GLOBALLY SOLVING QUADRATIC PROGRAMS WITH
CONVEX OBJECTIVE AND COMPLEMENTARITY
CONSTRAINTS VIA COMPLETELY POSITIVE PROGRAMMING

ZHI-BIN DENG
School of Economics and Management, University of Chinese Academy of Sciences
Key Laboratory of Big Data Mining and Knowledge Management
Chinese Academy of Sciences, Beijing 100190, China

YE TIAN
School of Business Administration, Southwestern University of Finance and Economics
Chengdu 611130, China

CHENG LU*
School of Economics and Management, North China Electric Power University
Beijing 102206, China

WEN-XUN XING
Department of Mathematical Sciences, Tsinghua University
Beijing 100084, China

(Communicated by Shu-Cherng Fang)

ABSTRACT. Quadratic programs with complementarity constraints (QPCC) are NP-hard due to the nonconvexity of complementarity relation between the pairs of nonnegative variables. Most of the existing solvers are capable of solving QPCC by finding stationary solutions, which are not able to be verified as global or local optimal ones. In this paper, we aim to globally solve QPCC by a branch-and-bound algorithm, in which the doubly nonnegative (DNN) relaxation in each node is efficiently solved via an augmented Lagrangian method. The method is practically efficient due to the fact that the augmented Lagrangian function can be decomposed into two easy-to-solve subproblems. Computational results demonstrate the effectiveness of the proposed algorithm, with a particular highlight in only a few nodes for some instances.

1. Introduction. We consider the following quadratic program with complementarity constraints (QPCC):

\[
\begin{align*}
\min \quad & f(x) = x^T Q x + q^T x \\
\text{s.t.} \quad & A x = b, \\
& x \geq 0, \\
& x_i x_j = 0, \ \forall (i, j) \in \mathcal{E},
\end{align*}
\]

(PQCC)
where \( x \in \mathbb{R}^n \) is the \( n \)-dimensional decision variable, \( Q \in \mathbb{R}^{n \times n} \), \( q \in \mathbb{R}^n \), \( A \in \mathbb{R}^{m \times n} \), \( b \in \mathbb{R}^m \), and \( E \subseteq \{(i,j)|1 \leq i < j \leq n\} \) are given data. Moreover, we assume that the matrix \( Q \) is symmetric and positive semidefinite, the polyhedral set \( \mathcal{P} = \{ x \in \mathbb{R}^n | Ax = b, x \geq 0 \} \) is bounded with a nonempty relative interior, and the feasible region \( \mathcal{F} = \{ x \in \mathbb{R}^n | Ax = b, x \geq 0, x_i x_j = 0, (i,j) \in E \} \neq \emptyset \). Denote \( \mathcal{H} = \{ x \in \mathbb{R}^n | x_i x_j = 0, \forall (i,j) \in E \} \), then \( \mathcal{F} = \mathcal{P} \cap \mathcal{H} \). Problem \( \text{(QPCC)} \) is NP-hard because the classical linear programs with complementarity constraints (LPCC), as a special subclass of \( \text{(QPCC)} \), is proved to be NP-hard in [14]. Moreover, the notorious nonconvex and disjunctive features of the complementarity constraints make the standard constraint qualifications, such as Mangasarian-Fromovitz constraint qualification [19], do not hold. These features prevent the direct application of nonlinear programming (NLP) solvers except under restrictive circumstances. In order to overcome the difficulty, the constraints \( x_i x_j = 0, (i,j) \in E \), are usually relaxed to \( x_i x_j \leq \phi \) or penalized into the objective function as additional terms \( \phi x_i x_j \) for some parameter \( \phi \). The resulting relaxations then often satisfy the constraint qualification required by NLP solvers. When the parameter \( \phi \) goes to the limit or beyond some threshold, one can obtain a sequence of solutions of the relaxations converging to a stationary point of problem (QPCC). A number of approaches have been proposed under this framework. For example, Huang et al. [12] investigate necessary and sufficient conditions that guarantee the convergence of the relaxation solutions. Hao and Liu [9] propose a trust-region filter-SQP method for solving the relaxations in order to promote the global convergence, while Chang et al. [6] adopt the generalized Newton method. For the theoretical and numerical comparison of different regularization methods for mathematical programs with complementarity constraints, we refer the readers to [10].

However, the regularization methods are incapable of ascertaining the quality of the solution, either local or global optimality. This motives us to undertake a systematic investigation on the global resolution of problem (QPCC), which has extensive applications in engineering and economics [8].

Recently, Hu et al. [11] propose an integer programming (IP) based algorithm for the global resolution of LPCC. They reformulate LPCC as a minmax integer program that provides the certificates for the three outcomes—ineasibility, unboundedness, or solvability—of LPCC. They characterize the extreme rays of the reformulated IP problem and derived valid inequalities in the form of satisfiability constraints in the spirit of Benders decomposition. The satisfiability constraints are then sparsiﬁed by a special procedure in the algorithm for the purpose of efficiency. Following the same research line, Bai et al. [1] solve QPCC by a two-stage approach, wherein the first stage solves a mixed integer quadratic program with preset upper bounds on complementarity variables, and the second stage solves the program outside the bounded region by the Benders scheme. Another global solution method is based on conic approximations. Zhou et al. [25] reformulate QPCC as a conic programming problem and then use a sequence of semidefinite relaxations to generate lower bounds converging to the optimal value.

The present paper aims to solve QPCC via completely positive programming approach. Specifically, we first reformulate QPCC to a completely positive problem (CPP). The contribution of this paper are twofold. First, this is the first time to use SDP relaxations within a branch-and-bound framework to solve QPCC. The difficulty lies in that the computational burden for SDP relaxations becomes excruciating when the problem size increases. We overcome this difficulty by solving
the relaxations via an augmented Lagrangian method, which is efficient in small and medium scale. Secondly, although the SDP relaxations are polynomial-time tractable, they are not required to be solved exactly in every node, except for the leaf node due to the finiteness of the algorithm, for the purpose of overall efficiency. Instead, a good lower bound is obtained in an efficient way if the relaxation in one node cannot be fully solved by our method. Computational results show that the proposal algorithm can solve medium-sized instances within reasonable amount of time.

The rest of this paper is arranged as follows. Section 2 presents the completely positive reformulation for QPCC and derives the doubly nonnegative program (DNP) as an SDP relaxation. Section 3 proposes the augmented Lagrangian method for efficiently solving the DNP relaxations. Section 4 states the branch-and-bound algorithm for globally solving QPCC and the implementation details for improving the efficiency. Results of numerical experiment are presented in Section 5, and the conclusions are given in Section 6.

The following notations are adopted in this paper. \( S_n \) denotes the cone of \( n \times n \) real symmetric matrices, \( S_n^+ \) denotes the cone of \( n \times n \) real positive semidefinite matrices, and \( N_n^+ \) denotes the cone of \( n \times n \) real symmetric matrices with nonnegative components. \( C_n \) denotes the cone of completely positive matrices, that is \( C^n = \{ A \in S^n \mid \exists \text{ integer } r \geq 0, A = \sum_{k=1}^{r} v_k v_k^T, v_k \in \mathbb{R}^n \} \). \( \text{diag}(A) \) denotes the diagonal vector of matrix \( A \). For two matrices \( A, B \in S^n \), the inner product between \( A \) and \( B \) is defined as \( A \cdot B = \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} B_{ij} \) where \( A_{ij} \) and \( B_{ij} \) are the \((i,j)\)-th components of matrices \( A \) and \( B \), respectively. For two vectors \( a, b \in \mathbb{R}^n \), the Hadamard product between \( a \) and \( b \), denoted by \( a \odot b \), is the component-wise product of \( a \) and \( b \).

2. Copositive reformulation and doubly nonnegative relaxation. In [4], Burer has shown that problem (QPCC) is equivalent to the following completely positive problem in terms of optimal value:

\[
\begin{align*}
\min & \quad \hat{Q} \cdot Y, \\
\text{s.t.} & \quad Ax = b, \ \text{diag}(AXAT) = b \odot b, \\
& \quad X_{ij} = 0, \ (i,j) \in \mathcal{E}, \\
& \quad Y = \begin{bmatrix} 1 & x^T \\ x & X \end{bmatrix} \in C^{n+1},
\end{align*}
\]  
(CPP)

where \( \hat{Q} = \begin{bmatrix} 1 & q^T/2 \\ q/2 & Q \end{bmatrix} \in S^{n+1} \). Since the completely positive cone \( C^{n+1} \) is intractable (ref. [7]), problem (CPP) is NP-hard in general. Due to the fact \( C^{n+1} \subseteq S^{n+1} \cap N^{n+1} \), problem (CPP) naturally has the following SDP relaxation:

\[
\begin{align*}
\min & \quad \hat{Q} \cdot Y, \\
\text{s.t.} & \quad Ax = b, \ \text{diag}(AXAT) = b \odot b, \\
& \quad X_{ij} = 0, \ (i,j) \in \mathcal{E}, \\
& \quad Y = \begin{bmatrix} 1 & x^T \\ x & X \end{bmatrix} \in S^{n+1} \cap N^{n+1},
\end{align*}
\]  
(DNP)

which is a doubly nonnegative program (DNP) because the eigenvalues and components of matrix \( Y \) are both nonnegative. By the theory of interior point method (IPM), DNP can be solved in polynomial time to any fixed precision. However, due to the existence of doubly nonnegative constraint, the Newton system in every iteration of IPM is of size \( O(n^4) \). Hence, IPMs are limited to fairly small or medium
problem sizes in practice because the CPU time will increase dramatically as the problem size \( n \) goes larger. In recent years, alternating direction methods of multipliers (ADMMs) are developed to solve large-scale SDPs (ref. [15], [22] and [23]). In general, ADMMs decompose the augmented Lagrangian function into several subproblems, each of which is relatively easy to handle. In particular, several ADMMs are specially designed for DNP problem. For example, Lu et al. [17] proposed a primal ADMM for the DNP relaxations of a class of mixed integer quadratic programming problem, whose domain is separable for each \( x_i \). This special structure of feasible domain is utilized in the design of algorithm so that the solutions of decomposed subproblems have closed form. Burer [5] proposed an augmented Lagrangian method for DNP relaxation of completely positive programs. We highlight here that Burer used coordinate descent method for solving the decomposed subproblem from the augmented Lagrangian function. It has been shown that this procedure is actually equivalent to the primal ADMM in [17].

Now, we derive a property of problem (DNP), which allows us to establish the finite branch-and-bound algorithm in Section 4.

**Property 1.** Suppose \( \bar{Y} = \begin{bmatrix} 1 & \bar{x}^T \\ \bar{x} & \bar{X} \end{bmatrix} \) is an optimal solution of (DNP). If \( \bar{x} \in \mathcal{H} \), then \( \bar{x} \) is an optimal solution of problem (QPCC).

**Proof.** If \( \bar{x} \in \mathcal{H} \) and \( \bar{x} \) is part of an optimal solution of (DNP), then \( \bar{x} \) is a feasible solution of (QPCC). It follows that \( \tilde{Y} = \begin{bmatrix} 1 & \bar{x}^T \\ \bar{x} & \bar{X} \end{bmatrix} \) is a feasible solution of (DNP) and \( Q \cdot \tilde{Y} \leq \bar{Q} \cdot \tilde{Y} \). On the other hand, following from the fact \( \bar{X} - \bar{x} \bar{x}^T \in S^n_+ \), \( \bar{Q} \cdot (\tilde{Y} - \bar{Y}) = Q \cdot (\bar{X} - \bar{x} \bar{x}^T) \geq 0 \). Hence, \( \hat{Q} \cdot \bar{Y} \geq \bar{Q} \cdot \tilde{Y} = \tilde{x}^T \bar{Q} \bar{x} + q^T \tilde{x} \). Note that \( \hat{Q} \cdot \bar{Y} \) is a lower bound on the optimal value of (QPCC), it follows that \( \bar{x} \) is an optimal solution of (QPCC).

Based on Property 1, we can design a finite branch-and-bound algorithm in which \( \bar{x} \in \mathcal{H} \) is enforced gradually. The finiteness is ensured by fixing \( \bar{x}_i = 0 \) or \( \bar{x}_j = 0 \) whenever \( \bar{x}_i \bar{x}_j \neq 0 \) in the branching process.

3. **An augmented Lagrangian method.** In this section, we investigate the special structure of doubly nonnegative relaxation (DNP) in details and design the decomposition method for augmented Lagrangian function.

3.1. **Properties of (DNP).** Let \( M = [b, -A] \) be the coefficient matrix of the linear system \( Ax = b \), and \( N \) the be an orthonormal basis of the null space of \( M \). Let \( r \) be the rank of \( M \), then \( N \in \mathbb{R}^{(n+1) \times (m-r)} \), and \( N^T N = I \). The following property is useful in reformulating problem (DNP).

**Lemma 3.1** (Proposition 1 in [5]). Suppose \( Y = \begin{bmatrix} 1 & \bar{x}^T \\ \bar{x} & \bar{X} \end{bmatrix} \in S^{n+1}_+ \) and \( M = [b, -A] \). Then, the following three conditions are equivalent:

(i) \( Ax = b, \ diag(AXA^T) = b \circ b \).
(ii) \( MYM^T = 0 \).
(iii) \( MY = 0 \).

Lemma 3.1 allows us to restate problem (DNP) as the following formulation:

\[
\begin{align*}
\min & \quad Q \cdot X + q^T x, \\
\text{s.t.} & \quad MYM^T = 0,
\end{align*}
\]
GLOBALLY SOLVING QPCC VIA CPP

\[ X_{ij} = 0, \ (i, j) \in \mathcal{E}, \]
\[ Y = \begin{bmatrix} 1 \\ x^T \\ X \end{bmatrix} \in \mathcal{S}_n^{n+1} \cap \mathcal{N}_n^{n+1}. \]  
\text{(DNP-New)}

In the next subsection, we will design a decomposition method based on this new formulation.

3.2. Augmented Lagrangian and decomposition. Define the following closed and convex cone:

\[ \mathcal{K} = \{ Y \in \mathcal{S}_n^{n+1} | MYM^T = 0 \}. \tag{1} \]

Let \( \mathcal{K}^\ast \) be the dual cone of \( \mathcal{K} \). It is easy to verify that \( \mathcal{K} = \{ Y = NPNT | P \in \mathcal{S}_m^{m-r} \} \). Hence, for any given matrix \( R \in \mathcal{S}_n^{n+1} \), the projection of \( R \) into the cone \( \mathcal{K} \) is (ref. [21])

\[ \text{proj}_{\mathcal{K}}(R) = \text{proj}_{\mathcal{S}_m^{m-r}}(N^TRN)N^T, \tag{2} \]

and

\[ \text{proj}_{\mathcal{K}^\ast}(R) = R + \text{proj}_{\mathcal{K}}(-R). \tag{3} \]

Now, we are ready to derive the augmented Lagrangian function. Introducing an auxiliary variable \( Z \) and consider the following auxiliary problem:

\[
\begin{align*}
\min & \quad Q \bullet X + q^T x, \\
\text{s.t.} & \quad 0 \leq Y = \begin{bmatrix} 1 \\ x^T \\ X \end{bmatrix} \leq U, \\
& \quad X_{ij} = 0, \ (i, j) \in \mathcal{E}, \\
& \quad Z \in \mathcal{K}, \\
& \quad Z = Y,
\end{align*}
\]

(Auxiliary)

where matrix \( U = \begin{bmatrix} 1 & u^T \\ u & uu^T \end{bmatrix} \) with \( u \) being the upper bounds on \( x \) implied by the assumption that feasible region \( \mathcal{F} \) is bounded. The idea of the decomposition is to relax the constraint \( Z = Y \) and then apply the standard augmented Lagrangian method. We introduce the multiplier \( S \) and the penalty parameter \( \sigma > 0 \) to yield the augmented Lagrangian function

\[ L_{S,\sigma}(Y, Z) = \hat{Q} \bullet Y - S \bullet (Y - Z) + \frac{\sigma}{2} \| Y - Z \|^2_F, \tag{4} \]

where \( \| \cdot \|_F \) is the Frobenius norm. The associated subproblem is then

\[
\begin{align*}
\min & \quad L_{S,\sigma}(Y, Z) = \hat{Q} \bullet Y - S \bullet (Y - Z) + \frac{\sigma}{2} \| Y - Z \|^2_F, \\
\text{s.t.} & \quad 0 \leq Y = \begin{bmatrix} 1 \\ x^T \\ X \end{bmatrix} \leq U, \\
& \quad X_{ij} = 0, \ (i, j) \in \mathcal{E}, \\
& \quad Z \in \mathcal{K}, \\
& \quad Z = Y.
\end{align*}
\]

(AL-Sub)

After we get an optimal solution \( (Y, Z) \) of problem (AL-Sub), we update the multiplier \( S \) in the standard way [23]

\[ S \leftarrow \text{proj}_{\mathcal{K}^\ast}(S - \sigma(Y - Z)). \tag{5} \]

In fact, the constraint \( Z \in \mathcal{K} \) implies that \( S \) must be in \( \mathcal{K}^\ast \), otherwise the optimal value of problem (AL-Sub) is unbounded. In the next subsection, we will discuss how to solve problem (AL-Sub) in an approximate but efficient way, and the approach for calculating the lower bound of problem (DNP).
3.3. Solving the subproblem (AL-Sub) and calculating lower bound. As in [24], we adopt the block coordinate descent over $Y$ and $Z$ for solving problem (AL-Sub). When $Z$ is held constant, problem (AL-Sub) over $Y$ becomes

$$\begin{align*}
\min & \quad (\tilde{Q} - S - \sigma Z) \cdot Y + \frac{\sigma}{2} \|Y\|_F^2 \\
\text{s.t.} & \quad 0 \leq Y = \begin{bmatrix} 1 & x^T \\ x & X \end{bmatrix} \leq U, \\
& \quad X_{ij} = 0, \quad (i, j) \in E,
\end{align*}$$

(Y-Block)

where a constant term $\|Z\|_F^2$ is ignored in the objective function. Obviously, problem (Y-Block) can be immediately decomposed into $n(n+1)/2 - |E| - 1$ one-dimensional strictly convex quadratic problems of the form $\min_y \{\alpha y + \sigma y^2/2 | 0 \leq y \leq \mu\}$, whose solution is analytic. We remark that the upper bound $\mu$ is finite and positive $\sigma$ ensures the boundedness of each one-dimensional subproblem.

On the other hand, when $Y$ is held constant, problem (AL-Sub) over $Z$ becomes

$$\begin{align*}
\min & \quad (S - \sigma Y) \cdot Z + \frac{\sigma}{2} \|Z\|_F^2 \\
\text{s.t.} & \quad Z \in K,
\end{align*}$$

(Z-Block)

where a constant term $(\tilde{Q} - S) \cdot Y + \frac{\sigma}{2} \|Y\|_F^2$ is neglected. Completing the square and dividing by $\sigma$, the objective function of problem (Z-Block) is $\|Y - \sigma^{-1} S - Z\|_F^2/2$. So the optimal solution of this problem is just the projection of $Y - \sigma^{-1} S$ onto $K$, which can be calculated according to (2). The mathematical operations for completing this projection include matrix production and a projection onto $S^m_{m-r}$. In total, it requires $O(n^2(m-r) + (m-r)^3) = O(n^2(m-r))$. We point out that this is the most dominant computational burden in the proposed algorithm and we implemented it in C programming language and connected to MATLAB using MEX library.

In order to calculate a valid lower bound $v$ for problem (DNP), for a given multiplier $S \in K^*$, we consider the standard Lagrangian relaxation of problem (DNP-New), i.e., setting $\sigma$ to be 0 in problem (AL-Sub). In this case, problem (AL-Sub) is separable over $Y$ and $Z$. Moreover, problem (Z-Block) has optimal solution $Z = 0$ because $S \in K^*$. Hence, problem (Y-Block) with $\sigma = 0$ and $Z = 0$ can be decomposed into $O(n^2)$ one-dimensional linear optimization problems of the form $\min_y \{\alpha y | 0 \leq y \leq \mu\} = \min \{0, \alpha \mu\}$. In all, the total cost for calculating a lower bound $v$ requires $O(n^2)$ mathematical operations.

We state the augmented Lagrangian method for problem (DNP) as Algorithm 1. Some details are not specified here, such as how to initialize the parameter $\sigma$ and the termination criterion. The reason is that these strategies are closely related to the branch-and-bound method that will be adopted. Therefore, we only state a basic framework of the algorithm here and leave the discussion on the details in Section 4. We end this section with two remarks.

Remark 1. Algorithm 1 may not converge because problem (DNP) is not strictly feasible. However, we only need to solve problem (DNP) approximately to get a valid lower bound $v$ in an efficient way. Solving problem (DNP) exactly, especially in the first few nodes of the branch-and-bound tree, could be time consuming while its optimal value may not provide a tight lower bound of problem (CPP).

Remark 2. The formula (5) for updating $S$ can be further simplified as $S \leftarrow S - \sigma(Y - Z)$. Following from (3), formula (5) is equivalent to $(S - \sigma(Y - Z)) +$
Algorithm 1 Augmented Lagrangian Method for Problem (DNP)

**Input:** Data \((Q, q, A, b, E)\). Initial dual multiplier \(S_0\) and parameter \(\sigma_0\).

**Output:** An approximate solution \(Y\) and a valid lower bound \(v\) of problem (DNP).

1. Set \((S, \sigma) = (S_0, \sigma_0)\) and \(v = -\infty\). Initialize \(Z = 0\).
2. for \(k = 0, 1, \ldots\) do
3. Solve problem (Y-Block) to get \(Y\).
4. Solve problem (Z-Block) to get \(Z\).
5. Update \(S\) according to (5).
6. Update \(v\) by optimizing problem (Y-Block) with \(Z = 0\) and \(\sigma = 0\).
7. Update \(\sigma\) if necessary.
8. STOP, if termination criteria are met.
9. end for

\(\sigma \text{proj}_K((Y - \sigma^{-1} S) - Z)\). Note that, \(Z\) is the solution of problem (Z-Block), i.e., \(Z = \text{proj}_K((Y - \sigma^{-1} S))\). Hence, \(\text{proj}_K((Y - \sigma^{-1} S) - Z) = 0\) according to the standard convex analysis [21].

4. A branch-and-bound algorithm. In this section, we present implementation details about the branch-and-bound algorithm for globally solving problem (QPCC). We also prove the finiteness and correctness of the proposed algorithm under the mild assumption.

In each node, some variables of \(x_i\) are fixed to be zero, where \((i, j) \in E\) or \((j, i) \in E\) for some \(j \in \{1, \ldots, n\}\). The relaxation problem in this node is then

\[
\begin{align*}
\min \quad & Q \cdot Y \\
\text{s.t.} \quad & MYM^T = 0, Y \leq U, \\
& X_{ij} = 0, \forall (i, j) \in E, \\
& x_i = 0, \forall i \in I \\
& Y = \begin{bmatrix} 1 & x^T \end{bmatrix} X \in S_n^{n+1} \cap N_n^{n+1},
\end{align*}
\]

where \(I\) is the set of indices of the variables fixed to be zero. It is obvious that Property 1 and Algorithm 1 still work for the above problem. The upper bound \(U = \begin{bmatrix} 1 & u^T \end{bmatrix} u u^T\) is obtained in the preprocessing phase, in which a sequence of linear programs (LP) in the form of \(u_i = \arg \max \{x_i | Ax = b, x \geq 0\}\) are solved for each variable \(x_i, i = 1, \ldots, n\). We used warm start to speed up the solution of these LPs. Specifically, the row and column basis information of previous LP is used in the next one.

The practical performance of Algorithm 1 depends immensely on how \(\sigma\) and \(S\) is initialized. We discuss the rule for updating and initializing these parameters. The default initialization is \(\sigma_0 = \max_{i,j=1,\ldots,n} \{|Q_{ij}|, |q_i|\}\) and \(S_0 = 0\). We calculate the lower bound \(v\) every 10 iterations, and then update \(\sigma\) according to the rule proposed by Burer [5]:

\[
\sigma \leftarrow \left(1 + \frac{v - \bar{v}}{1 + |\bar{v}|}\right) \sigma \quad (6)
\]

where \(\bar{v}\) is the best lower bound found so far. If the ratio \(1 + (v - \bar{v})/(1 + |\bar{v}|)\) is nonpositive, then we do not update \(\sigma\). We terminate Algorithm 1 if (i) The relative change in the last 5 updates of \(v\) is less than \(10^{-5}\); (ii) the maximal iteration 10,000.
is achieved, or (iii) the relative error \( \|Y - Z\|_F / (\|Y\|_F + \|Z\|_F) \) is less than \( 10^{-5} \).

At the end of Algorithm 1, the optimal \( \sigma \) and \( S \) are saved as the initialization for the child nodes.

When \( I \) is a vertex cover of \( E \) (that is \( x_i x_j = 0 \) for all \( (i, j) \in E \)), the node becomes a leaf, which has to be fathomed for the seek of correctness of a branch-and-bound algorithm. The problem in the leaf node has the following form

\[
\begin{align*}
\min & \quad x^T \bar{Q} x + \bar{q}^T x \\
\text{s.t.} & \quad \bar{A} x = \bar{b}, x \geq 0,
\end{align*}
\]

(Leaf)

where \( \bar{Q}, \bar{q}, \bar{A} \) and \( \bar{b} \) are the data obtained by deleting corresponding rows and columns indexed by \( I \). Obviously, problem (Leaf) is a convex quadratic program, whose feasible region is a polyhedron with nonempty relative interior. In this case, we switch to use the interior-point method (IPM) to solve the leaf node exactly. The optimal value of problem (Leaf) provides a valid global upper bound (GUB) for (QPCC).

A node is fathomed if (i) the lower bound obtained by solving the relaxation problem is within a prespecified tolerance of GUB; (ii) the solution of relaxation problem satisfies the condition of Property 1; or (iii) the relaxation problem is infeasible.

If a node cannot be fathomed, then branching step involves enforcing the complementarity constraints \( x_i x_j = 0 \) for \( (i, j) \in E \) by fixing some variable to be zero. We choose a maximum violation rule to select the complementarity constraint to branch on. Specifically, for the current optimal solution \( \bar{x} \), we choose \((\tilde{i}, \tilde{j}) = \arg \max \{\bar{x}_i \bar{x}_j | (i, j) \in E, \bar{x}_i \bar{x}_j > 0\}\). Then, two children nodes are created: one enforces \( \bar{x}_i = 0 \) while the other one \( \bar{x}_j = 0 \).

More details about the branch-and-bound algorithm are stated as follows:

- When selecting a node to solve, we choose the one with the smallest lower bound.
- In each node, the MATLAB function \texttt{fmincon} is used to find a local optimal solution. The global upper bound GUB is then updated if the optimal value is less than GUB.
- We use the relative optimality condition for fathoming. That is, for a given tolerance \( \epsilon > 0 \), a node with lower bound \( v \) is fathomed if \( (\text{GUB} - v) / \max \{1, |\text{GUB}|\} < \epsilon \), and Algorithm 1 is stopped before the termination criteria are met.

Remark 3. The \texttt{fmincon} function in MATLAB can find a good upper bound most of the time, thus helping to obtain a good GUB and eventually reduce the number of nodes. Our preliminary computational results show that the number of nodes increases dramatically if \texttt{fmincon} is not used. Other efficient solution methods that can find a good feasible solution of problem (QPCC) could be alternatives of \texttt{fmincon} for updating the GUB.

We are now ready to formally state the proposed branch-and-bound method for problem (QPCC) as Algorithm 2. The main steps of the algorithm are presented here and the details of each step can be found in previous paragraphs.

**Theorem 4.1.** Algorithm 2 for (QPCC) is finite and correct.

**Proof.** The finiteness of Algorithm 2 is ensure by the branching rule because there are finite possible vertex covers for the set \( E \). In order to have a correct branch-and-bound algorithm, the relaxation problem should have no gap at any leaf node.
Algorithm 2 A Branch-and-Bound Algorithm for Globally Solving Problem (QPCC)

Input: Data \((Q, q, A, b, E)\).
Output: A global solution of \((QPCC)\).

1. **Preprocessing Step:** Calculate the upper bound for variable \(x\).
2. **Initialization Step:** The list \(L\) to be explored is initialized to contain the root node. Upper bound is set to \(+\infty\), and lower bound is set to \(-\infty\).
3. **while** \(L\) is not empty **do**
   4. **Node Selection Step:** Select and remove the best-first node from \(L\).
   5. **Bounding Step:** Solve the relaxation problem of the node by Algorithm 1. Update upper and lower bound if possible. If the node is not fathomed, go to the next step.
   6. **Branching Step:** Branch the node on the most violated complementarity constraint, generate two children nodes and add them to the list \(L\).
4. **end while**

This is ensured in Algorithm 2 because the problem in the leaf node is a quadratic convex program, which can be solved exactly when feasible.

5. **Numerical experiment.** In this section, we test the proposed algorithm on various types of instances. The algorithm has been implemented in MATLAB R2015a on a PC with Windows 7, 3.60GHz Intel Dual Core CPU processors and 4GB RAM.

5.1. **Instances on MacMPEC.** We first solved some QPCC instances from MacMPEC [16]. Note that the instances are not in the standard form we presented in this paper, thus we transformed them by adding some auxiliary variables and constraints. Table 1 displays the computational results for six instances. The first column “Id” shows the problem name on the web site, the second column shows the size of the transformed problem, the column “nodes” shows the nodes explored by the proposed algorithm, and the forth column “time” displays the total CPU time in seconds. We confirmed, by our approach, that the global optimal values given on the MacMPEC were correctly identified. Especially, we can solve the largest instance “flp4-4” on MacMPEC in about one minute.

| Id     | \((m, n, |E|)\) | nodes | time (sec.) |
|--------|----------------|-------|-------------|
| bilevel2 | \((29,13,12)\) | 13    | 4.86        |
| bilevel2m | \((9,21,8)\) | 5     | 1.74        |
| flp4-1 | \((60,190,80)\) | 3     | 8.41        |
| flp4-2 | \((110,270,110)\) | 1     | 15.21       |
| flp4-3 | \((170,380,140)\) | 1     | 30.03       |
| flp4-4 | \((250,550,200)\) | 1     | 67.71       |

5.2. **Random instances.** We also tested the proposed algorithm on randomly generated instances. QPECgen [13] is an instance generator for QPCC problems. It allows the user to control the properties of the generated QPCC instances and gives stationary points, which are often the global solutions of the instances. Table
contains the computational results for a collection of QPCC instances of small to medium size. For each given problem size, ten instances are generated, and the average number of explored nodes and average CPU time required by the proposed algorithm are displayed in the second and third column, respectively. The results clearly show that the proposed algorithm can efficiently solve the randomly generated instances in a reasonable time. It is worthwhile to point out that the average number of explored nodes is still small, even if \( n = 100 \) and \(|E| = 40\).

Table 2. Computation results for randomly generated QPCC Instances

| \((m, n, |E|)\) | Ave. nodes | Ave. CPU time (sec.) |
|----------------|------------|----------------------|
| \((4, 12, 3)\) | 3          | 1.21                 |
| \((15, 45, 10)\) | 5          | 9.43                |
| \((25, 55, 20)\) | 26         | 65.21               |
| \((30, 100, 40)\) | 121        | 310.66              |

5.3. Application to binary quadratic programs. The binary quadratic program (BQP) is defined by

\[
\min_{x} x^T Qx \\
\text{s.t. } x \in \{0, 1\}^n.
\]  

(BQP)

Here, the matrix \( Q \) is usually not positive semidefinite. This problem is notable for embracing a wide range of applications, including set partition, resource allocation, and job assignment problems. Although the form of BQP is simple, it has been proved to be NP-hard by Pardalos and Jha [20]. One of the solving methods for BQP is quadratic convexification reformulation (QCR) [3]. Specifically, by using the fact that \( x_i = x_i^2 \) for \( i = 1, \ldots, n \), BQP can be reformulated as

\[
\min_{x} x^T(Q + \Lambda)x - e^T\Lambda x \\
\text{s.t. } x \in \{0, 1\}^n,
\]  

(RBQP)

where \( \Lambda \) is a diagonal matrix and \( e \) is a vector of all ones. QCR tries to find a \( \Lambda \) such that \( Q + \Lambda \) is positive semidefinite and the continuous relaxation of (RBQP) provides the best lower bound for BQP. Lu and Guo [18] proposed another QCR method by choosing \( \Lambda \) such that the trace of \( \Lambda \) is minimized while keeping the positive semidefiniteness of \( Q + \Lambda \). Our preliminary experiment shows that the proposed algorithm works better with Lu and Guo’s QCR method. Thus, we adopted their QCR method in this numerical experiment. Note that problem (RBQP) can be transformed to the form of (QPCC) by introducing an auxiliary variable \( y = e - x \) and adding complementarity constraints \( x_iy_i = 0 \) for \( i = 1, \ldots, n \). The data sets used for the experiment are from the OR-Library [2]. All instances have 10% density and the elements of \( Q \) are integral values in the [-100, 100] interval. The problem name, optimal value, the number of explored nodes and CPU time are given in Table 3. The proposed algorithm can solve 8 out of 10 instances within half an hour. For the rest 2 instances (“bqp50-1” and “bqp50-10”), the algorithm terminated due to the time limit of 1,800 seconds, and the best upper bound, instead of optimal value, is reported in Table 3. It is notable that, for instance “bqp50-3”, the number of explored nodes is only 69 while it took 114 iterations for CPLEX 12.6 to solve the same instance.
6. Conclusion. In this paper, we propose a branch-and-bound algorithm based on the doubly nonnegative relaxation for quadratic program with complementarity constraints. The doubly nonnegative relaxation in each node is efficiently solved by an augmented Lagrangian method due to the fact that the Lagrangian function can be decomposed into two subproblems that are easy to solve. The computational results show that the proposed algorithm can solve problems of small and medium size within a reasonable amount of time, with a particular highlight in a few nodes for some instances.

Acknowledgments. Deng’s research has been supported by National Natural Science Foundation of China Grant #11501543 and University of Chinese Academy of Sciences Research Foundation for Young Faculty Grant #Y551037Y00, Y65201VY00 and #Y65302V1G4. Xing’s research has been supported by National Natural Science Foundation of China Grant #11371216 and #11571029. Tian’s research has been supported by National Natural Science Foundation of China Grant #11401485 and #71331004. Lu’s research has been supported by Fundamental Research Funds for the Central Universities #2017MS058.

REFERENCES

[1] L. Bai, J. E. Mitchell and J.-S. Pang, On convex quadratic programs with linear complementarity constraints, *Computational Optimization and Applications*, 54 (2013), 517–554.
[2] J. Beasley, OR-Library: distributing test problems by electronic mail, *Journal of the Operational Research Society*, 41 (1990), 1069–1072.
[3] A. Billionnet, S. Elloumi and M.-C. Plateau, Improving the performance of standard solvers for quadratic 0-1 programs by a tight convex reformulation: The QCR method, *Discrete Applied Mathematics*, 157 (2009), 1185–1197.
[4] S. Burer, On the copositive representation of binary and continuous nonconvex quadratic programs, *Mathematical Programming*, 120 (2009), 479–495.
[5] S. Burer, Optimizing a polyhedral-semidefinite relaxation of completely positive programs, *Mathematical Programming Computation*, 2 (2010), 1–19.
[6] Y.-L. Chang, J.-S. Chen and J. Wu, Proximal point algorithm for nonlinear complementarity problem based on the generalized fischer-burmeister merit function, *Journal of Industrial and Management Optimization*, 9 (2013), 153–169.
[7] P. J. Dickinson and L. Gijben, On the computational complexity of membership problems for the completely positive cone and its dual, *Computational Optimization and Applications*, 57 (2014), 403–415.
[8] M. C. Ferris and J. S. Pang, Engineering and economic applications of complementarity problems, *SIAM Review*, 39 (1997), 669–713.

[9] C. Hao and X. Liu, A trust-region filter-sqp method for mathematical programs with linear complementarity constraints, *Journal of Industrial and Management Optimization*, 7 (2011), 1041–1055.

[10] T. Hoheisel, C. Kanzow and A. Schwartz, Theoretical and numerical comparison of relaxation methods for mathematical programs with complementarity constraints, *Mathematical Programming*, 137 (2013), 257–288.

[11] J. Hu, J. E. Mitchell, J.-S. Pang, K. P. Bennett and G. Kunapuli, On the global solution of linear programs with linear complementarity constraints, *SIAM Journal on Optimization*, 19 (2008), 445–471.

[12] X. X. Huang, D. Li and X. Q. Yang, Convergence of optimal values of quadratic penalty problems for mathematical programs with complementarity constraints, *Journal of Industrial and Management Optimization*, 2 (2006), 287–296.

[13] H. Jiang and D. Ralph, QPECgen, a MATLAB generator for mathematical programs with quadratic objectives and affine variational inequality constraints, *Computational Optimization and Applications*, 13 (1999), 25–59.

[14] J. J. Júdice and A. Faustino, The linear-quadratic bilevel programming problem, *Information Systems and Operational Research*, 32 (1994), 87–98.

[15] S. Kim, M. Kojima and K.-C. Toh, A lagrangian-dnn relaxation: A fast method for computing tight lower bounds for a class of quadratic optimization problems, *Mathematical Programming*, 156 (2016), 161–187.

[16] S. Leyffer, MacMPEC: AMPL collection of mathematical problems with equilibrium constraints, 2015, URL http://wiki.mcs.anl.gov/leyffer/index.php/MacMPEC.

[17] C. Lu, W. Xing, S.-C. Fang and Z. Deng, Doubly non-negative relaxation solution based branch-and-bound algorithms for mixed integer quadratic programs, Working paper.

[18] C. Lu and X. Guo, Convex reformulation for binary quadratic programming problems via average objective value maximization, *Optimization Letters*, 9 (2015), 523–535.

[19] O. Mangasarian and S. Fromovitz, The Fritz John necessary optimality conditions in the presence of equality and inequality constraints, *Journal of Mathematical Analysis and Applications*, 17 (1967), 37–47.

[20] P. Pardalos and S. Iha, Complexity of uniqueness and local search in quadratic 0-1 programming, *Operations Research Letters*, 11 (1992), 119–123.

[21] T. Rockafellar, *Convex Analysis*, Princeton University Press, Princeton, N.J., 1970.

[22] J. Sun and S. Zhang, A modified alternating direction method for convex quadratically constrained quadratic semidefinite programs, *European Journal of Operational Research*, 207 (2010), 1210–1220.

[23] Z. Wen, D. Goldfarb and W. Yin, Alternating direction augmented lagrangian methods for semidefinite programming, *Mathematical Programming Computation*, 2 (2010), 203–230.

[24] X.-Y. Zhao, D.-F. Sun and K.-C. Toh, A Newton-CG augmented Lagrangian method for semidefinite programming, *SIAM Journal on Optimization*, 20 (2010), 1737–1765.

[25] J. Zhou, S.-C. Fang and W. Xing, Conic approximation to quadratic optimization with linear complementarity constraints, *Computational Optimization and Applications*, 66 (2017), 92–122.

Received August 2016; revised April 2017.

E-mail address: zhibindeng@ucas.ac.cn
E-mail address: tianye7272@gmail.com
E-mail address: lucheng1983@163.com
E-mail address: wxing@math.tsinghua.edu.cn