQCR`, we have implemented the associated software called `Solution of Mixed Integer Quadratic Programs (SMIQP)` [94]. To conclude, we summarize in Table 5 the general features of the algorithms mentioned in this thesis, each of them having a different scope.

We end this manuscript by giving some perspectives of our work that include current and future research directions. Our main objective is to open new efficient theoretical perspectives on the solution of classes of nonlinear optimization problems that will meet the industrial challenges of tomorrow. To this end, we propose three lines of theoretical research : quadratic programming, polynomial optimization and applications. The second objective

Perspectives for quadratic programming

Method `MIQCR` generalizes and improves the state of the art, especially for the *bounding* step of the branch-and-bound. A natural perspective is to focus on the improvement of the *branching* step in order to address increasingly complex problems. Thus, a future research is to establish an innovative association between our methods and techniques from the field of constraint programming. More precisely, we collaborate with researchers who have developed algorithms for solving general nonlinear optimization problems implemented in the solver `IbexOpt` [81]. The success of their approach is based on a very refined separation mechanism. The first step of this project is to demonstrate the theoretical and experimental contribution of this association on the quadratic case. Then, their algorithms being able to solve general non-linear optimization problems, we plan in a second time to mix our approaches for wider classes of problems.

Another future research comes from the results related to the construction of a compact relaxation of the OPF which is a QCQP. It is indeed possible from this relaxation to deduce a SOCP relaxation of the OPF. Thus, the idea is to compute sharp SOCP relaxations, instead of convex quadratic relaxations, by use of semidefinite

Type	*Algorithm*	*Main features*	*Scope*
LINEARIZATION	`BIL (Binary Integer Linearization)` [23]	Linearization of each product of two integer variables by replacing only one of them by its binary decomposition	IQP
CONVEXIFICATION	`CQCR (Compact Quadratic Convex Reformulation)` [25]	Perturbation of diagonal terms of the Hessian matrix with the best non uniform vector and use of the RLT constraints	IQP
	`MIQCR (Mixed Integer Quadratic Convex Reformulation)`[24, 26]	Perturbation of all the terms of the Hessian matrix with the best non uniform matrix and use of the RLT constraints	MIQP[1]
	`MIQCR-Quad (Mixed Integer Quadratic Convex Reformulation for quadratic constraints)` [27, 28]	Extension of `MIQCR` to the case of quadratic constraints by complete linearization of them	MIQCP[1]
	`MIQCR-BB (Mixed Integer Quadratic Convex Reformulation with spatial branch-and-bound)` [51]	Extension of `MIQCR-Quad` to the general mixed case by use of spatial branch-and-bound	MICQP
	`MIQCR-T (Mixed Integer Quadratic Convex Reformulation with General Triangle)` [95]	Improvement of `MIQCR-BB` by use of general triangle inequalities into the SDP and the spatial branch-and-bound	MICQP
	`PQCR(Polynomial Quadratic Convex Reformulation)` [53]	Quadratization followed by a tailored convexification that perturbs the diagonal and some non-diagonal terms of the Hessian matrix	UBP

TABLE 5: Type, Algorithm, Main Features and Scope of each contributed algorithm
[1] convexity assumption on pure continuous quadratic terms

programming. The efficiency of SOCP solvers combined to the compactness of the tight relaxations would allow us to handle larger problems.

Perspectives for polynomial optimization

We now focus on our results obtained for polynomial optimization problems in binary variables. Indeed, we have introduced a two-step solution algorithm. The first step rewrites the polynomial objective function of any degree into a quadratic function, then, in the second step, we make it convex. The strength of this approach lies in the fact that we are able to compute the best convexification for a given

quadratization. A recent work [48] has focused on the design of "good" quadratizations. A natural perspective is to combine our two approaches in order to characterize "good" quadratizations for method PQCR. This collaboration, beyond the practical improvements, opens up many theoretical perspectives, and in particular that of determining a convexification that would be stable for any quadratization.

Another line of research deals with unconstrained polynomial problems where the variables can be continuous. The main idea is to determine for any multivariate polynomial a separable underestimator, which can be used as a bound in a classical branch-and-bound algorithm. Indeed, separability is a weaker property than convexity, but such an optimization problem can be solved efficiently by determining separately the optimal solution of each univariate problem. Thus the aim is to compute the coefficients of a separable polynomial which, over the feasible domain of variables, minimizes on average the deviation from the value of the initial polynomial. Forcing the separable function to be an underestimator amounts to calculating a certificate of positivity of a polynomial and this constitutes the main difficulty of the approach. To overcome it, we use the sum-of-square polynomial which can be computed using semidefinite programming. By construction, this approach has two main advantages: a compact formulation since there is no additional variable in the relaxation, and a scope that covers more general problems. The first results are promising and open many perspectives.

Applications

Unit Commitment problem is a variant of problem OPF where the optimization process determines which generator nodes are switched to on or off. A classical modelling is to introduce a binary variable for each generator, which role is to activate or not the appropriate constraints into the QCQP. Since, quadratic convex reformulation methods are initially devoted to integer programming, a natural perspective is to consider an extension of our methods to this problem.

Another line of research

A final research and application perspective concerns the use of non-linear optimization within supervised learning algorithms. If many researches currently aim at exploiting artificial intelligence techniques to improve the resolution of optimization problems, we are interested in the contribution of optimization to the construction of classification trees. Among the classical supervised learning methods, classification trees provide a good compromise between the accuracy of the prediction and the interpretability of the result obtained. Interpretability [44] is an essential notion for many applications, such as medical diagnostic assistance, or the automation of vehicle driving. Classification trees are based on a simple concept where the data follows a path, starting from its root, and whose branching rules lead to the possible decisions located at the leaves. Thus, the challenge is to build the structure of the tree, i.e. to define the rules that define the branches. Classical algorithms for building classification trees are generally heuristic. In order to obtain classifiers that are both interpretable and efficient, recent works [20, 143] have focused on the formulation, via linear optimization, of the problem of designing the optimal structure of classification trees. The resulting classification tree is based on branching rules that are described by linear functions. In order to improve the prediction quality of such trees, the objective of our research project is to design optimization models where the branching rules can be described by separable or non-linear functions.

Bibliography

[1] T. Achterberg. Scip : solving constraint integer programs. *Mathematical Programming Computation*, (1):1–41, 2009.

[2] W.P. Adams, M. Guignard, P.M. Hahn, and W.L. Hightower. A level-2 reformulation-linearization technique bound for the quadratic assignment problem. *European Journal of Operational Research*, 180:983–996, 2007.

[3] W.P. Adams and T.A. Johnson. Improved linear programming-based lower bounds for the quadratic assignment problem. In *Proceedings of the DIMACS Workshop on Quadratic Assignment Problems*, volume 16, pages 43–75. American Mathematical Society, 1994.

[4] N. Adhya, M. Tawarmalani, and N.V. Sahinidis. A lagrangian approach to the pooling problem. *Industrial & Engineering Chemistry Research*, 38(5):1956–1972, 1999.

[5] C.S. Adjiman, I. Androulakis, and C.A. Floudas. A global optimization method, αbb, for general twice-differentiable constrained nlps-ii. implementation and computational results. *Computers & Chemical Engineering*, 22(9):1159–1179, 1998.

[6] C.S. Adjiman, S. Dallwig, C.A. Floudas, and A. Neumaier. A global optimization method, αbb, for general twice-differentiable constrained nlps-i. theoretical advances. *Computers and Chemical Engineering*, 22(9):1137–1158, 1998.

[7] A. A. Ahmadi and A. Majumdar. Dsos and sdsos optimization: Lp and socp-based alternatives to sum of squares optimization. In *2014 48th Annual Conference on Information Sciences and Systems (CISS)*, pages 1–5, March 2014.

[8] C. Aholt, S. Agarwal, and R. Thomas. A qcqp approach to triangulation. In Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, editors, *Computer Vision – ECCV 2012*, pages 654–667, Springer Berlin Heidelberg, 2012.

[9] I.P. Androulakis, C.D. Maranas, and C.A. Floudas. abb : A global optimization method for general con- strained nonconvex problems. *Journal of Global Optimization*, 7:337–363, 1995.

[10] K. M. Anstreicher. Semidefinite programming versus the reformulation-linearization technique for nonconvex quadratically constrained quadratic programming. *Journal of Global Optimization*, 43(2):471–484, 2009.

[11] K. M. Anstreicher and N.W. Brixius. A new bound for the quadratic assignment problem based on convex quadratic programming. *Mathematical Programming*, 89:341–357, 2001.

[12] M. Anthony, E. Boros, Y. Crama, and A. Gruber. Quadratic reformulations of nonlinear binary optimization problems. *Mathematical Programming*, 162:115–144, 2017.

[13] C. Audet, J. Brimberg, P. Hansen, S. Le Digabel, and N. Mladenović. Pooling problem: Alternate formulations and solution methods. *Management science*, 50(6):761–776, 2004.

[14] X. Bai and H. Wei. Semi-definite programming-based method for security-constrained unit commitment with operational and optimal power flow constraints. *IET Generation, Transmission & Distribution*, 3:182–197(15), February 2009.

[15] X. Bai, H. Wei, K. Fujisawa, and Y. Wang. Semidefinite programming for optimal power flow problems. *International Journal of Electrical Power & Energy Systems*, 30(6):383 – 392, 2008.

[16] X. Bao, N.v. Sahinidis, and M. Tawarmalani. Multiterm polyhedral relaxations for nonconvex, quadratically constrained quadratic programs. *Optimization Methods Software*, 24(4-5):485–504, 2009.

[17] X. Bao, N.V. Sahinidis, and M. Tawarmalani. Semidefinite relaxations for quadratically constrained quadratic programming: A review and comparisons. *Mathematical Programming*, 129:129–157, 2011.

[18] P. Belotti, J. Lee, L. Liberti, F. Margot, and A. Wächter. Branching and bounds tightening techniques for non-convex minlp. *Optimization Methods and Software*, 4–5(24):597–634, 2009.

[19] J. Bernasconi. Low autocorrelation binary sequences: statistical mechanics and configuration space analysis. *J. Physique*, 141(48):559–567, 1987.

[20] Dimitris Bertsimas and Jack Dunn. Optimal classification trees. *Machine Learning*, 106(7):1039–1082, 2017.

[21] A. Billionnet and S. Elloumi. Best reduction of the quadratic semi-assignment problem. *DAMATH: Discrete Applied Mathematics and Combinatorial Operations Research and Computer Science*, 109:197–213, 2001.

[22] A. Billionnet and S. Elloumi. Using a mixed integer quadratic programming solver for the unconstrained quadratic 0-1 problem. *Mathematical Programming*, 109(1):55–68, 2007.

[23] A. Billionnet, S. Elloumi, and A. Lambert. Linear reformulations of integer quadratic programs. In *MCO 2008, september 8-10*, pages 43–51, 2008.

[24] A. Billionnet, S. Elloumi, and A. Lambert. Extending the QCR method to the case of general mixed integer program. *Mathematical Programming*, 131(1):381–401, 2012.

[25] A. Billionnet, S. Elloumi, and A. Lambert. An efficient compact quadratic convex reformulation for general integer quadratic programs. *Computational Optimization and Applications*, 54(1):141–162, 2013.

[26] A. Billionnet, S. Elloumi, and A. Lambert. A branch and bound algorithm for general mixed-integer quadratic programs based on quadratic convex relaxation. *Journal of Combinatorial Optimization*, 2(28):376–399, 2014.

[27] A. Billionnet, S. Elloumi, and A. Lambert. Exact quadratic convex reformulations of mixed-integer quadratically constrained problems. *Mathematical Programming*, 158(1):235–266, 2016.

[28] A. Billionnet, S. Elloumi, A. Lambert, and A. Wiegele. Using a Conic Bundle method to accelerate both phases of a Quadratic Convex Reformulation. *INFORMS Journal on Computing*, 29(2):318–331, 2017.

[29] A. Billionnet, S. Elloumi, and M. C. Plateau. Improving the performance of standard solvers for quadratic 0-1 programs by a tight convex reformulation: The QCR method. *Discrete Applied Mathematics*, 157(6):1185 – 1197, 2009.

[30] P. Bonami, L. Biegler, A. Conn, G. Cornuéjols, I. Grossmann, C. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, and A. Waechter. An Algorithmic Framework for Convex Mixed Integer Nonlinear Programs. *Discrete Optimization*, 5(2):186–204, 2008.

[31] P. Bonami, O. Günlük, and J. Linderoth. Globally solving nonconvex quadratic programming problems with box constraints via integer programming methods. *Mathematical Programming Computation*, 10:333–382, 2018.

[32] B. Borchers. CSDP, A C Library for Semidefinite Programming. *Optimization Methods and Software*, 11(1):613–623, 1999.

[33] E. Boros and A. Gruber. On quadratization of pseudo-Boolean functions. ISAIM, International Symposium on Artificial Intelligence and Mathematics, 2012.

[34] F. Boukouvala, R. Misener, and C.A. Floudas. Global optimization advances in mixed-integer nonlinear programming, minlp, and constrained derivative-free optimization, cdfo. *European Journal of Operational Research*, 252(3):701–727, 2016.

[35] C. Buchheim and C. D'Ambrosio. Monomial-wise optimal separable underestimators for mixed-integer polynomial optimization. *Journal of Global Optimization*, pages 1–28, 2016.

[36] C. Buchheim and G. Rinaldi. Efficient reduction of polynomial zero-one optimization to the quadratic case. *SIAM Journal on Optimization*, 18(4):1398–1413, 2007.

[37] S. Burer and A.N. Letchford. On nonconvex quadratic programming with box constraints. *SIAM Journal on Optimization*, 20(2):1073–1089, 2009.

[38] R.E Burkard, S. Karisch, and F. Rendl. QAPLIB - A quadratic assignment problem library. *Journal of Global Optimisation*, 10:391–403, 1997.

[39] C. Coffrin and P. Van Hentenryck. A linear-programming approximation of ac power flows. *INFORMS Journal on Computing*, 26:718–734, 2014.

[40] C. Coffrin, H. L. Hijazi, and P. Van Hentenryck. The QC relaxation: A Theoretical and Computational Study on Optimal Power Flow. *IEEE Transactions on Power Systems*, 31(4):3008–3018, 2016.

[41] C. Coffrin, H.L. Hijazi, and P. Van Hentenryck. Convex quadratic relaxations for mixed-integer nonlinear programs in power systems. *Mathematical Programming Computation*, 9(3):321–367, 2017.

[42] Y. Crama and E. Rodriguez-Heck. A class of valid inequalities for multilinear 0-1 optimization problems. *Discrete Optimization*, pages 28–47, 2017.

[43] Y. Cui. Dynamic programming algorithms for the optimal cutting of equal rectangles. *Appl.Math. Model*, 29:1040–1053, 2005.

[44] Dam H.,Tran T.,Ghose A. Explainable software analytics. https://arxiv.org/pdf/1802.00603.pdf, 2019. Accessed: 2021-08-03.

[45] J.W. Dickey and J.W. Hopkins. Campus building arrangement using topaz. *Transp. Res.*, 6:59–68, 1972.

[46] M.C. Dorneich and N.V. Sahinidis. Global optimization algorithms for chip layout and compaction. *Engineering Optimization*, 25(2):131–154, 1995.

[47] Zvi Drezner. *The Quadratic Assignment Problem*, pages 345–363. Springer International Publishing, Cham, 2015.

[48] Y. Crama E. Boros and E. Rodriguez-Heck. Quadratizations of symmetric pseudo-Boolean functions: sub-linear bounds on the number of auxiliary variables. ISAIM, International Symposium on Artificial Intelligence and Mathematics, 2018.

[49] S. Elloumi, , H. Godard, A. Lambert, J. Maegth, and M. Ruiz. Global optimality of optimal power flow using quadratic convex optimization. *6th International Conference on Control, Decision and Information Technologies, CODIT*, pages 1–6, 2019.

[50] S. Elloumi and A. Lambert. Comparison of quadratic convex reformulations to solve the quadratic assignment problem. *COCOA*, pages 726–734, 2016.

[51] S. Elloumi and A. Lambert. Global solution of non-convex quadratically constrained quadratic programs. *Optimization Methods and Software*, 34(1):98–114, 2019.

[52] S. Elloumi, A. Lambert, and A. Lazare. Etude de différentes quadratisations pour la reformulation quadratique convexe des programmes polynomiaux en variables binaires. *ROADEF*, pages 1–2, 2019.

[53] S. Elloumi, A. Lambert, and A. Lazare. Solving unconstrained 0-1 polynomial programs through quadratic convex reformulation. *Journal of Global Optimization*, page to appear, 2021.

[54] A.N. Elshafei. Hospital layout as a quadratic assignment problem. *Operational Research Quarterly*, 28:167–179, 1977.

[55] J.E. Falk and R.M. Soland. An algorithm for separable nonconvex programming problems. *Management Science*, 15:550–560, 1969.

[56] G. Finke, R.E. Burkard, and F. Rendl. Quadratic assignment problems. *Annals of Discrete Mathematics*, 31:61–82, 1987.

[57] I. Fischer, G. Gruber, F. Rendl, and R. Sotirov. Computational experience with a bundle approach for semidefinite cutting plane relaxations of Max-Cut and equipartition. *Mathematical Programming*, 105(2-3, Ser. B):451–469, 2006.

[58] C.A. Floudas and C.E. Gounaris. A review of recent advances in global optimization. *Journal of Global Optimization*, 45(1):3–38, 2009.

[59] C.A. Floudas and V. Visweswaran. A global optimization algorithm (gop) for certain classes of nonconvex nlps-i. theory. *Computers & chemical engineering*, 14(12):1397–1417, 1990.

[60] C.A. Floudas and V. Visweswaran. Primal-relaxed dual global optimization approach. *Journal of Optimization Theory and Applications*, 78(2):187–225, 1993.

[61] R. Fortet. L'algèbre de Boole et ses Applications en Recherche Opérationnelle. *Cahiers du Centre d'Etudes de Recherche Opérationnelle*, 4:5–36, 1959.

[62] A. Frangioni and C. Gentile. Perspective cuts for a class of convex 0-1 mixed integer programs. *Mathematical Programing*, 106:225–236, 2006.

[63] A.M. Frieze and J. Yadegar. On the quadratic assignment problem. *Discrete Applied Mathematics*, 5:89–98, 1983.

[64] L. Gan, N. Li, U. Topcu, and S. H. Low. Exact convex relaxation of optimal power flow in radial networks. *IEEE Transactions on Automatic Control*, 60(1):72–87, 2015.

[65] M.R. Garey and D.S. Johnson. Computers and Intractability: A guide to the theory of NP-Completness. *W.H. Freeman, San Francisco, CA*, 1979.

[66] S. Gershgorin. Uber die abgrenzung der eigenwerte einer matrix. *Bulletin de l'Académie des Sciences de l'URSS.*, 6:749–754, 1931.

[67] B. Ghaddar, J. C. Vera, and M. F. Anjos. A dynamic inequality generation scheme for polynomial programming. *Mathematical Programming*, 156(1):21–57, Mar 2016.

[68] H. Godard. *Résolution exacte du problème de l'optimisation des flux de puissance*. Thèse de doctorat en informatique, Conservatoire National des Arts et Métiers, Paris, 2019.

[69] LLC Gurobi Optimization. Gurobi optimizer reference manual 911. "http://www.gurobi.com", 2021.

[70] S.W. Hadley. Continuous optimization approaches for the quadratic assignment problem. *PhD thesis, University of Waterloo, Ontario*, 1989.

[71] S.W. Hadley, F. Rendl, and H. Wolkowicz. Bounds for the quadratic assignment problem using continuous optimization techniques. *Proceedings of the 1-st Integer Programming and Combinatorial Optimization Conference (IPCO), University of Waterloo Press, Waterloo*, pages 237–248, 1990.

[72] S.W. Hadley, F. Rendl, and H. Wolkowicz. A new lower bound via projection for the quadratic assignment problem. *Mathematics of Operations Research*, 17:727–739, 1992.

[73] S.W. Hadley, F. Rendl, and H. Wolkowicz. Nonsymmetric quadratic assignment problems and the hoffman-wielandt inequality. *Linear Algebra Appl.*, 58:109–124, 1992.

[74] W.W. Hager and J.T. Hungerford. Continuous quadratic programming formulations of optimization problems on graphs. *European Journal of Operational Research*, 240(2):328 – 337, 2015.

[75] P.L. Hammer and A.A. Rubin. Some remarks on quadratic programming with 0-1 variables. *Revue Française d'Informatique et de Recherche Opérationnelle*, 4:67–79, 1970.

[76] M. Hanan and J.M. Kurtzberg. A review of the placement and quadratic assignment problems. *SIAM Rev.*, 14:324–342, 1973.

[77] I. Harjunkoski, R. Pörn, T. Westerlund, and H. Skrifvars. Different strategies for solving bilinear integer non-linear programming problems with convex transformations. *Computers & Chemical Engineering*, 21:S487 – S492, 1997.

[78] C.A. Haverly. Studies of the behaviour of recursion for the pooling problem. *ACM SIGMAP Bulletin*, 26, 1978.

[79] C. Helmberg. *Conic Bundle v0.3.10*, 2011.

[80] Z.S Hua and P. Banerjee. Aggregate line capacity design for pwb assembly systems. *Int. J.Prod. Res.*, 38(11):2417–2441, 2000.

[81] IbexOpt. Ibexopt : un module d'optimisation globale sous contraintes fiable. "http://www.ibex-lib.org/doc/optim.html", 2012-2021.

[82] IBM-ILOG. IBM ILOG CPLEX 12.9 Reference Manual, 2020.

[83] N. Ito, S. Kim, and M. Kojima nad A.Takeda K.C. Toh. Algorithm 996: Bbcpop: A sparse doubly nonnegative relaxation of polynomial optimization problems with binary, box, and complementarity constraints. *ACM Trans. Math. Softw.*, 45(3), July 2019.

[84] R. A. Jabr. Radial distribution load flow using conic programming. *IEEE Transactions on Power Systems*, 21(3):1458–1459, 2006.

[85] R. A. Jabr. Exploiting sparsity in sdp relaxations of the opf problem. *IEEE Transactions on Power Systems*, 27(2):1138–1139, 2012.

[86] R. A. Jabr, A. H. Coonick, and B. J. Cory. A primal-dual interior point method for optimal power flow dispatching. *IEEE Transactions on Power Systems*, 17(3):654–662, 2002.

[87] C. Josz, J. Maeght, P. Panciatici, and J. C. Gilbert. Application of the moment-sos approach to global optimization of the opf problem. *IEEE Transactions on Power Systems*, 30(1):463–470, 2015.

[88] S. E. Karisch. Nonlinear approaches for quadratic assignment and graph partition problems. *Ph.D. Thesis, Graz University of Technology, Austria*, 1995.

[89] O. Kariv and S.L. Hakimi. An algorithm for the quadratic assignment problem using benders' decomposition. *Eurorean Journal of Operation Research*, 2:204–211, 1978.

[90] R.M. Karp. Reducibility among combinatorial problems. *Complexity of Computer Computations: Proceedings of a symposium on the Complexity of Computer Computations*, pages 85–103, 1972.

[91] B. Kocuk, S. S. Dey, and X. A. Sun. Strong socp relaxations for the optimal power flow problem. *Operations Research*, 64(6):1177–1196, December 2016.

[92] T.C. Koopmans and M.J. Beckmann. Assignment problems and the location of economic activities. *Econometrica*, 25:53–76, 1957.

[93] X. Kuang, , B. Ghaddar, J . Naoum-Sawaya, and L.F. Zuluaga. Alternative SDP and SOCP Approximations for Polynomial Optimization. *EURO Journal on Computational Optimization*, 7:153–175, 2019.

[94] A. Lambert. Solution of Mixed Integer Quadratic Programs (SMIQP). "https://github.com/amelie-lambert/SMIQP", 2020.

[95] A. Lambert. Using general triangle inequalities within quadratic convex reformulation method. preprint https://arxiv.org/abs/2005.02667, 2020.

[96] A. Lambert. A tight compact quadratically constrained convex relaxation of the optimal power flow problem. preprint https://arxiv.org/abs/2105.00453, 2021.

[97] J.B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001.

[98] J.B. Lasserre. An explicit equivalent positive semidefinite program for nonlinear 0-1 programs. *SIAM Journal on Optimization*, 12(3):756–769, 2002.

[99] J.B. Lasserre. *An Introduction to Polynomial and Semi-Algebraic Optimization*. Cambridge University Press, Cambridge, 2015.

[100] J.B. Lasserre and T.P. Thanh. Convex underestimators of polynomials. *Journal of Global Optimization*, pages 1–25, 2013.

[101] J. Lavaei and S. H. Low. Zero duality gap in optimal power flow problem. *IEEE Transactions on Power Systems*, 2012.

[102] E.L. Lawler. The quadratic assignment problem. *Management Science*, 9:586–599, 1963.

[103] A. Lazare. *Global optimization of polynomial programs with mixed-integer variables*. Thèse de doctorat en informatique, Univérsité Paris Saclay, 2019.

[104] L. Liberti, C. Lavor, N. Maculan, and A. Mucherino. Euclidean distance geometry and applications. *SIAM Review*, 56(1):3–69, 2014.

[105] F. Liers, E. Marinari, U. Pagacz, F. Ricci-Tersenghi, and V. Schmitz. A non-disordered glassy model with a tunable interaction range. *Journal of Statistical Mechanics: Theory and Experiment*, page L05003, 2010.

[106] M.L. Liu and N.V. Sahinidis. Process planning in a fuzzy environment. *European Journal of Operational Research*, 100(1):142 – 169, 1997.

[107] M. Locatelli and U. Raber. Packing equal circles in a square: a deterministic global optimization approach. *Discrete Applied Mathematics*, 122(1):139 – 166, 2002.

[108] M. Madani and M. Van Vyve. Computationally efficient MIP formulation and algorithms for european day-ahead electricity market auctions. *European Journal of Operational Research*, 242(2):580 – 593, 2015.

[109] R. Madani, S. Sojoudi, and J. Lavaei. Convex relaxation for optimal power flow problem: Mesh networks. *IEEE Transactions on Power Systems*, 30:199–211, 2015.

[110] C.D. Maranas and C.A. Floudas. Global minimum potential energy conformations for small molecules. *Journal of Global Optimization*, 4:135–170, 1994.

[111] G.P. McCormick. Computability of global solutions to factorable non-convex programs: Part i - convex underestimating problems. *Mathematical Programming*, 10(1):147–175, 1976.

[112] MINLPLib. Library of mixed integer non linear programs. "http://www.minlplib.org/", 2012.

[113] R. Misener and C.A. Floudas. Global optimization of mixed-integer quadratically-constrained quadratic programs (MIQCQP) through piecewise-linear and edge-concave relaxations. *Mathematical Programming B*, 136(1):155–182, 2012. http://www.optimization-online.org/DB_HTML/2011/11/3240.html.

[114] R. Misener and C.A. Floudas. GloMIQO: Global mixed-integer quadratic optimizer. *Journal of Global Optimization*, 57(1):3–50, 2013.

[115] R. Misener and C.A. Floudas. Antigone: algorithms for continuous/integer global optimization of nonlinear equations. *Journal of Global Optimization*, 59(2-3):503–526, 2014.

[116] R. Misener and C.A. Floudas. A framework for globally optimizing mixed-integer signomial programs. *Journal of Optimization Theory and Applications*, 161(3):905–932, 2014.

[117] R. Misener, J.B. Smadbeck, and C.A. Floudas. Dynamically generated cutting planes for mixed-integer quadratically constrained quadratic programs and their incorporation into GloMIQO 2. *Optimization Methods and Software*, 30(1):215–249, 2015.

[118] D. K. Molzahn and I. A. Hiskens. Sparsity-Exploiting Moment-Based Relaxations of the Optimal Power Flow Problem. *IEEE Transactions on Power Systems*, 30(6):3168–3180, November 2015.

[119] D. K. Molzahn and I. A. Hiskens. *A Survey of Relaxations and Approximations of the Power Flow Equations*. now, 2019.

[120] D. K. Molzahn, J. T. Holzer, B. C. Lesieutre, and C. L. DeMarco. Implementation of a large-scale optimal power flow solver based on semidefinite programming. *IEEE Transactions on Power Systems*, 28(4):3987–3998, 2013.

[121] J.J. Moré and G. Toraldo. Algorithms for bound constrained quadratic programming problems. *Numerische Mathematik*, 55(4):377–400, 1989.

[122] MOSEK ApS. *The MOSEK optimization toolbox for MATLAB manual. Version 9.2.*, 2019.

[123] M. Kilinç P. Bonami and J. Linderoth. Algorithms and software for convex mixed integer nonlinear programs. In *Mixed integer nonlinear programming*, pages 1–39. Springer New York, 2012.

[124] M. Padberg. The boolean quadric polytope: some characteristics, facets and relatives. *Mathematical programming*, 45(1):139–172, 1989.

[125] P. A. Parrilo. Semidefinite programming relaxations for semialgebraic problems. *Mathematical Programming Ser B.*, 96(2):293–320, 2003.

[126] PGLib Optimal Power Flow Benchmarks. *The IEEE PES Task Force on Benchmarks for Validation of Emerging Power System Algorithms*, "accessed 02/2021".

[127] R. Pörn, O. Nissfolk, A. Skjäl, and T. Westerlund. Solving 0-1 quadratic programs by reformulation techniques. *Industrial & Engineering Chemistry Research*, 56(45):13444–13453, 2017.

[128] F. Rendl. Ranking scalar products to improve bounds for the quadratic assignment problem. *European Journal of Operational Research.*, 20:363–372, 1985.

[129] F. Rendl and R. Sotirov. Bounds for the quadratic assignment problem using the bundle method. *Mathematical Programming B*, 109:505–524, 2007.

[130] F. Rendl and H. Wolkowicz. Applications of parametric programming and eigenvalue maximization to the quadratic assignment problem. *Mathematical Programming*, 53:63–78, 1992.

[131] F. Roupin. From linear to semidefinite programming: an algorithm to obtain semidefinite relaxations for bivalent quadratic problems. *Journal of Combinatorial Optimization*, 8(4):469–493, 2004.

[132] F. Roupin. Semidefinite relaxations of the quadratic assignment problem in a lagrangian framework. *International Journal of Mathematics in Operational Research*, 1:144–162, 2009.

[133] N.V. Sahinidis and M. Tawarmalani. Baron 9.0.4: Global optimization of mixed-integer nonlinear programs. *User's Manual*, 2010.

[134] S. Sahni and T. Gonzalez. P-complete approximation problems. *J. Assoc. Comput. Mach.*, 23:555–565, 1976.

[135] A. Saxena, P. Bonami, and J. Lee. Convex relaxations of non-convex mixed integer quadratically constrained programs: projected formulations. *Mathematical Programming*, 130:359–413, 2011.

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

[137] H.D. Sherali and C.H. Tuncbilek. A global optimization algorithm for polynomial programming using a reformulation-linearization technique. *Journal of Global Optimization*, 2:101–112, 1992.

[138] A. Sutou and Y. Dai. Global optimization approach to unequal global optimization approach to unequal sphere packing problems in 3d. *Journal of Optimization Theory and Applications*, 114(3):671–694, Sep 2002.

[139] M. Tawarmalani and N.V. Sahinidis. Convexification and global optimization in continuous and mixed-integer nonlinear programming. *Kluwer Academic Publishing, Dordrecht, The Netherlands*, 2002.

[140] M. Tawarmalani and N.V. Sahinidis. Global optimization of mixed-integer nonlinear programs: A theoretical and computational study. *Mathematical programming*, 99(3):563–591, 2004.

[141] M. Tawarmalani and N.V. Sahinidis. A polyhedral branch-and-cut approach to global optimization. *Mathematical Programming*, 103(2):225–249, 2005.

[142] G. L. Torres and V. Quintana. An interior-point method for nonlinear optimal power flow using voltage rectangular coordinates. *IEEE Transactions on Power Systems*, 13:1211–1218, 1998.

[143] Sicco Verwer and Yingqian Zhang. Learning optimal classification trees using a binary linear program formulation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1625–1632, 2019.

[144] S. Vigerske and Ambros G. Scip: Global optimization of mixed-integer nonlinear programs in a branch-and-cut framework. *Optimization Methods and Software*, pages 1–31, 2017.

[145] V. Visweswaran and C.A. Floudas. A global optimization algorithm (gop) for certain classes of nonconvex nlps-ii. application of theory and test problems. *Computers & chemical engineering*, 14(12):1419–1434, 1990.

[146] V. Visweswaran and C.A. Floudas. New properties and computational improvement of the gop algorithm for problems with quadratic objective functions and constraints. *Journal of Global Optimization*, 3(4):439–462, 1993.

[147] A. Wächter and L.T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, Mar 2006.

[148] H. Wang, C.E. Murillo-Sánchez, R.D. Zimmerman, and R.J. Thomas. On computational issues of market-based optimal power flow. *IEEE Transactions on Power Systems*, 22(3):1185–1193, 2007.

[149] Y.C. Wu, A. Debs, and R. Marsten. A direct nonlinear predictor-corrector primal-dual interior point algorithm for optimal power flows. *IEEE Transactions on Power Systems*, 9:876–883, 1993.

[150] W. Xie and N.V. Sahinidis. A branch-and-bound algorithm for the continuous facility layout problem. *Computers & Chemical Engineering*, 32(4):1016 – 1028, 2008. Festschrift devoted to Rex Reklaitis on his 65th Birthday.

[151] Y. Yajima and T. Fujie. A polyhedral approach for nonconvex quadratic programming problems with box constraints. *Journal of Global Optimization*, 13(2):151–170, 1998.

[152] Q. Zhao. Semidefinite programming for assignment and partitioning problems. *Ph.D. Thesis, University of Waterloo, Ontario*, 1996.

[153] Q. Zhao, S.E. Karisch, F. Rendl, and H. Wolkowicz. Semidefinite relaxations for the quadratic assignment problem. *Journal of Combinatorial Optimization*, 2:71–109, 1998.

[154] K. Zorn and N.V. Sahinidis. Computational experience with applications of bilinear cutting planes. *Industrial & Engineering Chemistry Research*, 52(22):7514–7525, 2013.

[155] K. Zorn and N.V. Sahinidis. Global optimization of general non-convex problems with intermediate bilinear substructures. *Optimization Methods and Software*, 29(3):442–462, 2014.