Quadratization and convexification in polynomial binary optimization

Yves Crama, Sourour Elloumi, Amélie Lambert, Elisabeth Rodríguez-Heck

Quadratization and convexification in polynomial binary optimization

Yves Crama[1]*, Sourour Elloumi[2], Amélie Lambert[3], Elisabeth Rodríguez-Heck[4]

October 13, 2025

[1]HEC - Management School of the University of Liège, 14 rue Louvrex, 4000 Liège, Belgium. yves.crama@uliege.be. *Corresponding author.

[2]Unité de Mathématiques Appliquées, ENSTA Paris, Institut Polytechnique de Paris, 91120 Palaiseau, France. sourour.elloumi@ensta-paris.fr

[3]CEDRIC-Cnam, 292 rue Saint Martin, F-75141 Paris Cedex 03, France. amelie.lambert@cnam.fr

[4]RWTH Aachen University, The Chair of Operations Research, Kackertstraße 7, 52072 Aachen, Germany. erodriguezheck@gmail.com

Abstract

In this paper, we discuss several reformulations and solution approaches for the problem of minimizing a polynomial in binary variables (P). We review and integrate different literature streams to describe a methodology consisting of three distinct phases, together with several possible variants for each phase. The first phase determines a recursive decomposition of each monomial of interest into pairs of submonomials, down to the initial variables. The decomposition gives rise to a so-called *quadratization scheme*. The second phase builds a *quadratic reformulation* of (P) from a given quadratization scheme, by associating a new auxiliary variable with each submonomial that appears in the scheme. A quadratic reformulation of (P) is obtained by enforcing relations between the auxiliary variables and the monomials that they represent, either through linear constraints or through penalty terms in the objective function. The resulting quadratic problem (QP) is nonconvex in general and is still difficult to solve. At this stage we introduce the third phase of the resolution process, which consists in *convexifying* (QP). We consider different types of convexification methods, including complete linearization or quadratic convex reformulations. Mathematical properties of the different phases are formally established and some relations between them are clarified. Finally, we present some experimental results which illustrate the discussion and which support the practical relevance of quadratic reformulation methods.

Keywords: nonlinear binary optimization; quadratic binary optimization; reformulation; convexification.

1 Introduction

We consider the *Polynomial Unconstrained Binary Programming* (P) problem

$$(\mathrm{P}) \quad \begin{cases} \min \quad f(x) \triangleq \sum_{M \in \mathcal{M}} a_M \prod_{i \in M} x_i & (1) \\ \text{s. t.} \quad x_i \in \{0,1\} \qquad \forall i \in [n] & (2) \end{cases}$$

where the objective function $f(x)$ is a polynomial defined on binary variables $x \in \{0,1\}^n$. We denote the set of variable indices by $[n] = \{1, ..., n\}$. Without loss of generality, we can assume that $f(x)$ is a multilinear polynomial since $x_i^2 = x_i$ for any binary variable $x_i \in \{0,1\}$, for $i \in [n]$. Hence, $f(x)$ consists of a sum of *monomials* $\prod_{i \in M} x_i$ weighted by non-zero coefficients $a_M \in \mathbb{Q}$, for $M \in \mathcal{M} \subseteq 2^{[n]}$. For the sake of simplicity, we will also use the word *monomial* to refer to a subset of indices $M \subseteq 2^{[n]}$. The *degree of a monomial* M is $|M|$, and the *degree of the polynomial* f is defined as $d_f = \max_{M \in \mathcal{M}} |M|$.

Problem (P) is a very general model that allows the formulation of many well-known problems in optimization. When $d_f = 2$, the problem becomes a quadratic unconstrained binary optimization (QUBO) problem. This important special case encompasses classical combinatorial optimization problems such as maximum cut or stable set problems; it has recently been identified as playing a central role in quantum computing (see for example [14, 19, 37, 58]). When $d_f \geq 3$, (P) can be used to formulate many classical problems such as uncapacitated facility location in operations research, 3-SAT and maximum satisfiability in computer science, or applications in different fields such as the construction of binary sequences with low aperiodic correlation, a very challenging problem in signal design theory, or the restoration of blurred images in computer vision; see, e.g., [7, 14, 19, 12, 33, 45, 48], as well as Section 6.1 and Section 6.2 hereunder.

Since (P) generalizes unconstrained binary quadratic optimization, it is NP-hard for any fixed $d_f \geq 2$ [35]. Practical difficulties arise from the non-convexity of f and from the integrality of the variables. Various approaches have been proposed to handle these difficulties. They will be briefly reviewed in Section 2.

In this paper, we focus on a class of methods based on quadratic reformulations of problem (P), which are subsequently solved via convexification techniques. These methods have their roots in a seminal paper by Rosenberg [61] and have been more recently revived by several researchers (see, e.g., [6, 12, 17]; more references will be provided in subsequent sections).

The methods considered in this paper consist of three phases, and we will describe different techniques for each phase in subsequent sections. *Phase I* consists in determining a recursive decomposition of each monomial into a product of two sub-monomials, where each sub-monomial is again decomposed into a product of two, and so forth down to the initial variables. This decomposition is called a *quadratization scheme*. *Phase II* consists in building a *quadratic reformulation*, that is, a quadratic problem equivalent to (P), from any given quadratization scheme. For this purpose, each element of the given quadratization scheme is associated with an auxiliary variable that models a submonomial. The basic idea is then to enforce relations between auxiliary variables and the product that they represent, which can be done using either linear constraints as in [32], or quadratic penalties in the objective function as in [6, 61]. The resulting quadratic problem is non-convex in general and still difficult to solve. At this stage we consider *Phase III* of the resolution process, or *convexification* phase, which consists in computing a new formulation equivalent to the quadratic problem obtained in Phase II, but this time with a convex continuous relaxation. We

consider two types of convexification approaches. In the first one, we simply produce a standard linearization of the quadratic problem obtained in Phase II. In the second type, we compute an equivalent formulation which is quadratic and convex using semi-definite programming [8, 29]. Both approaches can be applied to any unconstrained quadratic binary optimization problem, whether it is a quadratic reformulation of a polynomial problem of type (P) or not. However, in the particular case at hand, where we deal with a quadratic reformulation of a higher-degree problem, we can also use the recent `PQCR` [29] method, which improves the convex reformulation.

Our contribution. Research on quadratic reformulations of (P) has frequently examined one or two of the phases in isolation, e.g., by focusing on the choice of the quadratic reformulation obtained from Phases I and II, and then relying on generic solvers to handle Phase III [17, 60], or conversely, by picking a simple quadratic reformulation, and then focusing on the choice of the convexification method in Phase III [29]. The aim of the present paper is to revisit the three-phase approach from a more holistic perspective. It aims at examining the joint impact of the phases, from quadratic reformulation to convexification. On the theoretical side, we provide a unified framework for various definitions of quadratic reformulations that were independently introduced in the literature. We rigorously establish some mathematical properties of these reformulations and we clarify the relationships among them (e.g., the quality of the bounds that they deliver). We discuss which quadratic reformulations can be meaningfully combined with the convexification methods presented in [8, 29]. From a computational perspective, we conduct a set of experiments which illustrate the difficulty of choosing the best combination of a quadratic reformulation with a convexification technique, as this choice may depend on the type of instances considered. The experimental results also demonstrate that quadratic reformulations are at the very least competitive with other classes of approaches, such as linearization or outer approximation, and may even provide the most efficient solution approach in some contexts.

Outline of the paper. Section 2 provides a brief literature review and lays out the fundamental concepts required to formally describe the three-phase approach considered in this paper. Sections 3, 4 and 5 respectively discuss Phases I, II and III in detail. Section 6 presents the results of computational tests aimed at experimentally comparing different combinations of quadratic reformulations and convexification methods, and Section 7 concludes the paper.

2 Literature review and basic concepts

This section reviews the literature on algorithms that can be applied to solve polynomial unconstrained binary programming problems. We mostly focus on the case of polynomials of degree larger than 2, since the paper is primarily concerned with this case. We also exclude from our discussion some specialized combinatorial approaches based on dynamic programming, like the *basic algorithm* proposed in [41, 42], and closely related methods described in [18, 20, 23].

2.1 Generic approaches

Problem (P) can be viewed as a special case of a mixed-integer nonlinear program (MINLP), therefore, generic approaches to solve MINLPs can be applied to (P). Most of these methods are branch-and-bound algorithms based on either linear or quadratic convex under-estimators that

are usually defined in an extended space of variables. Several solvers implement this approach, for instance, `Baron` [63] or `SCIP` [1, 10]. It is also possible to obtain a convex under-estimator in the original space of variables. The αBB algorithm is based on this idea, and obtains a convex relaxation by perturbing the diagonal terms of the Hessian of the objective function [2]. The solver `Antigone` [56] implements this algorithm.

In the case where the objective function is a polynomial, but the variables are continuous, Lasserre proposed an algorithm based on a hierarchy of semi-definite relaxations [50]. The idea is, at each rank of the hierarchy, to successively tighten semi-definite relaxations until an optimal solution value is reached. In the binary case, this hierarchy converges to the optimal solution of the problem in a finite number of iterations [64]. Later, this work has been extended to hierarchies of second-order cone programs [3, 36, 49] and of sparse doubly non-negative relaxations [46]. Finally, in [16, 51], the authors build separable or convex under-estimators. Although these algorithms were not originally tailored for binary programming, they can handle (P) by imposing $x_i^2 = x_i$ for $i \in [n]$.

2.2 Linear reformulations

A classical approach to solve (P) consists in applying branch-and-bound to an appropriate linear reformulation of (P), such as the *standard linearization*, based on the pioneering work published in [32, 39, 66, 67]. This reformulation is proposed by most commercial solvers, like [40] or [44], as an option to the user. The fundamental idea behind standard linearization is to introduce an auxiliary variable y_M for every monomial $M \in \mathcal{M}$, and to force the equality $y_M = \prod_{i \in M} x_i$ by means of linear constraints. This leads to the following mixed-integer linear programming reformulation (SL):

$$
\text{(SL)} \quad \left\{ \begin{array}{lll}
\min & \displaystyle\sum_{i=1}^{n} a_i x_i + \sum_{M \in \mathcal{M}: |M| \geq 2} a_M y_M & \\
\text{s. t.} & & \\
& y_M \leq x_i & M \in \mathcal{M} : |M| \geq 2,\ i \in M \qquad (3) \\
& y_M \geq \displaystyle\sum_{i \in M} x_i - (|M| - 1) & M \in \mathcal{M} : |M| \geq 2 \qquad (4) \\
& y_M \geq 0 & M \in \mathcal{M} : |M| \geq 2 \qquad (5) \\
& x \in \{0,1\}^n & \qquad (6)
\end{array} \right.
$$

It is easy to check that the inequalities (3), (4), (5) imply $y_M = \prod_{i \in M} x_i$ when x is a binary vector. Therefore, problem (SL) is equivalent to (P).

Problem (SL) has been studied by many researchers. When $f(x)$ consists of a single monomial or has a specific acyclic structure [15, 26], its continuous relaxation is exact (i.e., its extreme points are binary). For the general case, several recent publications present valid inequalities for the convex hull of the feasible binary solutions, see, e.g., [21, 24, 25, 26, 27, 28, 30]. In [17], Buchheim and Rinaldi established a polyhedral description based on a quadratic reformulation of (P). They also proved the equivalence between separation algorithms for (SL) and for the cut polytope. Compact linearizations requiring a smaller number of additional variables than (SL) have been introduced in [38, 39] and in other papers.

2.3 Quadratic reformulations

Finally, another stream of literature focuses on quadratic reformulations of problem (P). Since these type of reformulations are the main focus of this paper, we will formally introduce several definitions before reviewing the related literature.

Phases I and II of the algorithms that we consider in the present paper are aimed at reformulating (P) as an equivalent quadratic problem of the form

$$\text{(QP)} \quad \begin{cases} \min & G(z) \triangleq z^T Q z + c^T z \\ \text{s. t.} & A_i z = b_i \quad \forall i \in [l] \\ & z^T Q_i z + c_i^T z = b_i \quad \forall i \in [q] \\ & z \in \{0,1\}^{n+m}. \end{cases}$$

The dimension of the binary vector z is $n+m$: the first n components of z correspond to the *original* variables $x \in \{0,1\}^n$ of problem (P) and the last m components correspond to a vector of *auxiliary* variables $y \in \{0,1\}^m$ which are introduced to define the quadratic reformulation. The correspondence between auxiliary and original variables is enforced by using either linear constraints $A_i z = b_i$, for $i \in [l]$, or quadratic constraints $z^T Q_i z + c_i^T z = b_i$, for $i \in [q]$, or penalties in the quadratic objective function (in which case l or q may be 0), or both penalties and constraints. In the special case of a (QP) formulation with $l = q = 0$, the problem becomes a quadratic unconstrained binary optimization (QUBO) problem.

We now formally define the concept of quadratic reformulation.

Definition 1 (**Quadratic reformulation of a polynomial optimization problem**)**.** (QP) is a *quadratic reformulation* of (P) if and only if, for every optimal solution $\hat{z} = (\hat{x}, \hat{y}) \in \{0,1\}^{n+m}$ of (QP), $\hat{x}$ is an optimal solution of (P).

In the quadratic reformulations to be discussed in this paper, it will always be the case that (P) and (QP) have the same optimal value, i.e., $f(\hat{x}) = G(\hat{z})$, although this condition is not strictly required in Definition 1.

The following related definition was previously introduced in several papers; see [5, 6, 13]. It applies to any *pseudo-Boolean function*, that is, to any real-valued function $f(x)$ defined on $x \in \{0,1\}^n$.

Definition 2 (**Quadratization of a pseudo-Boolean function**)**.** Given a pseudo-Boolean function $f(x)$ on $\{0,1\}^n$, a function $h(z) = h(x,y)$ is a *quadratization* of $f(x)$ if h is a quadratic polynomial depending on the original variables $x \in \{0,1\}^n$ and on a set of auxiliary variables $y \in \{0,1\}^m$ such that

$$f(x) = \min\{h(x,y) : y \in \{0,1\}^m\} \quad \forall x \in \{0,1\}^n.$$

Clearly, when $h(z)$ is a quadratization of the pseudo-Boolean function $f(x)$, then the problem: $\min\{h(z) : z \in \{0,1\}^{n+m}\}$ is a straightforward unconstrained quadratic reformulation of the polynomial optimization problem (P). This type of reformulation will be of special interest in the sequel, as the quadratic reformulations that we consider are closely related to quadratizations of the objective function.

It is well-known that every function $f(x)$ has a quadratization, and that a quadratization can be built in time polynomial in the size of f (see for instance Remark 3 in Section 4.2, and [6] for a detailed discussion). Common goals for finding good quadratic reformulations in the literature

have been: introducing as few auxiliary variables as possible in order to limit the dimension of the reformulated problem, or introducing a small number of (non-submodular) positive quadratic terms in the objective function, as this is often a good property for computational performance [6, 11, 33, 45, 60]. Other goals have been to define quadratic reformulations for particular classes of functions, such as symmetric functions [5, 11].

In this paper, we restrict our attention to quadratic reformulations resulting from the application of Phases I and II, that is, quadratic reformulations constructed on the basis of a quadratization scheme. Such quadratic reformulations have been previously considered in the literature and presented in various ways [6, 17, 29, 33, 61]. In the next sections, we provide a unified presentation of these methods and clarify the relations between them.

3 Phase I: Construction of quadratization schemes

Let us consider the first phase of the construction of a convex reformulation of (P), which consists in the elaboration of a *quadratization scheme*. The goal is to determine a valid decomposition of the monomials of the objective function f that will allow us to produce a quadratic reformulation in Phase II. Concretely, each monomial is recursively decomposed into exactly two products, down to the initial variables. Then, each element of the decomposition will be substituted by an auxiliary variable in the quadratic reformulation phase, thus reducing the degree of f. We further link this definition with related concepts existing in the literature.

3.1 Definition of quadratization schemes

We start with the definition of a scheme for a single monomial.

Definition 3 (**Quadratization scheme of a monomial**)**.** A *quadratization scheme* for a monomial M, where $|M| \geq 3$, is a directed acyclic graph $G_M = (V_M, A_M)$ with the following properties:

1. $V_M \subseteq 2^M$, i.e., each vertex in V_M is a subset of M;
2. $M \in V_M$; M is the *root* of G_M: it has indegree 0, and all other vertices have nonzero indegree;
3. $\{i\} \in V_M$ for all $i \in M$; the vertices $\{i\}$, $i \in M$, are the *leaves* of G_M: they have outdegree 0;
4. when a vertex $E \in V_M$ is not a leaf of G_M, its outdegree is 2, and
   - the arcs leaving E are denoted as $(E, l_M(E)) \in A_M$ and $(E, r_M(E)) \in A_M$; $l_M(E)$ and $r_M(E)$ are the *left child* and *right child* of E, respectively;
   - $l_M(E)$ and $r_M(E)$ define a nontrivial decomposition of E into two subsets: $0 < |l_M(E)| < |E|$, $0 < |r_M(E)| < |E|$, and $E = l_M(E) \cup r_M(E)$.

For simplicity, we assume that each subset of M can appear at most once as a vertex in V_M, although this assumption could be lifted at the expense of heavier notations. In the sequel, it will be useful to have a distinct notation for the set $\mathcal{E}_M \triangleq \{E \in V_M : |E| \geq 2, E \neq M\}$ which contains all vertices of G_M except the root M and the leaves $\{i\}$, $i \in M$. As a matter of terminology, remember that a *hypergraph* $\mathcal{H}$ on a ground set V is a set of subsets of V, that is, a set $\mathcal{H} \subseteq 2^V$. The elements of $\mathcal{H}$ are the *edges* of the hypergraph. So, V_M and $\mathcal{E}_M$ are hypergraphs on M.

We look at l_M and r_M as applications on $\mathcal{E}_M \cup \{M\}$, and we sometimes find it convenient to extend them to V_M by defining $l_M(\{i\}) = r_M(\{i\}) = \emptyset$ for all $i \in M$. Also, note that the quadratization scheme $G_M = (V_M, A_M)$ is equivalently defined by the triplet (V_M, l_M, r_M) or by the triplet $(\mathcal{E}_M, l_M, r_M)$; we will use either of these notations.

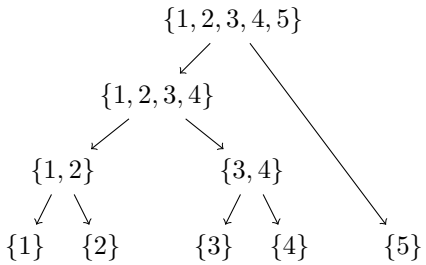
To illustrate the definitions, let us give three examples of quadratization schemes for a degree 5 monomial.

Example 1. Consider the monomial $M = \{1, 2, 3, 4, 5\}$.

We define (V_M^1, l_M^1, r_M^1), a first quadratization scheme of M, by $V_M^1 = \Big\{\{1\},\{2\},\{3\},\{4\},\{5\},\{1,2\},\{3,4\},\{1,2,3,4\},\{1,2,3,4,5\}\Big\}$ with

- $l_M^1(\{1,2\}) = \{1\}$, $r_M^1(\{1,2\}) = \{2\}$
- $l_M^1(\{3,4\}) = \{3\}$, $r_M^1(\{3,4\}) = \{4\}$
- $l_M^1(\{1,2,3,4\}) = \{1,2\}$, $r_M^1(\{1,2,3,4\}) = \{3,4\}$
- $l_M^1(\{1,2,3,4,5\}) = \{1,2,3,4\}$, $r_M^1(\{1,2,3,4,5\}) = \{5\}$

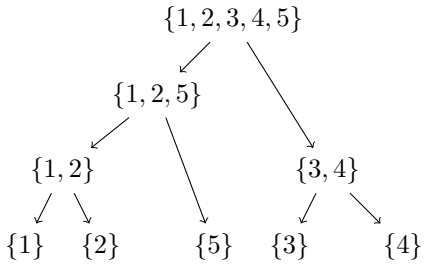
and $\mathcal{E}_M^1 = \Big\{\{1,2\},\{3,4\},\{1,2,3,4\}\Big\}$.



Next, we define (V_M^2, l_M^2, r_M^2), a second quadratization scheme of M, by $V_M^2 = \Big\{\{1\},\{2\},\{3\},\{4\},\{5\},\{1,2\},\{3,4\},\{1,2,5\},\{1,2,3,4,5\}\Big\}$ with

- $l_M^2(\{1,2\}) = \{1\}$, $r_M^2(\{1,2\}) = \{2\}$
- $l_M^2(\{3,4\}) = \{3\}$, $r_M^2(\{3,4\}) = \{4\}$
- $l_M^2(\{1,2,5\}) = \{1,2\}$, $r_M^2(\{1,2,5\}) = \{5\}$
- $l_M^2(\{1,2,3,4,5\}) = \{1,2,5\}$, $r_M^2(\{1,2,3,4,5\}) = \{3,4\}$

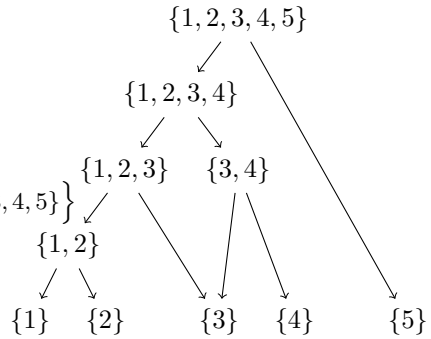
and $\mathcal{E}_M^2 = \Big\{\{1,2\},\{3,4\},\{1,2,5\}\Big\}$.



Finally, a third quadratization scheme (V_M^3, l_M^3, r_M^3) is given by $V_M^3 = \Big\{\{1\},\{2\},\{3\},\{4\},\{5\},\{1,2\},\{3,4\},\{1,2,3\},\{1,2,3,4\},\{1,2,3,4,5\}\Big\}$ with

- $l_M^3(\{1,2\}) = \{1\}$, $r_M^3(\{1,2\}) = \{2\}$
- $l_M^3(\{3,4\}) = \{3\}$, $r_M^3(\{3,4\}) = \{4\}$
- $l_M^3(\{1,2,3\}) = \{1,2\}$, $r_M^3(\{1,2,3\}) = \{3\}$
- $l_M^3(\{1,2,3,4\}) = \{1,2,3\}$, $r_M^3(\{1,2,3,4\}) = \{3,4\}$
- $l_M^3(\{1,2,3,4,5\}) = \{1,2,3,4\}$, $r_M^3(\{1,2,3,4,5\}) = \{5\}$

and $\mathcal{E}_M^3 = \Big\{\{1,2\},\{3,4\},\{1,2,3\},\{1,2,3,4\}\Big\}$.



As illustrated by Example 1, a quadratization scheme provides a decomposition of monomials

into pairs of subsets, which are themselves decomposed into pairs, and so forth, until singletons are obtained. Reading a scheme bottom-up, starting from the leaves, suggests how pairs of original or auxiliary variables can be recursively substituted by new ones. We will return to this interpretation in more detail in Section 4.

In Definition 3, we did not assume that $l_M(E)$ and $r_M(E)$ form a partition of E.

Definition 4. The scheme G_M is called *disjoint* if $l_M(E) \cap r_M(E) = \emptyset$ for every $E \in V_M$, $|E| \geq 2$.

For example, the first two schemes in Example 1 are disjoint, the third one is not. Let us call G_M a *rooted binary tree* if every vertex of G_M other than the root has indegree equal to 1.

Proposition 1. *A quadratization scheme is disjoint if and only if it is a rooted binary tree.*

Proof. If $l_M(E) \cap r_M(E) \neq \emptyset$ holds for some E, say, $i \in l_M(E) \cap r_M(E)$, then there is a directed path from $l_M(E)$ to $\{i\}$ and from $r_M(E)$ to $\{i\}$. These two paths meet in a first vertex which therefore has indegree at least 2, and hence G_M is not a tree.

Conversely, assume that G_M is not a rooted binary tree, meaning that there is a vertex $E \in V_M$ with indegree larger than 1. So, there exist two distinct paths from the root M to E. Let $F \in V_M$ be a vertex where these two paths diverge; say, $l_M(F)$ is on the first path and $r_M(F)$ is on the second one. Since E is a descendant of both $l_M(F)$ and $r_M(F)$, it follows that $E \subseteq l_M(F) \cap r_M(F)$, and the scheme is not disjoint. □

We now characterize the size of a quadratization scheme.

Proposition 2 (Size of a quadratization scheme). *If $G_M = (V_M, A_M)$ is a quadratization scheme for the monomial M, where $|M| \geq 3$, then:*

1. *$|V_M| \geq 2|M| - 1$ and $|\mathcal{E}_M| \geq |M| - 2$.*
2. *If the scheme is disjoint, $|V_M| = 2|M| - 1$ and $|\mathcal{E}_M| = |M| - 2$.*

Proof. By definition of $\mathcal{E}_M$, $|\mathcal{E}_M| = |V_M| - |M| - 1$ for every quadratization scheme. So, we only need to establish the claims about $|V_M|$. Let $M = \{1, \ldots, n\}$. If $n = 3$, the claims are easily checked by enumeration. If $n > 3$, let $E_1, \ldots, E_k$ be the parent nodes of the leaf $\{n\}$ and let F_i be the second child of E_i for $i = 1, \ldots, k$, i.e., $F_i = E_i \setminus \{n\}$. Consider the graph obtained from G_M by deleting node $\{n\}$ and by replacing every node $J \in V_M$ by $J \setminus \{n\}$. This graph contains two copies of each node F_i. By contracting the arc between these two copies, for each i, we obtain a quadratization scheme $G = (V, A)$ for the monomial $\{1, \ldots, n-1\}$ with $|V| = |V_M| - 1 - k$. The claims follow by induction on n since $k = 1$ when the scheme G_M is disjoint. □

Let us now turn to the case of a generic polynomial function f, as in Equation (1). Since a polynomial is defined by a set of monomials $\mathcal{M}$ (together with their coefficients), we define a quadratization scheme for a set of monomials or equivalently, for a hypergraph $\mathcal{M}$.

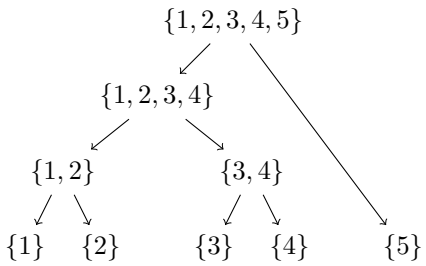
Definition 5 (Quadratization scheme for a set of monomials). A *quadratization scheme* for a hypergraph $\mathcal{M} \subseteq 2^{[n]}$ is a collection of quadratization schemes $\mathcal{S} = \Big\{G_M = (V_M, A_M) : M \in \mathcal{M}\Big\}$, where each G_M is a quadratization scheme for the corresponding monomial $M \in \mathcal{M}$.

Observe that a same subset $E \subseteq [n]$ can appear in several of the schemes G_M. Therefore, the collection of sets $(E : E \in V_M, M \in \mathcal{M})$ is a multiset, or a hypergraph with repeated edges. Nevertheless, we find it convenient to look at $\mathcal{S}$ as a collection of vertex disjoint digraphs, as illustrated in Example 2 hereunder. Moreover, we denote as $\mathcal{E}$ the hypergraph (without repeated edges)

$$\mathcal{E} \triangleq \bigcup_{M \in \mathcal{M}} \mathcal{E}_M = \bigcup_{M \in \mathcal{M}} (V_M \setminus (\{M\} \cup \bigcup_{i \in M} \{i\})). \tag{7}$$

Note that even though $M \notin \mathcal{E}_M$, $\mathcal{M} \cap \mathcal{E}$ might be nonempty if a monomial $M \in \mathcal{M}$ appears in the quadratization scheme of another monomial, that is, if $M \in \mathcal{E}_{M'}$ for $M' \in \mathcal{M}, M' \neq M$.

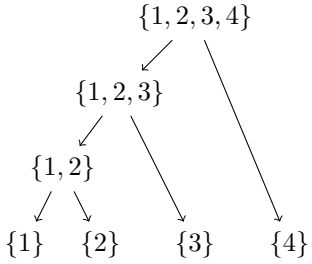
Example 2. Let $f(x) = a_1 x_1 x_2 x_3 x_4 x_5 + a_2 x_1 x_2 x_3 x_4$ be a polyomial containing two monomials, $\mathcal{M} = \{M_1 = \{1,2,3,4,5\}, M_2 = \{1,2,3,4\}\}$, and $n = 5$. A quadratization scheme $\mathcal{S} = \Big\{G_{M_1}, G_{M_2}\Big\}$ for $\mathcal{M}$ can be defined as follows:



$V_{M_1} = \Big\{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{1,2\}, \{3,4\}, \{1,2,3,4\}, \{1,2,3,4,5\}\Big\}$

- $l_{M_1}(\{1,2\}) = \{1\}$, $r_{M_1}(\{1,2\}) = \{2\}$
- $l_{M_1}(\{3,4\}) = \{3\}$, $r_{M_1}(\{3,4\}) = \{4\}$
- $l_{M_1}(\{1,2,3,4\}) = \{1,2\}$, $r_{M_1}(\{1,2,3,4\}) = \{3,4\}$
- $l_{M_1}(\{1,2,3,4,5\}) = \{1,2,3,4\}$, $r_{M_1}(\{1,2,3,4,5\}) = \{5\}$

and $\mathcal{E}_{M_1} = \Big\{\{1,2\}, \{3,4\}, \{1,2,3,4\}\Big\}$.



$V_{M_2} = \Big\{\{1\}, \{2\}, \{3\}, \{4\}, \{1,2\}, \{1,2,3\}, \{1,2,3,4\}\Big\}$

- $l_{M_2}(\{1,2\}) = \{1\}$, $r_{M_2}(\{1,2\}) = \{2\}$
- $l_{M_2}(\{1,2,3\}) = \{1,2\}$, $r_{M_2}(\{1,2,3\}) = \{3\}$
- $l_{M_2}(\{1,2,3,4\}) = \{1,2,3\}$, $r_{M_2}(\{1,2,3,4\}) = \{4\}$

and $\mathcal{E}_{M_2} = \Big\{\{1,2\}, \{1,2,3\}\Big\}$.

The hypergraph $\mathcal{E}$ without repeated edges is $\mathcal{E} = \Big\{\{1,2\}, \{3,4\}, \{1,2,3\}, \{1,2,3,4\}\Big\}$.

3.2 Related concepts

Definitions closely related to Definitions 3 and 5 have been previously introduced in the literature, going back to the work of McCormick [54] or El-Khayyal and Falk [4] on factorable programs.

Closer to our framework, specific (disjoint) quadratization schemes for polynomial unconstrained binary programs have been discussed by Ryoo and Sahinidis [62] and more recently by Khajavirad [47] or Raghunathan et al. [59]. These authors focus on *linear* reformulations of problem (P) under the name of *recursive McCormick* linearizations.

In the context of *quadratic* reformulations, quadratization schemes have been implicitly used by Rosenberg [61]. Anthony et al. [6] have considered *pairwise covers* of the hypergraph associated with a polynomial, and *reducible* sets of monomials have been introduced by Buchheim and Rinaldi in [17]. Let us briefly discuss and clarify some of the relations between these concepts.

Definition 6. When $\mathcal{M}$, $\mathcal{H}$ are two hypergraphs, we say that $\mathcal{H}$ is a *pairwise cover* of $\mathcal{M}$ if, for every set $M \in \mathcal{M}$ with $|M| \geq 2$, there are two sets $l(M), r(M) \in \mathcal{H}$ such that $0 < |l(M)| < |M|$, $0 < |r(M)| < |M|$ and $l(M) \cup r(M) = M$.

The original definition in [6] is restricted to $|M| \geq 3$, but the adaptation is harmless and is more coherent with the present paper.

Example 3. Consider again the hypergraph $\mathcal{M} = \{M_1 = \{1,2,3,4,5\}, M_2 = \{1,2,3,4\}\}$ of Example 2. Then, $\mathcal{H} = \{\{4\}, \{5\}, \{1,2,3\}, \{1,2,3,4\}\}$ is a pairwise cover of $\mathcal{M}$ with $l(\{1,2,3,4,5\}) = \{1,2,3,4\}$, $r(\{1,2,3,4,5\}) = \{5\}$, $l(\{1,2,3,4\}) = \{1,2,3\}$, and $r(\{1,2,3,4\}) = \{4\}$.

The definition of pairwise covers is obviously related to the definition of quadratization schemes and in fact, in Example 3, the decomposition of the monomials M_1, M_2 is the same as in the quadratization scheme of Example 2. A main difference, however, is that the definition of pairwise covers does not require a left-right decomposition of each subset $S \in \mathcal{H}$: the condition only applies to subsets $M \in \mathcal{M}$. In other words, pairwise covers only relate to the first level of quadratization schemes, just under the root monomials.

The exact relation between pairwise covers and quadratization schemes is clarified by the next result.

Proposition 3. *If* $\mathcal{S} = \left\{(V_M, A_M) : M \in \mathcal{M}\right\}$ *is a quadratization scheme for* $\mathcal{M}$*, then* $\mathcal{H} = \bigcup_{M\in\mathcal{M}}\{l_M(M), r_M(M)\}$ *is a pairwise cover of* $\mathcal{M}$*. Conversely, if* $\mathcal{H}$ *is a pairwise cover of* $\mathcal{M}$ *such that* $\mathcal{H} \subseteq \mathcal{M}$*, then* $\mathcal{M}$ *has a quadratization scheme* $\mathcal{S} = \left\{(V_M, A_M) : M \in \mathcal{M}\right\}$ *such that* $\bigcup_{M\in\mathcal{M}} V_M \subseteq \mathcal{H}$.

Proof. The first claim is obvious. For the second one, assume that $\mathcal{H}$ is a pairwise cover of $\mathcal{M}$ such that $\mathcal{H} \subseteq \mathcal{M}$. For each monomial $M \in \mathcal{M}$, define V_M and A_M inductively as follows:

1. $M \in V_M$;
2. $(M, l(M)) \in A_M$ and $(M, r(M)) \in A_M$;
3. if $(E_1, E_2) \in A_M$, then $E_2 \in V_M$, and if $|E_2| \geq 2$, then $(E_2, l(E_2)) \in A_M$, $(E_2, r(E_2)) \in A_M$.

Then, (V_M, A_M) is a quadratization scheme for M and $V_M \subseteq \mathcal{H}$. □

Remark 1. The quadratization scheme defined for $\mathcal{M}$ in the previous proof has the property that, for all $M, M', M'' \in \mathcal{M}$, $l_{M'}(M) = l_{M''}(M) = l(M)$, and $r_{M'}(M) = r_{M''}(M) = r(M)$: in other words, the left-right decomposition of each set M is unique in the scheme, as opposed to what is illustrated in Example 2 where the set $\{1,2,3,4\}$ is decomposed in two different ways. This shows

that quadratization schemes, as we have introduced them in Definition 3 and Definition 5, offer more flexibility than decompositions based on pairwise covers. Using different decompositions for a same set may tighten the formulation by capturing additional aspects of the structure of the instance, but can also result in creating redundant constraints when building the quadratic reformulation in Phase II. Whether decomposing a set in two or more different ways is helpful or not, and how to best exploit this possibility, are questions for future research.

The second statement in Proposition 3 explains why Anthony et al. [6] restrict their attention to special types of pairwise covers. More precisely, Theorem 4 in [6] establishes constructively that, if $\mathcal{H}$ is a pairwise cover of $\mathcal{M}$ such that $\mathcal{H} \subseteq \mathcal{M}$, then the function $f(x) = \sum_{M \in \mathcal{M}} a_M \prod_{i \in M} x_i$ has a quadratization (recall Definition 2) using at most $|\mathcal{H}|$ auxiliary variables. Even though this is not explicitly mentioned in [6], it is easy to check that the existence of a pairwise cover $\mathcal{H}$ such that $\mathcal{H} \subseteq \mathcal{M}$ is equivalent to the condition that $\mathcal{M}$ is a pairwise cover of itself (say, a *pairwise self-cover*). Therefore, as a consequence of Theorem 4 in [6], we can state that if $\mathcal{M}$ is a pairwise self-cover, then $f(x)$ has a quadratization using at most $|\mathcal{M}|$ auxiliary variables. Given a function f, the pairwise cover quadratization methods to be described in subsequent sections will concentrate on extending $\mathcal{M}$ (that is, on adding monomials with coefficient zero in f) until $\mathcal{M}$ becomes a pairwise self-cover and hence, until a quadratization scheme becomes available by virtue of Proposition 3. Anthony et al. [6] observed that such a pairwise self-cover can be constructed in polynomial time.

In a different framework, Buchheim and Rinaldi [17] had previously proposed a heuristic procedure in order to extend $\mathcal{M}$ to a pairwise self-cover (which they call a *reducible set* of monomials). They used these notions to provide a quadratic reformulation of problem (P), and to show that a complete description of the convex hull of the (binary) solutions of the standard linearization formulation (3)-(6) can be derived from a complete description of the quadratic reformulation.

4 Phase II: Building quadratic reformulations

We now present an adaptation of several approaches of the literature for reformulating problem (P) into an equivalent quadratic optimization problem, in the sense of Definition 1. Different types of reformulations that do not rely on quadratization schemes have been proposed, for instance, in [5, 6, 11, 13, 31, 33, 45]. In this paper, however, we restrict our attention to reformulations derived from an arbitrary quadratization scheme $\mathcal{S}$.

To do this, the basic idea is to introduce a set of $|\mathcal{E}|$ *auxiliary variables*, where $\mathcal{E}$ is defined by equation (7), so that each additional variable corresponds to an "intermediate" vertex in one of the graphs G_M. Let $z \in \{0,1\}^N$, where $N = n + |\mathcal{E}|$, the first n components of z correspond to the vector of *original variables* x, and the last $|\mathcal{E}|$ components correspond to the auxiliary variables. If the equalities $z_E = z_{l_M(E)} z_{r_M(E)}$ hold for all $E \in \mathcal{E}_M$, then it follows from the definition of quadratization schemes that $z_E = \prod_{i \in E} z_i$ also holds for all $E \in \mathcal{E}_M$. When this is the case, we say that z defines a *consistent assignment* of values to the variables, or simply, that z is *consistent*. It follows that, given a scheme $\mathcal{S}$, we can reformulate (P) as the equivalent quadratically constrained quadratic programming problem ($\texttt{QCQP}^{\mathcal{S}}$):

$$(\texttt{QCQP}^{\mathcal{S}})\begin{cases} \min\limits_{z\in\{0,1\}^N} & g(z) \triangleq \sum\limits_{\substack{M\in\mathcal{M}:\\ |M|\leq 2}} a_M \prod\limits_{i\in M} z_i + \sum\limits_{\substack{M\in\mathcal{M}:\\ |M|\geq 3}} a_M z_{l_M(M)} z_{r_M(M)} & (8) \\ \text{s.t.} & \\ & z_E = z_{l_M(E)} z_{r_M(E)} \qquad \forall E\in\mathcal{E}, \forall M\in\mathcal{M} : E\in\mathcal{E}_M. & (9) \end{cases}$$

Both the objective function and the constraints of ($\texttt{QCQP}^{\mathcal{S}}$) are usually non-convex. In this section, we focus on rewriting the quadratic constraints (9) and we show that they can be enforced either using linear inequalities or using penalties in the objective function. Doing so, we accordingly obtain either a linearly constrained quadratic problem, or an unconstrained quadratic problem. The resulting problems are still non-convex, in general, but now the non-convexity can only arise from the quadratic objective function and from the integrality of the variables, not from the additional constraints.

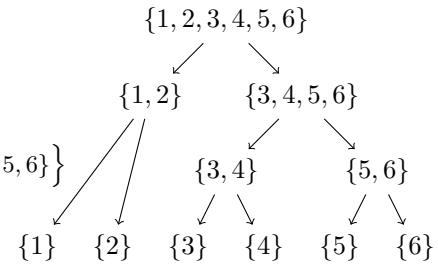
4.1 A quadratic reformulation based on standard linearizations

Similarly to the standard linearization of the objective function, the quadratic equations (9) can be reformulated by a set of linear constraints. This leads to the following quadratic reformulation ($\texttt{FOR}^{\mathcal{S}}$) (for Fortet [32]) of ($\texttt{QCQP}^{\mathcal{S}}$) and of (P):

$$(\texttt{FOR}^{\mathcal{S}})\begin{cases} \min\limits_{z\in\{0,1\}^N} & g(z) & \\ \text{s.t.} & & \\ & z_E \leq z_{l_M(E)},\ z_E \leq z_{r_M(E)} \qquad \forall E\in\mathcal{E}, \forall M\in\mathcal{M} : E\in\mathcal{E}_M & (10) \\ & z_E \geq z_{l_M(E)} + z_{r_M(E)} - 1,\ z_E \geq 0 \quad \forall E\in\mathcal{E}, \forall M\in\mathcal{M} : E\in\mathcal{E}_M & (11) \end{cases}$$

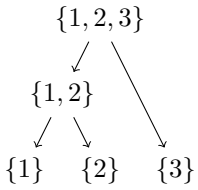
where $g(z)$ is defined by Equation (8).

Example 4. Let $f(x) = -6.5x_1x_2x_3x_4x_5x_6 - 5.6x_1x_2x_3$, with $\mathcal{M} = \{M_1 = \{1,2,3,4,5,6\}, M_2 = \{1,2,3\}\}$, and $n = 6$. Consider the quadratization scheme $\mathcal{S} = \left\{G_{M_1}, G_{M_2}\right\}$, where:



$V_{M_1} =$
$\Big\{\{1\},\{2\},\{3\},\{4\},\{5\},\{6\},\{1,2\},\{3,4\},\{5,6\},\{3,4,5,6\},\{1,2,3,4,5,6\}\Big\}$

- $l_{M_1}(\{1,2\}) = \{1\}$, $r_{M_1}(\{1,2\}) = \{2\}$
- $l_{M_1}(\{3,4\}) = \{3\}$, $r_{M_1}(\{3,4\}) = \{4\}$
- $l_{M_1}(\{5,6\}) = \{5\}$, $r_{M_1}(\{5,6\}) = \{6\}$
- $l_{M_1}(\{3,4,5,6\}) = \{3,4\}$, $r_{M_1}(\{3,4,5,6\}) = \{5,6\}$
- $l_{M_1}(M_1) = \{1,2\}$, $r_{M_1}(M_1) = \{3,4,5,6\}$



$V_{M_2} = \Big\{\{1\},\{2\},\{3\},\{1,2\},\{1,2,3\}\Big\}$

- $l_{M_2}(\{1,2\}) = \{1\}$, $r_{M_2}(\{1,2\}) = \{2\}$
- $l_{M_2}(M_2) = \{1,2\}$, $r_{M_2}(M_2) = \{3\}$

with $\mathcal{E}_{M_1} = \Big\{\{1,2\},\{3,4\},\{5,6\},\{3,4,5,6\}\Big\}$, $\mathcal{E}_{M_2} = \Big\{\{1,2\}\Big\}$, and $\mathcal{E} = \Big\{\{1,2\},\{3,4\},\{5,6\},\{3,4,5,6\}\Big\}$.

Standard linearization consists in performing the following substitutions $z_{\{1,2\}} = z_1 z_2$, $z_{\{3,4\}} = z_3 z_4$, $z_{\{5,6\}} = z_5 z_6$, $z_{\{3,4,5,6\}} = z_{\{3,4\}} z_{\{5,6\}}$ in $f(x)$ and adding the appropriate set of constraints. The final linearly constrained quadratic reformulation is:

$$(\texttt{FOR}^{\mathcal{S}})\begin{cases} \min\limits_{z\in\{0,1\}^{10}} g(z) = -6.5 z_{\{1,2\}} z_{\{3,4,5,6\}} - 5.6 z_{\{1,2\}} z_3 \\ \text{s.t. } z_{\{1,2\}} \le z_1,\ z_{\{1,2\}} \le z_2,\ z_{\{1,2\}} \ge z_1 + z_2 - 1,\ z_{\{1,2\}} \ge 0 \\ \quad z_{\{3,4\}} \le z_3,\ z_{\{3,4\}} \le z_4,\ z_{\{3,4\}} \ge z_3 + z_4 - 1,\ z_{\{3,4\}} \ge 0 \\ \quad z_{\{5,6\}} \le z_5,\ z_{\{5,6\}} \le z_6,\ z_{\{5,6\}} \ge z_5 + z_6 - 1,\ z_{\{5,6\}} \ge 0 \\ \quad z_{\{3,4,5,6\}} \le z_{\{3,4\}},\ z_{\{3,4,5,6\}} \le z_{\{5,6\}},\ z_{\{3,4,5,6\}} \ge z_{\{3,4\}} + z_{\{5,6\}} - 1,\ z_{\{3,4,5,6\}} \ge 0 \end{cases}$$

4.2 A quadratic reformulation based on Rosenberg's method

We first provide an explicit quadratic reformulation of (P) by applying the central idea of Rosenberg's method [61]. The original algorithm is iterative and produces a quadratic reformulation by selecting a pair of (original or auxiliary) variables say, z_i, z_j, and by substituting a new variable $z_{i,j}$ for the product $z_i z_j$ in the objective function. The equality $z_{i,j} = z_i z_j$ is enforced by adding to the function a quadratic penalty term of the form $p_{i,j}(3z_{i,j} - 2z_i z_{i,j} - 2z_j z_{i,j} + z_i z_j)$ with a large enough multiplier $p_{i,j}$ (it suffices to set $p_{i,j}$ equal to the sum of the absolute values of the coefficients of the terms containing the product $z_i z_j$; see, e.g., [19]). This step is repeated until the penalized objective function becomes quadratic.

The procedure can be interpreted as implicitly defining a quadratic scheme $\mathcal{S}$ bottom-up, starting from the leaves. In order to integrate it in our framework, we can actually translate it to produce

an explicit reformulation of (P) based on ($\texttt{QCQP}^{\mathcal{S}}$). Given a scheme $\mathcal{S}$, for each monomial M and $E \in \mathcal{E}_M$, let us introduce the penalty function

$$q_M^R(E) \triangleq 3z_E - 2z_{l_M(E)}z_E - 2z_{r_M(E)}z_E + z_{l_M(E)}z_{r_M(E)} \tag{12}$$

which possesses the following (easily verified) properties: for all binary values of z_E, $z_{l_M(E)}$, $z_{r_M(E)}$,

$$q_M^R(E) \geq 0, \tag{13}$$

$$z_E = z_{l_M(E)}z_{r_M(E)} \iff q_M^R(E) = 0. \tag{14}$$

If we repeatedly apply Rosenberg's approach to reduce the degree of M according to the quadratization scheme $\mathcal{S}$, we eventually find that the monomial $a_M \prod_{i\in M} x_i$ can be substituted in the objective function by

$$\rho_M(z) \triangleq a_M z_{l_M(M)}z_{r_M(M)} + |a_M| \sum_{E\in\mathcal{E}_M} q_M^R(E). \tag{15}$$

If we now apply the scheme $\mathcal{S}$ to the complete set of monomials $\mathcal{M}$ (i.e., to the function $f(x)$) and if we recall the definition of the function $g(z)$ in equation (8), we can transform the objective function into the quadratic function:

$$g^R(z) \triangleq \sum_{\substack{M\in\mathcal{M}:\\ |M|\leq 2}} a_M \prod_{i\in M} z_i + \sum_{\substack{M\in\mathcal{M}:\\ |M|\geq 3}} \rho_M(z) \tag{16}$$

$$= g(z) + \sum_{\substack{M\in\mathcal{M}:\\ |M|\geq 3}} |a_M| \sum_{E\in\mathcal{E}_M} q_M^R(E). \tag{17}$$

The resulting quadratic reformulation of ($\texttt{QCQP}^{\mathcal{S}}$) and of (P) is the following unconstrained problem:

$$(\texttt{ROS}^{\mathcal{S}}) \begin{cases} \min\limits_{z\in\{0,1\}^N} g^R(z). \end{cases}$$

For the sake of completeness, let us formally establish the correctness of the transformation (remember Definition 2).

Theorem 4. *For every quadratization scheme $\mathcal{S} = \{G_M = (V_M, l_M, r_M) : M \in \mathcal{M}\}$, the function $g^R(z)$ is a quadratization of $f(x)$ and ($\texttt{ROS}^{\mathcal{S}}$) is a quadratic reformulation of (P).*

Proof. Consider any binary minimizer of $g^R(z)$, say z^*, and let $x_i^* = z_i^*$ for $i \in [n]$. If z^* is consistent, then in view of Equation (14), $g^R(z^*) = f(x^*)$ as required for a quadratization.

On the other hand, if z^* is not consistent, then there is $M \in \mathcal{M}$ and $E \in \mathcal{E}_M$ such that $z_E^* \neq z_{l_M(E)}^* z_{r_M(E)}^*$. We say that z^* is inconsistent for this monomial M. In view of Equations (13)-(14), for any such M,

$$\rho_M(z^*) \geq a_M z_{l_M(M)}^* z_{r_M(M)}^* + |a_M| \geq \max(a_M, 0).$$

Next, define z^+ to be consistent with x^*, that is, let $z_S^+ = \prod_{i\in S} z_i^* = \prod_{i\in S} x_i^*$ for all subsets $S \in \bigcup_{\substack{M\in\mathcal{M}:\\ |M|\geq 3}} V_M$. For any monomial M, if z^* is consistent for M, then $\rho_M(z^+) = \rho_M(z^*)$. On the other hand, if z^* is inconsistent for M, then there are two cases:

1. if $\prod_{i \in M} x_i^* = 1$, then $\rho_M(z^+) = a_M z^+_{l_M(M)} z^+_{r_M(M)} = a_M \leq \max(a_M, 0) \leq \rho_M(z^*)$;

2. if $\prod_{i \in M} x_i^* = 0$, then $\rho_M(z^+) = a_M z^+_{l_M(M)} z^+_{r_M(M)} = 0 \leq \max(a_M, 0) \leq \rho_M(z^*)$.

It easily follows from these inequalities that $g^R(z^+) \leq g^R(z^*)$. Thus, $g^R(z)$ has a minimizer which is consistent (namely, z^+), and we conclude that $g^R(z)$ is a quadratization of $f(x)$ as in the first part of the proof. This implies, in turn, that $(\texttt{ROS}^{\mathcal{S}})$ is a quadratic reformulation of (P). □

Remark 2. For the record, it may be interesting to notice that in his original paper, Rosenberg [61] did not use the penalty multipliers $|a_M|$ that we use in $(\texttt{ROS}^{\mathcal{S}})$, but introduced a different, more flexible way to compute valid multipliers. As a result, the reformulation $g(z)$ produced by his original method does not necessarily yield a quadratization, contrary to what was suggested a bit hastily, for example in [6, 13].

Remark 3. The number of auxiliary variables in $g^R(z)$ is equal to the size of $\mathcal{E}$ in the quadratization scheme. Together with Proposition 2, this implies that a quadratization of the form $(\texttt{ROS}^{\mathcal{S}})$ can be built in polynomial time. Moreover, since Anthony et al. [6] observed that there exist quadratization schemes of size $O(2^{\frac{n}{2}})$ for every pseudo-Boolean function on n variables, it follows that every function has a quadratization involving $O(2^{\frac{n}{2}})$ auxiliary variables. This reasoning provides an alternative, easier proof of Theorem 4 in [6].

Remark 4. The function $g^R(z)$ can be rewritten as

$$g^R(z) = g(z) + \sum_{E \in \mathcal{E}} \Big(\sum_{\substack{M \in \mathcal{M}: \\ |M| \geq 3 \text{ and } E \in \mathcal{E}_M}} |a_M| \, q^R_M(E) \Big).$$

If each monomial has a unique decomposition in $\mathcal{S}$, that is, if $l_M(E) = l_{M'}(E) = l(E)$ and $r_M(E) = r_{M'}(E) = r(E)$ for all M, M' such that $E \in \mathcal{E}_M \cap \mathcal{E}_{M'}$, then this simply becomes

$$g^R(z) = g(z) + \sum_{E \in \mathcal{E}} \Big(\sum_{\substack{M \in \mathcal{M}: \\ |M| \geq 3 \text{ and } E \in \mathcal{E}_M}} |a_M| \Big) (3z_E - 2z_{l(E)} z_E - 2z_{r(E)} z_E + z_{l(E)} z_{r(E)}).$$

Here, the coefficient of the penalty function associated with variable z_E in $(\texttt{ROS}^{\mathcal{S}})$ is the sum of the absolute values of the coefficients of the terms of f whose decomposition involves E. This case arises, in particular, when $\mathcal{S}$ arises from a pairwise cover as explained in Proposition 3 (see Remark 1).

Example 5 (Example 4 continued)**.** To illustrate Remark 4, consider again the objective function $f(x) = -6.5x_1x_2x_3x_4x_5x_6 - 5.6x_1x_2x_3$, and the quadratization scheme $\mathcal{S}$. To build the penalized objective function we add a penalty function to $g(z)$ for each $E \in \mathcal{E}$:

- $P_{\{1,2\}}(3z_{\{1,2\}} - 2z_1 z_{\{1,2\}} - 2z_2 z_{\{1,2\}} + z_1 z_2)$ with $P_{\{1,2\}} = \sum_{\substack{M \in \mathcal{M}: \\ |M| \geq 3, \{1,2\} \in \mathcal{E}_M}} |a_M| = 6.5 + 5.6 = 12.1$
- $P_{\{3,4\}}(3z_{\{3,4\}} - 2z_3 z_{\{3,4\}} - 2z_4 z_{\{3,4\}} + z_3 z_4)$ with $P_{\{3,4\}} = \sum_{\substack{M \in \mathcal{M}: \\ |M| \geq 3, \{3,4\} \in \mathcal{E}_M}} |a_M| = 6.5$

- $P_{\{5,6\}}(3z_{\{5,6\}} - 2z_5 z_{\{5,6\}} - 2z_6 z_{\{5,6\}} + z_5 z_6)$ with $P_{\{5,6\}} = \sum_{\substack{M\in\mathcal{M}:\\ |M|\geq 3, \{5,6\}\in\mathcal{E}_M}} |a_M| = 6.5$
- $P_{\{3,4,5,6\}}(3z_{\{3,4,5,6\}} - 2z_{\{3,4\}} z_{\{3,4,5,6\}} - 2z_{\{5,6\}} z_{\{3,4,5,6\}} + z_{\{3,4\}} z_{\{5,6\}})$ with $P_{\{3,4,5,6\}} = \sum_{\substack{M\in\mathcal{M}:\\ |M|\geq 3,\\ \{3,4,5,6\}\in\mathcal{E}_M}} |a_M| = 6.5$

We obtain as final quadratic reformulation:

$$(\texttt{ROS}^{\mathcal{S}}) \begin{cases} \min\limits_{z\in\{0,1\}^{10}} & -6.5 z_{\{1,2\}} z_{\{3,4,5,6\}} - 5.6 z_{\{1,2\}} z_3 + 12.1(3z_{\{1,2\}} - 2z_1 z_{\{1,2\}} - 2z_2 z_{\{1,2\}} + z_1 z_2) \\ & +6.5(3z_{\{3,4\}} - 2z_3 z_{\{3,4\}} - 2z_4 z_{\{3,4\}} + z_3 z_4) + 6.5(3z_{\{5,6\}} - 2z_5 z_{\{5,6\}} - 2z_6 z_{\{5,6\}} + z_5 z_6) \\ & +6.5(3z_{\{3,4,5,6\}} - 2z_{\{3,4\}} z_{\{3,4,5,6\}} - 2z_{\{5,6\}} z_{\{3,4,5,6\}} + z_{\{3,4\}} z_{\{5,6\}}) \end{cases}$$

4.3 A quadratic reformulation based on ABCG quadratization

We call ABCG a quadratization procedure due to Anthony et al. [6] . Given a scheme $\mathcal{S}$, for each monomial M and $E \in \mathcal{E}_M$, let

$$q_M^A(E) \triangleq \big(2|E| - 1 - 2\sum_{j\in E} z_j\big) z_E + z_{l_M(E)} z_{r_M(E)}. \tag{18}$$

Let again $g(z)$ be given by Equation (8), and define

$$g^A(z) \triangleq g(z) + \sum_{\substack{M\in\mathcal{M}:\\ |M|\geq 3}} \sum_{E\in\mathcal{E}_M} \beta_M(E)\, q_M^A(E) \tag{19}$$

where the coefficients $\beta_M(E)$ are given by the following "top-down" recursion in the graph G_M:

- if $E \in \{l_M(M), r_M(M)\}$, then
$$\beta_M(E) = |a_M|; \tag{20}$$
- if $E \in \mathcal{E}_M \setminus \{l_M(M), r_M(M)\}$, then
$$\beta_M(E) = \sum_{\substack{S\in\mathcal{E}_M:\\ E\in\{l_M(S), r_M(S)\}}} \beta_M(S). \tag{21}$$

The recursive definition (20)-(21) has a simple graphical interpretation: for all $M \in \mathcal{M}$ and $E \in \mathcal{E}_M$, $\beta_M(E)$ is equal to $|a_M|\pi_{M,E}$, where $\pi_{M,E}$ is the number of directed paths from the root M to vertex E in the quadratization scheme G_M. In particular, $\beta_M(E) \geq |a_M| \geq 0$.

The quadratic expression $q_M^A(E)$ plays a similar role for ABCG as the quadratic expression $q_M^R(E)$ in Equation (12) for Rosenberg's procedure. However, contrary to $q_M^R(E)$, there is no guarantee that $q_M^A(E)$ is nonnegative for all assignments of values to the variables, so that it does not act as a classical penalty. The original motivation to consider $q_M^A(E)$ is that, for all binary values of z_j, $j \in E$:

$$\min_{z_E\in\{0,1\}} \big(2|E| - 1 - 2\sum_{j\in E} z_j\big) z_E + \prod_{j\in E} z_j = 0$$

(or equivalently, $\big(2|E|-1-2\sum_{j\in E} z_j\big)z_E$ is a quadratization of the negative monomial $(-\prod_{j\in E} z_j)$; see [34]). For our purpose, the following property will prove sufficient and is easily verified: for all binary values of $z_j, z_E, z_{l_M(E)}, z_{r_M(E)}$,

$$\text{if } z_E = z_{l_M(E)} z_{r_M(E)} \text{ and } z_E = \prod_{j\in E} z_j \text{ then } q_M^A(E) = 0. \tag{22}$$

Anthony et al. [6] proved that $g^A(z)$ is a quadratization of the original function $f(x)$ when the scheme $\mathcal{S} = \{G_M = (V_M, l_M, r_M) : M \in \mathcal{M}\}$ is associated with a pairwise cover. For the sake of completeness, we formally establish the next, more general statement, where ($\texttt{ABCG}^{\mathcal{S}}$) denotes the unconstrained quadratic minimization problem

$$(\texttt{ABCG}^{\mathcal{S}}) \left\{ \min_{z\in\{0,1\}^N} g^A(z) \right.$$

and $g^A(z)$ is defined by Equation (19).

Theorem 5. *For every quadratization scheme $\mathcal{S} = \{G_M = (V_M, l_M, r_M) : M \in \mathcal{M}\}$, the function $g^A(z)$ is a quadratization of $f(x)$ and ($\texttt{ABCG}^{\mathcal{S}}$) is a quadratic reformulation of (P).*

Proof. It suffices to prove that $g^A(z)$ is a quadratization of $f(x)$. As in the proof of Theorem 4, consider any binary minimizer of g^A, say z^*, and let $x_i^* = z_i^*$ for $i \in [n]$. If z^* is consistent with x^*, then in view of Equation (22), $g^A(z^*) = g(z^*) = f(x^*)$ as required from a quadratization.

We are now going to show that we can actually assume without loss of generality that z^* is consistent. Let us rewrite $g^A(z)$ in the form

$$g^A(z) = \sum_{\substack{M\in\mathcal{M}:\\ |M|\le 2}} a_M \prod_{i\in M} z_i + \sum_{\substack{M\in\mathcal{M}:\\ |M|\ge 3}} \alpha_M(z) \tag{23}$$

where

$$\alpha_M(z) \triangleq a_M z_{l_M(M)} z_{r_M(M)} + \sum_{E\in\mathcal{E}_M} \beta_M(E) \left((2|E| - 1 - 2\sum_{j\in E} z_j) z_E + z_{l_M(E)} z_{r_M(E)} \right). \tag{24}$$

Consider now any variable z_E which appears in $g^A(z)$ with $|E| \ge 2$, $E \in \mathcal{E}_M$ for some $M \in \mathcal{M}$. (Note that E may actually appear in several schemes G_M, but this will not affect the reasoning hereunder.) We would like to identify the multiplier of z_E in $\alpha_M(z)$. Let us distinguish two cases.

Case 1: E appears in the decomposition of M, that is, $E \in \{l_M(M), r_M(M)\}$. Assume without loss of generality that $E = l_M(M)$. Then, in (24), z_E is multiplied by

$$c_{M,E}(z) = a_M z_{r_M(M)} + \beta_M(E)(2|E| - 1 - 2\sum_{j\in E} z_j),$$

where the second term results from the decomposition of E in G_M. Consider again two subcases.

1.1. If $\prod_{j\in E} z_j^* = 1$, or equivalently $\sum_{j\in E} z_j^* = |E|$, then by definition (20):

$$c_{M,E}(z^*) = a_M z^*_{r_M(M)} - \beta_M(E) = a_M z^*_{r_M(M)} - |a_M| \le 0.$$

Since z^* is a minimizer of g^A, we can assume that $z^*_E = 1$, meaning that z^*_E is consistent with the value of $\prod_{j\in E} z^*_j$. (As already observed, E may appear in the quadratization scheme of several monomials M, but as we will see below, the conclusion $c_{M,E}(z) \le 0$ will hold in all cases.)

1.2. If $\prod_{j\in E} z^*_j = 0$, then $2|E| - 1 - 2\sum_{j\in E} z^*_j \ge 1$. From this and from definition (20), it follows that

$$c_{M,E}(z^*) \ge a_M z^*_{r_M(M)} + \beta_M(E) = a_M z^*_{r_M(M)} + |a_M| \ge 0.$$

So, in this case, we can safely assume that $z^*_E = 0 = \prod_{j\in E} z^*_j$.

Let us now turn to the second case.

Case 2: E appears in the decomposition of one or several sets $S \in \mathcal{E}_M$, $S \ne M$. For notational simplicity and without loss of generality, let us assume that for all such sets S, there holds $E = l_M(S)$. Then, in (24), z_E is multiplied by

$$c_{M,E}(z) = \beta_M(E)(2|E| - 1 - 2\sum_{j\in E} z_j) + \sum_{\substack{S\in\mathcal{E}_M:\\ E\in\{l_M(S),r_M(S)\}}} \beta_M(S) z_{r_M(S)}.$$

2.1. If $\prod_{j\in E} z^*_j = 1$, then in view of definition (21),

$$c_{M,E}(z^*) = -\beta_M(E) + \sum_{\substack{S\in\mathcal{E}_M:\\ E\in\{l_M(S),r_M(S)\}}} \beta_M(S) z^*_{r_M(S)} \le -\beta_M(E) + \sum_{\substack{S\in\mathcal{E}_M:\\ E\in\{l_M(S),r_M(S)\}}} \beta_M(S) = 0.$$

Since z^* is a minimizer of g^A, we can assume that $z^*_E = 1 = \prod_{j\in E} z^*_j$.

2.2. If $\prod_{j\in E} z^*_j = 0$, then

$$c_{M,E}(z^*) \ge \beta_M(E) + \sum_{\substack{S\in\mathcal{E}_M:\\ E\in\{l_M(S),r_M(S)\}}} \beta_M(S) z^*_{r_M(S)} \ge 0.$$

So, in this case, we can safely assume that $z^*_E = 0 = \prod_{j\in E} z^*_j$.

In all cases, we conclude that z^* can be assumed to be consistent. Hence, as argued in the first part of the proof, $g^A(z^*) = f(x^*)$ and $g^A(z)$ is a quadratization of $f(x)$. □

As observed earlier, $\beta_M(E)$ is equal to $|a_M|\pi_{M,E}$, where $\pi_{M,E}$ is the number of directed paths from the root M to vertex E in the quadratization scheme G_M. As a consequence, we immediately obtain the following simplified form of Theorem 5 for disjoint schemes (compare with Remark 4):

Proposition 6. *For every disjoint quadratization scheme $\mathcal{S} = \{G_M = (V_M, l_M, r_M) : M \in \mathcal{M}\}$, the function*

$$g^A(z) = g(z) + \sum_{E\in\mathcal{E}} \Big(\sum_{\substack{M\in\mathcal{M}:\\ |M|\ge 3 \text{ and } E\in\mathcal{E}_M}} |a_M| \Big)\, q^A_M(E).$$

is a quadratization of $f(x)$.

Proof. When $\mathcal{S}$ is disjoint, every vertex $E \neq M$ has indegree 1 in G_M (see Proposition 1). Hence $\pi_{M,E} = 1$ and $\beta_M(E) = |a_M|$. □

Remark 5. Proposition 6 shows that in the disjoint case, the penalty coefficients are the same in Rosenberg's procedure and in ABCG. However this is not the case for the penalty *functions* $q_M^R(E)$ and $q_M^A(E)$, which can be different as illustrated by the next example.

Example 6. Consider again the function $f(x) = -6.5x_1x_2x_3x_4x_5x_6 - 5.6x_1x_2x_3$ and the quadratization scheme $\mathcal{S}$ as in Example 4 and Example 5. We construct the penalty functions $q_M^A(E)$ for every $M \in \mathcal{M}$ with $|M| \geq 3$ and every $E \in \mathcal{E}_M$.

Since the quadratization scheme $\mathcal{S}$ is disjoint, we have that for $M_1 = \{1,2,3,4,5,6\}$, $\beta_{M_1}(E) = |a_{M_1}| = 6.5$ for every $E \in \mathcal{E}_{M_1}$ and for $M_2 = \{1,2,3\}$, $\beta_{M_2}(E) = |a_{M_2}| = 5.6$ for every $E \in \mathcal{E}_{M_2}$.

For $M_1 = \{1,2,3,4,5,6\}$, the sets $E \in \mathcal{E}_{M_1}$ and their corresponding penalty functions are

- When $E = \{3,4,5,6\}$, the penalty $q_{M_1}^A(E)$ is
  $6.5\left(7z_{\{3,4,5,6\}} - 2z_3z_{\{3,4,5,6\}} - 2z_4z_{\{3,4,5,6\}} - 2z_5z_{\{3,4,5,6\}} - 2z_6z_{\{3,4,5,6\}} + z_{\{3,4\}}z_{\{5,6\}}\right)$.
- When $E = \{1,2\}$, $q_{M_1}^A(E) = 6.5\left(3z_{\{1,2\}} - 2z_1z_{\{1,2\}} - 2z_2z_{\{1,2\}} + z_1z_2\right)$.
- When $E = \{3,4\}$, $q_{M_1}^A(E) = 6.5\left(3z_{\{3,4\}} - 2z_3z_{\{3,4\}} - 2z_4z_{\{3,4\}} + z_3z_4\right)$.
- When $E = \{5,6\}$, $q_{M_1}^A(E) = 6.5\left(3z_{\{5,6\}} - 2z_5z_{\{5,6\}} - 2z_6z_{\{5,6\}} + z_5z_6\right)$.

For $M_2 = \{1,2,3\}$, there is only one set $E \in \mathcal{E}_{M_2}$:

- $E = \{1,2\}$, and $q_{M_2}^A(E) = 5.6\left(3z_{\{1,2\}} - 2z_1z_{\{1,2\}} - 2z_2z_{\{1,2\}} + z_1z_2\right)$.

We finally obtain the quadratic reformulation:

$$(\texttt{ABCG}^{\mathcal{S}}) \begin{cases} \min\limits_{z\in\{0,1\}^{10}} & -6.5z_{\{1,2\}}z_{\{3,4,5,6\}} - 5.6z_{\{1,2\}}z_3 + 12.1(3z_{\{1,2\}} - 2z_1z_{\{1,2\}} - 2z_2z_{\{1,2\}} + z_1z_2) \\ & +6.5(3z_{\{3,4\}} - 2z_3z_{\{3,4\}} - 2z_4z_{\{3,4\}} + z_3z_4) + 6.5(3z_{\{5,6\}} - 2z_5z_{\{5,6\}} - 2z_6z_{\{5,6\}} + z_5z_6) \\ & +6.5(7z_{\{3,4,5,6\}} - 2z_3z_{\{3,4,5,6\}} - 2z_4z_{\{3,4,5,6\}} - 2z_5z_{\{3,4,5,6\}} - 2z_6z_{\{3,4,5,6\}} + z_{\{3,4\}}z_{\{5,6\}}). \end{cases}$$

Observe the difference between functions $g^R(z)$ and $g^A(z)$ for this example. In Rosenberg's procedure, the penalties for a variable z_E involve the direct successors of monomial E, while in ABCG they involve the leaves of the quadratization scheme (i.e., the initial variables). Compare, for example, the penalties associated with $z_{\{3,4,5,6\}}$ in Example 5 and in the current Example 6.

5 Phase III: Convexifying the quadratic reformulation

In the previous section, from a given quadratization scheme $\mathcal{S}$, we have built three quadratic formulations which are equivalent to (P), all of them sharing the same set of N binary variables z: a linearly constrained binary quadratic problem ($\texttt{FOR}^{\mathcal{S}}$), and two unconstrained binary quadratic programs ($\texttt{ROS}^{\mathcal{S}}$) and ($\texttt{ABCG}^{\mathcal{S}}$). In these three formulations, the quadratic objective function is non-convex. So, to solve these optimization problems, a convexification step would be typically required. We propose in this section a linearization and a convexification method for each of the three problems, and we compare the optimal values of their continuous relaxations.

5.1 Linearization of the objective function

In this section, we propose to linearize the three quadratic reformulations. For this purpose, the basic idea is to apply once again the standard linearization described in Section 4 to the nonlinear terms of the quadratic functions $g(z)$, $g^R(z)$, or $g^A(z)$ (as mentioned in Section 2.2, this strategy is also implemented in commercial solvers like CPLEX [44] or Gurobi [40]). We thus obtain three mixed integer linear reformulations of (P). We then compare their continuous relaxation bounds.

Linearization of ($\texttt{FOR}^{\mathcal{S}}$). To linearize problem ($\texttt{FOR}^{\mathcal{S}}$), we first introduce a new variable $Z_{\{i_1\},\{i_2\}}$ for each monomial $M = \{i_1, i_2\} \in \mathcal{M}$ of degree 2. Then, we introduce a variable $Z_{l_M(M),r_M(M)}$ to represent the product $z_{l_M(M)} z_{r_M(M)}$ in ($\texttt{FOR}^{\mathcal{S}}$) for each monomial $M \in \mathcal{M}$ of degree at least three. This variable also represents the monomial M itself. We obtain the following mixed integer linear programming reformulation of ($\texttt{FOR}^{\mathcal{S}}$) and therefore of (P):

$$
(\texttt{FOR-L}^{\mathcal{S}}) \begin{cases}
\min \quad g_L(z,Z) \triangleq \displaystyle\sum_{i=1}^{n} a_i z_i + \sum_{\substack{M \in \mathcal{M} \\ M=\{i_1,i_2\}}} a_M Z_{\{i_1\},\{i_2\}} + \sum_{\substack{M \in \mathcal{M} \\ |M| \geq 3}} a_M Z_{l_M(M),r_M(M)} & (25) \\
\text{s.t. } (10)-(11) & \\
\quad Z_{\{i_1\},\{i_2\}} \leq z_{i_1},\ Z_{\{i_1\},\{i_2\}} \leq z_{i_2},\ Z_{\{i_1\},\{i_2\}} \geq z_{i_1} + z_{i_2} - 1 \quad \forall M \in \mathcal{M} : M = \{i_1,i_2\} & (26) \\
\quad Z_{l_M(M),r_M(M)} \leq z_{l_M(M)},\ Z_{l_M(M),r_M(M)} \leq z_{r_M(M)} \quad \forall M \in \mathcal{M} : |M| \geq 3 & (27) \\
\quad Z_{l_M(M),r_M(M)} \geq z_{l_M(M)} + z_{r_M(M)} - 1 \qquad \forall M \in \mathcal{M} : |M| \geq 3 & (28) \\
\quad z \in \{0,1\}^N, \quad 0 \leq Z \leq 1,
\end{cases}
$$

where the shorthand notation $0 \leq Z \leq 1$ means that each component of Z is between 0 and 1.

The number of auxiliary variables in ($\texttt{FOR-L}^{\mathcal{S}}$) is equal to $|\mathcal{M}| + |\mathcal{E}|$. In contrast with this, the standard linearization (SL) introduced in Section 2 contains $|\mathcal{M}|$ auxiliary variables, and does not dependent on any quadratization scheme.

A result initially established by Lazare [52] (Corollary 4.1.1), and more recently by Khajavirad [47] (Proposition 2) in a closely related framework, states that the bound obtained by linear relaxation of the standard linearization (SL) is the same as that of ($\texttt{FOR-L}^{\mathcal{S}}$) when the quadratization scheme is *non-overlapping* in the sense that $\mathcal{E}_{M_1} \cap \mathcal{E}_{M_2} = \emptyset$ for all $(M_1, M_2) \in \mathcal{M}$ (this property is called *independence* in [52]). We record this result in the following proposition.

Proposition 7 ([52, 47])**.** *For every non-overlapping quadratization scheme $\mathcal{S}$, the LP bound provided by the continuous relaxation of ($\texttt{FOR-L}^{\mathcal{S}}$) is equal to the LP bound provided by the continuous relaxation of the standard linearization (SL).*

We now present a new, different result that holds when $\mathcal{S}$ is a *disjoint* quadratization scheme (in the sense of Definition 4).

Proposition 8. *For every disjoint quadratization scheme $\mathcal{S}$, the LP bound provided by the continuous relaxation of ($\texttt{FOR-L}^{\mathcal{S}}$) is at least as good as the LP bound provided by the continuous relaxation of the standard linearization (SL).*

Proof. Let (z^*, Z^*) be an optimal solution of the relaxation of ($\texttt{FOR-L}^{\mathcal{S}}$). We build the following point $(\tilde{x}, \tilde{y})$: $\tilde{x}_i = z_i^*$ for $i = 1, ..., n$, $\tilde{y}_{\{i_1,i_2\}} = Z^*_{\{i_1\},\{i_2\}}$ for every degree-2 monomial $\{i_1, i_2\} \in \mathcal{M}$, and $\tilde{y}_M = Z^*_{l_M(M),r_M(M)}$ for every monomial $M \in \mathcal{M}$ with $|M| \geq 3$.

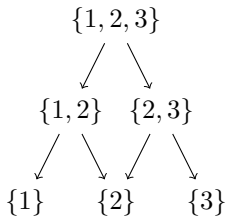


Figure 4: A non-disjoint quadratization scheme for a cubic monomial

Let us check that $(\tilde{x}, \tilde{y})$ is a feasible solution of (SL). For any degree-2 monomial $M = \{i_1, i_2\}$, Constraints (3) and (4) follow from (26). Moreover, Constraints (10)-(11) iteratively enforce $z^*_E \leq z^*_i$ and $z^*_E \geq \sum_{i \in E} z^*_i - (|E| - 1)$ for all $E \in \mathcal{E}$ and all $i \in E$. Therefore, for any monomial M with $|M| \geq 3$, Constraints (3) follow from (10)-(11),(27), and $M = l_M(M) \cup r_M(M)$. To verify Constraints (4), note that it follows from (28) that:

$$\tilde{y}_M \geq z^*_{l_M(M)} + z^*_{r_M(M)} - 1 \geq \sum_{i \in l_M(M)} z^*_i - (|l_M(M)| - 1) + \sum_{i \in r_M(M)} z^*_i - (|r_M(M)| - 1) - 1$$

and the last expression is equal to $\sum_{i \in M} \tilde{x}_i - (|M| - 1)$ as $l_M(M), r_M(M)$ form a partition of M.

Finally, solution $(\tilde{x}, \tilde{y})$ has the same objective value in (SL) as (z^*, Z^*) in (FOR-L$^{\mathcal{S}}$). We conclude that the optimal value of (FOR-L$^{\mathcal{S}}$) is greater than or equal to that of (SL). □

The inequality in the previous proposition can be strict as illustrated by the numerical results in Section 6.

In contrast with Proposition 8, the LP bound provided by the (FOR-L$^{\mathcal{S}}$) linearization associated with *non-disjoint* quadratization schemes may be worse than the standard linearization bound, and hence, than the bound provided by any disjoint scheme.

Example 7. Consider the objective function $f = x_1x_2x_3 - x_1 - x_2 - x_3$. Since f contains a single nonlinear monomial, the continuous relaxation of (SL) yields a (tight) lower bound equal to -2 and by virtue of Proposition 8, (FOR-L$^{\mathcal{S}}$) yields the same bound for any disjoint quadratization scheme $\mathcal{S}$. However, when $\mathcal{S}$ is the non-disjoint quadratization scheme represented in Figure 4, (FOR-L$^{\mathcal{S}}$) yields a weaker bound equal to -2.5. Indeed, in this case, (FOR-L$^{\mathcal{S}}$) involves two additional variables $z_{\{12\}}$ and $z_{\{23\}}$ in comparison with the standard linearization. It is easy to check that by projecting out these two variables of (FOR-L$^{\mathcal{S}}$), we obtain a polyhedron which strictly contains the polyhedron associated with (SL).

Linearization of (ROS$^{\mathcal{S}}$). We next introduce the linearization of problem (ROS$^{\mathcal{S}}$). Observe that the penalty functions of the Rosenberg procedure are composed of quadratic terms that do not appear in (FOR$^{\mathcal{S}}$). Therefore, the linearization of problems (ROS$^{\mathcal{S}}$) requires more auxiliary variables than (FOR-L$^{\mathcal{S}}$): in addition to the Z variables already in (FOR-L$^{\mathcal{S}}$), the products $z_E z_{l_M(E)}$, $z_E z_{r_M(E)}$ and $z_{l_M(E)} z_{r_M(E)}$ are respectively linearized by new auxiliary variables that we denote by $Z_{E,l_M(E)}$, $Z_{E,r_M(E)}$ and $Z_{l_M(E),r_M(E)}$, $\forall E \in \mathcal{E}, \forall M \in \mathcal{M} : E \in \mathcal{E}_M$. (Nevertheless, in order to simplify the presentation, we use the same letter Z to denote the vector of auxiliary variables in (ROS-L$^{\mathcal{S}}$) and in (FOR-L$^{\mathcal{S}}$).) We build the following mixed integer linear problem equivalent to (ROS$^{\mathcal{S}}$) and to (P):

$$
(\texttt{ROS-L}^{\mathcal{S}})\begin{cases}
\min \quad g_{LR}(z,Z) \triangleq g_L(z,Z) + \displaystyle\sum_{\substack{M\in\mathcal{M}:\\ |M|\geq 3}} \sum_{E\in\mathcal{E}_M} |a_M|\Big(3z_E - 2Z_{E,l_M(E)} - 2Z_{E,r_M(E)} + Z_{l_M(E),r_M(E)}\Big) & \\
\text{s.t. } (26)-(28) & \\
\quad Z_{E,l_M(E)} \leq z_{l_M(E)},\ Z_{E,l_M(E)} \leq z_E \quad \forall E\in\mathcal{E}, \forall M\in\mathcal{M}: E\in\mathcal{E}_M. & (29)\\
\quad Z_{E,r_M(E)} \leq z_{r_M(E)},\ Z_{E,r_M(E)} \leq z_E \quad \forall E\in\mathcal{E}, \forall M\in\mathcal{M}: E\in\mathcal{E}_M & (30)\\
\quad Z_{l_M(E),r_M(E)} \geq z_{l_M(E)} + z_{r_M(E)} - 1 \quad \forall E\in\mathcal{E}, \forall M\in\mathcal{M}: E\in\mathcal{E}_M & (31)\\
\quad z_E \leq z_{l_M(E)},\ z_E \leq z_{r_M(E)} \qquad \forall E\in\mathcal{E}, \forall M\in\mathcal{M}: E\in\mathcal{E}_M & (32)\\
\quad z\in\{0,1\}^N \quad 0\leq Z\leq 1. &
\end{cases}
$$

The function $g_L(z,Z)$ in the objective function is given by Equation (25). Constraints (29) are enough to enforce the equality $Z_{E,l_M(E)} = z_E z_{l_M(E)}$ in any optimal solution because variable $Z_{E,l_M(E)}$ appears in the objective function with a negative coefficient, and nowhere else. The same holds for Constraints (30) which enforce the equality $Z_{E,r_M(E)} = z_E z_{r_M(E)}$. Constraints (31) enforce the equality $Z_{l_M(E),r_M(E)} = z_{l_M(E)} z_{r_M(E)}$ in any optimal solution as this variable has a positive coefficient in the objective function whenever $|E| \geq 3$. This property may not hold, however, when E is a degree-2 monomial and $Z_{l_M(E),r_M(E)}$ also appears with a negative coefficient in $g_L(z,Z)$. But in this case, Constraints (31) are redundant with (26), which impose the required equalities. Constraints (32) are valid inequalities (identical to (10)) that express that in any optimal solution z of $(\texttt{ROS}^{\mathcal{S}})$ and therefore of $(\texttt{ROS-L}^{\mathcal{S}})$, z_E must be equal to the product $z_{l_M(E)} z_{r_M(E)}$.

We next show that the addition of Constraints (32) allows us to significantly reduce the size of $(\texttt{ROS-L}^{\mathcal{S}})$ (the letter C in $(\texttt{ROS-C}^{\mathcal{S}})$ refers to a *compact* reformulation):

Proposition 9. *Let $(\texttt{ROS-C}^{\mathcal{S}})$ be the following problem:*

$$
(\texttt{ROS-C}^{\mathcal{S}})\begin{cases}
\min \quad g'_{LR}(z,Z) \triangleq g_L(z,Z) + \displaystyle\sum_{\substack{M\in\mathcal{M}:\\ |M|\geq 3}} \sum_{E\in\mathcal{E}_M} |a_M|(Z_{l_M(E),r_M(E)} - z_E) & \\
s.t.\ (26)-(28),\ (31),\ (32) & \\
\quad z\in\{0,1\}^N \quad 0\leq Z\leq 1. &
\end{cases}
$$

Problems $(\texttt{ROS-L}^{\mathcal{S}})$ and $(\texttt{ROS-C}^{\mathcal{S}})$ have the same optimal value. The same property holds for their LP relaxations.

Proof. In an optimal solution of $(\texttt{ROS-L}^{\mathcal{S}})$, $Z_{E,l_M(E)}$ is as large as possible, therefore it is equal to $\min(z_E, z_{l_M(E)})$ which is in turn equal to z_E by Constraints (32). The same reasoning applies for variable $Z_{E,r_M(E)}$ which is equal to z_E in any optimal solution. One can therefore fix both $Z_{E,l_M(E)}$ and $Z_{E,r_M(E)}$ to z_E in $(\texttt{ROS-L}^{\mathcal{S}})$ and straightforwardly get $(\texttt{ROS-C}^{\mathcal{S}})$. Integrality plays no role in this argument, so the same conclusion holds for the linear relaxations of $(\texttt{ROS-L}^{\mathcal{S}})$ and $(\texttt{ROS-C}^{\mathcal{S}})$. □

The compact problem $(\texttt{ROS-C}^{\mathcal{S}})$ has the same variables as $(\texttt{FOR-L}^{\mathcal{S}})$ and additional variables $Z_{l_M(E),r_M(E)}$. We now compare the LP bounds of linearizations $(\texttt{FOR-L}^{\mathcal{S}})$ and $(\texttt{ROS-C}^{\mathcal{S}})$.

Proposition 10. *For every quadratization scheme $\mathcal{S}$, the LP bound provided by the continuous relaxation of $(\texttt{FOR-L}^{\mathcal{S}})$ is at least as good as the LP bound provided by the continuous relaxation of $(\texttt{ROS-C}^{\mathcal{S}})$.*

Proof. Let (z^*, Z^*) be an optimal solution of the relaxation of $(\texttt{FOR-L}^{\mathcal{S}})$. Let us build a solution $(\tilde{z}, \tilde{Z})$ of the relaxation of $(\texttt{ROS-C}^{\mathcal{S}})$. We set $\tilde{z} = z^*$ and $\tilde{Z} = Z^*$ for the components of Z that are common to both problems. We also set $\tilde{Z}_{l_M(E),r_M(E)} = z^*_E$. One can easily check that $(\tilde{z}, \tilde{Z})$ is a feasible solution of the linear relaxation of $(\texttt{ROS-C}^{\mathcal{S}})$ with the same value as (z^*, Z^*) in the linear relaxation of $(\texttt{FOR-L}^{\mathcal{S}})$. □

Linearization of $(\texttt{ABCG}^{\mathcal{S}})$. Let us now turn to the ABCG procedure. We need the Z variables present in $(\texttt{FOR-L}^{\mathcal{S}})$, variables $Z_{l_M(E),r_M(E)}$ already used in $(\texttt{ROS-L}^{\mathcal{S}})$, and we add a new variable denoted by $Z_{E,\{j\}}$ for each product $z_j z_E$, for all $M \in \mathcal{M}$, $E \in \mathcal{E}_M$ and $j \in E$. We build the following mixed integer linear problem equivalent to $(\texttt{ABCG}^{\mathcal{S}})$ and to (P):

$$(\texttt{ABCG-L}^{\mathcal{S}}) \begin{cases} \min \quad g_{LA}(z,Z) \triangleq g_L(z,Z) + \displaystyle\sum_{\substack{M\in\mathcal{M}:\\ |M|\geq 3}} \sum_{E\in\mathcal{E}_M} \beta_M(E)\Big((2|E|-1)z_E - 2\sum_{j\in E} Z_{E,\{j\}} + Z_{l_M(E),r_M(E)}\Big) \\ \text{s.t. } (26)-(28),\ (31),\ (32) \\ \quad Z_{E,\{j\}} \leq z_j,\ Z_{E,\{j\}} \leq z_E \qquad \forall E \in \mathcal{E}, \forall j \in E \\ \quad z \in \{0,1\}^N \quad 0 \leq Z \leq 1. \end{cases} \tag{33}$$

Here again, variables $Z_{E,\{j\}}$ are weighted by negative coefficients in $g_{LA}(z,Z)$, so Constraints (33) are enough to linearize $z_j z_E$ by $Z_{E,\{j\}}$. For the same reasons as in $(\texttt{ROS-L}^{\mathcal{S}})$, Constraints (32) are valid inequalities in $(\texttt{ABCG-L}^{\mathcal{S}})$. We now show that variables $Z_{E,\{j\}}$ can be fixed and Constraints (33) can be dropped to obtain a compact reformulation of $(\texttt{ABCG-L}^{\mathcal{S}})$.

Proposition 11. *Let* $(\textit{ABCG-C}^{\mathcal{S}})$ *be the following problem:*

$$(\textit{ABCG-C}^{\mathcal{S}}) \begin{cases} \min \quad g'_{LA}(z,Z) \triangleq g_L(z,Z) + \displaystyle\sum_{\substack{M\in\mathcal{M}:\\ |M|\geq 3}} \sum_{E\in\mathcal{E}_M} \beta_M(E)(Z_{l_M(E),r_M(E)} - z_E) \\ \textit{s.t. } (26)-(28),\ (31),\ (32) \\ \quad z \in \{0,1\}^N \quad 0 \leq Z \leq 1 \end{cases}$$

Problems $(\textit{ABCG-L}^{\mathcal{S}})$ *and* $(\textit{ABCG-C}^{\mathcal{S}})$ *have the same optimal value. The same property holds for their LP relaxations.*

Proof. The proof is similar to the proof of Proposition 9. It is enough to observe that, in an optimal solution of $(\texttt{ABCG-L}^{\mathcal{S}})$, $Z_{E,\{j\}}$ is as large as possible and is therefore equal to $\min(z_E, z_j)$. Constraints (32) can be applied iteratively starting from z_E until reaching the leaf $\{j\}$. Hence, $\min(z_E, z_j) = z_E$ and $Z_{E,\{j\}} = z_E$. It follows that $(2|E|-1)z_E - 2\sum_{j\in E} Z_{E,\{j\}} = -z_E$. □

The following proposition states that the LP bounds of linearizations $(\texttt{FOR-L}^{\mathcal{S}})$ and $(\texttt{ABCG-C}^{\mathcal{S}})$ compare similarly to above.

Proposition 12. *For every quadratization scheme* $\mathcal{S}$*, the LP bound provided by the continuous relaxation of* $(\textit{FOR-L}^{\mathcal{S}})$ *is at least as good as the LP bound provided by the continuous relaxation of* $(\textit{ABCG-C}^{\mathcal{S}})$.

Proof. Similar to the proof of Proposition 10. □

Finally, we can observe that the compact mixed integer linear programs (ROS-C$^{\mathcal{S}}$) and (ABCG-C$^{\mathcal{S}}$) differ only by the penalty coefficient in their objective functions. As proved in Proposition 6, these coefficients are identical when the quadratization scheme is disjoint. This observation is recorded in the following statement.

Corollary 13. *When $\beta_M(E) = |a_M|$ for all $M \in \mathcal{M}$, $E \in \mathcal{E}_M$, problems (ROS-C$^{\mathcal{S}}$) and (ABCG-C$^{\mathcal{S}}$) are identical. This happens in particular when the quadratization scheme $\mathcal{S}$ is disjoint.*

As a conclusion of this section, we can assert that the linearized forms of (ROS$^{\mathcal{S}}$) and (ABCG$^{\mathcal{S}}$) are weaker than (FOR$^{\mathcal{S}}$), in terms of quality of the bounds they deliver. In fact, an important difference between the quadratic reformulation (FOR$^{\mathcal{S}}$) and the penalized quadratic programs (ROS$^{\mathcal{S}}$) and (ABCG$^{\mathcal{S}}$) is that in (FOR$^{\mathcal{S}}$), due to Constraints (10)-(11), the equivalence with (P) holds at each binary point, while for problems (ROS$^{\mathcal{S}}$) and (ABCG$^{\mathcal{S}}$), it only holds at the optimum. For the latter two formulations, the link between the variables z_E and the products $z_{l_M(E)}z_{r_M(E)}$ is somehow lost. (If we reestablish this relation by imposing the equality $z_E = z_{l_M(E)}z_{r_M(E)}$ in their linearized versions, then the penalties vanish in (ROS-C$^{\mathcal{S}}$) and (ABCG-C$^{\mathcal{S}}$), which both simply reduce to (FOR-L$^{\mathcal{S}}$).)

5.2 Quadratic convex reformulations

In this section, we apply convexification techniques to the quadratic formulations (FOR$^{\mathcal{S}}$), (ROS$^{\mathcal{S}}$) and (ABCG$^{\mathcal{S}}$) described in Section 4. The general idea is to replace the functions $g(z)$, $g^R(z)$ and $g^A(z)$ by equivalent quadratic convex functions. This provides an alternative approach to the linearization step described in Section 5.1.

5.2.1 Convexification for quadratic problems: the smallest eigenvalue method

As in Section 2.3, consider a general quadratic problem of the form (QP) with an objective function $G(z)$ (this could be, for example, problem (FOR$^{\mathcal{S}}$) or any other quadratic reformulation). Since $z^2 = z$ when z is binary, the function $G(z)$ can be viewed as a pure quadratic form. From now on, we write its Hessian matrix representation as

$$G(z) = \frac{1}{2} z^T Q z$$

where Q is a symmetric matrix of order N.

One common method to get a quadratic convex reformulation of (QP) is then to compute λ_{min}, that is, the smallest eigenvalue of Q, and to replace $G(z)$ by

$$G_{\lambda_{min}}(z) \triangleq \frac{1}{2}\left(z^T Q z - \lambda_{min} \sum_{E \in \mathcal{E}} (z_E^2 - z_E) \right).$$

The equality $G(z) = G_{\lambda_{min}}(z)$ holds when each z_E is a binary variable. The Hessian matrix of this equivalent form of the objective function is $Q - \lambda_{min} I_N$, where I_N is the identity matrix of order N. Since $Q - \lambda_{min} I_N$ is positive semidefinite, the function $G_{\lambda_{min}}(z)$ is convex. Thus, the continuous relaxation of the reformulated problem is a convex quadratic problem which can be solved by branch-and-bound. This basic convexification method is implemented, for example, in Gurobi [40] and in CPLEX [44]. Different improvements of this approach are presented hereunder.

5.2.2 Convexification for quadratic problems: the QCR method

QCR is a quadratic convex reformulation method proposed by Billionnet and Elloumi [8]. It is an improvement of the smallest eigenvalue method based on two key ideas. First, it considers a parameterized non-uniform diagonal convexification. Second, it uses semidefinite programming to determine the parameters that maximize the optimal value of the continuous relaxation of the resulting problem.

QCR method applied to (FOR$^{\mathcal{S}}$). We can rewrite (FOR$^{\mathcal{S}}$) as the following equivalent quadratic problem, parameterized by any vector $\alpha \in \mathbb{R}^N$:

$$(\texttt{FOR-Q}^{\mathcal{S}}_{\alpha}) \begin{cases} \min \quad g_\alpha(z) \triangleq g(z) + \displaystyle\sum_{E \in \mathcal{E} \cup [n]} \alpha_E \left(z_E^2 - z_E\right) \\ \text{s.t.} \quad (10) - (11) \\ \quad z \in \{0,1\}^N. \end{cases}$$

It is clear that $g_\alpha(z) = g(z)$ for any $z \in \{0,1\}^N$. Since there exist many choices of the parameter vector $\alpha \in \mathbb{R}^N$ that make $g_\alpha(z)$ convex, we next try to determine such a vector that maximizes the continuous relaxation value of (FOR-Q$^{\mathcal{S}}_{\alpha}$). For this purpose, we compute the dual optimal solution of the following semi-definite relaxation of (FOR$^{\mathcal{S}}$):

$$(\texttt{SDP}) \begin{cases} \min \quad g_L(z, Z) & \\ \text{s.t.} \quad (10) - (11) & \\ \quad Z_{E,E} - z_E = 0 \qquad E \in \mathcal{E} \cup [n] & (34) \\ \quad \begin{pmatrix} 1 & z^T \\ z & Z \end{pmatrix} \succeq 0 & (35) \\ \quad z \in \mathbb{R}^N,\ Z \in \mathbb{S}_N & (36) \end{cases}$$

where the Z variables are introduced in the same spirit as in the linear reformulations of Section 5.1 to replace the product of two z variables, $g_L(z, Z)$ is defined in Equation (25), and $\mathbb{S}_N$ is the set of symmetric matrices of order N. Constraints (35) are the Shor relaxation of the equality $Z = zz^T$. It is proven in [8] that an optimal vector α^* can be deduced from the optimal dual variables associated with Constraints (34). The continuous relaxation of (FOR-Q$^{\mathcal{S}}_{\alpha^*}$) ((FOR-Q*$^{\mathcal{S}}$) for short) is a quadratic convex problem leading to the highest possible dual bound in this framework. It is also proven in [8] that it has the same optimal value as problem (SDP).

QCR method applied to (ROS$^{\mathcal{S}}$). We can similarly apply the QCR method to convexify problem (ROS$^{\mathcal{S}}$). The main difference with the previous case (beside the fact that the objective functions are different) is that (ROS$^{\mathcal{S}}$) is unconstrained. Therefore, Constraints (10)-(11) are not present, which means that (ROS$^{\mathcal{S}}$) is reformulated as the following problem:

$$(\texttt{ROS-Q*}^{\mathcal{S}}) \begin{cases} \min \quad g^R(z) + \displaystyle\sum_{E \in \mathcal{E} \cup [n]} \mu_E^* (z_E^2 - z_E) \\ \text{s.t.} \quad z \in \{0,1\}^N \end{cases}$$

where the optimal parameters μ_E^* are the optimal dual variables of Constraints (34) in the semidefinite program:

$$(\texttt{SDP}_\texttt{R})\begin{cases}\min & g_{LR}(z,Z)\\ \text{s.t.} & (34)-(36).\end{cases}$$

QCR method applied to ($\texttt{ABCG}^{\mathcal{S}}$). Finally, the application of method QCR to ($\texttt{ABCG}^{\mathcal{S}}$) consists in solving the following quadratic convex problem:

$$(\texttt{ABCG-Q*}^{\mathcal{S}})\begin{cases}\min & g^A(z)+\sum\limits_{E\in\mathcal{E}\cup[n]}\nu_E^*(z_E^2-z_E)\\ \text{s.t.} & z\in\{0,1\}^N\end{cases}$$

where the optimal parameters ν_E^* are the optimal dual variables of Constraints (34) in the semidefinite program:

$$(\texttt{SDP}_\texttt{A})\begin{cases}\min & g_{LA}(z,Z)\\ \text{s.t.} & (34)-(36).\end{cases}$$

5.2.3 Convexification for polynomial problems: the PQCR method

We now describe a more elaborate convex quadratic reformulation approach called PQCR [29], which is more specifically adapted to binary polynomial optimization problems and takes deeper advantage of the specific quadratization scheme. This method has an initial step that can be viewed as yielding a quadratization scheme $\mathcal{S}$ together with the associated reformulation by the quadratic problem with linear constraints ($\texttt{FOR}^{\mathcal{S}}$). Next, it applies a convexification step to ($\texttt{FOR}^{\mathcal{S}}$), based on a stronger SDP relaxation than QCR. We next detail this convexification step, following the presentation given in [29].

We start by introducing three sets of quadratic forms that, in addition to the forms $z_E^2-z_E$, vanish on the feasible domain of ($\texttt{FOR}^{\mathcal{S}}$) (i.e., when z is binary and Constraints (10)-(11) are satisfied):

$$\begin{cases} z_E - z_{l_M(E)}z_{r_M(E)} = 0 & \forall E\in\mathcal{E},\forall M\in\mathcal{M}: E\in\mathcal{E}_M \quad (37)\\ z_{E_1} - z_{E_1}z_{E_2} = 0 & \forall E_1,E_2\in\mathcal{E}\cup[n]:\ E_2\subset E_1 \quad (38)\\ z_{E_1}z_{E_2} - z_{E_3}z_{E_4} = 0 & \forall E_1,..,E_4\in\mathcal{E}\cup[n]:\ E_1\cup E_2 = E_3\cup E_4. \quad (39)\end{cases}$$

With each of these equations, we associate a real scalar multiplier, say δ_E for Constraints (37), β_{E_1,E_2} for Constraints (38), and $\lambda_{E_1,E_2,E_3,E_4}$ for Constraints (39). We next add to $g_\alpha(z)$ the quadratic forms in (37)–(39) multiplied by their associated coefficient. This yields the following parameterized function:

$$g_{\alpha,\delta,\beta,\lambda}(z) \triangleq g_\alpha(z) + \sum_{\substack{E\in\mathcal{E},M\in\mathcal{M}:\\ E\in\mathcal{E}_M}}\delta_E^M(z_E - z_{l_M(E)}z_{r_M(E)}) + \sum_{\substack{E_1,E_2\in\mathcal{E}\cup[n]:\\ E_2\subset E_1}}\beta_{E_1,E_2}(z_{E_1}-z_{E_1}z_{E_2}) + \sum_{\substack{E_1,..,E_4\in\mathcal{E}\cup[n]:\\ E_1\cup E_2=E_3\cup E_4}}\lambda_{E_1,E_2,E_3,E_4}(z_{E_1}z_{E_2}-z_{E_3}z_{E_4}).$$

Function $g_{\alpha,\delta,\beta,\lambda}(z)$ has the same value as $g(z)$ for any binary z satisfying the standard inequalities (10)-(11). Moreover, there exist parameters α, β, δ and λ such that $g_{\alpha,\delta,\beta,\lambda}$ is a convex function (take, for instance, $\alpha = -\lambda_{min}$ and $\beta = \delta = \lambda = 0$).

Replacing g by this new objective function, we obtain the following family of convex mixed-integer quadratic equivalent formulations of ($\texttt{FOR}^{\mathcal{S}}$):

$$(\texttt{PQCR}^{\mathcal{S}}_{\alpha,\delta,\beta,\lambda}) \begin{cases} \min g_{\alpha,\delta,\beta,\lambda}(z) \\ \text{s.t.} \quad (10)-(11) \\ \qquad z \in \{0,1\}^N. \end{cases}$$

It is proved in [29, 52] that optimal parameters $(\alpha^*, \delta^*, \beta^*, \lambda^*)$ that make $g_{\alpha,\delta,\beta,\lambda}$ convex and maximize the continuous relaxation bound of ($\texttt{PQCR}^{\mathcal{S}}_{\alpha,\delta,\beta,\lambda}$) can be deduced from the dual solution of the following semidefinite program ($\texttt{SDP}'$):

$$(\texttt{SDP}') \begin{cases} \min \quad g_L(z,Z) & & \\ \text{s.t.} \quad (34)-(36) & & \\ \quad z_E - Z_{l_M(E),r_M(E)} = 0 & E \in \mathcal{E}, M \in \mathcal{M} : E \in \mathcal{E}_M & (40) \\ \quad z_{E_1} - Z_{E_1,E_2} = 0 & E_1, E_2 \in \mathcal{E} \cup [n] : \ E_2 \subset E_1 & (41) \\ \quad Z_{E_1,E_2} - Z_{E_3,E_4} = 0 & E_1, .., E_4 \in \mathcal{E} \cup [n] : \ E_1 \cup E_2 = E_3 \cup E_4 \ . & (42) \end{cases}$$

It is interesting to note that, as proved in [52], Constraints (10)-(11) become redundant when adding Constraints (40)-(42). However, ($\texttt{SDP}'$) has many more constraints than ($\texttt{SDP}$). The optimal parameters $(\alpha^*, \delta^*, \beta^*, \lambda^*)$ can be obtained as optimal values of the dual variables respectively associated with Constraints (34), (40), (41), and (42). Here again the continuous relaxation bound of ($\texttt{PQCR}^{\mathcal{S}}_{\alpha^*,\delta^*,\beta^*,\lambda^*}$) (($\texttt{PQCR*}^{\mathcal{S}}$) for short) is as tight as the semidefinite programming bound.

Note also that non-disjoint quadratization schemes may prove worthwhile when applying PQCR, in the sense that they may provide better bounds than disjoint schemes. (Compare with Example 7 where a non-disjoint scheme is worse than any disjoint one in the linearization framework.)

Example 8. Let $f(x) = 2x_1 + 3x_2x_3 - 2x_2x_3x_4 - 3x_1x_2x_3x_4$, and let $M_3 = \{2,3,4\}$, $M_4 = \{1,2,3,4\}$. We define a disjoint quadratization scheme $\mathcal{S}^1$ and a non-disjoint one $\mathcal{S}^2$:

- in $\mathcal{S}^1$, $l^1_{M_3}(M_3) = \{2,3\}$, $r^1_{M_3}(M_3) = \{4\}$, $l^1_{M_4}(M_4) = \{1,2\}$, $r^1_{M_4}(M_4) = \{3,4\}$;
- in $\mathcal{S}^2$, $l^2_{M_3}(M_3) = \{2,3\}$, $r^2_{M_3}(M_3) = \{3,4\}$, $l^2_{M_4}(M_4) = \{1,2,3\}$, $r^2_{M_4}(M_4) = \{2,3,4\}$, $l^2_{M_4}(\{1,2,3\}) = \{1,2\}$, $r^2_{M_4}(\{1,2,3\}) = \{2,3\}$, $l^2_{M_4}(\{2,3,4\}) = \{2,3\}$, $r^2_{M_4}(\{2,3,4\}) = \{3,4\}$.

($\mathcal{S}^1$ is the scheme QA and $\mathcal{S}^2$ is the scheme QD defined in Section 6; see Figure 5.) For these schemes, the continuous relaxation bound delivered by $\texttt{PQCR*}^{\mathcal{S}^2}$ is equal to 0 (and is tight), whereas $\texttt{PQCR*}^{\mathcal{S}^1}$ provides the weaker bound -0.625.

Remark 14. *Following the definitions in [52], let us say that a scheme* $\mathcal{S}^1 = \left\{(V^1_M, A^1_M) : M \in \mathcal{M}\right\}$ is included in *a scheme* $\mathcal{S}^2 = \left\{(V^2_M, A^2_M) : M \in \mathcal{M}\right\}$ *if* $\bigcup_{M\in\mathcal{M}} V^1_M \subseteq \bigcup_{M\in\mathcal{M}} V^2_M$*, or in other words, if every monomial that appears in* $\mathcal{S}^1$ *also appears in* $\mathcal{S}^2$*. Note that this is the case for Example 8. In a slightly different framework, Lazare [52] (Proposition 4.2.2) showed that when* $\mathcal{S}^1$ *is included in* $\mathcal{S}^2$*, then the bound provided by* $\texttt{PQCR*}^{\mathcal{S}^2}$ *is at least as good as the bound provided by* $\texttt{PQCR*}^{\mathcal{S}^1}$*. Although we do not prove it explicitly here, the result is likely to extend to our setting.*

6 Computational comparison of quadratization schemes

In this section, we start by summing up the different three-phase reformulations that one can derive from a quadratization scheme, as discussed in previous sections. Next, we present four quadratization schemes in order to examine the impact of the quadratization scheme on the reformulations. Then, in Subsections 6.1 and 6.2, we experimentally compare the performance of different combinations of quadratization schemes, quadratic reformulations and convexification techniques in terms of CPU time and root-node gap of a branch-and-bound algorithm on two families of instances of polynomial binary optimization problems. Finally, in Subsection 6.3, we compare the performance of our best combinations of quadratization schemes, quadratic reformulations and convexification techniques with that of the standard linearization (SL), of a cutting-plane approach based on strengthened formulations of the multilinear polytope [28], and of the MINLP solver `SCIP 9.1` [10].

We summarize in Table 1 the methods to be discussed. Note again that, as pointed out in Section 5.1, (`FOR-L`$^{\mathcal{S}}$), (`ROS-L`$^{\mathcal{S}}$), and (`ABCG-L`$^{\mathcal{S}}$) are exactly the MILP models that would be generated by `Gurobi` when solving the quadratic reformulations (`FOR`$^{\mathcal{S}}$), (`ROS`$^{\mathcal{S}}$), and (`ABCG`$^{\mathcal{S}}$), respectively, with a standard choice of the user parameter `PreQLinearize` equal to 1 (which simply linearizes the quadratic model).

On the other hand, we do not explicitly report on the performance of the smallest eigenvalue convexification method. `Gurobi` applies this method when its parameter `PreQLinearize` is set to 0. In all our experiments, the performance of `Gurobi` was much better (sometimes, by orders of magnitude) when `PreQLinearize`=1. In fact, as might be expected, the computing times obtained when applying the smallest eigenvalue method are generally worse than those obtained by `QCR` which, as we will see hereunder, is itself one of the slowest methods in our tests.

Finally, we did not consider the `PQCR` convexification applied to penalty-based quadratic reformulations ((`ROS`$^{\mathcal{S}}$) or (`ABCG`$^{\mathcal{S}}$)), as this method only works if the linearization inequalities are explicitly added to the constraint set.

Phase 1	Phase 2	Phase 3		
Quadratization scheme $\mathcal{S}$	Quadratic reformulation	Convexification		
		`Linearization`	`QCR`	`PQCR`
-	-	(`SL`)	–	–
`QA`, `QB`, `QC`, `QD`	(`FOR`$^{\mathcal{S}}$)	(`FOR-L`$^{\mathcal{S}}$)	(`FOR-Q*`$^{\mathcal{S}}$)	(`PQCR*`$^{\mathcal{S}}$)
	(`ROS`$^{\mathcal{S}}$)	(`ROS-L`$^{\mathcal{S}}$) (`ROS-C`$^{\mathcal{S}}$)	(`ROS-Q*`$^{\mathcal{S}}$)	×
	(`ABCG`$^{\mathcal{S}}$)	(`ABCG-L`$^{\mathcal{S}}$) (`ABCG-C`$^{\mathcal{S}}$)	(`ABCG-Q*`$^{\mathcal{S}}$)	×

Table 1: Summary of the methods considered in our experiments with the associated phases.

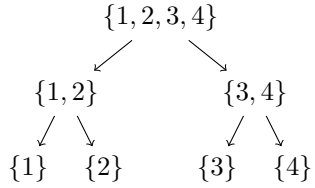
Let us now describe the four specific quadratization schemes that we consider. The case of monomials of degree 2 is trivial, so we focus on the longer ones.

- `QA` [29, 52] is a "lexicographic" quadratization scheme. To obtain it, we start by sorting the monomials in non increasing order of their degrees, and we next sort the monomials of the same degree in lexicographic order. Then, in this order, iteratively: *(i)* Select the first monomial $M = \{i_1, \ldots, i_d\}$ with $i_1 < \ldots < i_d$ and $d \geq 3$. *(ii)* Set $l_M(M) = \{i_1, i_2\}$ and

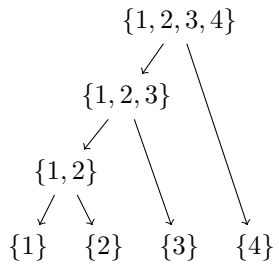
$r_M(M) = \{i_3, \ldots, i_d\}$. *(iii)* Repeat step *(ii)* with $M' = r_M(M)$ until the right child is of degree 2 or less.

- `QB` (Heuristic 2 in Chapter 7 of [60]). Here, we *(i)* consider the product $x_i x_j$ which appears most frequently in the monomial set. *(ii)* For any monomial M containing i and j, set $l_M(M) = \{i, j\}$ and $r_M(M) = M \setminus \{i, j\}$. *(iii)* Add $l_M(M)$ and $r_M(M)$ to the monomial set.
- `QC`: Recursively split any monomial $M = \{i_1, \ldots, i_d\}$ with $i_1 < \ldots < i_d$ and $d \geq 3$ into $l_M(M) = \{i_1, \ldots, i_{d-1}\}$ and $r_M(M) = \{i_d\}$.
- `QD`: Recursively split any monomial $M = \{i_1, \ldots, i_d\}$ with $i_1 < \ldots < i_d$ and $d \geq 3$ into $l_M(M) = \{i_1, \ldots, i_{d-1}\}$ and $r_M(M) = \{i_2, \ldots, i_d\}$. This is our only non-disjoint scheme.

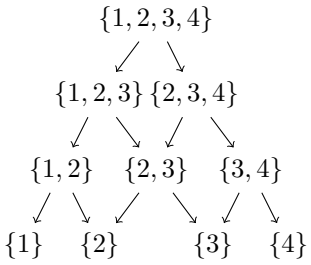
One can easily check that these quadratization schemes fulfill Definition 1. We illustrate them in Figure 5 for the monomial $x_1x_2x_3x_4$.



(a) Quadratization schemes `QA` and `QB`



(b) Quadratization scheme `QC`



(c) Quadratization scheme `QD`

Figure 5: Illustration of the quadratization schemes `QA`, `QB`, `QC`, `QD`

Experimental environment
Our experiments were carried out on a server with 2 CPU Intel Xeon, each of them having 16 cores and 32 threads of 2.3 GHz and $8 * 16$ GB of RAM, under a Linux operating system. For all our experiments, we set a total time limit of 3 hours.

- To solve the linearized formulations (`FOR-L`$^{\mathcal{S}}$), (`ROS-L`$^{\mathcal{S}}$), (`ROS-C`$^{\mathcal{S}}$), (`ABCG-L`$^{\mathcal{S}}$), and (`ABCG-C`$^{\mathcal{S}}$), we relied on the MILP solver of `Gurobi 9.1` [40].
- We used the solver `Mosek` [57] to solve the semi-definite programs (`SDP`), (`SDP`$_{\mathtt{R}}$), and (`SDP`$_{\mathtt{A}}$) that yield the convex quadratic problems (`FOR-Q*`$^{\mathcal{S}}$), (`ROS-Q*`$^{\mathcal{S}}$), and (`ABCG-Q*`$^{\mathcal{S}}$). Then,

the resulting convex problems were solved by the MIQP solver `Gurobi 9.1` [40] with the parameter `PreQLinearize` set to 0 in order to prevent linearization or other reformulations that `Gurobi` might otherwise perform.

- When applying `PQCR`, the quadratization schemes considered in our experiments lead to a very large number of Constraints (42) in (`SDP'`). This raises two numerical difficulties: first, generating all the Constraints (42) is very time consuming, and second, no standard solver is able to directly handle the resulting problem (`SDP'`). To overcome this difficulty, we solved a relaxation of (`SDP'`) where we considered only the subset of Constraints (42) with at least one of E_1 or E_2, and one of E_3 or E_4, corresponding to an original variable x_i. Then we used the solver `Mosek` together with the Conic Bundle library [43] to heuristically solve the relaxation of (`SDP'`) obtained in this way within a Lagrangian duality framework (as described in [9]). Finally, we used the MIQP solver `Gurobi` 9.1 [40] to solve (`PQCR*`$^{\mathcal{S}}$) with the option `PreQLinearize` set to 0.

The quadratizations `QA`, `QC` and `QD` obey the inclusion relations `QA` $\subseteq$ `QD` and `QC` $\subseteq$ `QD` in the sense of Remark 14. So, we generally expect the lower bound provided by (`PQCR*`$^{\texttt{QD}}$) to be better than those provided by (`PQCR*`$^{\texttt{QA}}$) or (`PQCR*`$^{\texttt{QC}}$). However, since we only consider a subset of Constraints (42) in our experiments, these relations are not always verified in the numerical results.

6.1 The image restoration problem

The first class of instances that we consider stems from the *image restoration* problem [12, 33, 45, 48]. The goal is to reconstruct an original sharp base image from a blurred image. An image is a rectangle containing $n = l \times h$ (black or white) pixels. This rectangle is modeled as a binary matrix of the same dimension. The problem can be written as the minimization of a degree-4 polynomial of binary variables where each variable represents a pixel. The coefficients of the monomials are indicative of how likely a configuration is to appear in the sharp base image. The size of the considered instances is $l \times h = 10 \times 10$ and $l \times h = 10 \times 15$, or in the polynomial formulation $n = 100$ and 150, with a number of monomials equal to $|\mathcal{M}| = 668$ and 1033, respectively. In our experiments, 15 instances of each size are considered, for a total of 30 instances. Note that the 15 instances of the same size have identical monomials with different coefficients. The name of each image restoration instance describes its characteristics `v.n.r`: `n` is the number of binary variables and `r` is the index of the instance with the same characteristics. The instances can be found at `https:/github.com/amelie-lambert/ImageRestorationInstances`.

For each quadratization (`QA`, `QB`, `QC`, and `QD`), we display in Table 2 the number of auxiliary variables, namely $|\mathcal{E}|$, and the number of constraints that we actually incorporated in (`SDP'`), denoted here as Σ. As mentioned above, Σ only accounts for a subset of Constraints (42), which means that the root node gap of `PQCR` could in principle be significantly improved by generating more constraints.

We can observe in Table 2 that the number of auxiliary variables is always smaller than the number $|\mathcal{M}|$ of original monomials. This is particularly significant for the quadratization schemes `QA` and `QB` which, for each original monomial of degree 4, only add auxiliary intermediate monomials of degree two. They therefore add at most $n(n-1)/2$ monomials, regardless of the number of original monomials. This also implies that the convexification methods of Table 1 (which all have $n + |\mathcal{E}|$ variables) will have a significantly smaller number of variables than the linearization methods (which use at least $|\mathcal{M}| + |\mathcal{E}|$ variables). The same observation holds for Table 3 in Section 6.2.

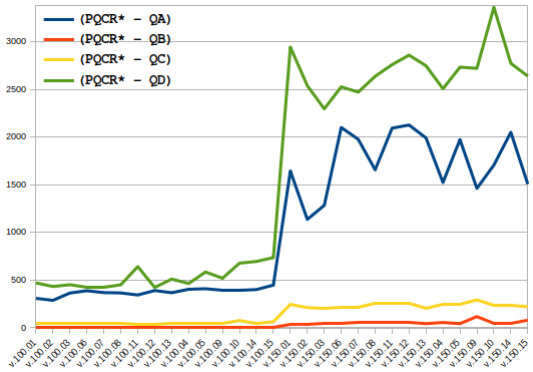
instance	$\|\mathcal{M}\|$	$\|\mathcal{E}\|$				Σ			
		QA	QB	QC	QD	QA	QB	QC	QD
v.100.r	668	252	159	324	423	2990	1317	3446	4687
v.150.r	1033	392	249	504	653	4730	2097	5436	7317

Table 2: Characteristics of the image restoration instances

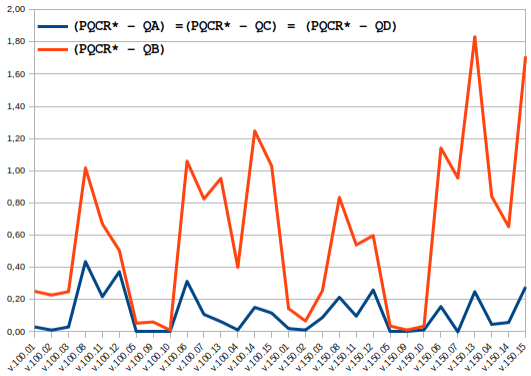
In Figures 6 and 7, we present a comparison of the CPU time required by different approaches to obtain the optimal (binary) solution of the image restoration instances, as well as the initial gaps provided by the corresponding relaxations. In Figure 6, for a given quadratic reformulation and a given convexification, we compare the quadratization schemes (QA, QB, QC, and QD). We only report the detailed results of the best performing methods (PQCR*$^{\mathcal{S}}$), (FOR-L$^{\mathcal{S}}$) and (ROS-C$^{\mathcal{S}}$). In particular, methods (FOR-Q*$^{\mathcal{S}}$), (ROS-Q*$^{\mathcal{S}}$), and (ABCG-Q*$^{\mathcal{S}}$) were not able to solve any of the 30 instances within the time limit of 3 hours. The x-axis represents the instances and the y-axis the CPU times in seconds or the initial gaps $= 100 * \frac{|\text{LB}-\text{S}|}{\text{S}}$, with LB the continuous relaxation value of the method, and S the value of the best known solution. Note that the scale of the vertical axis for the initial gap differs significantly from figure to figure.

We observe that the quadratization scheme has a limited impact on the CPU times of (FOR-L$^{\mathcal{S}}$) (Figure 6c) and (ROS-C$^{\mathcal{S}}$) (Figure 6e), while its choice strongly affects the CPU times of (PQCR*$^{\mathcal{S}}$) (Figure 6a). Indeed, the total CPU time of (PQCR*$^{\mathcal{S}}$) goes on average from about 1600 seconds for QD to 32 seconds for QB, and the CPU time of (FOR-L$^{\mathcal{S}}$) (respectively, (ROS-C$^{\mathcal{S}}$)) goes from 5 seconds for QA (respectively, 5 seconds for QB and QC) to 21 seconds for QD (respectively, 26 seconds for QA). On these sparse instances, the quadratization scheme QB, which has the smallest number of auxiliary variables, is the best scheme for (PQCR*$^{\mathcal{S}}$). Note that most of the CPU time of (PQCR*$^{\mathcal{S}}$) is used to solve (SDP$'$) since the average CPU time for the branch-and-bound is 5 seconds for QB within an average total CPU time of 32 seconds. Concerning the initials gaps, we observe that (PQCR*$^{\mathcal{S}}$) (Figure 6b) always has an initial gap lower than 2%, for all quadratization schemes, whereas (FOR-L$^{\mathcal{S}}$) (Figure 6d) and (ROS-C$^{\mathcal{S}}$) (Figure 6f) always have extremely large gaps, larger than 175%. Moreover, for (PQCR*$^{\mathcal{S}}$) the initial gaps are identical for the quadratization schemes QA, QC and QD and it is also the case for (FOR-L$^{\mathcal{S}}$) with the quadratization schemes QA and QB. This is most likely due to the specific structure of the instances.

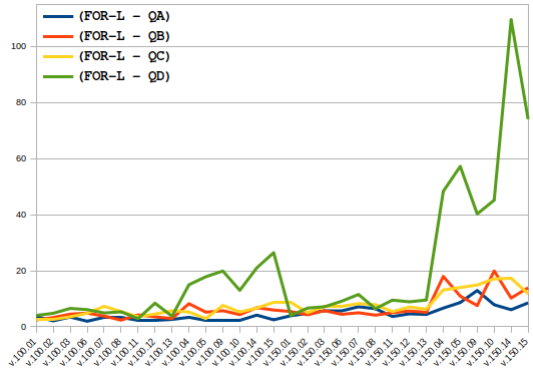
The quadratization scheme that is most efficient globally on all the methods is QB, we therefore compare in Figure 7 the performance of different methods when QB is selected. As in Section 6.2, we compare the CPU time (Figure 7a) and the initial gaps (Figure 7b) for the three best performing methods: (PQCR*$^{\text{QB}}$), (ROS-C$^{\text{QB}}$) and (FOR-L$^{\text{QB}}$) with the performance of the standard linearization (SL). We observe again that rather simple linearization methods like (ROS-C$^{\text{QB}}$) and (FOR-L$^{\text{QB}}$) outperform (PQCR*$^{\text{QB}}$) and (SL) in terms of CPU times, in spite of the fact that (PQCR*$^{\text{QB}}$) has a significant smaller initial gap than the other methods. Finally, we observe that the standard linearization (SL) is the slowest method in spite of an initial gap comparable to that of (FOR-L$^{\text{QB}}$). Then, we present in Figure 7d a comparison of the initial gaps of all the considered methods and we observe that the quadratic reformulation and convexification steps have a clear impact on these gaps. Finally, we present in Figure 7c an illustration of the improvement in computing time obtained with the compact linearization (ROS-C$^{\text{QB}}$) when compared with the reformulation (ROS$^{\text{QB}}$) (which is identical to (ABCG$^{\text{QB}}$) for the scheme QB). We observe that the new linearization is slightly faster on average.



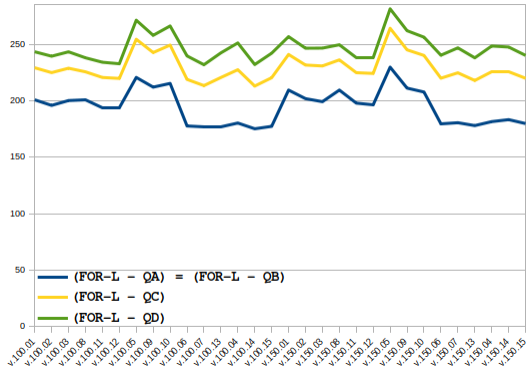
(a) CPU times: (PQCR*$^{\mathcal{S}}$) with QA, QB, QC and QD



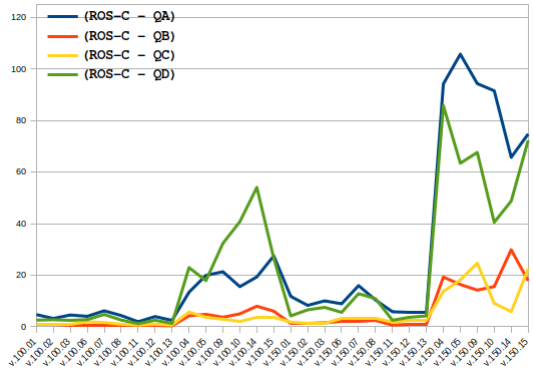
(b) Initial gaps: (PQCR*$^{\mathcal{S}}$) with QA, QB, QC and QD



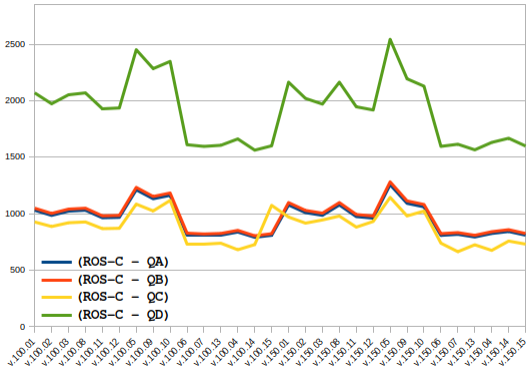
(c) CPU times: (FOR-L$^{\mathcal{S}}$) with QA, QB, QC and QD



(d) Initial gaps: (FOR-L$^{\mathcal{S}}$) with QA, QB, QC and QD

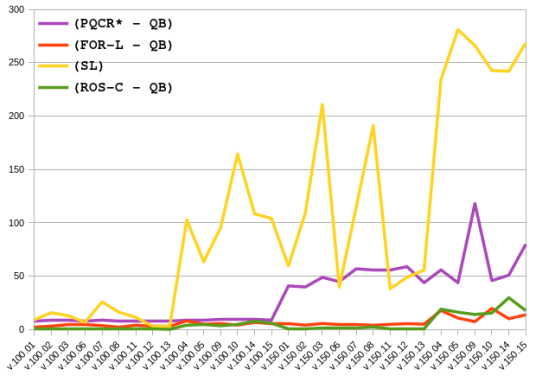


(e) CPU times: (ROS-C$^{\mathcal{S}}$) with QA, QB, QC and QD

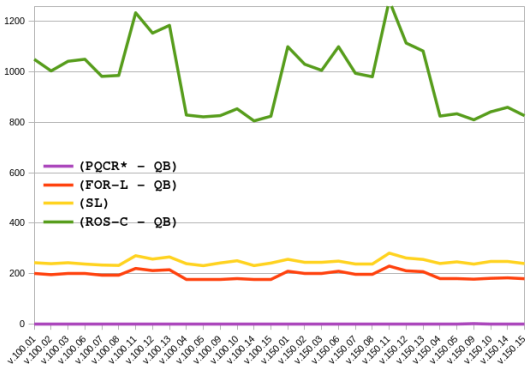


(f) Initial gaps: (ROS-C$^{\mathcal{S}}$) with QA, QB, QC and QD

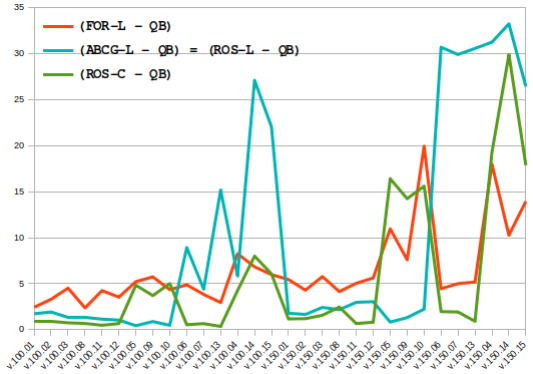
Figure 6: Image restoration instances: CPU time and initial gaps comparison of the quadratizations QA, QB, QC and QD. Time limit: 10800 seconds.



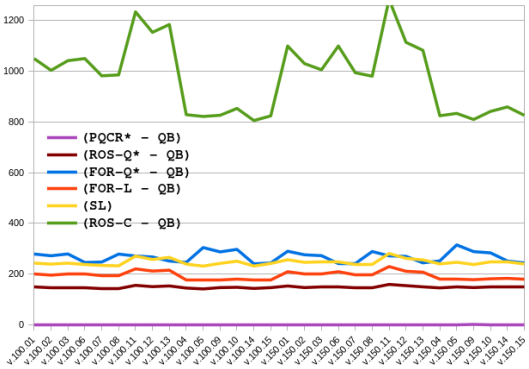
(a) CPU times: best performing methods with QB



(b) Initial gaps: best performing methods with QB



(c) CPU times for linearizations with QB



(d) Initial gaps: all methods with QB

Figure 7: Image restoration: CPU times and initial gaps with quadratization scheme QB

6.2 The Low Auto-correlation Binary Sequence problem

The second set of instances that we consider is associated with the *Low Auto-correlation Binary Sequence* (LABS) problem [7],[53]. Given two integers n_0 and n, with $n_0 \leq n$ the problem is to find a $(-1,+1)$-sequence σ of length n that minimizes the degree-4 polynomial

$$E_{n_0}(\sigma) = \sum_{i=1}^{n-n_0+1} \sum_{d=1}^{n_0-1} \left(\sum_{j=i}^{i+n_0-1-d} \sigma_j \sigma_{j+d} \right)^2 .$$

In order to apply our methods, we convert the variables from $\{-1,1\}$ to $\{0,1\}$ using the standard transformation $x = \frac{\sigma+1}{2}$. The problem admits a lot of symmetries. In particular the value of $E_{n_0}(\sigma)$ is identical for a sequence σ and for its complement. We exploit this symmetry by fixing to 0 the variable that appears most often.

The LABS instances that we consider have been downloaded from the MINLPLib website [55]. These instances have a smaller number of initial variables, but are much more dense than the image restoration instances. Solution values of some hard instances can be found in [29] and [18].

We present in Table 3 the characteristics of our instances, where each instance is labeled $\texttt{b.n.n}_0$ (column *Instance*) and $|\mathcal{M}|$ is the number of monomial of the objective function. Here again, we only generate a subset of Constraints (42), leading to a total number of Σ constraints included in (SDP′). As an illustration, the real number of constraints of (SDP′) for instance b.25.6 is 1525 for QA, 1071 for QB, 41944, for QC, and 53595 for QD.

Instance	$\lvert\mathcal{M}\rvert$	$\lvert\mathcal{E}\rvert$				Σ			
		QA	QB	QC	QD	QA	QB	QC	QD
b.20.5	207	44	25	68	86	498	163	750	1034
b.20.10	833	103	118	245	320	2477	3152	5395	7966
b.20.15	1494	144	157	447	625	4652	5271	13603	22299
b.25.6	407	79	69	144	169	1187	873	2102	2619
b.25.13	1782	180	202	540	717	5900	7234	15942	24347
b.25.19	3040	240	258	934	1312	10088	11200	37170	61312
b.25.25	3677	264	294	1140	1665	12012	14136	50532	87621
b.30.4	223	51	33	73	76	433	189	631	658
b.30.8	926	143	141	324	401	3125	2923	6574	8997
b.30.15	2944	265	299	924	1228	10497	13009	32702	50120
b.35.4	263	61	38	88	91	523	214	766	793
b.35.9	1381	198	197	493	616	5064	4819	11423	15928
b.40.5	447	104	45	168	206	1278	263	1970	2634
b.45.5	507	119	50	193	236	1473	288	2275	3034
b.50.6	882	179	144	344	394	2887	1848	5302	6444
b.55.6	977	199	159	384	439	3227	2043	5942	7209
b.60.8	2036	323	291	774	941	7625	6073	16684	22437

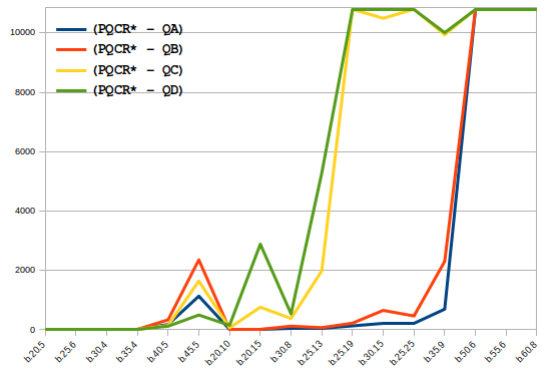
Table 3: Characteristics of the LABS instances

We present the same graphs as in Section 6.1 (Figures 8-10), comparing the CPU time of the optimization process (in binary variables), and the initial gap for different combinations of the quadratization schemes, quadratic reformulations and convexifications. We only report the detailed results of the best performing methods (PQCR$*^{\mathcal{S}}$), (FOR-Q$*^{\mathcal{S}}$), (FOR-L$^{\mathcal{S}}$) and (ROS-C$^{\mathcal{S}}$) (methods (ROS-Q$*^{\mathcal{S}}$) and (ABCG-Q$*^{\mathcal{S}}$), were only able to solve one instance out of 17 within the time limit of 3 hours).

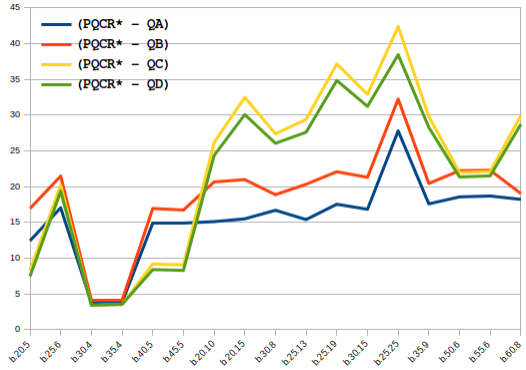
We observe that the CPU times (Figures 8a and 8c, respectively) and the initial gaps (Figures 8b and 8d, respectively) for the convexification methods based on quadratic convex reformulation ((PQCR$*^{\mathcal{S}}$) and (FOR-Q$*^{\mathcal{S}}$), respectively) are impacted by the choice of the quadratization. Note that the solution time of (SDP′) is included in the total time, and is on average about 25 seconds for QA and QB, 500 for QC, and 1450 for QD. On the other side, the impact of the quadratization scheme on the linearization methods is much smaller. It barely affects the CPU time, as shown in Figures 9a and 9c. For (FOR-L$^{\mathcal{S}}$), the initial gap is very large but quite stable (Figure 9b) and even identical for the quadratizations QA, QC, and QD. For (ROS-C$^{\mathcal{S}}$), however, the choice of the quadratization has a real impact on the initial gaps, with a factor of about 2 between the gaps of the quadratization schemes QC and QD (Figure 9d).

The quadratization scheme that is globally most efficient, irrespective of the methods, is QA. For this reason, we compare the performance of the different methods when QA is applied; see Figure 10.

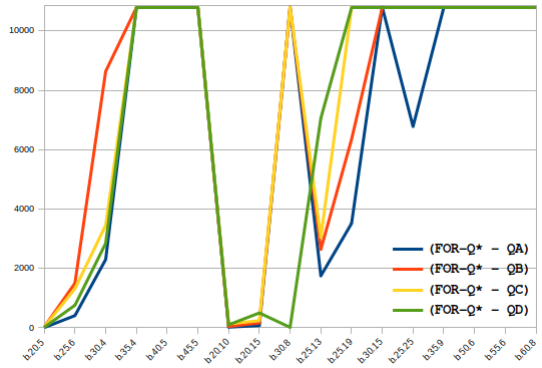
More precisely, we compare the CPU time (Figure 10a) and the initial gaps (Figure 10b) for the three best performing methods: (PQCR*$^{\texttt{QA}}$), (ROS-C$^{\texttt{QA}}$) and (FOR-L$^{\texttt{QA}}$) with the performance of the standard linearization (SL). We observe that (PQCR*$^{\texttt{QA}}$) outperforms the other methods in terms of both CPU time and initial gaps. The linearizations (FOR-L$^{\texttt{QA}}$) and (ROS-C$^{\texttt{QA}}$) have about the same CPU times, while the initial gaps with (FOR-L$^{\texttt{QA}}$) are almost twice as small as those with (ROS-C$^{\texttt{QA}}$). Finally, for the standard linearization (SL), we observe that despite an initial gap comparable to that of (FOR-L$^{\texttt{QA}}$), it is the slowest formulation. Figure 10d displays a comparison of the initial gaps of all the considered methods; we observe that the quadratic reformulation and convexification steps have a clear impact on these gaps. Finally, Figure 10c illustrates the improvement in computing time brought by the compact linearization (ROS-C$^{\texttt{QA}}$) when compared to the linearization (ROS-L$^{\texttt{QA}}$) (which is identical to (ABCG-L$^{\texttt{QA}}$)). We observe that it performs comparably to the linearization (FOR-L$^{\texttt{QA}}$).



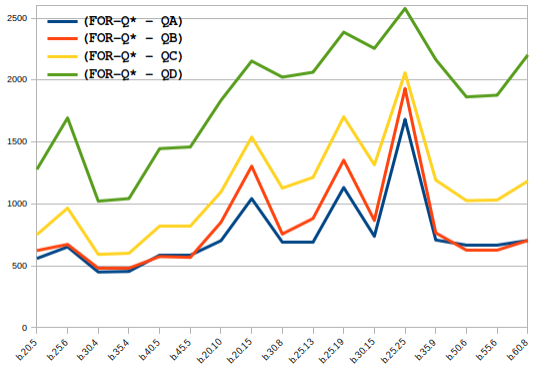
(a) CPU times: (PQCR*$^{\mathcal{S}}$) with QA, QB, QC and QD



(b) Initial gaps: (PQCR*$^{\mathcal{S}}$) with QA, QB, QC and QD

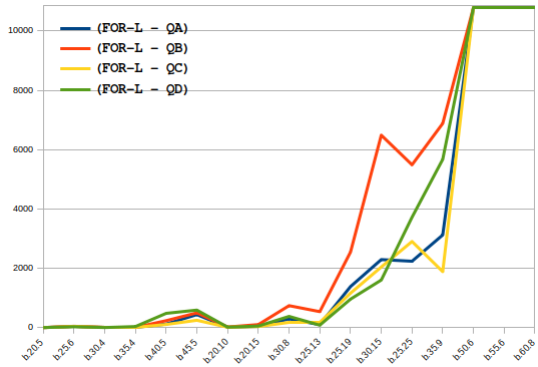


(c) CPU times: (FOR-Q*$^{\mathcal{S}}$) with QA, QB, QC and QD

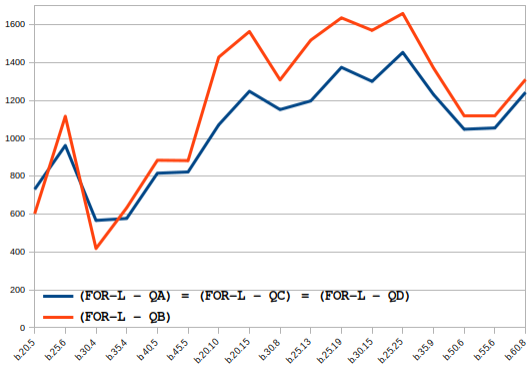


(d) Initial gaps: (FOR-Q*$^{\mathcal{S}}$) with QA, QB, QC and QD

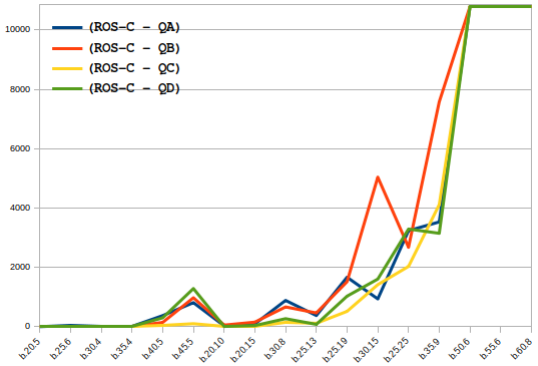
Figure 8: LABS: CPU times and initial gaps for QCR and PQCR applied to (FOR$^{\mathcal{S}}$).



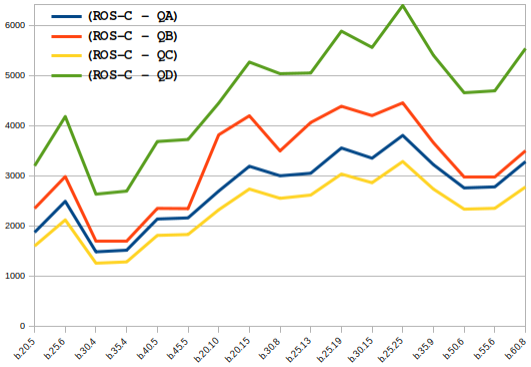
(a) CPU times: (FOR-L$^{\mathcal{S}}$) with QA, QB, QC and QD



(b) Initial gaps: (FOR-L$^{\mathcal{S}}$) with QA, QB, QC and QD

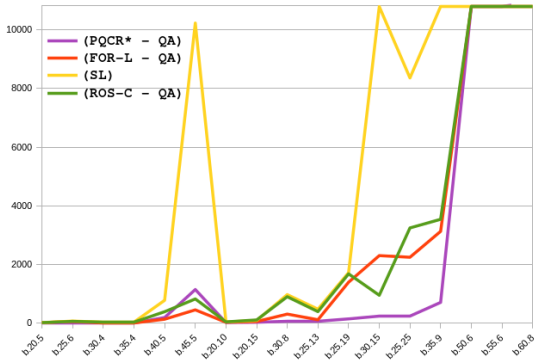


(c) CPU times: (ROS-C$^{\mathcal{S}}$) with QA, QB, QC and QD

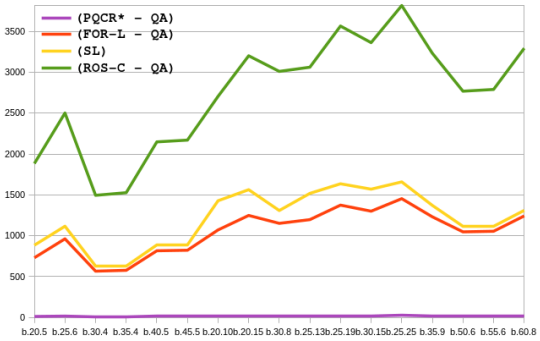


(d) Initial gaps: (ROS-C$^{\mathcal{S}}$) with QA, QB, QC and QD

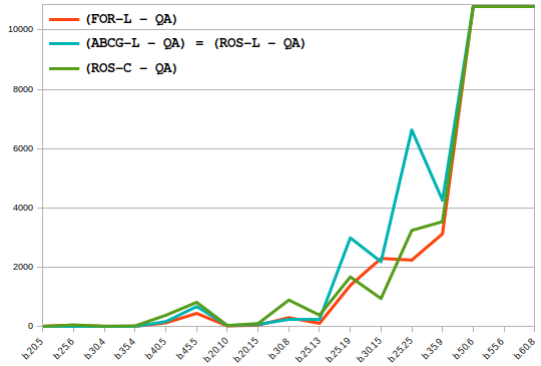
Figure 9: LABS: CPU times and initial gaps for the linearizations.



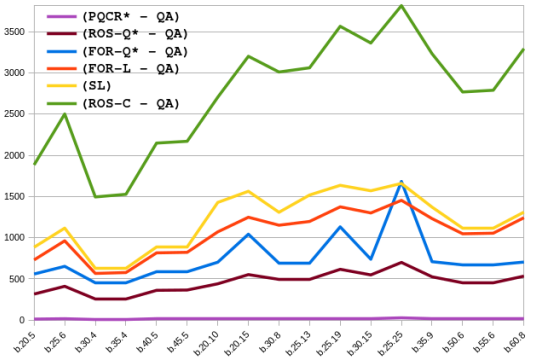
(a) CPU times: best performing methods with `QA`



(b) Initial gaps: best performing methods with `QA`



(c) CPU times for linearizations with `QA`



(d) Initial gaps: all methods with `QA`

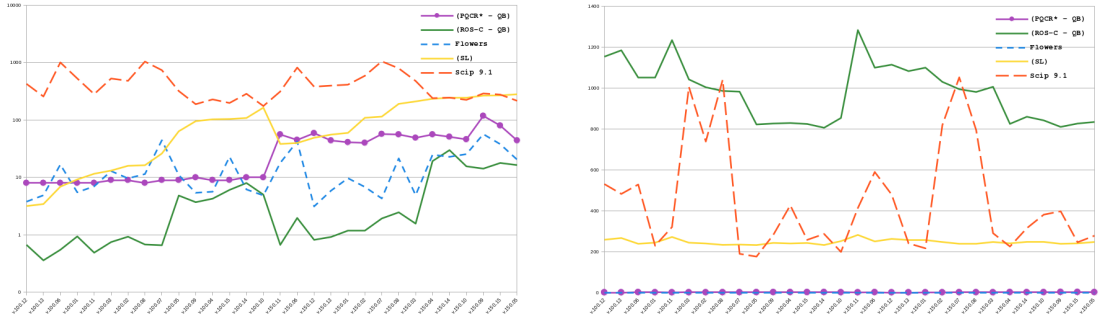
Figure 10: LABS: CPU times and initial gaps with quadratization scheme `QA`

6.3 A comparison with direct linearization methods and `SCIP`

In this section, we compare the performance of best combinations of quadratization schemes, quadratic reformulations and convexification techniques on the one hand (namely, (`PQCR*`$^{\mathcal{S}}$) and (`ROS-C`$^{\mathcal{S}}$)), and the following exact solution methods on the other hand:

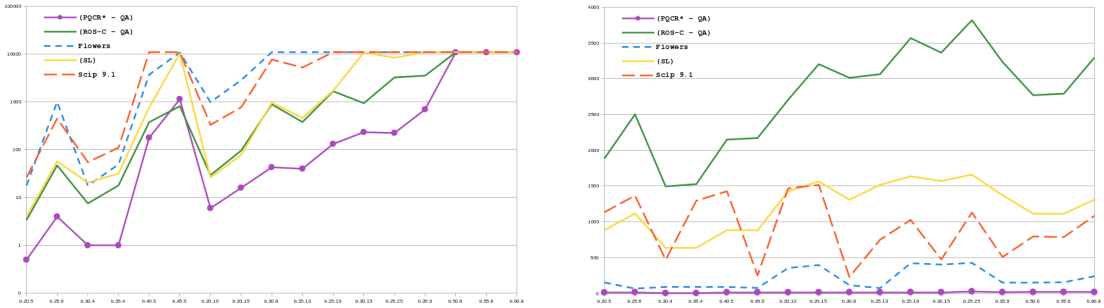
- `SL`: solution by `Gurobi 9.1` of the standard linearization (SL);
- `Flowers`: solution of the standard linearization (SL) enriched by "flower inequalities with two neighbors" introduced in [24, 26], and by "simple odd β-cycle inequalities" [28], a subset of the odd β-cycle inequalities introduced in [22]; this method uses the branch-and-cut code implemented by Del Pia and Walter [28] within the `SCIP` framework [10];
- `SCIP`: direct solution of the polynomial unconstrained binary program (P) by the MINLP solver `SCIP 9.1` [10]. The solver uses a spatial branch-and-bound algorithm based on a linear outer-approximation, which is computed by convex over- and under-estimation of nonconvex functions [65].

We present in Figures 11 and 12 the CPU time and initial gap obtained by each method for the image restoration and LABS instances, respectively.



(a) CPU times (log scale) (b) Initial gaps of the root node relaxation

Figure 11: Image restoration: comparison with linearization and SCIP



(a) CPU times (log scale) (b) Initial gaps of the root node relaxation

Figure 12: LABS: comparison with linearization and SCIP

As in previous experiments, the different methods do not perform equally well when applied to the two classes of instances, namely, image restoration and LABS. For the sparser instances (image restoration, Figure 11), which are solved within the time limit by all methods, the (ROS-C^{QB}) linearization is generally more efficient than the other approaches. In particular, (ROS-C^{QB}) is the fastest method for 90% of the instances. The cutting-plane approach Flowers and the quadratic convex method (PQCR*^{QB}) are the second fastest, while SL and SCIP are clearly slower. Surprisingly, the quality of the initial gaps is not directly related to the efficiency of the approaches. In particular, the bounds provided by (PQCR*^{QB}) and Flowers are extremely tight (their duality gaps are smaller than 1%), and the (ROS-C^{QB}) linearization has the worst initial gaps (on average, 1000 times larger than those of Flowers and (PQCR*^{QB})). But the computation of the initial bounds for (PQCR*^{QB}) and Flowers can be quite expensive, whereas it is very fast for (ROS-C^{QB}).

The densest LABS instances (Figure 12) are significantly harder to solve since only 14 out of 17 instances are solved to global optimality within the time limit of 3 hours by at least one of the

methods considered here. We observe that the convexification method (PQCR*^{QA}) is clearly the most efficient one: it is fastest for 93% of our instances. The (ROS-C^{QA}) linearization comes in second, followed by SL, the solver SCIP, and Flowers that only solve 11, 8 and 8 instances, respectively, within the time limit. We observe that the (PQCR*^{QA}) convexification, which is clearly the most efficient method on these difficult instances, also has the smallest duality gap (always less than 20%). For the linearization approaches, however, there is again no clear relationship between the overall efficiency and the quality of the initial gap. Indeed, the time spent at the root node, which allows Flowers to reach a duality gap 100 times smaller, on average, than the gap of (ROS-C^{QA}), results in long computational times for the overall branch-and-bound process.

It is not completely clear why denser instances like LABS are so much more difficult to solve than sparser ones, like image restoration instances. However, the density of the instances has a direct impact on the number of additional variables of the standard linearization, and intuitively, the greater the number of products is approximated, the weaker the continuous relaxation will be. This is what we observe for these two types of instances since their initial integrality gaps are of different orders of magnitude. For the set of sparse image restoration instances, the initial integrality gap of the standard linearization is about 200%, as shown in Figure 11b, while for the denser LABS instances, Figure 12b shows that the initial integrality gap of the standard linearization is about 1000%.

As a conclusion of this section, and although the results vary according to the class of instances that are solved, it can be confidently stated that the quadratic reformulation methods discussed in this paper are at the very least competitive with more classical linearization methods. For certain types of problems and instances, they even deliver the best results obtained in our experiments.

7 Conclusion

In this paper, we have examined a generic framework for the exact solution of the polynomial unconstrained binary programming problem. The framework consists of three phases: the first one determines a *quadratization scheme* of the polynomial which is used, in a second phase, to produce a *quadratic reformulation* of the initial problem. The resulting quadratic problem is in general non-convex. The third phase of the solution process is the *convexification* of the quadratic reformulation. For each phase, we have presented in a unified way several possible approaches that were previously introduced independently of each other, and we have clarified their relations from a theoretical point of view.

We have illustrated our findings through a set of computational experiments. Our numerical results clearly demonstrate that the choice of the quadratization scheme, of the quadratic reformulation step and of the convexification method should not be made arbitrarily and independently of each other. In particular, in our experiments, the smaller quadratization schemes QA and QB rather consistently outperformed the denser schemes QC and QD in terms of CPU time. Regarding the quadratic reformulation methods, the constraint-based approaches proved more robust than the penalty approaches. Finally, for the convexification methods, QCR struggled and was never competitive, neither in terms of gap nor in terms of CPU time, whereas PQCR was best able to reduce the duality gap and to provide a tight reformulation. However, this small gap comes at the cost of long computing times, mostly spent in setting up the large-size reformulation. For the (relatively easy) image restoration instances, this time was prohibitively large, so that the reduction of the gap and the associated tighter relaxation were not sufficient to make PQCR really competitive: it was

generally outperformed by simpler, more primitive methods like (FOR-L$^{\mathcal{S}}$) or (ROS-C$^{\mathcal{S}}$), in spite of their much larger initial gap.

In summary, we conclude that within the three-phase framework for polynomial unconstrained binary optimization, the performance of a global solution method depends very much on the combination of its constituting elements, as well as on the features of the instances to be solved. The combined method should always be designed accordingly. In particular, for sparse instances, applying linearization methods that do not require expensive pre-processing and whose size remains moderate is probably a wise choice. On the other hand, for very dense instances, the size of linearizations increases drastically, and thus convexifications that are very stable in both size and gap are likely to perform more efficiently. Another lesson drawn from the computational experiments is that quadratic reformulation methods frequently provide a very competitive alternative to generic MINLP algorithms, like outer-approximation, or to classical linearization approaches. In particular, for some of the instances that we considered, quadratization phases *à la* Fortet or *à la* Rosenberg, followed by a linearization step or by a PQCR convexification step, provided the best numerical results. Although our computational experiments were mostly meant to illustrate the three-phase framework and are limited in scope, they clearly suggest that quadratic reformulations of polynomial unconstrained binary optimization problems should be further investigated in the future.

Acknowledgements

The authors are grateful to the anonymous reviewers for their helpful comments, and to Matthias Walter for gracefully sharing his prototype code. This research has benefited from the support of the FMJH Program Gaspard Monge for optimization and operations research and their interactions with data science.

Data availability

All numerical instances used in this research are available at `http://www.minlplib.org/` and `https:/github.com/amelie-lambert/ImageRestorationInstances`.

Statements and Declarations

The authors have no competing interests to declare that are relevant to the content of this article.

References

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

[2] 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.

[3] A.A. Ahmadi and A. Majumdar. DSOS and SDSOS optimization: LP and SOCP-based alternatives to sum of squares optimization. In *48th Annual Conference on Information Sciences and Systems (CISS)*, pages 1–5, March 2014.

[4] F.A. Al-Khayyal and J.E. Falk. Jointly constrained biconvex programming. *Mathematics of Operations Research*, 8(2):273–286, 1983.

[5] M. Anthony, E. Boros, Y. Crama, and A. Gruber. Quadratization of symmetric pseudo-Boolean functions. *Discrete Applied Mathematics*, 203:1 – 12, 2016.

[6] M. Anthony, E. Boros, Y. Crama, and A. Gruber. Quadratic reformulations of nonlinear binary optimization problems. *Mathematical Programming*, 162(1-2):115–144, 2017.

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

[8] 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.

[9] 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.

[10] S. Bolusani, M. Besançon, K. Bestuzheva, A. Chmiela, J. Dionísio, T. Donkiewicz, J. van Doornmalen, L. Eifler, M. Ghannam, A. Gleixner, C. Graczyk, K. Halbig, I. Hedtke, A. Hoen, C. Hojny, R. van der Hulst, D. Kamp, T. Koch, K. Kofler, J. Lentz, J. Manns, G. Mexi, E. Mühmer, M. E. Pfetsch, F. Schlösser, F. Serrano, Y. Shinano, M. Turner, S. Vigerske, D. Weninger, and L. Xu. The SCIP Optimization Suite 9.0. Technical report, Optimization Online, February 2024.

[11] E. Boros, Y. Crama, and E. Rodríguez-Heck. Compact quadratizations for pseudo-boolean functions. *Journal of Combinatorial Optimization*, 39:687–707, 2020.

[12] E. Boros, A. Fix, A. Gruber, and R. Zabih. A hypergraph-based reduction for higher-order binary Markov random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(7):1387–1395, 2015.

[13] E. Boros and A. Gruber. On quadratization of pseudo-Boolean functions. In *ISAIM*, Fort Lauderdale, 2012. ISAIM, International Symposium on Artificial Intelligence and Mathematics. Open Source: arXiv:1404.6538v1.

[14] E. Boros and P.L. Hammer. Pseudo-Boolean optimization. *Discrete Applied Mathematics*, 123(1):155–225, 2002.

[15] C. Buchheim, Y. Crama, and E. Rodríguez-Heck. Berge-acyclic multilinear 0-1 optimization problems. *European Journal of Operational Research*, 271(1):102–107, 2019.

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

[17] 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.

[18] J.V. Clausen, Y. Crama, R. Lusby, E. Rodríguez-Heck, and S. Ropke. Solving unconstrained binary polynomial programs with limited reach: Application to low autocorrelation binary sequences. *Computers and Operations Research*, 165:106586, 2024.

[19] Y. Crama and P.L. Hammer. *Boolean Functions: Theory, Algorithms, and Applications*. Cambridge University Press, New York, N.Y., 2011.

[20] Y. Crama, P. Hansen, and B. Jaumard. The basic algorithm for pseudo-Boolean programming revisited. *Discrete Applied Mathematics*, 29(2-3):171–185, 1990.

[21] Y. Crama and E. Rodríguez-Heck. A class of valid inequalities for multilinear 0-1 optimization problems. *Discrete Optimization*, 25:28–47, 2017.

[22] A. Del Pia and S. Di Gregorio. Chvátal rank in binary polynomial optimization. *INFORMS Journal on Optimization*, 3(4):315–349, 2021.

[23] A. Del Pia and S. Di Gregorio. On the complexity of binary polynomial optimization over acyclic hypergraphs. *Algorithmica*, https://doi.org/10.1007/s00453-022-01086-9, 2022.

[24] A. Del Pia and A. Khajavirad. A polyhedral study of binary polynomial programs. *Mathematics of Operations Research*, 42(2):389–410, 2016.

[25] A. Del Pia and A. Khajavirad. On decomposability of multilinear sets. *Mathematical Programming: Series A*, 170:387–415, 2017.

[26] A. Del Pia and A. Khajavirad. The multilinear polytope for acyclic hypergraphs. *SIAM Journal on Optimization*, 28(2):1049–1076, 2018.

[27] A. Del Pia, A. Khajavirad, and N.V. Sahinidis. On the impact of running intersection inequalities for globally solving polynomial optimization problems. *Mathematical Programming Computation*, 12(2):165–191, 2020.

[28] A. Del Pia and M. Walter. Simple odd β-cycle inequalities for binary polynomial optimization. *Mathematical Programming: Series B*, https://doi.org/10.1007/s10107-023-01992-y, 2023.

[29] S. Elloumi, A. Lambert, and A. Lazare. Solving unconstrained 0–1 polynomial programs through quadratic convex reformulation. *Journal of Global Optimization*, 80(2):231–248, 2021.

[30] A. Fischer, F. Fischer, and S.T. McCormick. Matroid optimisation problems with nested non-linear monomials in the objective function. *Mathematical Programming*, 169(2):417–446, 2018.

[31] A. Fix, A. Gruber, E. Boros, and R. Zabih. A hypergraph-based reduction for higher-order binary Markov random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(7):1387–1395, 2015.

[32] R. Fortet. L'algèbre de Boole et ses applications en recherche opérationnelle. *Cahiers du Centre d'Études de Recherche Opérationnelle*, 4:5–36, 1959.

[33] D. Freedman and P. Drineas. Energy minimization via graph cuts: settling what is possible. *IEEE Conference on Computer Vision and Pattern Recognition*, 2:939–946, 2005.

[34] D. Freedman and P. Drineas. Energy minimization via graph cuts: settling what is possible. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, 2:939–946 vol. 2, 2005.

[35] 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.

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

[37] F. Glover, G. Kochenberger, and Y. Du. Quantum bridge analytics I: A tutorial on formulating and using QUBO models. *4OR*, 17(4):335–371, 2019.

[38] F. Glover and E. Woolsey. Further reduction of zero-one polynomial programming problems to zero-one linear programming problems. *Operations Research*, 21(1):156–161, 1973.

[39] F. Glover and E. Woolsey. Technical note: Converting the 0-1 polynomial programming problem to a 0-1 linear program. *Operations Research*, 22(1):180–182, 1974.

[40] Gurobi Optimization LLC. Gurobi Optimizer 9.1. https://www.gurobi.com, 2021.

[41] P.L. Hammer, I.G. Rosenberg, and S. Rudeanu. Application of discrete linear programming to the minimization of Boolean functions. *Revue Roumaine de Mathématiques Pures et Appliquées*, 8:459–475, 1963.

[42] P.L. Hammer, I.G. Rosenberg, and S. Rudeanu. On the determination of the minima of pseudo-Boolean functions. *Studii si Cercetari Matematice*, 14:359–364, 1963. in Romanian.

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

[44] IBM. IBM ILOG CPLEX Optimizer 12.6 and 12.7. http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/.

[45] H. Ishikawa. Transformation of general binary MRF minimization to the first-order case. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(6):1234–1249, 2011.

[46] N. Ito, S. Kim, M. Kojima, and A.Takeda K.C. Toh. BBCPOP: A Sparse Doubly Nonnegative Relaxation of Polynomial Optimization Problems with Binary, Box and Complementarity Constraints. *ArXiv e-prints*, April 2018.

[47] A. Khajavirad. On the strength of recursive McCormick relaxations for binary polynomial optimization. *Operations Research Letters*, 51(2):146–152, 2023.

[48] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):147–159, 2004.

[49] X. Kuang, B. Ghaddar, J. Naoum-Sawaya, and L.F. Zuluaga. Alternative SDP and SOCP Approximations for Polynomial Optimization. *ArXiv e-prints*, October 2015.

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

[51] J.B. Lasserre and T.P. Thanh. Convex underestimators of polynomials. *Journal of Global Optimization*, 56:1–25, 2013.

[52] A. Lazare. *Global optimization of polynomial programs with mixed-integer variables*. PhD thesis, Université Paris-Saclay, https://www.theses.fr/2019SACLY011.pdf, 2019.

[53] 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.

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

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

[56] 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.

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

[58] A.P. Punnen, editor. *The Quadratic Unconstrained Binary Optimization Problem: Theory, Algorithms, and Applications*. Springer Nature, 2022.

[59] A.U. Raghunathan, C. Cardonha, D. Bergman, and C.J. Nohra. Recursive McCormick linearization of multilinear programs. Technical Report arXiv:2207.08955, 2022.

[60] E. Rodríguez-Heck. *Linear and quadratic reformulations of nonlinear optimization problems in binary variables*. PhD thesis, Université de Liège, 2018.

[61] I.G. Rosenberg. Reduction of bivalent maximization to the quadratic case. *Cahiers du Centre d'Études de Recherche Opérationnelle*, 17:71–74, 1975.

[62] H. S. Ryoo and N. V. Sahinidis. Analysis of bounds for multilinear functions. *Journal of Global Optimization*, 19(4):403–424, 2001.

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

[64] S. Sakaue, A. Takeda, S. Kim, and N. Ito. Exact semidefinite programming relaxations with truncated moment matrix for binary polynomial optimization problems. *SIAM Journal on Optimization*, 27(1):565–582, 2017.

[65] S. Vigerske and A. Gleixner. SCIP: Global optimization of mixed-integer nonlinear programs in a branch-and-cut framework. *Optimization Methods and Software*, 33(3):563–593, 2018.

[66] L.J. Watters. Reduction of integer polynomial programming problems to zero-one linear programming problems. *Operations Research*, 15:1171–1174, 1967.

[67] W.I. Zangwill. Media selection by decision programming. *Journal of Advertising Research*, 5:30–36, 1965.