A provable better Branch and Bound method for a nonconvex integer quadratic programming problem

Wenxing Zhu

Department of Computer Science and Technology, Fuzhou University, Fuzhou 350002, P. R. China

Received 3 February 2004; received in revised form 18 July 2004
Available online 27 August 2004

Abstract

This paper presents a Branch and Bound method for a nonconvex integer quadratic programming problem with a separable objective function over a bounded box. For this problem, a special branch method is constructed, which has a property that if a box has been partitioned into $2^n$ sub-boxes, then at least one sub-box can be deleted. We analyze the complexity of the algorithm, and prove that it is better than that of the complete enumeration method in the worst case if the solution space is large enough.

© 2004 Elsevier Inc. All rights reserved.

Keywords: Nonconvex integer quadratic programming; Branch and Bound; Complete enumeration

1. Introduction

The Branch and Bound method, a general scheme, has been extensively used for solving many NP-hard discrete and combinatorial optimization problems. The effectiveness of the method is always demonstrated by massive numerical experiments. It is often understood that in the worst case the Branch and Bound method is as worse as the complete enumeration method. So in this paper we are interested in a question that if we can design a Branch and Bound method for an NP-hard problem which is provable better than the complete enumeration method in the worst case.

Research supported by the National Natural Science Foundation of China under Grant 10301009, and the Natural Science Foundation of Fujian Province under Grant A0310013.

E-mail address: wxzhu@fzu.edu.cn (W. Zhu).

0022-0000/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
Most combinatorial problems can be formulated as integer programming problems. However, most of integer programming problems are NP-hard [4]. Due to the inherent difficulty of integer programming problems, the Branch and Bound method has been used to solve them, e.g., the linear integer programming problem [5], the nonconvex integer quadratic programming problem [3] and some general integer programming problems [1]. However, to our knowledge, no theoretical result has been presented to demonstrate that the Branch and Bound method is better than the complete enumeration method in the worst case for these problems.

In this paper, we consider the following NP-hard nonconvex integer quadratic programming problem with a separable objective function

\[
(QP)_I \left\{ \begin{array}{ll}
\min & \sum_{i=1}^{n} (q_i x_i^2 - r_i x_i) \\
\text{s.t.} & a_j^T x - b_j \leq 0, \ j = 1, \ldots, m, \\
& x \in X \cap I^n,
\end{array} \right.
\]

where \( q_i, r_i, i = 1, 2, \ldots, n, b_j, j = 1, 2, \ldots, m \) are integers, \( a_j = (a_{j1}, a_{j2}, \ldots, a_{jn})^T, \ j = 1, 2, \ldots, m \) are integer vectors, \( x = (x_1, x_2, \ldots, x_n)^T, X = ([x_1^L, x_1^U] \times \cdots \times [x_n^L, x_n^U]) \) is a bounded box with \( x_i^L, x_i^U, i = 1, \ldots, n \) integers, and \( I^n \) is the set of integer points in Euclidean space \( R^n \). We design a special branch method for problem \((QP)_I\) in Section 2. The branch method has a property that once a box has been partitioned along every coordinate direction, at least one sub-box can be deleted. We present a prototype Branch and Bound method for problem \((QP)_I\), and analyze its complexity in Section 3, which is proved better than that of the complete enumeration method in the worst case if the width of the initial box \( X \) is large enough. At last, in Section 4 we give an implementable algorithm of the prototype Branch and Bound method and present two numerical examples to illustrate the algorithm.

2. Branch method

In this section, before using the Branch and Bound method to solve problem \((QP)_I\), we define a special partition technique, and analyze its properties.

Let \( S = X \cap \{x : a_j^T x - b_j \leq 0, \ j = 1, \ldots, m\} \).

We partition the box \( X \) along the \( i \)th coordinate direction as follows.

(1) If \( q_i \geq 0 \), then set \( x_i^{\text{mid}} = \lfloor \frac{x_i^L + x_i^U}{2} \rfloor \), where \( \lfloor \frac{x_i^L + x_i^U}{2} \rfloor \) is the largest integer less than or equal to \( \frac{x_i^L + x_i^U}{2} \), and partition the box \( X \) into two parts \( X_1, X_2 \) as follows:

\[
X_1 = [x_1^L, x_1^{mid}] \times \cdots \times [x_{i-1}^L, x_{i-1}^{mid}] \times [x_i^{mid}, x_i^U] \times [x_{i+1}^L, x_{i+1}^{mid}] \times \cdots \times [x_n^L, x_n^U],
\]

\[
X_2 = [x_1^L, x_1^{mid}] \times \cdots \times [x_{i-1}^L, x_{i-1}^{mid}] \times [x_i^{mid}, x_i^U] \times [x_{i+1}^L, x_{i+1}^{mid}] \times \cdots \times [x_n^L, x_n^U].
\]

Obviously, \( X \cap I^n = (X_1 \cap I^n) \cup (X_2 \cap I^n) \).

With this partition method, for \( q_i > 0 \), if \( \frac{r_i}{2q_i} \leq x_i^{\text{mid}} \), then

\[
\min_{x_i \in [x_i^{\text{mid}} + 1, x_i^U] \cap I^i} q_i x_i^2 - r_i x_i = q_i (x_i^{\text{mid}} + 1)^2 - r_i (x_i^{\text{mid}} + 1) \quad (1)
\]
and if \( \frac{r_i}{2q_i} \geq x_i^{\text{mid}} \), then
\[
\min_{x_i \in [x_i^L, x_i^{\text{mid}}] \cap I_i} q_i x_i^2 - r_i x_i = q_i (x_i^{\text{mid}})^2 - r_i x_i^{\text{mid}},
\]
for \( q_i = 0 \), if \( r_i \geq 0 \), then
\[
\min_{x_i \in [x_i^L, x_i^{\text{mid}}] \cap I_i} q_i x_i^2 - r_i x_i = q_i (x_i^{\text{mid}})^2 - r_i x_i^{\text{mid}}
\]
and if \( r_i \leq 0 \), then
\[
\min_{x_i \in [x_i^{\text{mid}}+1, x_i^U]} q_i x_i^2 - r_i x_i = q_i (x_i^{\text{mid}} + 1)^2 - r_i (x_i^{\text{mid}} + 1).
\]

(2) If \( q_i < 0 \) and \( \frac{r_i}{2q_i} \notin (x_i^L, x_i^U) \), then set \( x_i^{\text{mid}} = \lfloor \frac{x_i^L + x_i^U}{2} \rfloor \) and partition the box \( X \) into two parts \( X_1, X_2 \) as follows:
\[
X_1 = [x_1^L, x_1^U] \times \cdots \times [x_{i-1}^L, x_{i-1}^U] \times [x_i^L, x_i^{\text{mid}}] \times [x_{i+1}^L, x_{i+1}^U] \times \cdots \times [x_n^L, x_n^U],
\]
\[
X_2 = [x_1^L, x_1^U] \times \cdots \times [x_{i-1}^L, x_{i-1}^U] \times [x_i^{\text{mid}} + 1, x_i^U] \times [x_{i+1}^L, x_{i+1}^U] \times \cdots \times [x_n^L, x_n^U].
\]
Obviously, \( X \cap I^n = (X_1 \cap I^n) \cup (X_2 \cap I^n) \).

With this partition method, for \( q_i < 0 \), if \( \frac{r_i}{2q_i} \geq x_i^{\text{mid}} \), i.e., \( \frac{r_i}{2q_i} \geq x_i^U \), then
\[
\min_{x_i \in [x_i^{\text{mid}}+1, x_i^U]} q_i x_i^2 - r_i x_i = q_i (x_i^{\text{mid}} + 1)^2 - r_i (x_i^{\text{mid}} + 1)
\]
and if \( \frac{r_i}{2q_i} < x_i^{\text{mid}} \), i.e., \( \frac{r_i}{2q_i} \leq x_i^L \), then
\[
\min_{x_i \in [x_i^L, x_i^{\text{mid}}] \cap I_i} q_i x_i^2 - r_i x_i = q_i (x_i^{\text{mid}})^2 - r_i x_i^{\text{mid}}.
\]

(3) If \( q_i < 0 \) and \( \frac{r_i}{2q_i} \in (x_i^L, x_i^U) \), then set \( x_i^{\text{mid}} = \lfloor \frac{r_i}{2q_i} \rfloor \), and partition the box \( X \) into two parts \( X_1, X_2 \) as follows:
\[
X_1 = [x_1^L, x_1^U] \times \cdots \times [x_{i-1}^L, x_{i-1}^U] \times [x_i^L, x_i^{\text{mid}}] \times [x_{i+1}^L, x_{i+1}^U] \times \cdots \times [x_n^L, x_n^U],
\]
\[
X_2 = [x_1^L, x_1^U] \times \cdots \times [x_{i-1}^L, x_{i-1}^U] \times [x_i^{\text{mid}} + 1, x_i^U] \times [x_{i+1}^L, x_{i+1}^U] \times \cdots \times [x_n^L, x_n^U].
\]
Obviously, \( X \cap I^n = (X_1 \cap I^n) \cup (X_2 \cap I^n) \).

For the above partition technique, we have the following theorems.

**Theorem 1.** Without loss of generality, suppose that \( x_i^U - x_i^L \geq 1, i = 1, \ldots, n \), and suppose that \( \frac{r_i}{2q_i} \notin (x_i^L, x_i^U) \) for any \( q_i < 0, i = 1, \ldots, n \). Then if the box \( X = [x_1^L, x_1^U] \times \cdots \times [x_n^L, x_n^U] \) has been partitioned
along every coordinate direction by the defined partition technique, we get \(x_{\text{mid}}\) and \(2^n\) sub-boxes \(X_l\), \(l = 1, \ldots, 2^n\) as follows:

\[
x_{\text{mid}} = \left(\frac{x_1^L + x_1^U}{2}, \ldots, \frac{x_n^L + x_n^U}{2}\right)^T,
\]

\[
X_l \in \{[x_1^L, x_{\text{mid}}^1], [x_1^L + 1, x_1^U]\} \times \cdots \times \{[x_n^L, x_{\text{mid}}^n], [x_n^L + 1, x_n^U]\}.
\]

Then there exists a sub-box \(X_k\) with a vertex \(y = (y_1, \ldots, y_n)^T\), where \(y_i \in \{x_{\text{mid}}^i, x_{\text{mid}}^i + 1\}\) such that

\[
\min_{x \in X_k \cap I^n} \sum_{i=1}^{n} (q_i x_i^2 - r_i x_i) = \sum_{i=1}^{n} (q_i y_i^2 - r_i y_i).
\]

**Proof.** By the defined partition techniques and their analyzes (1)–(6), it is obvious that (7) is true. \(\square\)

**Theorem 2.** In Theorem 1, if \(y \in S\), then

\[
\min_{x \in X_k \cap I^n} \sum_{i=1}^{n} (q_i x_i^2 - r_i x_i) = \sum_{i=1}^{n} (q_i y_i^2 - r_i y_i).
\]

**Proof.** Obviously,

\[
\min_{x \in X_k \cap I^n} \sum_{i=1}^{n} (q_i x_i^2 - r_i x_i) \geq \min_{x \in X_k \cap I^n} \sum_{i=1}^{n} (q_i x_i^2 - r_i x_i) = \sum_{i=1}^{n} (q_i y_i^2 - r_i y_i).
\]

But by the assumption that \(y \in S\) and \(y \in X_k \cap I^n\), we have

\[
\sum_{i=1}^{n} (q_i y_i^2 - r_i y_i) \geq \min_{x \in X_k \cap I^n} \sum_{i=1}^{n} (q_i x_i^2 - r_i x_i).
\]

Hence Theorem 2 holds. \(\square\)

Theorem 2 means that if \(y\) is a feasible integer point, then \(y\) is a minimal solution of problem \((QP)_I\) over \(X_k\). Thus if we want to find a minimal solution of problem \((QP)_I\), then we only need to record \(y\), and delete \(X_k\), since it is not needed again.

**Theorem 3.** In Theorem 1, if \(y \notin S\), then there exists a sub-box \(X_k\) of \(X\) such that \(X_k \cap S = \emptyset\), i.e., \(X_k\) is infeasible.

**Proof.** Since \(S\) is a closed convex set, \(y \notin S\) indicates that there exists an \(x_s \in S\) such that

\[
(x - x_s)^T (y - x_s) \leq 0 \quad \text{for any } x \in S.
\]
Set \( c = y - x_s \). Obviously, \( c \neq 0 \), and
\[
c^T(x - x_s) \leq 0 \quad \text{for any } x \in S.
\]
Let \( H = \{ x \in \mathbb{R}^n : (c^T(x - x_s)) \leq 0 \} \), we have \( S \subseteq H \). Now translate the usual coordinate system to the point \( y \). Obviously, in the new coordinate system with the origin \( y \), the boxes \( X_l, l = 1, \ldots, 2^n \) generated in Theorem 1 are in different quadrants. Furthermore, suppose that the vector \( c \) points into a quadrant \( G \), then it holds that
\[
c^T(x - y) \geq 0 \quad \text{for any } x \in G.
\]
Thus, for any \( x \in G \),
\[
c^T(y - x_s) = c^T(x_i^{mid} - x_s + y - x_i^{mid}) = c^T c + c^T(y - x_i^{mid}) > 0.
\]
Therefore, \( G \subset \mathbb{R}^n \setminus H \), and \( G \cap S = \emptyset \). Hence the box located in \( G \), say \( X_k \), satisfies that \( X_k \cap S = \emptyset \).

By Corollary 1 and its proof, if \( y \notin \mathbb{R}^n \), and \( y \in X \), then there exists at least one index \( j \in \{1, 2, \ldots, m\} \) such that \( a_j^T y - b_j > 0 \), and there is a sub-box constructed by the partition method which is located in
\( \{ x \in R^n : a_j^T x - b_j > 0 \} \) and is infeasible. However, we need an efficient way to identify the sub-box located in \( \{ x \in R^n : a_j^T x - b_j > 0 \} \). In fact, this kind of sub-boxes can be characterized as follows.

Suppose that \( x_v \) is a vertex of a box \( Y_0 \), and the \( n \) edge directions of \( Y_0 \) starting from \( x_v \) are \( e_1, \ldots, e_n \), respectively, where \( e_i = (e_{i1}, \ldots, e_{in})^T, |e_{ii}| = 1, i \neq i, l = 1, 2, \ldots, n \). Furthermore, suppose that the \( i \)th edge length corresponding to \( e_i \) of the box \( Y_0 \) is \( l_i, i = 1, 2, \ldots, n \), and without loss of generality suppose that

\[
\begin{align*}
l_1 a_j^T e_1 < 0, & \ldots, l_k a_j^T e_k < 0, l_{k+1} a_j^T e_{k+1} \geq 0, \ldots, l_n a_j^T e_n \geq 0. \\
\end{align*}
\] (9)

Then we have the following theorem.

**Theorem 4.** The box \( Y_0 \) is contained in \( \{ x \in R^n : a_j^T x - b_j > 0 \} \) if and only if

\[
\sum_{i=1}^{k} l_i a_j^T e_i > b_j - a_j^T x_v, \quad a_j^T x_v > b_j.
\] (10)

**Proof.** Obviously, \( x_v + \sum_{i=1}^{l} l_i e_i, l = 1, 2, \ldots, n \) are vertices of the box \( Y_0 \). Thus \( Y_0 \) is contained in \( \{ x \in R^n : a_j^T x - b_j > 0 \} \) implies that

\[
a_j^T x_v - b_j > 0
\]
and

\[
a_j^T \left( x_v + \sum_{i=1}^{k} l_i e_i \right) - b_j > 0.
\]

So (10) holds.

Conversely, for any \( x \in Y_0 \), there exists \( t_i \) with \( 0 \leq t_i \leq 1, i = 1, \ldots, n \) such that

\[
x = x_v + \sum_{i=1}^{n} t_i l_i e_i
\]
and if (10) holds, then by (9) we have

\[
a_j^T e - b_j = a_j^T x_v + \sum_{i=1}^{n} t_i l_i a_j^T e_i - b_j \\
\geq a_j^T x_v + \sum_{i=1}^{k} t_i l_i a_j^T e_i - b_j \\
\geq a_j^T x_v + \sum_{i=1}^{k} l_i a_j^T e_i - b_j \\
> 0,
\]

which means that \( Y_0 \) is contained in \( \{ x \in R^n : a_j^T x - b_j > 0 \} \).

Hence Theorem 4 holds. \( \square \)
3. A prototype Branch-and-Bound algorithm

Now we present a prototype Branch-and-Bound algorithm for problem \((QP)_I\). We use the partition method presented in Section 2. For simplicity, the lower bound on the minimal value of problem \((QP)_I\) over a box \(X\) is set as the minimal value of problem \(\min_{x \in X \cap I^n} \sum_{i=1}^n (q_i x_i^2 - r_i x_i)\), although it might not be sharp and can be improved easily by the Lagrangian relaxation of problem \((QP)_I\). The algorithm is described as follows.

The prototype Branch and Bound algorithm

**Step 1:** Let \(L\) be a list, and set initially \(L = \{X\}\). Let \(f^*\) be the current minimal value found by this algorithm, and \(x^*\) be the corresponding minimal solution. Set initially \(f^* = +\infty\).

**Step 2:** For every box \(Y\) in the list \(L\), use the partition method defined to partition the box \(Y\) along every coordinate direction into \(2^n\) sub-boxes \(Y_l, l = 1, \ldots, 2^n\). Remove \(Y\) from list \(L\).

**Step 3:** For every obtained sub-box \(Y_l\) in Step 2.

3.1: Solve problem \(\min_{x \in Y_l \cap I^n} \sum_{i=1}^n (q_i x_i^2 - r_i x_i)\), and denote by \(f^*_1\) and \(x^*_1\) its minimal value and minimal solution respectively. If \(f^*_1 \geq f^*\), then delete \(Y_l\), since \(Y_l \cap I^n\) does not contain a solution lower than \(x^*\), otherwise if \(x^*_1\) is feasible, then delete \(Y_l\), set \(f^* = f^*_1\), and set \(x^* = x^*_1\). If \(Y_l\) cannot be deleted in this step, then go to Step 3.2.

3.2: Take a vertex \(x_v\) of \(Y_l\). If it is infeasible, then take all inequalities of the constraints of problem \((QP)_I\) violated by \(x_v\), check condition (10). Once satisfied, then \(Y_l\) is infeasible, and delete \(Y_l\).

**Step 4:** Enter the boxes undeleted in Step 3 into list \(L\).

**Step 5:** If list \(L\) is nonempty, then go to Step 2, otherwise stop the algorithm and output \(f^*\) and \(x^*\) as the minimal value and minimal solution of problem \((QP)_I\), respectively.

**Remark 1.** In Step 3.1 of the above algorithm, the problem \(\min_{x \in Y_l \cap I^n} \sum_{i=1}^n (q_i x_i^2 - r_i x_i)\) can be solved very efficiently since \(Y_l\) is a box and \(\sum_{i=1}^n (q_i x_i^2 - r_i x_i)\) is a separable function. In fact, we need \(O(n)\) basic arithmetic operations to find a minimal solution of problem \(\min_{x \in Y_l \cap I^n} \sum_{i=1}^n (q_i x_i^2 - r_i x_i)\).

**Remark 2.** If the above algorithm stops, and \(f^* = +\infty\), then problem \((QP)_I\) is infeasible.

**Remark 3.** Suppose that \(x_i^U - x_i^L \geq 1, i = 1, 2, \ldots, n\), and suppose that \(\frac{r_i}{2q_i} \notin (x_i^L, x_i^U)\), for all \(q_i < 0\), \(i \in \{1, 2, \ldots, n\}\). For the above Branch and Bound algorithm, if a box has been partitioned along every coordinate direction into \(2^n\) sub-boxes, then by Theorems 1–3, at least one sub-box can be deleted in Step 3.1 or in Step 3.2.

By the defined partition method and Theorems 1–3, it is obvious that the following theorem holds.

**Theorem 5.** The prototype Branch and Bound algorithm can terminate after finite steps and find an optimal solution of problem \((QP)_I\) or declare that problem \((QP)_I\) is infeasible.

Next we analyze the complexity of the above prototype Branch and Bound algorithm.
Without loss of generality, in the sequel we suppose that \( \frac{r_i}{2q_i} \notin (x^U_i, x^L_i) \), for all \( q_i < 0, i \in \{1, 2, \ldots, n\} \). For the defined partition method, we have the following result.

**Lemma 1.** Suppose that the maximum length of the edges of the initial box \( X \) is \( N \), \( N \) is an integer and \( N \leq 2^l - 1, l \geq 1 \), then by Step 2 of the prototype algorithm, with at most \( l \) partitions of \( X \) along all coordinate directions, we get all integer points in \( X \).

**Proof.** Without loss of generality, we prove Lemma 1 in one-dimensional case by the induction method.

In one-dimensional case, the box \( X \) is an interval, and without loss of generality suppose that \( X = [0, N] \).

For \( l = 1 \), \( N \leq 2^1 - 1 = 1 \). Obviously, in this case with one partition of \( X \) we get all integer points in \( X \), and Lemma 1 holds.

Now suppose that Lemma 1 is true for \( l = k \). Next we show that it also holds for \( l = k + 1 \).

The first partition of \([0, N]\) gets two intervals: \([0, \lfloor \frac{N}{2} \rfloor] \), \([\lfloor \frac{N}{2} \rfloor + 1, N]\). Since \( N \leq 2^{k+1} - 1 \), we have \( \lfloor \frac{N}{2} \rfloor \leq 2^{k-1} \) and \( N - (\lfloor \frac{N}{2} \rfloor + 1) \leq 2^{k-1} \). Thus by the assumption on \( l = k \), we need at most \( k \) partitions of \([0, \lfloor \frac{N}{2} \rfloor]\) and \([\lfloor \frac{N}{2} \rfloor + 1, N]\) to get all integer points in them. So in all, we need at most \( k + 1 \) partitions of \( X \) to get all integer points in it.

So by theory of the induction method, Lemma 1 holds.  

Next we analyze the complexity of the prototype Branch and Bound algorithm as follows.

**Theorem 6.** Suppose that any edge length of the initial box \( X \) is \( N \), \( N \) is an integer and \( N \leq 2^l - 1, l \geq 1 \), and suppose that \( n \geq 2 \). Then to solve problem \((QP)_I\), the algorithm needs at most \((2n - 1)^l O(mn)\) basic arithmetic operations.

**Proof.** During the first iteration of the algorithm, we partition the initial box \( X \) along all coordinate directions into \( 2^n \) sub-boxes by the defined partition method. By Theorems 1–3, at least one sub-box can be deleted, and at most \( 2^n - 1 \) sub-boxes are kept. So during the second iteration, we partition at most \( 2^n - 1 \) sub-boxes into at most \( 2^n (2^n - 1) \) sub-boxes, and for the same reason, at least \( 2^n - 1 \) sub-boxes are deleted. By Lemma 1, the algorithm gets all integer points in \( X \) after \( l \) partitions of \( X \). So in all, after the algorithm terminates, the number of considered boxes (including single integer points) is not more than

\[
1 + 2^n + 2^n (2^n - 1) + \cdots + 2^n (2^n - 1)^{l-1} \\
= \frac{2^{n-1}(2^n - 1)^l - 1}{2^n - 1} \leq \frac{2^{n-1}(2^n - 1)^l}{2^n - 1} \\
= \left(1 + \frac{1}{2^n - 1}\right)(2^n - 1)^l \\
\leq 2(2^n - 1)^l.
\]
Over any one of the sub-boxes, say $Y_k$, we must finish Step 3. The main work of the algorithm is in Step 3, which is to

1. solve problem $\min_{x \in Y_k \cap I^n} \sum_{i=1}^n (q_i x_i^2 - r_i x_i)$,
2. check if a vertex $x_v$ is infeasible,
3. check condition (10).

To solve problem $\min_{x \in Y_k \cap I^n} \sum_{i=1}^n (q_i x_i^2 - r_i x_i)$, we need $O(n)$ basic arithmetic operations since $Y_k$ is a box and $\sum_{i=1}^n (q_i x_i^2 - r_i x_i)$ is a separable function. To check if a vertex $x_v$ is infeasible, we need $O(mn)$ basic arithmetic operations. To check condition (10), we must calculate $l_i a_j^T e_i$, $i = 1, 2, \ldots, n$, and find all negative terms among them, which can be finished with $O(n)$ basic arithmetic operations, since $e_i = (e_{i1}, \ldots, e_{in})^T$, $|e_{ii}| = 1$, $e_{ik} = 0$, $k \neq i$, $k = 1, 2, \ldots, n$. Hence to check condition (10) we only need $O(n)$ basic arithmetic operations. So over one box, the algorithm does at most $O(mn)$ basic arithmetic operations.

Therefore, to solve problem $(QP)_I$, the algorithm needs at most $(2^n - 1)^l O(mn)$ basic arithmetic operations.

However, is the prototype Branch and Bound algorithm better than the complete enumeration method? In fact, we have the following result.

**Theorem 7.** Suppose that any edge length of the initial box $X$ is $N$, $N = 2^l - 1$, $l \geq 1$, and suppose that $n \geq 2$. If $l$ is large enough, then the prototype Branch and Bound algorithm is better than the complete enumeration method in the worst case.

**Proof.** Under the assumption of this theorem, the box $X$ has $(2^l)^n$ integer points. For every integer point, the complete enumeration method must first check if it is feasible, which can be done with $O(mn)$ basic arithmetic operations. If it is feasible, the method must calculate the value of the objective function $\sum_{i=1}^n (q_i x_i^2 - r_i x_i)$ at this point, which can be done with $O(n)$ basic arithmetic operations. So the complete enumeration method needs $2^n l O(mn)$ basic arithmetic operations to solve problem $(QP)_I$.

By Theorem 6, the prototype Branch and Bound algorithm needs at most $(2^n - 1)^l O(mn)$ basic arithmetic operations to solve problem $(QP)_I$. Since

$$\lim_{l \to +\infty} \frac{(2^n - 1)^l O(mn)}{2^n l O(mn)} = 0,$$

we can conclude that the prototype Branch and Bound algorithm is better than the complete enumeration method in the worst case.

4. **Implementable algorithm**

The prototype Branch and Bound algorithm is not suitable for practical implementation, since it partitions a box into $2^n$ sub-boxes. For practical considerations, a box should be partitioned into several sub-boxes, not $2^n$ sub-boxes. So the prototype Branch and Bound algorithm can be modified as follows.
Implementable algorithm

Step 1: Let \( L \) be a list, and set initially \( L = \{ X \} \). Let \( f^* \) be the current minimal value found by this algorithm, and \( x^* \) be the corresponding minimal solution. Set initially \( f^* = +\infty \).

Step 2: Denote by \( Y \) the box with the maximum width in list \( L \). Choose an edge of \( Y \) with the maximum length, and by the partition technique, partition \( Y \) into two sub-boxes \( Y_1, Y_2 \). Remove \( Y \) from list \( L \).

Step 3: For every \( Y_l, l = 1, 2 \),

Step 3.1: Solve problem \( \min_{x\in Y_l \cap I^n} \sum_{i=1}^{n} (q_i x_i^2 - r_i x_i) \), and denote by \( f_l^* \) and \( x_l^* \) its minimal value and minimal solution, respectively. If \( f_l^* \geq f^* \), then delete \( Y_l \), since \( Y_l \cap I^n \) does not contain a solution lower than \( x^* \), otherwise if \( x_l^* \) is feasible, then delete \( Y_l \), set \( f^* = f_l^* \), and set \( x^* = x_l^* \). If \( Y_l \) cannot be deleted in this step, then go to Step 3.2.

Step 3.2: Take a vertex \( x_v \) of \( Y_l \). If it is infeasible, then choose an inequality of the constraints of problem \((QP)_l\) violated by \( x_v \), check condition (10). If satisfied, then \( Y_l \) is infeasible, and delete \( Y_l \).

Step 4: Enter the boxes undeleted in Step 3 into list \( L \).

Step 5: If list \( L \) is nonempty, then go to Step 2, otherwise stop the algorithm and output \( f^* \) and \( x^* \) as the minimal value and minimal solution of problem \((QP)_l\), respectively.

To illustrate the above algorithm, we use it to solve the following 2 testing problems in [2, pp. 110, 121].

**Problem 1.**

\[
\begin{align*}
\min P &= -16x_1^2 - 18x_1 - 7x_2^2 + 12x_2, \\
\text{s.t.} & \quad 6x_1 - x_2 \leq 100, \\
& \quad x_1 + 2x_2 \leq 150, 000, \\
& \quad 0 \leq x_i \leq 99, 999, i = 1, 2 \\
& \quad x_i : \text{integer, } i = 1, 2.
\end{align*}
\]

Conley [2] used the Monte Carlo approach to examine 1,110,000 points and found the solution \( x_1 = 0, x_2 = 75, 000 \), and minimum \( P = -39, 374, 100, 000 \). Our algorithm examines 179 boxes (including single integer points) and finds the same solution.

**Problem 2.**

\[
\begin{align*}
\min P &= -x_1^2 - x_2^2 - 3x_3^2 - 4x_4^2 - 2x_5^2 + 8x_1 + 2x_2 + 3x_3 + x_4 + 2x_5, \\
\text{s.t.} & \quad x_1 + x_2 + x_3 + x_4 + x_5 \leq 400, \\
& \quad x_1 + 2x_2 + 2x_3 + x_4 + 6x_5 \leq 800, \\
& \quad 2x_1 + x_2 + 6x_3 \leq 200, \\
& \quad x_3 + x_4 + 5x_5 \leq 200, \\
& \quad 0 \leq x_i \leq 99, i = 1, \ldots, 5, \\
& \quad x_i : \text{integer, } i = 1, \ldots, 5.
\end{align*}
\]

Conley [2] used the Monte Carlo approach to examine 1,500,000 points and found the solution \( x_1 = 48, x_2 = 92, x_3 = 0, x_4 = 98, x_5 = 17 \) and minimum \( P = -49, 062 \).

Our algorithm examines 6327 boxes (including single integer points) and finds the solution \( x_1 = 50, x_2 = 99, x_3 = 0, x_4 = 99, x_5 = 20 \), and minimum \( P = -51, 568 \).
5. Discussions and conclusions

Basing on a special partition method, this paper has presented a Branch and Bound algorithm for a nonconvex integer quadratic programming problem with a separable objective function. Although the lower bound presented is not sharp, the algorithm is proved in the worst case better than the complete enumeration method if the solution space is large enough. Thus it is shown theoretically that the Branch and Bound method can be better than the complete enumeration method in the worst case. The algorithm presented in this paper can be improved empirically with tighter lower bounds, which can be obtained by the Lagrangian relaxation method. Future research along the direction of this paper is to present provable and empirical better Branch and Bound algorithms for the 0–1 quadratic programming problem, the general nonconvex integer quadratic programming problem and other NP-hard combinatorial optimization problems.

References