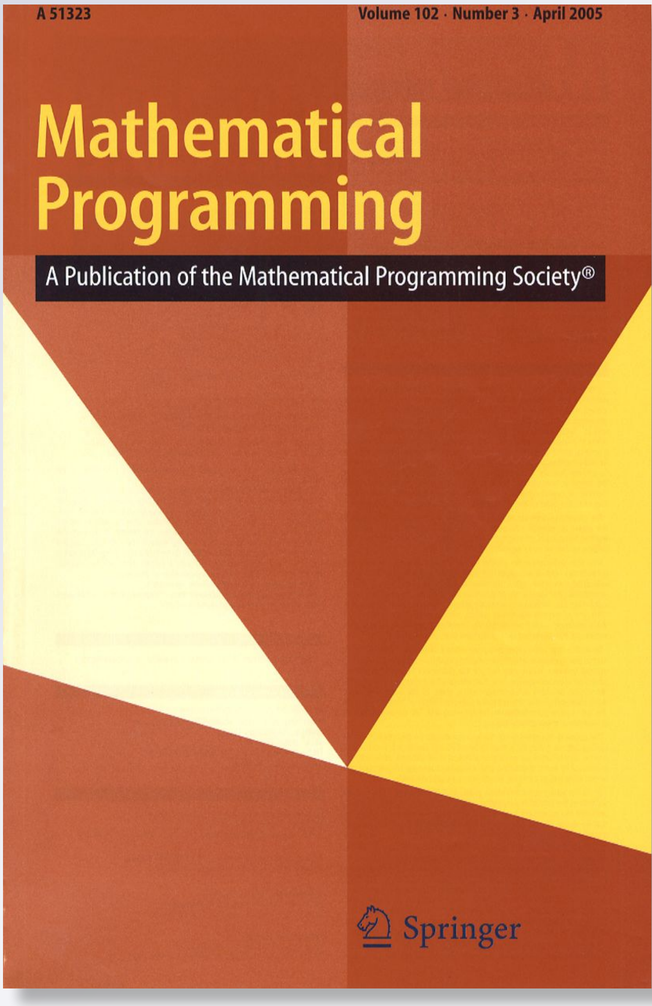
Two-stage quadratic integer programs with stochastic right-hand sides

Osman Y. Özaltın, Oleg A. Prokopyev & Andrew J. Schaefer

Mathematical Programming
A Publication of the Mathematical Optimization Society

ISSN 0025-5610
Volume 133
Combined 1-2

Math. Program. (2012) 133:121-158
DOI 10.1007/s10107-010-0412-4



Math. Program., Ser. A (2012) 133:121–158
DOI 10.1007/s10107-010-0412-4

FULL LENGTH PAPER

Two-stage quadratic integer programs with stochastic right-hand sides

Osman Y. Özaltın · Oleg A. Prokopyev · Andrew J. Schaefer

Received: 18 September 2009 / Accepted: 7 September 2010 / Published online: 1 October 2010

Abstract We consider two-stage quadratic integer programs with stochastic right-hand sides, and present an equivalent reformulation using value functions. We propose a two-phase solution approach. The first phase constructs value functions of quadratic integer programs in both stages. The second phase solves the reformulation using a global branch-and-bound algorithm or a level-set approach. We derive some basic properties of value functions of quadratic integer programs and utilize them in our algorithms. We show that our approach can solve instances whose extensive forms are hundreds of orders of magnitude larger than the largest quadratic integer programming instances solved in the literature.

Keywords Stochastic integer programming · Quadratic integer programming · Value functions · Superadditive duality

Mathematics Subject Classification (2000) 90C15 · 90C20 · 90C26 · 90C31

O. Y. Özaltın · O. A. Prokopyev (✉) · A. J. Schaefer
Department of Industrial Engineering, University of Pittsburgh, 1048 Benedum Hall, Pittsburgh, PA 15261, USA
e-mail: prokopyev@engr.pitt.edu

O. Y. Özaltın
e-mail: oyo1@pitt.edu

A. J. Schaefer
e-mail: schaefer@engr.pitt.edu

1 Introduction

We consider the following class of two-stage quadratic integer programs with stochastic right-hand sides:

$$(P1):\quad \max \quad \frac{1}{2}x^T \Lambda x + c^T x + \mathbb{E}_\omega \mathbf{Q}(x,\omega) \tag{1a}$$

$$\text{subject to} \quad x \in \mathbb{X}, \tag{1b}$$

where $\mathbb{X} = \{x \in \mathbb{Z}_+^{n_1} \mid Ax \leq b\}$ and,

$$\mathbf{Q}(x,\omega) = \max \quad \frac{1}{2}y^T \Gamma y + d^T y \tag{2a}$$

$$\text{subject to} \quad Wy \leq h(\omega) - Tx, \tag{2b}$$

$$y \in \mathbb{Z}_+^{n_2}. \tag{2c}$$

The random variable ω from probability space $(\Omega, \mathcal{F}, \mathcal{P})$ describes the realizations of uncertain parameters, known as *scenarios*. The numbers of constraints and decision variables in stage i are m_i and n_i, respectively, for $i = 1, 2$. The first-stage objective vector $c \in \mathbb{R}^{n_1}$, right-hand side vector $b \in \mathbb{R}^{m_1}$ and the second-stage objective vector $d \in \mathbb{R}^{n_2}$ are known column vectors. The first-stage constraint matrix $A \in \mathbb{R}^{m_1 \times n_1}$, technology matrix $T \in \mathbb{R}^{m_2 \times n_1}$ and recourse matrix $W \in \mathbb{R}^{m_2 \times n_2}$ are all deterministic. Furthermore, $\Lambda \in \mathbb{R}^{n_1 \times n_1}$ and $\Gamma \in \mathbb{R}^{n_2 \times n_2}$ are known, and possibly indefinite, symmetric matrices. The stochastic component consists of only $h(\omega) \in \mathbb{R}^{m_2}\ \forall \omega \in \Omega$.

The *extensive form* formulation of $(P1)$ is given by:

$$\max \quad \frac{1}{2}x^T \Lambda x + c^T x + \mathbb{E}_\omega \left[\frac{1}{2}y(\omega)^T \Gamma y(\omega) + d^T y(\omega)\right] \tag{3a}$$

$$\text{subject to} \quad x \in \mathbb{X}, \tag{3b}$$

$$Wy(\omega) \leq h(\omega) - Tx \qquad \forall \omega \in \Omega, \tag{3c}$$

$$y(\omega) \in \mathbb{Z}_+^{n_2} \qquad \forall \omega \in \Omega. \tag{3d}$$

In this paper we make the following assumptions:

A1 The random variable ω follows a discrete distribution with finite support.
A2 The first-stage feasibility set $\mathbb{X} = \{x \in \mathbb{Z}_+^{n_1} \mid Ax \leq b\}$ is nonempty and bounded.
A3 $\mathbf{Q}(x,\omega)$ is finite for all $x \in \mathbb{X}$ and $\omega \in \Omega$.
A4 The first-stage constraint matrix A, technology matrix T and recourse matrix W are all integral, i.e. $A \in \mathbb{Z}^{m_1 \times n_1}$, $T \in \mathbb{Z}^{m_2 \times n_1}$, $W \in \mathbb{Z}^{m_2 \times n_2}$.

Assumption **A1** is justified by Schultz [49], who showed that the optimal solution to any stochastic program with continuously distributed ω can be approximated within any desired accuracy using a discrete distribution. Assumption **A2** and integrality restrictions in the first stage ensure that $\mathbb{X}$ is a finite set. Assumption **A3** ensures that

$\mathbf{Q}(x, \omega)$ is feasible for all $x \in \mathbb{X}$ and $\omega \in \Omega$, i.e. relatively complete recourse [58]. Assumption **A4** is not too restrictive in a sense, as any rational matrix can be converted to an integral one. Most of the stochastic programming studies in the literature make assumptions similar to **A1**–**A3** [9,16,35,51] and **A4** [35]. Without loss of generality, we also assume that $b \in \mathbb{Z}^{m_1}$ and $h(\omega) \in \mathbb{Z}^{m_2}\ \forall \omega \in \Omega$, as A, T and W are all integer matrices. Note that all of the undesirable properties of stochastic integer programs, e.g. discontinuity and nonconvexity of $\mathbf{Q}(x, \omega)$, still exist in $(P1)$.

We reformulate $(P1)$ using the value functions of the first- and second-stage quadratic integer programs. The advantage of this reformulation is that it is relatively insensitive to the number of variables and scenarios. In the first phase of our solution approach, we construct the value functions in both stages. In the second phase, we use a global branch-and-bound algorithm or a level-set approach to optimize $(P1)$ over the set of feasible first-stage right-hand sides.

Our approach can solve very large instances of $(P1)$ as measured by the size of the extensive form. However, it is sensitive to the number of constraints in each stage and the magnitude of $h(\omega)$. Note that the number of quadratic integer programs that must be solved when constructing the value function grows exponentially in the number of constraints. A major contribution of this paper is to propose algorithms that can mitigate the effect of this exponential growth to some extent by exploiting the properties of value functions. Specifically, our approach can handle instances of $(P1)$ that have up to seven constraints in each stage.

The remainder of this paper is organized as follows. In Sect. 2, we review the literature on quadratic integer programming and stochastic integer programming. In Sect. 3, we present a value function reformulation of $(P1)$. In Sect. 4, we present a global branch-and-bound algorithm and a level-set approach to optimize the reformulation over the set of feasible first-stage right-hand sides. In Sect. 5, we identify various properties of value functions of quadratic integer programs, which are subsequently exploited in our algorithms. In Sect. 6, we propose four algorithms to construct the value function of a quadratic integer program. In Sect. 7, we discuss the details of our implementation and present computational results. We conclude and give future research directions in Sect. 8.

2 Literature review

2.1 Quadratic integer programming

Quadratic integer programs (QIPs) have been extensively studied, e.g. the quadratic assignment problem [36], the quadratic knapsack problem [22] and the discrete version of the bilinear programming problem [3].

Linearization is widely used for solving 0–1 QIPs [1,4,2,15,24,44,45,57]. The original problem is transformed into an equivalent linear mixed-integer program by introducing new variables and additional constraints. Major drawbacks of this method is the substantial increase of the problem size and the weakness of the LP relaxation [1,4]. There are various branch-and-bound [5,7,10,14,21,30,39,40,46,48,56] and cutting plane [6,11,27,28,37,38,47] algorithms proposed for general and 0–1

QIPs. These methods often rely on some simplifying assumptions, e.g. unconstrained 0–1 problems [30,46], or convex and separable objective functions [14,48].

Value functions of linear integer programs have been considered in [13,31–33,59]. However, the literature on QIP value functions is very sparse. Sensitivity of definite QIPs is studied in [18,25]. Granot and Skorin-Kapov [25] extends some of the linear integer programming proximity results derived in Schrijver et al. [17] to QIPs. Bank and Hansel [12] investigates the stability of indefinite mixed-integer quadratic programs. In terms of numerical and theoretical results of parameterized QIPs, an early computational study by McBride and Yormark [41] considers a sequence of 0–1 QIPs parameterized over the right-hand side of a single constraint. More recently, Dua et al. [20] develops a global optimization algorithm for solving a general class of nonconvex mixed-integer programs parameterized over the right-hand sides of a set of possibly nonlinear constraints. Their proposed algorithmic approach utilizes convex under- and over-estimators within a generic branch-and-bound algorithm.

2.2 Stochastic integer programming

Stochastic programs have many applications, including supply chain network design [53], telecommunications [34,52], server location [43] and dynamic capacity acquisition [8]. Imposing integrality restrictions on the second-stage variables increases the problem complexity significantly as the expected recourse function becomes nonconvex and discontinuous in general [55]. In the literature, algorithms developed for solving general stochastic programs with integer recourse utilize cutting planes and/or branch-and-bound techniques in combination with decomposition methods that exploit block-separability of the underlying problem structure. Here we describe two papers that are most closely related to our work. We refer the reader to Klein Haneveld and van der Vlerk [26] or Schultz [50] for detailed surveys.

Ahmed et al. [9] considers two-stage stochastic programs with discrete probability distributions, mixed-integer first-stage and pure-integer second-stage problems. A variable transformation is applied to make the discontinuities of the expected recourse function orthogonal to the variable axes. This structure is exploited through a rectangular branching strategy. Then a bounding strategy is employed to obtain the value function of the second-stage integer program in the absence of discontinuities. Finiteness of the method is established within a bounded search domain.

Kong et al. [35] considers a class of stochastic programs with stochastic right-hand sides, pure-integer first- and second-stage problems that have linear objective functions in both stages, i.e. $\Lambda = \Gamma = \mathbf{0}$ in $(P1)$. Similar to Ahmed et al. [9], their approach is based on an equivalent variable transformation that uses the value functions in both stages. Moreover, superadditive duality properties are exploited to characterize value functions efficiently. In contrast to [9], the value functions of both stages are calculated in advance, which are then utilized within a global branch-and-bound algorithm or an implicit exhaustive search procedure. The extensive forms of the stochastic linear integer programs that are solved by Kong et al. [35] are the largest ones reported in the literature so far. Our solution approach extends that of Kong et al. [35], in that we consider the more general problem of stochastic quadratic integer programs.

3 Value function reformulation

We reformulate $(P1)$ using the value functions of QIPs in both stages. Let $\mathbf{B}^1$ denote the set of vectors $\beta_1 \in \mathbb{R}^{m_2}$ such that there exists $x \in \mathbb{X}$ satisfying $\beta_1 = Tx$, i.e. $\mathbf{B}^1 = \{\beta_1 \in \mathbb{R}^{m_2} \mid \exists x \in \mathbb{X}, \beta_1 = Tx\}$, where $\mathbb{X} \subseteq \mathbb{Z}_+^{n_1}$ is the first-stage feasibility set. Furthermore, let $\mathbf{B}^2$ denote the set of vectors $\beta_2 \in \mathbb{R}^{m_2}$ such that there exists $\beta_1 \in \mathbf{B}^1$ and $\omega \in \Omega$ satisfying $\beta_2 = h(\omega) - \beta_1$, i.e. $\mathbf{B}^2 = \bigcup_{\beta_1 \in \mathbf{B}^1} \cup_{\omega \in \Omega} \{h(\omega) - \beta_1\}$. Note that all vectors in $\mathbf{B}^1$ are integral since $T \in \mathbb{Z}^{m_2 \times n_1}$. Together with the condition that $h(\omega) \in \mathbb{Z}^{m_2}\ \forall \omega \in \Omega$, all vectors in $\mathbf{B}^2$ are also integral.

For any $\beta_1 \in \mathbb{Z}^{m_2}$, we define the first-stage value function of $(P1)$ as:

$$\psi(\beta_1) = \max\left\{\frac{1}{2}x^T\Lambda x + c^T x \mid x \in S_1(\beta_1)\right\}, \quad S_1(\beta_1) = \{x \in \mathbb{X} \mid Tx \le \beta_1\}. \quad (4)$$

Note that the condition $Tx = \beta_1$ in the definition of $\mathbf{B}^1$ is replaced by $Tx \le \beta_1$ in (4). This is justified by nondecreasing property of the value function (see Proposition 8 in Sect. 5.2).

Next, for any $\beta_2 \in \mathbb{Z}^{m_2}$, we define the second-stage value function of $(P1)$ as:

$$\phi(\beta_2) = \max\left\{\frac{1}{2}y^T\Gamma y + d^T y \mid y \in S_2(\beta_2)\right\}, \quad S_2(\beta_2) = \left\{y \in \mathbb{Z}_+^{n_2} \mid Wy \le \beta_2\right\}. \quad (5)$$

We use $\psi(\cdot)$ and $\phi(\cdot)$ to reformulate $(P1)$ as:

$$(P2): \quad \max\left\{\psi(\beta) + \mathbb{E}_\omega \phi(h(\omega) - \beta) \mid \beta \in \mathbf{B}^1\right\}. \quad (6)$$

The variables β in $(P2)$ are known as the *tender variables* [9,35]. Instead of searching in $\mathbb{X}$, we search in the space of tender variables to obtain a global optimal solution. Theorem 1 establishes the correspondence between the optimal solutions of (P1) and (P2), and is similar to Theorem 3.2 in Ahmed et al. [9].

Theorem 1 *Let β^* be an optimal solution to (P2). Then, $\hat{x} \in argmax\left\{\frac{1}{2}x^T\Lambda x + c^T x \mid x \in S_1(\beta^*)\right\}$ is an optimal solution to (P1). Furthermore, the optimal values of the two problems are equal.*

Next, we present a global branch-and-bound algorithm and a level-set approach to optimize (P2) over the set of feasible first-stage right-hand sides given the value functions of QIPs in both stages. These two methods motivate us to study the properties of the QIP value function in Sect. 5, which are subsequently exploited in our algorithmic developments in Sect. 6.

4 Finding the optimal tender

The first method is a global branch-and-bound algorithm in which bounds are derived for hyper-rectangular partitions of $\mathbf{B}^1$. The second method is a level-set approach that evaluates the objective function in (P2) only for a subset of $\mathbf{B}^1$.

4.1 A global branch-and-bound algorithm

We propose a global branch-and-bound algorithm based on the framework described in Horst and Tuy [29]. The algorithm partitions $\mathbf{B}^1$ into hyper-rectangles $\mathcal{P}^k$. Each hyper-rectangle is associated with a subproblem of the form

$$f^k = \max\left\{\psi(\beta) + \mathbb{E}_\xi \phi(h(\omega) - \beta) \mid \beta \in \mathcal{P}^k \cap \mathbb{Z}^{m_2}\right\},$$

a lower bound $\mu^k \leq f^k$, and an upper bound $\upsilon^k \geq f^k$. A hyper-rectangle k is fathomed if $\mu^k \geq \upsilon^k$. We denote the list of unfathomed hyper-rectangles by $\mathcal{M}$.

Algorithm 5. A global branch-and-bound algorithm to solve (P2).

Step 0: (Initialization) Construct a hyper-rectangle $\mathcal{P}^0 := \left[\lambda^0, \eta^0\right] = \Pi_{i=1}^{m_2}\left[\lambda_i^0, \eta_i^0\right]$ such that $\mathbf{B}^1 \subseteq \mathcal{P}^0 \cap \mathbb{Z}^{m_2}$. Initialize list $\mathcal{M} \leftarrow \left\{\mathcal{P}^0\right\}$ and $k \leftarrow 1$. Set global lower bound $L = \psi(\beta^0) + \mathbb{E}_\xi \phi(h(\omega) - \beta^0)$ using an arbitrary $\beta^0 \in \mathbf{B}^1$. Set $\mu^0 = \psi(\lambda^0) + \mathbb{E}_\xi \phi(h(\omega) - \eta^0)$ and $\upsilon^0 = \psi(\eta^0) + \mathbb{E}_\xi \phi(h(\omega) - \lambda^0)$.

Step 1: (Subproblem selection) If $\mathcal{M} = \emptyset$, terminate with optimal solution β^*; otherwise, select and delete from $\mathcal{M}$ a hyper-rectangle $\mathcal{P}^k := \left[\lambda^k, \eta^k\right] = \Pi_{i=1}^{m_2}\left[\lambda_i^k, \eta_i^k\right]$.

Step 2: (Subproblem pruning)
- (2a) If $\upsilon^k \leq L$ or $\mathcal{P}^k \cap \mathbf{B}^1 = \emptyset$, go to Step 1.
- (2b) If $\mu^k < \upsilon^k$, i.e. $\mathcal{P}^k$ is an unfathomed hyper-rectangle, go to Step 3.
- (2c) If $\mu^k = \upsilon^k$ and $L < \mu^k$, update $L = \mu^k = \upsilon^k = f^k$, and arbitrarily select $\beta \in \mathcal{P}^k \cap \mathbf{B}^1$ and set $\beta^* = \beta$.
- (2d) Delete from $\mathcal{M}$ all hyper-rectangles $\mathcal{P}^{k'}$ with $\upsilon^{k'} \leq L$ and go to Step 1.

Step 3: (Subproblem partitioning) Choose a dimension i', $1 \leq i' \leq m_2$, such that $\lambda_{i'}^k < \eta_{i'}^k$. Divide $\mathcal{P}^k$ into two hyper-rectangles $\mathcal{P}^{k_1}$ and $\mathcal{P}^{k_2}$ along dimension i' as: $\mathcal{P}^{k_1} := \left[\lambda^{k_1}, \eta^{k_1}\right] = \left[\lambda_{i'}^k, \left\lfloor(\eta_{i'}^k + \lambda_{i'}^k)/2\right\rfloor\right] \times \Pi_{i \neq i'}\left[\lambda_i^k, \eta_i^k\right]$ and $\mathcal{P}^{k_2} := \left[\lambda^{k_2}, \eta^{k_2}\right] = \left[\left\lfloor(\eta_{i'}^k + \lambda_{i'}^k)/2\right\rfloor + 1, \eta_{i'}^k\right] \times \Pi_{i \neq i'}\left[\lambda_i^k, \eta_i^k\right]$. Add the two hyper-rectangles $\mathcal{P}^{k_i}, i = 1, 2$, to $\mathcal{M}$, i.e. $\mathcal{M} \leftarrow \mathcal{M} \cup \left\{\mathcal{P}^{k_1}, \mathcal{P}^{k_2}\right\}$. Set $\mu^{k_i} = \psi(\lambda^{k_i}) + \mathbb{E}_\xi \phi(h(\omega) - \eta^{k_i})$ and $\upsilon^{k_i} = \psi(\eta^{k_i}) + \mathbb{E}_\xi \phi(h(\omega) - \lambda^{k_i})$, $i = 1, 2$. Set $k \leftarrow k + 1$ and go to Step 1.

Theorem 2 *[35] There Algorithm 5 terminates with an optimal solution β^* to (P2) after a finite number of iterations.*

4.2 The minimal tender approach

In this section we describe a level-set approach to reduce the search space when T in (P1) is nonnegative so that $\mathbf{B}^1 \subset \mathbb{Z}_+^{m_2}$. In this case, there must exist an optimal right-hand side β^* to $(P2)$ satisfying that each smaller right-hand side has also a strictly smaller objective value in the first stage. We call such right-hand sides *minimal tenders*. Let $\Theta \subseteq \mathbf{B}^1$ be the set of all minimal tenders. The minimal tender approach is first introduced in Kong et al. [35] for linear integer programs (IPs). We extend their results to QIPs.

Definition 1 [35] There a vector $\beta \in \mathbf{B}^1$ is a minimal tender if, $\forall i = 1, \ldots, m_2$, either $\beta_i = 0$ or $\psi(\beta - e_i) < \psi(\beta)$, where e_i is the ith unit vector.

Theorem 3 *[35] There exists a minimal tender optimal solution to $(P2)$. That is,*

$$\max_{\beta \in \Theta} \left\{ \psi(\beta) + \mathbb{E}_\xi \phi(h(\omega) - \beta) \right\} = \max_{\beta \in \mathbf{B}^1} \left\{ \psi(\beta) + \mathbb{E}_\xi \phi(h(\omega) - \beta) \right\}.$$

Let $\rho = |\Theta|/|\mathbf{B}^1|$. Intuitively, as $\rho \to 1$, the computational benefit of searching Θ may be surpassed by the computational burden of determining Θ. Unfortunately, the value of ρ is typically not known until $\psi(\cdot)$ is completely determined, although in some special cases it can be identified analytically.

Remark 1 Suppose $\forall i\ c_i + \frac{1}{2}\Lambda_{ii} > 0$ and $\forall j \neq i\, \Lambda_{ij} \geq 0$. If T contains I_{m_2}, the m_2-dimensional identity matrix as a submatrix, then $\Theta = \mathbf{B}^1$ and $\rho = 1$.

Let $\overline{opt}(\beta) \subseteq \mathbb{Z}_+^{n_1}$ denote the set of optimal solutions to $\psi(\beta)$.

Lemma 1 *[35] There for any $\beta \in \Theta$ and $\hat{x} \in \overline{opt}(\beta)$, $T\hat{x} = \beta$.*

Proposition 1 *Let $\Lambda = diag(\Lambda_{11}, \ldots, \Lambda_{nn}) \succeq 0$ and $\hat{x} \in \overline{opt}(\beta)$. If $\beta \in \Theta \backslash \{\mathbf{0}\}$, then $Tx \in \Theta$ for all $x \neq \mathbf{0}$ such that $x_i = 0$ or $x_i = \hat{x}_i\ \forall i$.*

Proof $\psi(\cdot)$ is superadditive by Proposition 9 in Sect. 5.2. Let $x \neq \mathbf{0}$ be such that $x_i = \hat{x}_i$ or $x_i = 0\ \forall i$. Then $\psi(Tx) = \frac{1}{2}x^T\Lambda x + c^T x$ by Corollary 9 in Sect. 5.2. Suppose that $Tx \notin \Theta$. Then there exists an $i \in \{1, \ldots, m_2\}$ such that $Tx - e_i \geq 0$ and $\psi(Tx - e_i) = \psi(Tx)$. Let $y \in \overline{opt}(Tx - e_i)$. Thus $y \neq x$, $y \in S_1(Tx - e_i)$ and $\frac{1}{2}x^T\Lambda x + c^T x = \frac{1}{2}y^T\Lambda y + c^T y$. Consider $\tilde{x} = \hat{x} - x + y$. Then $T\tilde{x} = T(\hat{x} - x + y) \leq T\hat{x} - Tx + Tx - e_i = T\hat{x} - e_i$ and $\tilde{x} \in S_1(T\hat{x} - e_i)$.

Note that $\sum_i \Lambda_{ii}\hat{x}_i x_i = \sum_i \Lambda_{ii} x_i^2$ as $x_i = 0$ or $x_i = \hat{x}_i\ \forall i$. Then the objective value for $\tilde{x}$:

$$\begin{aligned}
&\frac{1}{2}(\hat{x} - x + y)^T\Lambda(\hat{x} - x + y) + c^T(\hat{x} - x + y) \\
&= \frac{1}{2}\sum_i \Lambda_{ii}(\hat{x}_i - x_i)^2 + \frac{1}{2}\sum_i \Lambda_{ii} y_i^2 + \sum_i \Lambda_{ii}(\hat{x}_i - x_i)y_i + c^T(\hat{x} - x + y) \\
&\geq \frac{1}{2}\sum_i \Lambda_{ii}\hat{x}_i^2 - \sum_i \Lambda_{ii}\hat{x}_i x_i + \frac{1}{2}\sum_i \Lambda_{ii} x_i^2 + \frac{1}{2}\sum_i \Lambda_{ii} y_i^2 + c^T(\hat{x} - x + y) \\
&= \frac{1}{2}\sum_i \Lambda_{ii}\hat{x}_i^2 + c^T\hat{x} - \frac{1}{2}\sum_i \Lambda_{ii} x_i^2 - c^T x + \frac{1}{2}\sum_i \Lambda_{ii} y_i^2 + c^T y \\
&= \frac{1}{2}\hat{x}^T\Lambda\hat{x} + c^T\hat{x},
\end{aligned}$$

which implies that $\psi(T\hat{x} - e_i) = \psi(T\hat{x})$ contradicting $T\hat{x} = \beta \in \Theta$. □

Corollary 1 *Let $\Lambda = diag(\Lambda_{11}, \ldots, \Lambda_{nn}) \succeq 0$. For any $\beta \in \Theta \backslash \{\mathbf{0}\}$ and $\hat{x} \in \overline{opt}(\beta)$, if $\hat{x}_\ell \geq 1$, then $\hat{x}_\ell t_\ell \in \Theta$.*

Proof Let $x \in \mathbb{Z}^n_+$ such that $x_\ell = \hat{x}_\ell$ and $x_i = 0$ for all $i \neq \ell$. Then the result follows directly from Proposition 1. □

Remark 2 Corollary 1 is a generalization of Kong et al.'s [35] result for linear integer programs, which states that for any $\beta \in \Theta \setminus \{\mathbf{0}\}$ and $\hat{x} \in \overline{opt}(\beta)$, if $\hat{x}_\ell \geq 1$, then $t_\ell \in \Theta$. Proposition 1 and Corollary 1 hold for linear IPs as well, however Kong et al.'s [35] result does not hold for diagonal QIPs. Consider the following instance:

$$z(\beta) = \max \left\{3x_1^2 + x_2^2 + 2x_1 + 4x_2 \mid 2x_1 + x_2 \leq \beta_1,\ 2x_1 + 2x_2 \leq \beta_2,\ x \in \mathbb{Z}^2_+\right\}.$$

Clearly, $\hat{x} = (2, 0)^T \in \overline{opt}((4, 4)^T)$ and $z((4, 4)^T) = 16$. $(4, 4)^T$ is a minimal tender since $z((3, 4)^T) = 12 < 16$ and $z((4, 3)^T) = 5 < 16$. Note that $t_1 = (2, 2)^T$ and $\hat{x}_1 t_1 = (4, 4)^T$. However, $t_1 = (2, 2)^T$ is not a minimal tender since $z((2, 2)^T) = z((1, 2)^T) = 5$.

Corollary 2 *Let* $\Lambda = diag(\Lambda_{11}, \ldots, \Lambda_{nn}) \succeq 0$. *For any* $\beta \in \Theta \setminus \{\mathbf{0}\}$ *and* $\hat{x} \in \overline{opt}(\beta)$, *if* $\hat{x}_\ell \geq 1$ *and* $\Lambda_{\ell\ell} = 0$, *then* $t_\ell \in \Theta$.

4.3 Reduction of the primal formulation using minimal tenders

In this section we assume that Λ is a diagonal matrix with nonnegative diagonal elements, i.e. $\forall i\, \Lambda_{ii} \geq 0$ and $\forall j \neq i\, \Lambda_{ij} = 0$. Let $\mathcal{T}$ be the index set of columns t_j in T such that $kt_j \in \Theta$ for some $k \in \mathbb{Z}^1_+$. For $\beta \in \mathbb{Z}^{m_2}_+$, we define

$$\psi'(\beta) = \max \left\{ \frac{1}{2} \sum_{j \in \mathcal{T}} \Lambda_{jj} x_j^2 + \sum_{j \in \mathcal{T}} c_j x_j \mid x \in S_1'(\beta) \right\}, \tag{7}$$

where

$$S_1'(\beta) = \left\{ x \in \mathbb{Z}^{|\mathcal{T}|}_+ \,\middle|\, \sum_{j \in \mathcal{T}} a_j x_j \leq b,\ \sum_{j \in \mathcal{T}} t_j x_j \leq \beta \right\}. \tag{8}$$

Then the reduced superadditive dual reformulation is

$$\max_{\beta \in \Theta} \left\{ \psi'(\beta) + \mathbb{E}_\xi \phi(h(\omega) - \beta) \right\}. \tag{9}$$

Lemma 2 *For* $\beta \in \Theta$, $\psi'(\beta) = \psi(\beta)$.

Proof The result is trivial for $\beta = \mathbf{0}$. For any $\beta \in \Theta \setminus \{\mathbf{0}\}$ it follows directly from Corollary 1. □

Theorem 4 *There exists an optimal solution to* $(P2)$ *that is an optimal solution to* (9). *That is,*

$$\max_{\beta \in \Theta} \left\{ \psi'(\beta) + \mathbb{E}_\xi \phi(h(\omega) - \beta) \right\} = \max_{\beta \in \mathbf{B}^1} \left\{ \psi'(\beta) + \mathbb{E}_\xi \phi(h(\omega) - \beta) \right\}.$$

The proof of Theorem 4 directly follows from Lemma 2 and Theorem 3.

Corollary 3 *Let β^* be an optimal solution to (9). Then $\hat{x} \in \overline{opt}(\beta^*)$ is an optimal solution to $(P1)$. Furthermore, the optimal objective values of the two problems are equal.*

Corollary 4 *There exists an optimal solution x^* to $(P1)$ where $x_j^* = 0$ for $j \notin \mathcal{T}$.*

5 The value function of a quadratic integer program

We first review basic properties of linear IP value functions. We then consider value functions of quadratic integer programs and describe various properties of them, which are subsequently utilized in algorithmic developments in Sect. 6. These properties may also be useful in other contexts, such as sensitivity analysis of quadratic integer programs [18,25].

5.1 Properties of linear IP value functions

Given $G \in \mathbb{Z}^{m\times n}$ and $\gamma \in \mathbb{Z}^n$, consider a family of parameterized linear IPs:

$$\text{(PIP)}: \quad \zeta(\beta) = \max\{\gamma^T x \mid x \in S(\beta)\}, \quad S(\beta) = \{x \in \mathbb{Z}_+^n \mid Gx \le \beta\} \quad \text{for } \beta \in \mathbb{Z}^m.$$

The function $\zeta(\cdot) : \mathbb{Z}^m \mapsto \mathbb{Z}$, is called the *value function* of (PIP). Define $\widehat{opt}(\beta) = \text{argmax}\{\gamma^T x \mid Gx \le \beta, x \in \mathbb{Z}_+^n\}$ and $S_{LP}(\beta) = \{x \in \mathbb{R}_+^n \mid Gx \le \beta\}$. Moreover, let $\zeta_{LP}(\beta) = \max\{\gamma^T x \mid x \in S_{LP}(\beta)\}$. The following results are proved in [42].

Proposition 2 $\zeta(\mathbf{0}) \in \{0, \infty\}$. *If $\zeta(\mathbf{0}) = \infty$, then $\zeta(\beta) = \pm\infty$ for all $\beta \in \mathbb{R}^m$. If $\zeta(\mathbf{0}) = 0$, then $\zeta(\beta) < \infty$ for all $\beta \in \mathbb{Z}^m$.*

Proposition 3 $\zeta(g_j) \ge \gamma_j$ *for $j = 1, \ldots, n$.*

Proposition 4 $\zeta(\cdot)$ *is nondecreasing in $\beta \in \mathbb{Z}^m$.*

Proposition 5 $\zeta(\cdot)$ *is superadditive over $D = \{\beta \in \mathbb{Z}^m \mid S(\beta) \neq \emptyset\}$.*

Proposition 6 *(Integer Complementary Slackness) If $\hat{x} \in \widehat{opt}(\beta)$, then $\zeta(Gx) = \gamma^T x$ and $\zeta(Gx) + \zeta(\beta - Gx) = \zeta(Gx) + \zeta(G(\hat{x} - x)) = \zeta(\beta)$, for all $x \in \mathbb{Z}_+^n$ such that $x \le \hat{x}$.*

Corollary 5 *If $\zeta(g_j) > \gamma_j$, then for all $\beta \in \mathbb{Z}^m$ and $\hat{x} \in \widehat{opt}(\beta)$, $\hat{x}_j = 0$.*

5.2 Properties of quadratic IP value function

Given a symmetric matrix $Q \in \mathbb{Z}^{n\times n}$, column vectors $c \in \mathbb{Z}^n$, $\beta \in \mathbb{Z}^m$ and constraint matrix $G \in \mathbb{Z}^{m\times n}$, we consider the following family of parametric QIPs:

$$(PQIP): \quad z(\beta) = \max\left\{\frac{1}{2}x^T Qx + c^T x \middle| \, x \in S(\beta)\right\} \quad \text{for } \beta \in \mathbb{Z}^m.$$

The function $z(\cdot) : \mathbb{Z}^m \mapsto \mathbb{Z}$, is called the value function of $(PQIP)$. Define $opt(\beta) = \text{argmax}\left\{\frac{1}{2}x^T Qx + c^T x \mid x \in S(\beta)\right\}$ and $z_{QP}(\beta) = \max\left\{\frac{1}{2}x^T Qx + c^T x \mid x \in S_{LP}(\beta)\right\}$. We assume that $z(\beta) = -\infty$ when $S(\beta) = \emptyset$; and $z_{QP}(\beta) = -\infty$ when $S_{LP}(\beta) = \emptyset$. Let q_i be ith column and q_{ij} be (i, j)th element of matrix Q, respectively.

Proposition 7 $z(g_j) \geq c_j + \frac{1}{2}q_{jj}$.

Proposition 8 *$z(\cdot)$ is nondecreasing in $\beta \in \mathbb{Z}^m$.*

Proposition 9 *If Q is nonnegative, then $z(\cdot)$ is superadditive over $D = \left\{\beta \in \mathbb{Z}^m \mid S(\beta) \neq \emptyset\right\}$. Otherwise, there exists a matrix G such that $z(\cdot)$ is not superadditive.*

Proof Let $x_1 \in opt(\beta_1)$ and $x_2 \in opt(\beta_2)$, then $x_1 + x_2 \in S(\beta_1 + \beta_2)$, and

$$\begin{aligned} z(\beta_1 + \beta_2) &\geq c^T(x_1 + x_2) + \frac{1}{2}(x_1 + x_2)^T Q(x_1 + x_2) \\ &= c^T x_1 + c^T x_2 + \frac{1}{2}x_1^T Qx_1 + \frac{1}{2}x_2^T Qx_2 + x_1^T Qx_2 \\ &= z(\beta_1) + z(\beta_2) + x_1^T Qx_2, \end{aligned}$$

which implies that $z(\beta_1 + \beta_2) \geq z(\beta_1) + z(\beta_2)$ since $q_{ij} \geq 0\ \forall i, j$.

Next, suppose that $\exists i, j$ such that $q_{ij} < 0$. If $i = j$, i.e. $q_{ii} < 0$, consider the following feasible region:

$$S(\beta) = \left\{ x \in \mathbb{Z}^n_+ \mid \sum_{k:k\neq i} x_k \leq \beta_1,\ x_i \leq \beta_2,\ -x_i \leq \beta_3 \right\}.$$

Let $\beta = (0, 1, -1)^T$. Then for any given c, we obtain $z(\beta) = c_i + \frac{1}{2}q_{ii}$ and

$$z(2\beta) = 2c_i + 2q_{ii} = z(\beta) + z(\beta) + q_{ii} < z(\beta) + z(\beta).$$

Otherwise, if $i \neq j$ and $q_{ij} < 0$. Consider the following feasible region:

$$S(\beta) = \left\{ x \in \mathbb{Z}^n_+ \mid \sum_{k:k\neq i, k\neq j} x_k \leq \beta_1,\ x_i \leq \beta_2,\ -x_i \leq \beta_3,\ x_j \leq \beta_4,\ -x_j \leq \beta_5 \right\}.$$

Let $\beta = (0, 1, -1, 0, 0)^T$ and $\beta' = (0, 0, 0, 1, -1)^T$. Then, for any given c, we obtain $z(\beta) = c_i + \frac{1}{2}q_{ii}$, $z(\beta') = c_j + \frac{1}{2}q_{jj}$ and

$$z(\beta + \beta') = c_i + c_j + \frac{1}{2}q_{ii} + \frac{1}{2}q_{jj} + q_{ij} = z(\beta) + z(\beta') + q_{ij} < z(\beta) + z(\beta').$$

□

Next, we investigate whether an analogue of Proposition 2 holds for QIPs. Note that Assumptions **A2** and **A3** ensure that $z(\cdot)$ is finite for all $\beta \in \mathbb{Z}^m$. We relax these two assumptions for the results that are directly related to the finiteness of $z(\cdot)$.

Remark 3 There are instances of $(PQIP)$ such that $z(\mathbf{0}) \notin \{0, \infty\}$. Consider the following instance:

$$z(\beta) = \max \left\{ 3x_1 - \frac{3}{2}x_1^2 + x_2 \mid x_2 \leq \beta \text{ and } x \in \mathbb{Z}_+^2 \right\}.$$

Obviously, $z(\mathbf{0}) = \frac{3}{2} \notin \{0, \infty\}$ with $\hat{x} = (1, 0)^T$. □

If $z(\cdot)$ is superadditive, then the first two properties of Proposition 2 extend to QIPs.

Proposition 10 *If $z(\cdot)$ is superadditive, then $z(\mathbf{0}) \in \{0, \infty\}$. Moreover, if $z(\mathbf{0}) = \infty$, then $z(\beta) = \pm\infty$ for all $\beta \in \mathbb{Z}^m$.*

Proof Suppose $z(\mathbf{0}) < \infty$. Then $z(\mathbf{0}) \leq 0$ [42], but also $z(\mathbf{0}) \geq 0$ as $\mathbf{0} \in S(\mathbf{0})$. This implies that $z(\mathbf{0}) = 0 \in \{0, \infty\}$.

Suppose $z(\mathbf{0}) = \infty$. If $S(\beta) = \emptyset$, then $z(\beta) = -\infty$. Otherwise, from superadditivity $z(\mathbf{0}) + z(\beta) \leq z(\beta) \Rightarrow \infty \leq z(\beta)$. □

Remark 4 There are instances of $(PQIP)$ such that $z(\beta) = +\infty$ for some $\beta \in \mathbb{Z}^m$ while $z(\mathbf{0}) = 0$, which implies that the last statement of Proposition 2 does not hold. Consider the following instance:

$$z(\beta) = \max \left\{ x_2 + x_1 x_2 \mid x_2 - x_1 \leq \beta_1, x_2 \leq \beta_2, x \in \mathbb{Z}_+^2 \right\}.$$

Note that $z(\mathbf{0}) = 0$ and $(1, 0)^T \in opt(\mathbf{0})$. If $\beta = (0, 1)^T$, then $z(\beta) = +\infty$, since $\hat{x} = (1, 1)^T + t(1, 0)^T \in S(\beta)\ \forall t \in \mathbb{Z}_+^1$. Note that $z(\cdot)$ is also superadditive as Q is nonnegative. □

Lemma 3 *If $z(\mathbf{0}) = 0$, then $v^T Q v \leq 0\ \forall v \in S(\mathbf{0})$.*

Proof If $v \in S(\mathbf{0})$, then $tv \in S(\mathbf{0})\ \forall t \in \mathbb{Z}_+^1$. Suppose that $v^T Q v > 0$. Then $c^T(tv) + \frac{1}{2}(tv)^T Q(tv) > 0$ as $t \to +\infty$, which contradicts $z(\mathbf{0}) = 0$. □

Proposition 11 *If $z(\mathbf{0}) = 0$ and $x^T Q v \leq 0\ \forall x \in S(\beta)$ and $\forall v \in S(\mathbf{0})$, then $z(\beta) < \infty\ \forall \beta \in \mathbb{Z}^m$.*

Proof Since $z(\mathbf{0}) = 0$ Lemma 3 holds. Suppose $z(\beta) = \infty$ for some $\beta \in \mathbb{Z}^m$. Then, $\exists x \in S(\beta)$, $v \in S(\mathbf{0})$ and $t \in \mathbb{Z}_+^1$ such that

$$\begin{aligned} &c^T (x + (t+1)v) + \frac{1}{2}(x + (t+1)v)^T Q (x + (t+1)v) \\ &\quad - c^T(x + tv) - \frac{1}{2}(x + tv)^T Q(x + tv) > 0, \\ &\Rightarrow c^T v + \frac{1}{2} v^T Q v + t v^T Q v + x^T Q v > 0, \end{aligned}$$

which is a contradiction because $v^T Qv \leq 0$ from Lemma 3 and $x^T Qv \leq 0$ by the assumption of the proposition. □

Remark 5 The superadditivity of $z(\cdot)$ does not imply that $x^T Qv \leq 0$. Consider the instance in Remark 4, where $x^T Qv = 1$ for $x = (1, 1)^T$ and $v = (1, 0)^T$. □

Remark 6 If $x^T Qv \leq 0$ for all $\beta \in \mathbb{Z}^m$, $x \in S(\beta)$ and $v \in S(\mathbf{0})$, then $z(\cdot)$ is not necessarily superadditive. Consider the following instance:

$$z(\beta) = \max \left\{ 3x - x^2 \mid x \leq \beta, x \in \mathbb{Z}^1_+ \right\}.$$

$z(\cdot)$ is not superadditive since $z(1) = 2$, $z(2) = 2$ and $z(3) = 2$. Note that $x^T Qv = 0$ for all x since $v = 0$ when $\beta = 0$. □

As a result of Remarks 5 and 6, we may conclude that there is no direct relation between the superadditivity of the value function $z(\cdot)$ and the sufficient condition for its finiteness given by Proposition 11.

Next, we consider the extensions of Proposition 6 (Integer Complementary Slackness) for QIPs.

Proposition 12 *Let $\hat{x} \in opt(\beta)$ for $\beta \in \mathbb{Z}^m$. Then $\forall x \leq \hat{x}$ and $x \in \mathbb{Z}^n_+$,*

$$z(G\hat{x}) - x^T Q(\hat{x} - x) \leq z(Gx) + z(G(\hat{x} - x)) \leq z(Gx) + z(\beta - Gx).$$

Proof The right inequality follows since $z(\cdot)$ is nondecreasing and $\hat{x} \in S(\beta)$. To show the left inequality,

$$\begin{aligned} z(Gx) + z(G(\hat{x} - x)) &\geq c^T x + \frac{1}{2} x^T Qx + c^T(\hat{x} - x) + \frac{1}{2}(\hat{x} - x)^T Q(\hat{x} - x) \\ &= c^T \hat{x} + \frac{1}{2} x^T Qx + \frac{1}{2} \hat{x}^T Q\hat{x} + \frac{1}{2} x^T Qx - x^T Q\hat{x} \\ &= c^T \hat{x} + \frac{1}{2} \hat{x}^T Q\hat{x} - x^T Q(\hat{x} - x) = z(G\hat{x}) - x^T Q(\hat{x} - x). \end{aligned}$$

□

Remark 7 Either bound in Proposition 12 can be tight. Consider the following instance:

$$z(\beta) = \max \left\{ -x^2 + 6x \mid x \leq \beta, x \in \mathbb{Z}^1_+ \right\}.$$

For $\beta = 3$, $\hat{x} = 3$ and $z(G\hat{x}) = z(3) = 9$. Let $x = 2 \leq \hat{x} = 3$. Then, $z(G(\hat{x} - x)) = z(1) = 5$, $z(Gx) = z(2) = 8$ and $z(G\hat{x}) - x^T Q(\hat{x} - x) = z(Gx) + z(G(\hat{x} - x)) = 13$, which implies that the left bound is tight. Note that $z(\beta - Gx) = z(G(\hat{x} - x)) = z(1)$ and the right bound is tight as well. □

Proposition 13 *Let $z(\cdot)$ be superadditive and $\hat{x} \in opt(\beta)$ for $\beta \in \mathbb{Z}^m$. Then $\forall x \leq \hat{x}$ and $x \in \mathbb{Z}^n_+$,*

$$z(Gx) \leq c^T x + \frac{1}{2} x^T Qx + x^T Q(\hat{x} - x).$$

Proof Suppose $z(Gx) > c^T x + \frac{1}{2} x^T Qx + x^T Q(\hat{x} - x)$ for some $x \leq \hat{x},\ x \in \mathbb{Z}^n_+$. Then,

$$\begin{aligned} z(Gx) + z(G(\hat{x} - x)) &> c^T x + \frac{1}{2} x^T Qx + x^T Q(\hat{x} - x) + c^T(\hat{x} - x) \\ &\quad + \frac{1}{2}(\hat{x} - x)^T Q(\hat{x} - x) \\ &= c^T \hat{x} + \frac{1}{2} x^T Qx + x^T Q(\hat{x} - x) + \frac{1}{2}\hat{x}^T Q\hat{x} \\ &\quad + \frac{1}{2} x^T Qx - x^T Q\hat{x} \\ &= c^T \hat{x} + \frac{1}{2}\hat{x}^T Q\hat{x} + x^T Q(\hat{x} - x) + x^T Q(x - \hat{x}) = z(G\hat{x}) \end{aligned}$$

which contradicts the superadditivity of $z(\cdot)$. □

Remark 8 The bound in Proposition 13 can be tight when $x^T Q(\hat{x} - x) \neq 0$. Consider the following instance:

$$z(\beta) = \max \left\{ x_1^2 + x_1 x_2 - x_1 + x_2 \mid 2x_1 + x_2 \leq \beta, x \in \mathbb{Z}_2^+ \right\}.$$

Clearly, $\hat{x} = (1, 2)^T \in opt(4)$ and $z(4) = 4$. Consider $x = (1, 1)^T \leq \hat{x} = (1, 2)^T$. Then, $Gx = 3$ and $(0, 3)^T \in opt(3)$ and $z(Gx) = z(3) = 3$. Moreover, $x^T Q(\hat{x} - x) = 1$, $c^T x + \frac{1}{2} x^T Qx = 2$ and $z(\cdot)$ is superadditive since Q is nonnegative. As a result, $c^T x + \frac{1}{2} x^T Qx + x^T Q(\hat{x} - x) = 3 = z(Gx)$. □

Remark 9 For linear IPs, $\zeta(\beta - Gx) = \zeta(G(\hat{x} - x))\ \forall x \leq \hat{x} \in \widehat{opt}(\beta)$ and $x \in \mathbb{Z}^n_+$. This property does not necessarily hold for QIPs even if the value function $z(\cdot)$ is superadditive. Consider the following instance:

$$z(\beta) = \max \left\{ x_1 + x_2^2 + 2x_2 + 2x_1 x_2 \mid x_1 + 3x_2 \leq \beta_1, x_1 + x_2 \leq \beta_2, x \in \mathbb{Z}_2^+ \right\}.$$

Clearly, $\hat{x} = (2, 2)^T \in opt((9, 4)^T)$ and $z((9, 4)^T) = 18$.$z(\cdot)$ is superadditive since Q is nonnegative. Let $x = (0, 2)^T \leq \hat{x} = (2, 2)^T$. Then, $Gx = (6, 2)^T$ and $\beta - Gx = (3, 2)^T$. Moreover, $G\hat{x} = (8, 4)^T$ and $G(\hat{x} - x) = (2, 2)^T$. We have $z(\beta - Gx) = z((3, 2)^T) = 3$ (with a solution of $(0, 1)^T$) and $z(G(\hat{x} - x)) = z((2, 2)^T) = 2$ (with a solution of $(2, 0)^T$). As a result, $z(\beta - Gx) > z(G(\hat{x} - x))$. □

However, if $z(\cdot)$ is superadditive, we can find an upper bound on the difference between $z(\beta - Gx) - z(G(\hat{x} - x))$, which also provides us with some necessary optimality conditions as a corollary.

Proposition 14 *Let $z(\cdot)$ be superadditive and $\hat{x} \in opt(\beta)$ for $\beta \in \mathbb{Z}^m$. Then $\forall x \leq \hat{x}$ and $x \in \mathbb{Z}^n_+$,*

$$0 \leq z(\beta - Gx) - z(G(\hat{x} - x)) \leq x^T Q(\hat{x} - x).$$

Proof The left inequality follows from Proposition 8. To show the right inequality, since $z(\cdot)$ is superadditive,

$$\begin{aligned} z(\beta - Gx) - z(G(\hat{x} - x)) &\leq z(\beta) - z(Gx) - z(G(\hat{x} - x)) \\ &\leq c^T \hat{x} + \frac{1}{2}\hat{x}^T Q\hat{x} - c^T x - \frac{1}{2}x^T Qx - c^T(\hat{x} - x) - \frac{1}{2}(\hat{x} - x)^T Q(\hat{x} - x) \\ &= \frac{1}{2}\hat{x}^T Q\hat{x} - \frac{1}{2}x^T Qx - \frac{1}{2}\hat{x}^T Q\hat{x} - \frac{1}{2}x^T Qx + x^T Q\hat{x} = x^T Q(\hat{x} - x). \end{aligned}$$

□

Corollary 6 *Let $z(\cdot)$ be superadditive and $\hat{x} \in opt(\beta)$ for $\beta \in \mathbb{Z}^m$. Then $x^T Q(\hat{x} - x) \geq 0 \; \forall x \leq \hat{x}$ and $x \in \mathbb{Z}^n_+$.*

Corollary 7 generalizes Proposition 6 (Integer Complementary Slackness) for QIPs.

Corollary 7 *Let $z(\cdot)$ be superadditive and $\hat{x} \in opt(\beta)$ for $\beta \in \mathbb{Z}^m$. Then $\forall x \leq \hat{x}$ and $x \in \mathbb{Z}^n_+$,*

$$c^T x + \frac{1}{2}x^T Qx \leq z(Gx) \leq c^T x + \frac{1}{2}x^T Qx + x^T Q(\hat{x} - x) \quad \text{and}$$
$$z(G\hat{x}) - x^T Q(\hat{x} - x) \leq z(Gx) + z(G(\hat{x} - x)) \leq z(Gx) + z(\beta - Gx) \leq z(G\hat{x}).$$

Note that when $Q = 0$, Proposition 6 (Integer Complementary Slackness) directly follows from Corollary 7. Next, Corollaries 8 and 9 allow us to get either exact values or perform some simple column/value elimination.

Corollary 8 *Let $z(\cdot)$ be superadditive and $\hat{x} \in opt(\beta)$ for $\beta \in \mathbb{Z}^m$. If there exists j such that $q_{ij} = q_{ji} = 0 \; \forall i \neq j$, then $z(\hat{x}_j g_j) = c_j \hat{x}_j + \frac{1}{2} q_{jj} \hat{x}_j^2$.*

Proof Consider vector $x = (0, \ldots, 0, \hat{x}_j, 0, \ldots, 0)^T$ with $x_j = \hat{x}_j$ and $x_i = 0 \; \forall i \neq j$. Then $x \leq \hat{x}$, $Gx = \hat{x}_j g_j$ and $x^T Q(\hat{x} - x) = 0$. The result follows from Corollary 7. □

Corollary 9 *Let $Q = diag(q_{11}, \ldots, q_{nn}) \succeq 0$ and $\hat{x} \in opt(\beta)$ for $\beta \in \mathbb{Z}^m$. Then $z(Gx) = c^T x + \frac{1}{2}x^T Qx$ for all x such that $x_i = 0$ or $x_i = \hat{x}_i \; \forall i$.*

Proof If x is such that $x_i = 0$ or $x_i = \hat{x}_i \; \forall i$, then $x \leq \hat{x}$ and $x^T Q(\hat{x} - x) = 0$ as Q is diagonal. The result follows from Corollary 7. □

Corollary 10 *Let $z(\cdot)$ be superadditive and $\hat{x} \in opt(\beta)$ for $\beta \in \mathbb{Z}^m$. If there exists j such that $q_{ij} = q_{ji} = 0 \; \forall i$ and $z(g_j) > c_j$, then $\hat{x}_j = 0$.*

Proof Suppose $\hat{x}_j \geq 1$, then from Proposition 13, $z(g_j) \leq c^T e_j + \frac{1}{2} e_j^T Q e_j + e_j^T Q(\hat{x} - e_j) = c_j$, which contradicts $z(g_j) > c_j$. □

Corollary 10 is analogous to Corollary 5 for the linear case. It holds when $z(\cdot)$ is superadditive and the respective variable does not appear in the nonlinear part of the objective function. Corollary 11 is a generalization of Corollary 10.

Corollary 11 *Let $Q = diag(q_{11}, \ldots, q_{nn}) \succeq 0$ and $\hat{x} \in opt(\beta)$ for $\beta \in \mathbb{Z}^m$. If there exists j such that $z(hg_j) > hc_j + h(k - \frac{1}{2}h)q_{jj}$ for $k \geq h \geq 1$, then $\hat{x}_j \notin [h, k]$.*

Proof Note that $z(\cdot)$ is superadditive from Proposition 9. If $\hat{x}_j \in [h, k]$, then from Proposition 13

$$z(hg_j) \leq hc^T e_j + \frac{1}{2} h^2 e_j^T Q e_j + h e_j^T Q(\hat{x} - he_j) = hc_j + h\left(\hat{x}_j - \frac{1}{2}h\right) q_{jj}.$$

□

6 Constructing the value function of a parameterized quadratic IP

Motivated by some of the ideas used in the linear case [35], we develop four algorithms to construct the QIP value function. Note that assumptions **A1** and **A2** ensure the finiteness of the feasible right-hand side set. Hence, we consider $(PQIP)$ parameterized over a finite set of right-hand sides $\beta \in \mathbf{B} \subseteq \mathbb{Z}^m$. Under assumptions **A2** and **A3**, the value function is finite, so we assume that $z(\beta) < \infty \; \forall \beta \in \mathbf{B}$.

Our first algorithm is based on the bounds derived in Sect. 5 for the superadditive QIP value function. The next three algorithms are designed for problems with non-negative constraint matrix G. The second algorithm applies to problems with diagonal $Q \succeq 0$, and the remaining two assume that the objective function can be decomposed into sum of a small number of products of linear functions.

6.1 An exact algorithm based on superadditivity

In this section we assume that $z(\cdot)$ is superadditive. Let $l(\cdot)$ and $u(\cdot)$ be lower and upper bounds of $z(\cdot)$, respectively. We maintain $l(\beta) \leq z(\beta) \leq u(\beta) \; \forall \beta \in \mathbf{B}$ and $z(\beta)$ is known when $l(\beta) = u(\beta)$. The algorithm terminates when $z(\beta)$ is determined $\forall \beta \in \mathbf{B}$. In addition to the bounds derived in Sect. 5, we utilize the following properties.

Lemma 4 *Let $\hat{x} \in opt(\beta)$ for $\beta \in \mathbb{Z}^m$. Then $\forall \bar{\beta} \in \mathbb{Z}^m$ and $0 \leq \bar{\beta} \leq \beta - G\hat{x}$, $z(\bar{\beta}) = 0$.*

Lemma 5 *Let $\hat{x} \in opt(\beta)$ for $\beta \in \mathbb{Z}^m$. Then $\forall \bar{\beta} \in \mathbb{Z}^m$ and $G\hat{x} \leq \bar{\beta} \leq \beta$, $z(\bar{\beta}) = z(\beta)$.*

At each iteration, we update $l(\beta)$ and $u(\beta)$ for some $\beta \in \mathbf{B}$ by performing the following two main operations:

1. Solve the quadratic integer program exactly for a given right-hand side $\beta \in \mathbf{B}$ (e.g. using dynamic programming or any QIP solver) to obtain optimal solution $\hat{x}$.
2. Update the lower and upper bounds for a subset of right-hand sides in $\mathbf{B}$ utilizing:
   i. properties of value functions given by Lemmas 4 and 5;
   ii. nondecreasing and superadditivity properties of $z(\cdot)$ (Propositions 8 and 9), bounds from Proposition 13 and some feasibility arguments (see details below).

Algorithm 1. *The Exact-Superadditive Algorithm.*

Step 0: Initialize the lower bound $l^0(\beta) = -\infty\ \forall \beta \in \mathbf{B}$. For $j = 1, \ldots, n$, if $g_j \in \mathbf{B}$, set $l^0(g_j) = \frac{1}{2}q_{jj} + c_j$. Without loss of generality, we assume that there are no duplicate columns. Initialize the upper bound $u^0(\beta) = +\infty\ \forall \beta \in \mathbf{B}$. Initialize $\mathcal{L}^k = \emptyset$ and set $k \leftarrow 1$.
Step 1: Set $l^k(\beta) \leftarrow l^{k-1}(\beta)$ and $u^k(\beta) \leftarrow u^{k-1}(\beta)\ \forall \beta \in \mathbf{B}$. Select $\beta^k \in \mathbf{B}\backslash\mathcal{L}^k$. Solve the QIP with right-hand side β^k to obtain an optimal solution $\hat{x}^k$.
(1a) For all $\beta \in \mathbf{B}\backslash\mathcal{L}^k$ such that $G\hat{x}^k \leq \beta \leq \beta^k$, set $l^k(\beta) = u^k(\beta) = c^T\hat{x}^k + \frac{1}{2}\hat{x}^{kT}Q\hat{x}^k$ and $\mathcal{L}^k \leftarrow \mathcal{L}^k \cup \{\beta\}$,
(1b) For all $\beta \in \mathbf{B}\backslash\mathcal{L}^k$ such that $0 \leq \beta \leq \beta^k - G\hat{x}^k$, set $l^k(\beta) = u^k(\beta) = 0$ and $\mathcal{L}^k \leftarrow \mathcal{L}^k \cup \{\beta\}$.
(1c) For all $\beta \in \mathbf{B}\backslash\mathcal{L}^k$ such that $\beta \geq \beta^k$, set $l^k(\beta) \leftarrow \max\left\{l^k(\beta), l^k(\beta^k)\right\}$. If $\beta - \beta^k \in \mathbf{B}\backslash\mathcal{L}^k$ then $l^k(\beta) \leftarrow \max\left\{l^k(\beta), l^k(\beta^k) + l^k(\beta - \beta^k)\right\}$.
(1d) For all $\beta \in \mathbf{B}\backslash\mathcal{L}^k$ such that $\beta \leq \beta^k$, set $u^k(\beta) \leftarrow \min\left\{u^k(\beta), u^k(\beta^k)\right\}$. If $\beta^k - \beta \in \mathbf{B}\backslash\mathcal{L}^k$ then $u^k(\beta) \leftarrow \min\left\{u^k(\beta), u^k(\beta^k) - l^k(\beta^k - \beta)\right\}$.
(1e) For all $\beta \in \mathbf{B}\backslash\mathcal{L}^k$, if $\beta + \beta^k \in \mathbf{B}\backslash\mathcal{L}^k$, $u^k(\beta) \leftarrow \min\left\{u^k(\beta), u^k(\beta + \beta^k) - l^k(\beta^k)\right\}$.
Step 2: Select all $x \in \mathbb{Z}^n_+$ such that $x \leq \hat{x}^k$.
If $Gx \in \mathbf{B}$ then
(2a) $l^k(Gx) \leftarrow \max\left\{l^k(Gx), c^Tx + \frac{1}{2}x^TQx\right\}$,
(2b) $u^k(Gx) \leftarrow \min\left\{u^k(Gx), c^Tx + \frac{1}{2}x^TQx + x^TQ(\hat{x} - x)\right\}$.
If $\beta^k - Gx \in \mathbf{B}$ then
(2c) $l^k(\beta^k - Gx) \leftarrow \max\left\{l^k(\beta^k - Gx), l^k(G\hat{x}^k) - x^TQ(\hat{x}^k - x) - c^Tx - \frac{1}{2}x^TQx\right\}$,
(2d) $u^k(\beta^k - Gx) \leftarrow \min\left\{u^k(\beta^k - Gx), u^k(G\hat{x}^k) - c^Tx - \frac{1}{2}x^TQx\right\}$.
Step 3: If $l^k(\beta) = u^k(\beta)$ for all $\beta \in \mathbf{B}$, terminate with solution $z(\cdot) = l^k(\cdot) = u^k(\cdot)$; otherwise, set $k \leftarrow k + 1$ and go to Step 1.

Lemma 6 *At any iteration k of the Exact-Superadditive Algorithm, $l^k(\beta) \leq z(\beta) \leq u^k(\beta)$ for all $\beta \in$* $\mathbf{B}$.

Proof The lower bounds in Step 0 follows from Proposition 7. Suppose that at iteration $k - 1 \geq 0$, Lemma 6 holds. Consider iteration k.

- Steps (1a) and (1b) are due to Lemmas 4 and 5, respectively.
- Steps (1c)-(1e) are due to the nondecreasing and superadditivity properties of $z(\cdot)$.
- Step (2a) holds since $x \in S(Gx)$.
- Step (2b) follows from Proposition 13.

– Step (2c) holds since $(\hat{x}^k - x) \in S(\beta^k - Gx)$ and $z(\cdot)$ is nondecreasing. Specifically, $z(\beta - Gx) \geq z(G(\hat{x} - x)) \geq c^T(\hat{x} - x) + \frac{1}{2}(\hat{x} - x)^T Q(\hat{x} - x)$.
– Step (2d) holds since $x \in S(Gx)$ and $z(\cdot)$ is superadditive. Specifically, $z(\beta - Gx) \leq z(G\hat{x}) - z(Gx) \leq z(G\hat{x}) - c^T x - \frac{1}{2}x^T Qx$. □

Proposition 15 *The Exact-Superadditive Algorithm terminates finitely with optimal* $z(\beta)\ \forall \beta \in \mathbf{B}$.

Proof Consider any iteration $k \geq 1$. After Step 1, there exists at least one $\beta \in \mathbf{B}$ such that $l^k(\beta) = u^k(\beta) = z(\beta)$ while $l^{k-1}(\beta) \neq z(\beta)$ or $u^{k-1}(\beta) \neq z(\beta)$. The proof follows since $l^k(\beta) \leq z(\beta) \leq u^k(\beta)\ \forall \beta \in \mathbf{B}$ at any iteration k and $\mathbf{B}$ is finite. □

The size of the instances that can be handled and the overall performance of the *Exact-Superadditive Algorithm* depend on the method used for solving the QIPs arising in Step 1. Our implementation, which is based on dynamic programming, can solve instances up to 400 variables. We leave the investigation of other methods for solving the QIP subproblems for future research.

6.2 A DP-based algorithm for diagonal $Q \succeq 0$

In this section we assume that $Q = diag(q_{11}, \ldots, q_{nn}) \succeq 0$, i.e. Q is a diagonal matrix and all q_{ii}'s are nonnegative. Therefore, by Proposition 9, $z(\cdot)$ is superadditive. We also assume that G is a nonnegative matrix. As $\mathbf{B}$ is finite, there exists a nonnegative hyper-rectangle $\mathcal{B}$ rooted at the origin that contains $\mathbf{B}$. Let $b = (b_1, \ldots, b_m)$ denote the largest vector in $\mathcal{B}$ componentwise. We set $\mathbf{B} = \mathcal{B} \cap \mathbb{Z}_+^m$ in the rest of Sect. 6, where $\mathcal{B} = \{[0, b_1] \times [0, b_2] \times \cdots \times [0, b_m]\}$. Let B_j denote the set of all $\beta \in \mathbf{B}$ such that $\beta \geq g_j$.

Lemma 7 *For all* $\beta \in \mathbf{B} \setminus \cup_{j=1}^n B_j$, $\zeta(\beta) = 0$ *and* $z(\beta) = 0$.

The following result is due to Gilmore and Gomory [23] for linear IPs:

Theorem 5 $\zeta(\beta) = \max\{0, \gamma_j + \zeta(\beta - g_j) \mid g_j \in \mathbf{B},\ j = 1, \ldots, n\}\ \forall \beta \in \cup_{j=1}^n B_j$.

We extend Theorem 5 to $(PQIP)$ with diagonal $Q \succeq 0$ as follows:

Lemma 8 *For all* $\beta \in \cup_{j=1}^n B_j$ *and for all* $j \in \{1, \ldots, n\}$ *such that* $g_j \in \mathbf{B}$,

$$z(\beta) = \max_{\mu \in \mathbb{Z}_+} \left\{ c_j\mu + \frac{1}{2}q_{jj}\mu^2 + z(\beta - \mu g_j) \mid \beta - \mu g_j \geq \mathbf{0} \right\}.$$

In contrast to the exact superadditive algorithm, the following algorithm only defines $l(\cdot)$ and does not solve any quadratic integer program. We update $l(\cdot)$ using the superadditive property of $z(\cdot)$. This approach is motivated by a DP-based algorithm proposed for finding the value function of linear IPs [35]. The major difference of our algorithm is the initialization of lower bounds in Step 0, which follows from Lemma 8.

Algorithm 2. *The Diagonal-Q Algorithm.*

Step 0: Initialize the lower bound $l^0(\beta) = 0\ \forall \beta \in \mathbf{B}$. For $j = 1, \ldots, n$, if $g_j \in \mathbf{B}$, $l^0(\mu g_j) \leftarrow \max\{l^0(\mu g_j), c_j\mu + \frac{1}{2}q_{jj}\mu^2\}\ \forall \mu \in \mathbb{Z}_+$ such that $b - \mu g_j \geq 0$. Insert μg_j into a vector list $\mathcal{L}$. Set $l^1(\beta) = l^0(\beta)$ for all $\beta \in \mathbf{B}$. Set $k \leftarrow 1$.
Step 1: Denote the kth vector in $\mathcal{L}$ by β^k and the ith element of a vector β by β_i. Let $\beta = \beta^k$. Update all vectors β' such that $\beta' \in \mathbf{B}$ and $\beta' \geq \beta^k$ with the following lexicographic order:

(1a) Set $\beta_1 \leftarrow \beta_1 + 1$ and $l^k(\beta) \leftarrow \max\{l^k(\beta), l^k(\beta^k) + l^k(\beta - \beta^k)\}$.
(1b) If $\beta_1 \geq b_1$, go to Step (1c); otherwise, go to Step (1a).
(1c) If for all $i = 1, \ldots, m$, $\beta_i \geq b_i$, go to Step 2. Otherwise, let $s = \min\{i : \beta_i < b_i\}$. Set $\beta_i \leftarrow \beta_i^k$ for $i = 1, \ldots, s-1$. Set $\beta_s \leftarrow \beta_s + 1$ and go to Step (1a).

Step 2: If $k = |\mathcal{L}|$, terminate with solution $z(\cdot) = l^k(\cdot)$. Otherwise, put $l^{k+1}(\beta) \leftarrow l^k(\beta)$ for all $\beta \in \mathbf{B}$, set $k \leftarrow k + 1$ and go to Step 1.

Let $\mu_{\max}$ denote the maximum feasible scalar value that any variable can take in a solution. Note that $\mu_{\max}$ is finite since G is nonnegative and $\mathbf{B}$ is finite.

Proposition 16 *The Diagonal-Q Algorithm terminates with optimal $z(\cdot)$ for all $\beta \in \mathbf{B}$ in at most $n\mu_{\max}$ iterations.*

Proof For any $\beta \in \mathbf{B} \setminus \cup_{j=1}^n B_j$, we initialize $l^0(\beta) = 0$ in Step 0 and do not update them subsequently. By Lemma 7, $z(\beta) = 0,\ \forall \beta \in \mathbf{B} \setminus \cup_{j=1}^n B_j$. Assume that the algorithm terminates at iteration $k^* = |\mathcal{L}|$. Then $l^{k^*}(\beta) = z(\beta),\ \forall \beta \in \mathbf{B} \setminus \cup_{j=1}^n B_j$. Suppose there exists $\beta \in \cup_{j=1}^n B_j$ such that $l^{k^*}(\beta) \neq z(\beta)$ and $l^{k^*}(\beta') = z(\beta')\ \forall \beta' \leq \beta, \beta' \in \cup_{j=1}^n B_j$. Then $l^{k^*}(\beta) < z(\beta)$ by construction of the algorithm. It follows that there exists a $j^* \in \{1, \ldots, n\}$ and $\mu_* \geq 1$ such that $l^{k^*}(\beta) < c_{j^*}\mu_* + \frac{1}{2}q_{j^*j^*}\mu_*^2 + z(\beta - g_{j^*}\mu_*)$ by Lemma 8. Since $l^{k^*}(g_{j^*}\mu_*) \geq c_{j^*}\mu_* + \frac{1}{2}q_{j^*j^*}\mu_*^2$ and $l^{k^*}(\beta - g_{j^*}\mu_*) = z(\beta - g_{j^*}\mu_*)$, it follows that $l^{k^*}(g_{j^*}\mu_*) + l^{k^*}(\beta - g_{j^*}\mu_*) \geq c_{j^*}\mu_* + \frac{1}{2}q_{j^*j^*}\mu_*^2 + z(\beta - g_{j^*}\mu_*) > l^{k^*}(\beta)$, which contradicts the superaddivity of $l^{k^*}(\cdot)$. Hence, $l^{k^*}(\beta) = z(\beta)\ \forall \beta \in \cup_{j=1}^n B_j$ and the result follows since $k^* = |\mathcal{L}| \leq n\mu_{\max}$. □

Proposition 17 *The running time of the Diagonal-Q Algorithm is $O(n\mu_{\max}|\mathbf{B}|)$.*

Proof Step 0 requires $O(n\mu_{\max})$ calculations. Step 1 of the algorithm requires at most $O(|\mathbf{B}|)$ calculations. Since Step 1 is executed at most $n\mu_{\max}$ times, the overall running time of the algorithm is $O(n\mu_{\max}|\mathbf{B}|)$. □

We note that in the linear case the running time of the most efficient algorithm of Kong et al. [35] is $O(n|\mathbf{B}|)$. Since $\mu_{\max} << |\mathbf{B}|$, the running time of the *Diagonal-Q Algorithm* is almost as good.

6.3 An iterative fixing algorithm for low-rank Q

For some $\ell \leq n$, the quadratic objective function of $(PQIP)$ can be represented as:

$$\frac{1}{2}x^T Qx + c^T x = \left(\chi_1^T x\right)\left(\sigma_1^T x\right) + \cdots + \left(\chi_\ell^T x\right)\left(\sigma_\ell^T x\right) + c^T x, \tag{10}$$

where χ_i and σ_i are vectors in $\mathbb{Z}^n$ for $i = 1, \ldots, \ell$. We are interested in classes of (10) with small $\ell << n$ and either all χ's or all σ's are nonnegative integer vectors. To motivate the representation in (10), note that any quadratic function can be written as:

$$\frac{1}{2}x^T Qx + c^T x = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} q_{ij}x_i x_j + c^T x = \frac{1}{2}\sum_{i=1}^{n}\left(x_i \cdot \sum_{j=1}^{n} q_{ij}x_j\right) + c^T x. \tag{11}$$

Suppose that for every nonlinear term $x_i x_j$ in (11) either i or $j \in \{1, \ldots, \ell\}$. Then due to symmetry of Q, quadratic function in (11) simplifies to

$$\frac{1}{2}x^T Qx + c^T x = \sum_{i=1}^{\ell}\left(x_i \cdot \sum_{j=1}^{n} q_{ij}x_j\right) + c^T x, \tag{12}$$

where $\chi_1 = (1, 0, \ldots, 0)^T$, ..., $\chi_\ell = (0, \ldots, 0, 1, 0, \ldots, 0)^T$, and $\sigma_1 = (q_{11}, \ldots, q_{1n})^T, \ldots, \sigma_\ell = (q_{\ell 1}, \ldots, q_{\ell n})^T$. That is, columns and rows of Q can be rearranged in a such way that only first ℓ rows and ℓ columns may contain nonzero elements for some small ℓ, while the rest of Q contains only zero elements.

Another motivation is that any symmetric matrix Q can be decomposed into

$$Q = U\ diag(\lambda_1, \ldots, \lambda_n)\ U^T, \tag{13}$$

where U is the matrix of eigenvectors $u_1, \ldots, u_n$ (stored as columns) and $\lambda_1, \ldots, \lambda_n$ are eigenvalues of Q, respectively. If only ℓ eigenvalues of Q are nonzero, which for small ℓ corresponds to a low-rank matrix Q, then

$$\frac{1}{2}x^T Qx + c^T x = \lambda_1\left(u_1^T x\right)^2 + \cdots + \lambda_\ell\left(u_\ell^T x\right)^2 + c^T x, \tag{14}$$

and we can set $\chi_1 = \lambda_1 u_1$, ..., $\chi_\ell = \lambda_\ell u_\ell$, and $\sigma_1 = u_1, \ldots, \sigma_\ell = u_\ell$.

Let $\mathcal{H}$ and Σ be $\ell \times n$ matrices with rows composed by $(\chi_i)^T$'s and $(\sigma_i)^T$'s for $i = 1, \ldots, \ell$, respectively. We assume that $\mathcal{H} \in \mathbb{Z}_+^{\ell \times n}$ and $\ell << n$. Note that we do not require $z(\cdot)$ to be superadditive in this section.

For $y \in \mathbb{Z}_+^\ell$ and $\beta \in \mathbf{B}$, we define problem $\mathbf{P}^y(\beta)$ by:

$$z^y(\beta) = \max_{x \in \mathbb{Z}_+^n}\left\{y^T \Sigma x + c^T x \mid Gx \leq \beta,\ \mathcal{H}x = y\right\}. \tag{15}$$

Moreover, for $\delta \in \mathbb{Z}_+^\ell$, we formulate an auxiliary problem $\mathbf{P}^{\delta,y}(\beta)$ where the equality constraint of $\mathbf{P}^y(\beta)$ is replaced with an inequality.

$$z^{\delta,y}(\beta) = \max_{x \in \mathbb{Z}_+^n} \left\{ \delta^T \mathcal{H}x + y^T \Sigma x + c^T x \mid Gx \leq \beta,\ \mathcal{H}x \leq y \right\}. \tag{16}$$

Let $\delta \in \mathbb{Z}_+^\ell$ be such that $\frac{1}{2}\delta^T e_i > \max_x \left\{ \left| y^T \Sigma x + c^T x \right| \mid Gx \leq b,\ \mathcal{H}x \leq y \right\}$ for all $i = 1 \ldots \ell$. Then the following results hold.

Lemma 9 *Let $\hat{x}$ be an optimal solution to $\mathbf{P}^{\delta,y}(\beta)$. Then $z^{\delta,y}(\beta) \geq \delta^T y - \frac{1}{2}\min_i \delta^T e_i$ if and only if $y = \mathcal{H}\hat{x}$.*

Proof "$\Leftarrow$" If $y = \mathcal{H}\hat{x}$ then $z^{\delta,y}(\beta) = \delta^T y + y^T \Sigma \hat{x} + c^T \hat{x}$. This value is at least as large as $\delta^T y - \frac{1}{2}\min_i \delta^T e_i$ because $\frac{1}{2}\delta^T e_i + y^T \Sigma x + c^T x > 0$ for all i and for all feasible x by definition of δ.

"$\Rightarrow$" If $z^{\delta,y}(\beta) \geq \delta^T y - \frac{1}{2}\min_i \delta^T e_i$ then

$$\begin{aligned} &\delta^T \mathcal{H}\hat{x} + y^T \Sigma \hat{x} + c^T \hat{x} \geq \delta^T y - \frac{1}{2}\min_i \delta^T e_i \\ &\quad \Rightarrow y^T \Sigma \hat{x} + c^T \hat{x} \geq \delta^T (y - \mathcal{H}\hat{x}) - \frac{1}{2}\min_i \delta^T e_i \Rightarrow y = \mathcal{H}\hat{x}. \end{aligned}$$

The last equality holds as $\mathcal{H}\hat{x} \leq y$ and $\frac{1}{2}\delta^T e_i > \max \left\{ \left| y^T \Sigma x + c^T x \right| |Gx \leq b, \mathcal{H}x \leq y \right\}$ for all i and for all feasible x by definition of δ. Note that if $\mathcal{H}\hat{x} < y$, then $\delta^T (y - \mathcal{H}\hat{x}) \geq \min_i \delta^T e_i$ since both y and $\mathcal{H}\hat{x}$ are integer vectors. □

Corollary 12 *If $z^{\delta,y}(\beta) \geq \delta^T y - \frac{1}{2}\min_i \delta^T e_i$, then $z^y(\beta) = z^{\delta,y}(\beta) - \delta^T y$.*

Lemma 10 *Let $R = \{y \in \mathbb{Z}_+^\ell \mid Gx \leq b,\ \mathcal{H}x = y,\ x \in \mathbb{Z}_+^n\}$. Then*

$$z(\beta) = \max_{y \in R} z^y(\beta)\ \forall \beta \in \mathbf{B}.$$

Lemma 10 directly follows since set R contains all possible values that $\mathcal{H}x$ can take for $\beta \in \mathbf{B}$. If ℓ is small and $\mathcal{H}$ is sparse, then all possible y's in set R might be enumerated. Otherwise, let $\bar{y}_i = \max_x \{(\mathcal{H}x)^T e_i \mid Gx \leq b\},\ i = 1, \ldots, \ell$ and define $R' \supseteq R$ by

$$R' = \{y \in \mathbb{Z}_+^\ell \mid \mathbf{0} \leq y \leq \bar{y}\}.$$

We first give an iterative algorithm that searches over R'. Then, in Sect. 6.4 we present a modification of this algorithm which enumerates all vectors in R. Define

$$\Pi_y = \left\{ \pi \in \mathbb{Z}_+^{m+\ell} | \pi_i \leq b_i, \forall i \leq m \text{ and } \pi_i \leq y_{i-m}, \forall i > m \right\} \quad \forall y \in R'.$$

Let $\mathfrak{G} = [G \quad \mathcal{H}]$ and Π_{yj} denote the set of vectors $\pi \in \Pi_y$ such that $\pi \geq \mathfrak{g}_j \in \mathfrak{G}$, where $\mathfrak{g}_j$ is the jth column of $\mathfrak{G}$.

Algorithm 3. *Sparse-Fixing Algorithm.*

Step 0: Set $\tau \leftarrow 1$. Denote the τth vector in R' by r^τ. Initialize the global bound $v^0(\beta) = 0$ for all $\beta \in \mathbf{B}$.
Step 1: Set $y \leftarrow r^\tau$. Let $\gamma^T = \delta^T \mathcal{H} + y^T \Sigma + c^T$ and $\hbar = [b \quad y]$. Initialize the lower bound $l^0(\pi) = 0$ for all $\pi \in \Pi_y$. For $j = 1, \ldots, n$, if $\mathfrak{g}_j \in \Pi_y$, $l^0(\mathfrak{g}_j) = \gamma_j$ and insert $\mathfrak{g}_j$ into a vector list $\mathcal{L}$. Set $l^1(\pi) = l^0(\pi)$ for all $\pi \in \Pi_y$. Set $k \leftarrow 1$.
Step 2: Denote the kth vector in $\mathcal{L}$ by π^k and the ith element of π by π_i. Let $\pi = \pi^k$. Update all vectors π' such that $\pi' \in \Pi_y$ and $\pi' \geq \pi^k$ with the following lexicographic order:

(2a) Set $\pi_1 \leftarrow \pi_1 + 1$ and $l^k(\pi) \leftarrow \max\left\{l^k(\pi), l^k\left(\pi^k\right) + l^k\left(\pi - \pi^k\right)\right\}$.
(2b) If $\pi_1 \geq \hbar_1$, go to Step (2c); otherwise, go to Step (2a).
(2c) If $\pi_i \geq \hbar_i$ for all $i = 1, \ldots, m + \ell$, go to Step 3. Otherwise, let $s = \min\{i : \pi_i < \hbar_i\}$. Set $\pi_i \leftarrow \pi_i^k$ for $i = 1, \ldots, s-1$. Set $\pi_s \leftarrow \pi_s + 1$ and go to Step (2a).

Step 3: If $k = |\mathcal{L}|$, go to Step 4. Otherwise, $l^{k+1}(\pi) \leftarrow l^k(\pi)$ for all $\pi \in \Pi_y$, set $k \leftarrow k + 1$ and go to Step 2.
Step 4: For all $\beta \in \cup_{j=1}^n B_j$, let $\beta_y = [\beta \quad y]$. If $l^k(\beta_y) \geq \delta^T y - \frac{1}{2} \min_i \delta^T e_i$, then update $v^\tau(\beta) \leftarrow \max\{l^k(\beta_y) - \delta^T y, v^{\tau-1}(\beta)\}$. If $\tau = |R'|$, stop. Otherwise, empty out vector list $\mathcal{L}$, set $\tau \leftarrow \tau + 1$ and go to Step 1.

Lemma 11 *If $\beta \in \mathbf{B} \setminus \cup_{j=1}^n B_j$, then $\beta_y = [\beta \quad y] \in \Pi_y \setminus \cup_{j=1}^n \Pi_{yj}$ for all $y \in R'$.*

Lemma 12 $z^{\delta,y}(\beta) = 0 \; \forall \beta \in \mathbf{B} \setminus \cup_{j=1}^n B_j$.

For $\pi \in \Pi_y$, we define π^+ to be the subvector composed by the first m elements of π; and π^- to be the subvector composed by the last ℓ elements of π.

Theorem 6 *The Sparse-Fixing Algorithm terminates with optimal solutions to $z(\cdot)$ in at most $|R'|$ iterations of τ.*

Proof Consider a $y \in R'$. For any $\pi \in \Pi_y \setminus \cup_{j=1}^n \Pi_{yj}$, we initialize $l^0(\pi) = 0$ in Step 1 and do not update them subsequently. By Lemma 12, $z^{\delta,y}(\beta) = 0 \; \forall \beta \in \mathbf{B} \setminus \cup_{j=1}^n B_j$. Assume the algorithm enters Step 4, when $k^* = |\mathcal{L}|$. Then $\forall \beta \in \mathbf{B} \setminus \cup_{j=1}^n B_j$, $l^{k^*}(\beta_y) = z^{\delta,y}(\beta)$ as $\beta_y = [\beta \quad y] \in \Pi_y \setminus \cup_{j=1}^n \Pi_{yj}$ by Lemma 11. Suppose there exists $\beta \in \cup_{j=1}^n B_j$ such that $l^{k^*}(\beta_y) \neq z^{\delta,y}(\beta)$ and $l^{k^*}(\pi) = z^{\delta,\pi^-}(\pi^+) \; \forall \pi \leq \beta_y, \pi \in \Pi_y$. Then $l^{k^*}(\beta_y) < z^{\delta,y}(\beta)$ by construction of the algorithm. It follows that there exists a $j^* \in \{1, \ldots, n\}$ such that $l^{k^*}(\beta_y) < \gamma_{j^*} + z^{\delta,(y-\mathfrak{g}_{j^*}^-)}(\beta - \mathfrak{g}_{j^*}^+)$ by Theorem 5. Since $l^{k^*}(\mathfrak{g}_{j^*}) \geq \gamma_{j^*}$ and $l^{k^*}(\beta_y - \mathfrak{g}_{j^*}) = z^{\delta,(y-\mathfrak{g}_{j^*}^-)}(\beta - \mathfrak{g}_{j^*}^+)$, it follows that $l^{k^*}(\mathfrak{g}_{j^*}) + l^{k^*}(\beta_y - \mathfrak{g}_{j^*}) \geq \gamma_{j^*} + z^{\delta,(y-\mathfrak{g}_{j^*}^-)}(\beta - \mathfrak{g}_{j^*}^+) > l^{k^*}(\beta_y)$, which contradicts the superadditivity of $l^{k^*}(\cdot)$. As a result, $l^{k^*}(\beta_y) = z^{\delta,y}(\beta) \; \forall \beta \in \cup_{j=1}^n B_j$ when the algorithm enters Step 4. From Lemma 9, $l^{k^*}(\beta_y) = z^{\delta,y}(\beta) \geq \delta^T y - \frac{1}{2} \min_i \delta^T e_i \; \forall \beta \in \cup_{j=1}^n B_j$ if and only if $y \in R$. When this condition is satisfied, in Step 4, we subtract $\delta^T y$ from $z^{\delta,y}(\beta)$ to obtain $z^y(\beta)$ by Corollary 12.

The algorithm searches over every $y \in R'$ by updating the global lower bound $v^\tau(\beta)$ if $y \in R$. As a result, $v^{|R'|}(\beta) = \max_{y \in R} z^y(\beta) = z(\beta)\ \forall \beta \in \cup_{j=1}^n B_j$ by Lemma 10. □

Proposition 18 *The Sparse-Fixing Algorithm runs in* $O\left(\sum_{y \in R'} n|\Pi_y|\right)$ *time.*

Proof For each $y \in R'$, the algorithm makes $O\left(n|\Pi_y|\right)$ calculations since $|\mathcal{L}|$ is at most n and Step 2 requires $O\left(|\Pi_y|\right)$ calculations. The algorithm iterates over all $y \in R'$, so the overall running time is $O\left(\sum_{y \in R'} n|\Pi_y|\right)$. □

6.4 An iterative fixing algorithm for low-rank Q with enumeration of the set R

In this section, we present a modified version of the sparse fixing algorithm which enumerates all y's in set R. Our computational experiments show that this version runs much faster than the previous one if ℓ is small and matrix $\mathcal{H}$ is sparse.

Let $\hat{\mathcal{H}}$ be the set of nonzero columns in $\mathcal{H}$. We define $\hat{G} \subseteq G$ and $\hat{\Sigma} \subseteq \Sigma$ to be the submatrices corresponding to the columns in $\hat{\mathcal{H}}$. Likewise, let $\hat{c}$ be the linear part of the objective function corresponding to the columns in $\hat{\mathcal{H}}$. We give the set of feasible solutions for the columns in $\hat{\mathcal{H}}$ by:

$$\hat{X} = \left\{ x \in \mathbb{Z}_+^{|\hat{\mathcal{H}}|} : \hat{G}x \le b \right\}.$$

Algorithm 4. *Sparse-Enumeration Algorithm.*

Step 0: Set $t \leftarrow 1$. Denote the tth vector in $\hat{X}$ by x^t. Initialize the global bound $v^0(\beta) = 0$ for all $\beta \in \mathbf{B}$.
Step 1: Set $y \leftarrow \hat{\mathcal{H}}x^t$. Let $\gamma^T = y^T\Sigma + c^T$ and $\hbar = b - \hat{G}x^t$. Define the set $\mathbf{B}^t = \{\beta \in \mathbb{Z}^m | \beta_i \le \hbar_i, \forall i \le m\}$ and $\forall j \notin \hat{\mathcal{H}}$ let B_j^t to be the set of all $\beta \in \mathbf{B}^t$ such that $\beta \ge g_j$. Initialize the lower bound $l^0(\beta) = 0\ \forall \beta \in \mathbf{B}^t$. For all $j \notin \hat{\mathcal{H}}$, if $g_j \in \mathbf{B}^t$, set $l^0(g_j) = \gamma_j$ and insert g_j into a vector list $\mathcal{L}$. Set $l^1(\beta) = l^0(\beta)$ for all $\beta \in \mathbf{B}^t$. Set $k \leftarrow 1$.
Step 2: Denote the kth vector in $\mathcal{L}$ by β^k and the ith element of a vector β by β_i. Let $\beta = \beta^k$. Update all vectors β' such that $\beta' \in \mathbf{B}^t$ and $\beta' \ge \beta^k$ with the following lexicographic order:
(2a) Set $\beta_1 \leftarrow \beta_1 + 1$ and $l^k(\beta) \leftarrow \max\{l^k(\beta), l^k(\beta^k) + l^k(\beta - \beta^k)\}$.
(2b) If $\beta_1 \ge \hbar_1$, go to Step (2c); otherwise, go to Step (2a).
(2c) If $\beta_i \ge \hbar_i$ for all $i = 1, \ldots, m$, go to Step 3. Otherwise, let $s = \min\{i : \beta_i < \hbar_i\}$. Set $\beta_i \leftarrow \beta_i^k$ for $i = 1, \ldots, s-1$. Set $\beta_s \leftarrow \beta_s + 1$ and go to Step (2a).
Step 3: If $k = |\mathcal{L}|$, go to Step 4. Otherwise, $l^{k+1}(\beta) \leftarrow l^k(\beta)$ for all $\beta \in \mathbf{B}^t$, set $k \leftarrow k + 1$ and go to Step 2.
Step 4: For all $\beta \in \cup_{j \notin \hat{\mathcal{H}}} B_j^t$, update $v^t(\beta + \hat{G}x^t) \leftarrow \max\{l^k(\beta) + y^T\hat{\Sigma}x^t + \hat{c}^T x^t, v^{t-1}(\beta + \hat{G}x^t)\}$. If $t = |\hat{X}|$, stop. Otherwise, empty out vector list $\mathcal{L}$, set $t \leftarrow t + 1$ and go to Step 1.

Theorem 7 *The Sparse-Enumeration Algorithm terminates with optimal solutions to* $z(\cdot)$ *in at most* $|\hat{X}|$ *iterations of* t.

Proposition 19 *The Sparse-Enumeration Algorithm runs in* $O(\sum_{x^t \in \hat{X}} n|\mathbf{B}^t|)$ *time.*

The proofs of Theorem 7 and Proposition 19 are similar to those of Theorem 6 and Proposition 18, respectively.

7 Computational experiments

7.1 Design of experiments

Our computational experiments consist of two main sections. In Sect. 7.3 we test the algorithms for constructing the value function of parameterized QIPs. Then in Sect. 7.4 we test the algorithms for finding the optimal tender of $(P2)$. Given value functions of the first- and second-stage QIPs, to find an optimal tender we consider using the branch-and-bound algorithm (B&B), the minimal tender approach (MT) and also exhaustive search over $\mathbf{B^1}$. In all of our computational tests, exhaustive search was several orders of magnitude slower than both the B&B algorithm and the MT approach. Hence, we do not report computational results on the exhaustive search approach.

In our implementation of the B&B algorithm, we follow standard strategies. In Step 1 (subproblem selection), we choose the hyper-rectangle k that has the smallest upper bound v^k. In Step 3 (subproblem partitioning), we choose the dimension i' that has the largest range, i.e. $i' \in \text{argmax}\{(\eta_i^k - \lambda_i^k) \mid i \in \{1, \ldots, m_2\}\}$. Moreover, we set the initial global lower bound to the maximum between the objective values of $(P2)$ with respect to $\mathbf{0}$ and b^1, i.e. $\max\{\mathbb{E}_\xi \phi(h(\omega)), \psi(b^1) + \mathbb{E}_\xi \phi(h(\omega) - b^1)\}$, where b^1 is the largest vector in $\mathbf{B}^1$ componentwise. Note that proper selection of the rules used in Step 1 and Step 3 as well as the method to generate the initial global lower bound may affect the performance of the B&B algorithm significantly. We leave further tuning of these parameters for future research.

With the MT approach, we do not explore the possible computational benefits of the reduced formulation (9). After obtaining the first-stage value function, we check if each $\beta \in \mathbf{B}^1$ is a minimal tender by definition, and then form the minimal tender set Θ. An optimal solution to $(P2)$ is obtained by evaluating the objective function with respect to each $\beta \in \Theta$.

Computational experiments are conducted on an SGI Altix 4700 shared-memory machine with a single 1.66 GHz CPU. All reported solution times in Tables 3, 4, 5, 6, 7, 8, 9 and 10 are in seconds.

7.2 Instance generation

We consider two-stage stochastic quadratic integer programming instances whose first- and second-stage objective functions are given by $(\chi_1^T \bar{x})(\sigma_1^T x) + c^T x$ and $(\chi_2^T \bar{y})(\sigma_2^T y) + d^T y$, respectively. We have that $\sigma_1 \in \mathbb{Z}^{n_1}, \sigma_2 \in \mathbb{Z}^{n_2}, \chi_1 \in \mathbb{Z}^{a_1}$ for

$a_1 \leq n_1$ and $\chi_2 \in \mathbb{Z}^{a_2}$ for $a_2 \leq n_2$. Note that $\bar{x} \in \mathbb{Z}^{a_1}$ and $\bar{y} \in \mathbb{Z}^{a_2}$ are variable vectors whose elements are composed of a_1 and a_2 dimensional subsets of the indices in x and y, respectively. We do not consider the first-stage constraints $Ax \leq b$ as they can be embedded into the technology matrix T by setting the corresponding rows of the recourse matrix W to $\mathbf{0}$. To test all four algorithms for finding the value function in the first phase and both approaches for finding the optimal tender in the second phase we assume that $\mathbf{B}^k = \mathcal{B}^k \cap \mathbb{Z}_+^{m_2}$, where $\mathcal{B}^k = \prod_{i=1}^{m_2}[0, b_i^k]$ for $k = 1, 2$ were constructed such that $b_i^1 = \min_{\omega \in \Omega} h_i(\omega)$ and $b_i^2 = \max_{\omega \in \Omega} h_i(\omega)$ for $i = 1, \ldots, m_2$.

We randomly generate two different testbeds under the assumptions **A1** − **A4**. Testbed 1 in Table 1 has 45 instance classes and Testbed 2 in Table 2 has 15 instance classes. There are more instances in Testbed 1 as we vary a_1 and a_2 between 10 and 20; whereas in Testbed 2 we always set $a_1 = n_1$ and $a_2 = n_2$. Note that the quadratic objective functions of Testbed 1 instances are specially structured (i.e. sparse) so that the Sparse-Fixing Algorithm and the Sparse-Enumeration Algorithm apply. However, instances in Testbed 2 do not exhibit any special structure.

The instances are named IC$m - KX$, where $m = 1, 2$ is the testbed index, $K = 1, \ldots, 45$ is the instance class index, and $X \in \{S, L\}$ is the instance size index. For a particular instance class K in testbed m, size index S denotes the smaller instance; whereas size index L denotes the larger instance. There are $|\Omega| = 279936$ scenarios for each instance in Testbed 1 and 2, which is equal to the largest number of scenarios in Kong et al. [35]. Deterministic parameters $c, \chi_1, \sigma_1, d, \chi_2, \sigma_2$ are generated from $U[1, 1000]$. We set the density of technology matrix T and recourse matrix W to 0.7, which is modeled by a Bernoulli distribution. In Tables 1 and 2, the numbers listed under T, W and $h(\omega)$ are the lower and upper bounds of the uniform distribution that is used to generate nonzero elements of these parameters. Our instances are available online [54].

7.3 Finding the value function

7.3.1 The diagonal-Q algorithm versus the exact-superadditive algorithm

In this section we test the Exact-Superadditive Algorithm and the Diagonal-Q Algorithm. To obtain diagonal instances we delete the off-diagonal elements of the instances in Testbed 1 and Testbed 2. First, we run both algorithms on small instances of Testbed 2, and present the computational results in Table 3.

Note that in Table 3 the Exact-Superadditive Algorithm is often faster than the Diagonal-Q Algorithm as the number of constraints increases. This is due to the fact that $|\mathbf{B}|$ grows exponentially as the number of constraints increases, and from Proposition 17 the running time of the Diagonal-Q Algorithm is $O(n\mu_{\max}|\mathbf{B}|)$.

We also test the Diagonal-Q Algorithm on large instances of Testbed 2. The goal of this test is to demonstrate that the size of diagonal instances of $(P1)$ (as measured by the size of the extensive form) that can be solved efficiently using the Diagonal-Q Algorithm is much larger than that of the Exact-Superadditive Algorithm. We present the computational results in Table 4. Note that the Exact-Superadditive Algorithm can

Table 1 Characteristics of instances in Testbed 1, e.g. instance IC1 − 1S has $n_1 = 200$, IC1 − 1L has $n_1 = 500$

IC1-KX	m_2	IC1-KS					IC1-KL				
		a_1/n_1	a_2/n_2	T	W	$h(\omega)$	a_1/n_1	a_2/n_2	T	W	$h(\omega)$
IC1-1X	3	10/200	20/200	[10,100]	[50,150]	[100,200]	10/500	20/500	[10,100]	[100,350]	[200,500]
IC1-2X	3	10/300	20/300	[10,100]	[50,200]	[100,300]	10/750	20/500	[10,100]	[200,475]	[200,750]
IC1-3X	3	10/400	20/400	[10,100]	[75,250]	[100,400]	10/1000	20/500	[10,100]	[500,750]	[200,1000]
IC1-4X	3	15/200	15/200	[10,100]	[50,150]	[100,200]	15/500	15/500	[10,100]	[100,350]	[200,500]
IC1-5X	3	15/300	15/300	[10,100]	[50,200]	[100,300]	15/750	15/500	[10,100]	[200,475]	[200,750]
IC1-6X	3	15/400	15/400	[10,100]	[75,250]	[100,400]	15/1000	15/500	[10,100]	[500,750]	[200,1000]
IC1-7X	3	20/200	10/200	[10,100]	[50,150]	[100,200]	20/500	10/500	[10,100]	[100,350]	[200,500]
IC1-8X	3	20/300	10/300	[10,100]	[50,200]	[100,300]	20/750	10/500	[10,100]	[200,475]	[200,750]
IC1-9X	3	20/400	10/400	[10,100]	[75,250]	[100,400]	20/1000	10/500	[10,100]	[500,750]	[200,1000]
IC1-10X	4	10/200	20/200	[10,50]	[25,62]	[50,75]	10/500	20/500	[1,25]	[25,75]	[50,100]
IC1-11X	4	10/300	20/300	[10,50]	[25,75]	[50,100]	10/750	20/500	[1,25]	[50,100]	[50,150]
IC1-12X	4	10/400	20/400	[10,50]	[25,87]	[50,125]	10/1000	20/500	[1,25]	[75,125]	[50,200]
IC1-13X	4	15/200	15/200	[10,50]	[25,62]	[50,75]	15/500	15/500	[1,25]	[25,75]	[50,100]
IC1-14X	4	15/300	15/300	[10,50]	[25,75]	[50,100]	15/750	15/500	[1,25]	[50,100]	[50,150]
IC1-15X	4	15/400	15/400	[10,50]	[75,87]	[50,125]	15/1000	15/500	[1,25]	[75,125]	[50,200]
IC1-16X	4	20/200	10/200	[10,50]	[25,62]	[50,75]	20/500	10/500	[1,25]	[25,75]	[50,100]
IC1-17X	4	20/300	10/300	[10,50]	[25,75]	[50,100]	20/750	10/500	[1,25]	[50,100]	[50,150]
IC1-18X	4	20/400	10/400	[10,50]	[25,87]	[50,125]	20/1000	10/500	[1,25]	[75,125]	[50,200]
IC1-19X	5	10/200	20/200	[1,20]	[15,30]	[20,35]	10/500	20/500	[1,10]	[15,30]	[20,50]
IC1-20X	5	10/300	20/300	[1,20]	[15,30]	[20,40]	10/750	20/500	[1,10]	[15,30]	[20,60]
IC1-21X	5	10/400	20/400	[1,20]	[15,30]	[20,45]	10/1000	20/500	[1,10]	[15,30]	[20,70]
IC1-22X	5	15/200	15/200	[1,20]	[15,30]	[20,35]	15/500	15/500	[1,10]	[15,30]	[20,50]
IC1-23X	5	15/300	15/300	[1,20]	[15,30]	[20,40]	15/750	15/500	[1,10]	[15,30]	[20,60]
IC1-24X	5	15/400	15/400	[1,20]	[15,30]	[20,45]	15/1000	15/500	[1,10]	[15,30]	[20,70]
IC1-25X	5	20/200	10/200	[1,20]	[15,30]	[20,35]	20/500	10/500	[1,10]	[15,30]	[20,50]
IC1-26X	5	20/300	10/300	[1,20]	[15,30]	[20,40]	20/750	10/500	[1,10]	[15,30]	[20,60]
IC1-27X	5	20/400	10/400	[1,20]	[15,30]	[20,45]	20/1000	10/500	[1,10]	[20,35]	[20,70]
IC1-28X	6	10/200	20/200	[1,10]	[10,20]	[10,20]	10/500	20/500	[1,5]	[10,20]	[10,25]
IC1-29X	6	10/300	20/300	[1,10]	[10,20]	[10,23]	10/750	20/500	[1,5]	[10,20]	[10,30]
IC1-30X	6	10/400	20/400	[1,10]	[10,20]	[10,25]	10/1000	20/500	[1,5]	[10,20]	[10,35]
IC1-31X	6	15/200	15/200	[1,10]	[10,20]	[10,20]	15/500	15/500	[1,5]	[10,20]	[10,25]
IC1-32X	6	15/300	15/300	[1,10]	[10,20]	[10,23]	15/750	15/500	[1,5]	[10,20]	[10,30]
IC1-33X	6	15/400	15/400	[1,10]	[10,20]	[10,25]	15/1000	15/500	[1,5]	[10,20]	[10,35]
IC1-34X	6	20/200	10/200	[1,10]	[10,20]	[10,20]	20/500	10/500	[1,5]	[10,20]	[10,25]
IC1-35X	6	20/300	10/300	[1,10]	[10,20]	[10,23]	20/750	10/500	[1,5]	[10,20]	[10,30]
IC1-36X	6	20/400	10/400	[1,10]	[10,20]	[10,25]	20/1000	10/500	[1,5]	[10,20]	[10,35]
IC1-37X	7	10/200	20/200	[1,5]	[5,10]	[5,10]	10/500	20/500	[1,5]	[5,10]	[5,10]
IC1-38X	7	10/300	20/300	[1,5]	[5,10]	[5,13]	10/750	20/500	[1,5]	[5,10]	[5,15]
IC1-39X	7	10/400	20/400	[1,5]	[5,10]	[5,15]	10/1000	20/500	[1,5]	[5,10]	[5,20]
IC1-40X	7	15/200	15/200	[1,5]	[5,10]	[5,10]	15/500	15/500	[1,5]	[5,10]	[5,10]

Table 1 continued

IC1-KX	m_2	IC1-KS					IC1-KL				
		a_1/n_1	a_2/n_2	T	W	$h(\omega)$	a_1/n_1	a_2/n_2	T	W	$h(\omega)$
IC1-41X	7	15/300	15/300	[1,5]	[5,10]	[5,13]	15/750	15/500	[1,5]	[5,10]	[5,15]
IC1-42X	7	15/400	15/400	[1,5]	[5,10]	[5,15]	15/1000	15/500	[1,5]	[5,10]	[5,20]
IC1-43X	7	20/200	10/200	[1,5]	[5,10]	[5,10]	20/500	10/500	[1,5]	[5,10]	[5,10]
IC1-44X	7	20/300	10/300	[1,5]	[5,10]	[5,13]	20/750	10/500	[1,5]	[5,10]	[5,15]
IC1-45X	7	20/400	10/400	[1,5]	[5,10]	[5,15]	20/1000	10/500	[1,5]	[5,10]	[5,20]

Table 2 Characteristics of instances in Testbed 2, e.g. instance IC2 − 1S has $a_1 = n_1 = 200$, IC2 − 1L has $a_1 = n_1 = 500$

IC2-KX	m_2	IC2-KS					IC2-KL				
		a_1/n_1	a_2/n_2	T	W	$h(\omega)$	a_1/n_1	a_2/n_2	T	W	$h(\omega)$
IC2-1X	3	200/200	200/200	[10,100]	[50,150]	[100,200]	500/500	500/500	[1,100]	[50,150]	[100,200]
IC2-2X	3	300/300	300/300	[10,100]	[50,200]	[100,300]	750/750	500/500	[1,100]	[50,200]	[100,300]
IC2-3X	3	400/400	400/400	[10,100]	[75,250]	[100,400]	1000/1000	500/500	[1,100]	[50,250]	[100,400]
IC2-4X	4	200/200	200/200	[10,50]	[25,62]	[50,75]	500/500	500/500	[1,50]	[25,62]	[50,75]
IC2-5X	4	300/300	300/300	[10,50]	[25,75]	[50,100]	750/750	500/500	[1,50]	[25,75]	[50,100]
IC2-6X	4	400/400	400/400	[10,50]	[25,87]	[50,125]	1000/1000	500/500	[1,50]	[25,87]	[50,125]
IC2-7X	5	200/200	200/200	[5,20]	[15,30]	[20,35]	500/500	500/500	[1,10]	[15,30]	[20,50]
IC2-8X	5	300/300	300/300	[5,20]	[15,30]	[20,40]	750/750	500/500	[1,10]	[15,30]	[20,60]
IC2-9X	5	400/400	400/400	[5,20]	[15,30]	[20,45]	1000/1000	500/500	[1,10]	[20,35]	[20,70]
IC2-10X	6	200/200	200/200	[5,10]	[10,20]	[10,20]	500/500	500/500	[1,5]	[10,20]	[10,25]
IC2-11X	6	300/300	300/300	[5,10]	[10,20]	[10,23]	750/750	500/500	[1,5]	[10,20]	[10,30]
IC2-12X	6	400/400	400/400	[5,10]	[10,20]	[10,25]	1000/1000	500/500	[1,5]	[10,20]	[10,35]
IC2-13X	7	200/200	200/200	[1,5]	[5,10]	[5,10]	500/500	500/500	[1,5]	[5,10]	[5,10]
IC2-14X	7	300/300	300/300	[1,5]	[5,10]	[5,13]	750/750	500/500	[1,5]	[5,10]	[5,15]
IC2-15X	7	400/400	400/400	[1,5]	[5,10]	[5,15]	1000/1000	500/500	[1,5]	[5,10]	[5,20]

not solve any of the instances reported in Table 4 in a reasonable amount of time as they have too many variables to solve using a DP algorithm.

7.3.2 The Sparse-enumeration Algorithm versus the Exact-Superadditive Algorithm

In this section we test the Exact-Superadditive Algorithm and the Sparse-Enumeration Algorithm. First, we run both algorithms on small instances of Testbed 1. Table 5 presents the number iterations as well as the total solution time required for constructing both value functions.

As seen in Table 5, the Exact-Superadditive Algorithm outperforms the Sparse-Enumeration Algorithm in only 2 instances in the first stage; and in 32 instances in

Table 3 Evaluating the value function of diagonal small instances in Testbed 2 using the Exact-Superadditive Algorithm and the Diagonal-Q Algorithm

IC2-KS	First stage				Second stage			
	Exact-sup.		Diagonal-Q		Exact-sup.		Diagonal-Q	
	Iters	Time	Iters	Time	Iters	Time	Iters	Time
IC2-1S	17	3	500	5	158	7	470	40
IC2-2S	47	85	814	7	31	150	938	313
IC2-3S	309	403	1042	9	389	806	1254	1015
IC2-4S	375	4	402	14	591	13	418	77
IC2-5S	323	9	652	25	575	35	686	576
IC2-6S	964	35	856	30	589	184	986	2313
IC2-7S	470	5	414	6	721	42	402	89
IC2-8S	1100	19	624	10	2731	86	614	354
IC2-9S	2864	60	820	11	258	69	894	1269
IC2-10S	711	4	400	2	1727	173	400	46
IC2-11S	1090	9	600	2	1408	79	604	244
IC2-12S	706	6	800	3	2518	215	804	638
IC2-13S	4588	19	400	1	2289	24	406	9
IC2-14S	5062	56	624	0	3087	85	604	159
IC2-15S	1713	23	822	0	1612	160	832	879

Table 4 Evaluating the value function of diagonal large instances in Testbed 2 using the Diagonal-Q Algorithm

IC2-KL	First stage		Second stage	
	Iters	Time	Iters	Time
IC2-1L	1476	14	1204	103
IC2-2L	2284	21	1506	508
IC2-3L	3154	30	1786	1526
IC2-4L	1316	75	1050	198
IC2-5L	1846	109	1140	1012
IC2-6L	2428	121	1292	3019
IC2-7L	2398	135	1246	3254
IC2-8L	3500	198	2026	12261
IC2-9L	4616	262	2008	24374
IC2-10L	2132	37	1006	815
IC2-11L	3246	64	1080	4481
IC2-12L	4282	82	1242	17621
IC2-13L	1006	1	1000	23
IC2-14L	1528	2	1044	1085
IC2-15L	2040	2	2006	17408

Table 5 Evaluating the value function of small instances in Testbed 1 using the Exact-Superadditive Algorithm and the Sparse-Enumeration Algorithm

IC1-KS	First stage				Second stage			
	Exact-sup.		Sparse-enum.		Exact-sup.		Sparse-enum.	
	Iters	Time	Iters	Time	Iters	Time	Iters	Time
IC1-1S	846	134	12	5	252	16	46	59
IC1-2S	220	2936	69	26	624	2009	219	1151
IC1-3S	395	792	12	10	446	4800	196	3537
IC1-4S	120	16	72	12	252	15	87	92
IC1-5S	151	529	113	31	405	1075	89	811
IC1-6S	654	1580	46	15	1243	10942	146	2777
IC1-7S	453	67	82	8	351	34	35	66
IC1-8S	537	1271	184	23	773	2375	122	929
IC1-9S	269	3953	95	32	762	3882	80	3097
IC1-10S	630	16	16	20	665	16	23	89
IC1-11S	1425	44	11	23	458	74	79	1074
IC1-12S	297	62	20	55	3074	3081	121	4700
IC1-13S	694	19	16	19	458	14	19	94
IC1-14S	617	37	19	31	1124	117	33	733
IC1-15S	820	119	29	39	2216	2068	58	3453
IC1-16S	1027	11	22	17	285	10	25	102
IC1-17S	579	36	31	34	990	120	14	803
IC1-18S	789	65	36	47	2337	2000	35	3368
IC1-19S	1363	155	14	13	654	33	21	96
IC1-20S	16523	6278	16	22	529	39	30	403
IC1-21S	3608	3002	36	43	2719	324	56	1330
IC1-22S	3092	125	33	16	594	27	16	105
IC1-23S	18763	2959	77	52	2227	75	18	388
IC1-24S	11934	13002	28	38	2005	311	22	1318
IC1-25S	7703	542	56	16	316	27	13	103
IC1-26S	17271	6100	119	47	876	53	14	421
IC1-27S	26982	8831	38	32	2404	233	17	1364
IC1-28S	7358	55	12	3	330	43	21	71
IC1-29S	17913	447	14	6	2442	140	21	296
IC1-30S	5168	813	22	12	2866	248	21	812
IC1-31S	11360	142	17	4	1180	111	16	65
IC1-32S	8139	172	20	6	1814	210	16	279
IC1-33S	15148	2817	48	12	3379	306	17	702
IC1-34S	4651	78	44	6	1683	130	11	62
IC1-35S	9442	425	36	6	1177	177	11	256
IC1-36S	5003	609	76	14	4894	298	11	730
IC1-37S	3566	22	12	1	1919	52	21	13

Table 5 continued

IC1-KS	First stage				Second stage			
	Exact-sup.		Sparse-enum.		Exact-sup.		Sparse-enum.	
	Iters	Time	Iters	Time	Iters	Time	Iters	Time
IC1-38S	6050	75	13	0	3106	108	21	191
IC1-39S	6626	279	16	1	16064	1099	31	969
IC1-40S	3330	23	20	1	2569	50	16	13
IC1-41S	1812	66	21	1	3120	118	19	186
IC1-42S	5424	213	18	1	12796	841	24	1088
IC1-43S	4776	23	22	1	2777	61	11	13
IC1-44S	2743	29	38	1	6308	130	11	183
IC1-45S	2203	173	36	1	26800	1638	15	1047

Table 6 Evaluating the value function of small instances in Testbed 2 using the Exact-Superadditive Algorithm

IC2-KS	First stage		Second stage	
	Iters	Time	Iters	Time
IC2-1S	17	3	158	7
IC2-2S	47	85	31	150
IC2-3S	309	403	389	806
IC2-4S	375	4	591	13
IC2-5S	323	9	575	35
IC2-6S	964	35	589	184
IC2-7S	470	5	721	42
IC2-8S	1100	19	2731	86
IC2-9S	2864	60	258	69
IC2-10S	711	4	1727	173
IC2-11S	1090	9	1408	79
IC2-12S	706	6	2518	215
IC2-13S	4588	19	2289	24
IC2-14S	5062	56	3087	85
IC2-15S	1713	23	1612	160

the second stage. This result is due to the fact that the first-stage variable ranges of Testbed 1 instances are relatively higher than that of the second-stage variables.

We further test the individual performance of the Exact-Superadditive Algorithm on small instances of Testbed 2 in which a_1 and a_2 increase up to 400. The computational results are reported in Table 6. Note that the Sparse-Enumeration Algorithm can not solve any one of these instances in a reasonable amount of time as their quadratic objective functions are not sparse.

Finally, we run the Sparse-Enumeration Algorithm on large instances of Testbed 1 in which the number of variables n_1 and n_2 increase up to 1000. The goal of this test is to demonstrate that the size of the instances (as measured by the size of the extensive

Table 7 Evaluating the value function of large instances in Testbed 1 using the Sparse-Enumeration Algorithm

IC1-KL	First stage		Second stage		IC1-KL	First stage		Second stage	
	Iters	Time	Iters	Time		Iters	Time	Iters	Time
IC1-1L	409	3755	202	7037	IC1-28L	223	143	22	930
IC1-2L	274	2493	123	14261	IC1-29L	301	251	41	5602
IC1-3L	93	1891	21	8674	IC1-30L	99	187	122	28444
IC1-4L	27	15	93	4661	IC1-31L	464	146	17	993
IC1-5L	600	3474	95	17395	IC1-32L	525	294	35	5856
IC1-6L	1165	12150	16	8981	IC1-33L	607	391	97	31619
IC1-7L	2992	10796	37	3276	IC1-34L	1289	245	11	867
IC1-8L	4060	13425	41	12320	IC1-35L	991	329	12	5401
IC1-9L	2516	17575	11	10337	IC1-36L	3221	1027	37	26350
IC1-10L	910	2820	75	1351	IC1-37L	13	1	21	26
IC1-11L	521	4172	71	6575	IC1-38L	13	2	41	1301
IC1-12L	399	3102	42	19571	IC1-39L	32	3	240	29852
IC1-13L	1053	2486	41	1440	IC1-40L	23	1	16	25
IC1-14L	2542	6877	67	7216	IC1-41L	22	1	21	1179
IC1-15L	2283	14964	43	21746	IC1-42L	20	2	140	28255
IC1-16L	1432	1942	25	1187	IC1-43L	77	2	11	24
IC1-17L	2673	6462	21	5295	IC1-44L	37	2	22	1362
IC1-18L	3525	10178	34	20144	IC1-45L	30	3	66	24037
IC1-19L	117	324	146	5900					
IC1-20L	164	630	250	25515					
IC1-21L	114	591	233	56721					
IC1-22L	1037	1104	49	3780					
IC1-23L	1017	1457	187	30283					
IC1-24L	854	1688	142	42716					
IC1-25L	2250	1292	42	3774					
IC1-26L	2868	3837	72	21241					
IC1-27L	3173	5122	66	34799					

form) that can be solved efficiently using the Sparse-Enumeration Algorithm is much larger than that of the Exact-Superadditive Algorithm. Table 7 presents the computational results. Note that the Exact-Superadditive Algorithm can not solve any one of these instances in a reasonable amount of time as they have too many variables to solve using a DP algorithm.

7.3.3 The Sparse-Fixing Algorithm versus the Sparse-Enumeration Algorithm

In this section we test the Sparse-Fixing Algorithm and the Sparse-Enumeration Algorithm. First, we run both algorithms on small instances of Testbed 1. Table 8 reports

Table 8 Evaluating the value function of small instances in Testbed 1 using the Sparse-Fixing and the Sparse-Enumeration Algorithms

IC1-KS	First stage				Second stage			
	Sparse fixing		Sparse enum.		Sparse fixing		Sparse enum.	
	Iter	Time	Iter	Time	Iter	Time	Iter	Time
IC1-1S	6	98	12	5	11	2575	46	60
IC1-2S	19	1338	69	26	12	–	219	1152
IC1-3S	7	263	12	10	13	–	196	3536
IC1-4S	25	1456	72	11	25	3683	87	91
IC1-5S	22	1682	113	32	–	–	89	811
IC1-6S	9	561	46	15	–	–	146	2781
IC1-7S	10	408	82	8	14	2354	35	66
IC1-8S	22	1681	184	23	–	–	122	931
IC1-9S	12	707	95	32	–	–	80	3096
IC1-10S	9	835	16	20	6	1827	23	89
IC1-11S	5	402	11	28	–	–	79	1073
IC1-12S	8	1263	20	47	–	–	121	4697
IC1-13S	6	385	16	18	8	3533	19	93
IC1-14S	9	1010	19	29	–	–	33	868
IC1-15S	10	2018	29	45	–	–	58	3446
IC1-16S	7	495	22	17	10	4955	25	101
IC1-17S	12	1790	31	32	–	–	14	627
IC1-18S	8	1164	36	40	–	–	35	3367
IC1-19S	7	333	14	13	5	1439	21	97
IC1-20S	8	709	16	21	–	–	30	405
IC1-21S	14	3483	36	43	–	–	56	1336
IC1-22S	15	1309	33	16	6	1919	16	105
IC1-23S	38	12675	77	52	–	–	18	391
IC1-24S	9	1063	28	37	–	–	22	1320
IC1-25S	7	976	56	17	13	2465	13	104
IC1-26S	14	1996	119	48	–	–	14	422
IC1-27S	10	1261	38	31	–	–	17	1372
IC1-28S	5	50	12	4	6	880	21	71
IC1-29S	8	207	14	6	–	–	21	294
IC1-30S	12	720	22	11	–	–	21	807
IC1-31S	6	75	17	4	6	838	16	64
IC1-32S	8	176	20	5	–	–	16	277
IC1-33S	12	530	48	11	–	–	17	697
IC1-34S	6	421	44	5	15	750	11	61
IC1-35S	8	180	36	6	–	–	11	255
IC1-36S	20	1558	76	14	–	–	11	724
IC1-37S	6	8	12	0	6	134	21	13

Table 8 continued

IC1-KS	First stage				Second stage			
	Sparse fixing		Sparse enum.		Sparse fixing		Sparse enum.	
	Iter	Time	Iter	Time	Iter	Time	Iter	Time
IC1-38S	7	15	13	1	6	3277	21	188
IC1-39S	9	50	16	1	–	–	31	947
IC1-40S	7	11	20	0	6	136	16	13
IC1-41S	8	21	21	0	–	–	19	182
IC1-42S	7	23	18	1	–	–	24	1061
IC1-43S	6	9	22	1	6	128	11	12
IC1-44S	8	18	38	1	6	3462	11	183
IC1-45S	13	70	36	1	–	–	15	1020

"–" means that the algorithm runs out of memory

Table 9 The Sparse-Fixing Algorithm outperforms the Sparse-Enumeration Algorithm when $\left(\chi^T \bar{x}\right)$ has a small range and large domain

m	$\beta_{\max}$	Sparse-fixing		Sparse-enum.	
		Iters	Time	Iters	Time
3	7	22	3	1735000	18
3	8	25	6	4506125	54
3	9	28	15	11079200	153
4	7	29	7	5778720	122
4	8	33	15	15130125	478
4	9	37	1959	38542900	7894
5	7	36	655	13475808	5415
5	8	41	2102	–	>20 h
5	9	46	6390	–	>20 h
6	7	43	1378	–	>20 h
6	8	49	4018	–	>20 h
6	9	55	13950	–	>20 h
7	7	50	7882	–	>20 h
7	8	57	32330	–	>20 h

the number iterations as well as the total solution time required for constructing value functions. The Sparse-Fixing algorithm is sensitive to the magnitudes of the deterministic parameters χ_1, χ_2. Hence, for the instances reported in Table 8, these parameters are generated from $U[1, 5]$. As seen in Table 8, the Sparse-Enumeration Algorithm runs faster than the Sparse-Fixing Algorithm for all considered instances, sometimes by several orders of magnitude.

Recall that the Sparse-Fixing Algorithm enumerates all possible $(\chi_1^T \bar{x})$ values, whereas the Sparse-Enumeration Algorithm iterates over a_1-dimensional candidate solution vectors to $\bar{x}$. In Table 9, we demonstrate that the Sparse-Fixing Algorithm may also outperform the Sparse-Enumeration Algorithm in some cases where $(\chi^T \bar{x})$

Table 10 Finding the optimal tender of large instances in Testbed 1

IC1-KL	Solution time		Search space size		
	B&B	MT	$\lvert\Theta\rvert$	$\lvert\mathbf{B}^1\rvert$	$\lvert\mathbf{B}^2\rvert$
IC1-1L	305	137	1489	8000000	125000000
IC1-2L	167	368	3675	8000000	421875000
IC1-3L	42	387	3196	8000000	1000000000
IC1-4L	14	46	512	8000000	125000000
IC1-5L	88	204	1989	8000000	421875000
IC1-6L	268	325	2673	8000000	1000000000
IC1-7L	655	111	1185	8000000	125000000
IC1-8L	188	207	2040	8000000	421875000
IC1-9L	122	1035	8620	8000000	1000000000
IC1-10L	32	19166	224027	6250000	100000000
IC1-11L	150	3334	33385	6250000	506250000
IC1-12L	59	13735	89910	6250000	1600000000
IC1-13L	89	4471	52202	6250000	100000000
IC1-14L	12	20849	210458	6250000	506250000
IC1-15L	165	3529	23448	6250000	1600000000
IC1-16L	41	5836	68224	6250000	100000000
IC1-17L	38	2481	24896	6250000	506250000
IC1-18L	20	33175	160343	6250000	1600000000
IC1-19L	33	5371	56196	3200000	312500000
IC1-20L	75	1554	14355	3200000	777600000
IC1-21L	85	17872	126713	3200000	1680700000
IC1-22L	99	1757	18605	3200000	312500000
IC1-23L	20	12532	117717	3200000	777600000
IC1-24L	24	26301	178154	3200000	1680700000
IC1-25L	174	1816	19227	3200000	312500000
IC1-26L	15	220	2032	3200000	777600000
IC1-27L	70	1121	7384	3200000	1680700000
IC1-28L	18	4609	47046	1000000	244140625
IC1-29L	70	2017	18148	1000000	729000000
IC1-30L	30	10369	66766	1000000	1838265625
IC1-31L	24	2174	22189	1000000	244140625
IC1-32L	126	4475	40083	1000000	729000000
IC1-33L	36	14815	75078	1000000	1838265625
IC1-34L	21	2530	25829	1000000	244140625
IC1-35L	22	4076	36442	1000000	729000000
IC1-36L	30	18059	68271	1000000	1838265625
IC1-37L	11	77	935	78125	10000000
IC1-38L	30	195	1806	78125	170859375
IC1-39L	59	288	1839	78125	1280000000

Table 10 continued

IC1-KL	Solution time		Search space size		
	B&B	MT	$\|\Theta\|$	$\|\mathbf{B}^1\|$	$\|\mathbf{B}^2\|$
IC1-40L	52	78	953	78125	10000000
IC1-41L	21	198	1075	78125	170859375
IC1-42L	42	419	2057	78125	1280000000
IC1-43L	526	64	784	78125	10000000
IC1-44L	18	96	891	78125	170859375
IC1-45L	22	273	1543	78125	1280000000

has a small range and large domain on a set of $(PQIP)$ instances in which $\frac{1}{2}x^T Qx = (\chi^T \bar{x})(\sigma^T x)$. We generate these instances by setting each element of $\chi \in \mathbb{Z}^a$ to 1, and each column of the G matrix that appears as a variable in the χ vector to a random $0-1$ unit vector. All other nonzero elements of the G matrix are generated from $U[3, 5]$. Furthermore, we set $a = 10$, $n = 200$, and generate σ and c vectors from $U[1, 1000]$. As seen in Table 9, the number iterations of the Sparse-Enumeration Algorithm is very large for such instances.

7.4 Finding the optimal tender

In this section we test the algorithms proposed for finding the optimal tender. These tests are conducted on large instances of Testbed 1 after computing value functions in both stages. Note that performances of the algorithms presented in this section do not depend on the number of variables. Therefore, we do not repeat our experiments neither with small instances of Tesbed 1 nor with any instance from Testbed 2.

Table 10 reports the time required for finding an optimal tender using the branch-and-bound algorithm (B&B) and the minimal tender approach (MT). We also report the size of the minimal tender set $|\Theta|$, and the sizes of the first- and second-stage feasible right-hand side sets $|\mathbf{B}^1|$ and $|\mathbf{B}^2|$.

Recall that the MT approach eliminates the first-stage right-hand sides that are not minimal tenders (see Theorem 3). Then it enumerates over all right-hand sides in the minimal tender set Θ. Not surprisingly, the MT approach tends to outperform the B&B algorithm when $|\Theta|$ is small, e.g. IC1 $-1L$, IC1 $-7L$ and IC1 $-43L$ instances. Generally, the B&B algorithm outperforms the MT approach in most instances. Furthermore, its performance does not vary a lot as $|\Theta|$ gets larger.

7.5 Observations from the computational experiments

The overall computational results show that our approach is relatively insensitive to the number of decision variables in both stages but sensitive to the number of constraints and the numbers of feasible right-hand sides in $\mathbf{B}^1$ and $\mathbf{B}^2$. We note that the portion of the total running time spent in the first and second phase varies depending on the algorithms used in those phases.

In the literature, the largest QIPs that have been solved so far are diagonal instances and they have no more than 2000 columns and 2000 rows [48]. By exploiting the special structure of the two-stage stochastic quadratic integer programs, we can solve (possibly indefinite) instances of $(P1)$ whose extensive forms (3) are hundreds of orders of magnitude larger than those instances solved in the literature.

Usually integer programs with (possibly indefinite) quadratic objectives are substantially harder to solve than their linear objective counterparts. However, extensive forms (3) of the instances that we solve in this paper have the similar order of magnitude as those of Kong et al. [35], which are the largest stochastic linear integer programs solved in the literature so far (as measured by the extensive form size). From this observation we conclude that our proposed two-phase solution framework alleviates the difficulties arising due to quadratic objective functions for the class of problems considered in this paper.

8 Concluding remarks

We present an algorithmic framework for a class of two-stage stochastic quadratic integer programs where the uncertainty only appears in the second-stage right-hand sides. The main contribution of the paper is twofold. First, we derive some theoretical properties of QIP value functions. These properties may be useful in sensitivity analysis of quadratic integer programs [18,25]. Second, we use these properties as well as superadditivity to develop efficient algorithms for computing value functions of QIPs. We then apply a dual reformulation and use a generic global branch-and-bound algorithm and a level-set approach to find an optimal tender.

This paper represents an important first step towards more general two-stage stochastic quadratic integer programs where uncertainty appears in the second-stage objective and constraint matrix, as well as the right-hand side. We note that our approach is amenable to solve general two-stage stochastic quadratic integer programs as long as the scenarios may be divided into relatively few groups that share the same objective functions and constraint matrices. For such instances, the value function must be found for the first stage and each group of scenarios.

The Exact-Superadditive Algorithm presented in Sect. 6.1 provides the flexibility for improvements that would be interesting for further investigation. We use a DP algorithm to solve the quadratic integer programs arising in Step 1, which limits the number of variables that can be handled. Various objectives regarding the computational preference between solving quadratic integer programs and applying superadditive dual properties may lead to different procedural selections.

The major limitation of our two-phase solution approach is the explicit storage of value functions in computer memory. This is why our computations are based on instances that have large number of columns and scenarios but relatively few rows. One approach to overcome this limitation is to seek more efficient ways to store value functions, such as using generating functions [19]. Another approach is to modify the global branch-and-bound algorithm to calculate the solution on a subset of right-hand sides so that only a portion of the value function needs to be stored at any time.

Acknowledgments We thank two anonymous referees for their insightful and constructive comments. The authors were supported by Air Force Office of Scientific Research grant FA9550-08-1-0268, National Science Foundation grants CMMI-0826141 and CMMI-0620780, and a grant from the Central Research Development Fund of the University of Pittsburgh.

References

1. Adams, W.P., Forrester, R.: Linear forms of nonlinear expressions: new insights on old ideas. Oper. Res. Lett. **35**(4), 510–518 (2007)
2. Adams, W.P., Sherali, H.D.: A tight linearization and an algorithm for zero-one quadratic programming problems. Manag. Sci. **32**(10), 1274–1290 (1986)
3. Adams, W.P., Sherali, H.D.: Mixed-integer bilinear programming problems. Math. Program. **59** (1–3), 279–305 (1993)
4. Adams, W.P., Forrester, R., Glover, F.: Comparisons and enhancement strategies for linearizing mixed 0-1 quadratic programs. Discret. Optim. **1**(2), 99–120 (2004)
5. Agrawal, S.C.: On integer solutions to quadratic programs by a branch-and-bound technique. Trabajos de Estadìstica y de Investigaciòn Operativa **25**(1–2), 65–70 (1974)
6. Agrawal, S.C.: On mixed-integer quadratic programs. Naval Res. Logist. Q. **21**(2), 289–297 (1974)
7. Agrawal, S.C.: An alternative method on integer solutions to quadratic programs by a branch-and-bound technique. Trabajos de Estadìstica y de Investigaciòn Operativa **27**(1–3), 185–192 (1976)
8. Ahmed, S., Garcia, R.: Dynamic capacity acquisition and assignment under uncertainty. Ann. Oper. Res. **124**(1–4), 267–283 (2003)
9. Ahmed, S., Tawarmalani, M., Sahinidis, N.V.: A finite branch and bound algorithm for two-stage stochastic integer programs. Math. Program. **100**(2), 355–377 (2004)
10. Al-Khayyal, F.A., Larsen, C.: Global optimization of a quadratic function subject to a bounded mixed integer constraint set. Ann. Oper. Res. **25**(1–4), 169–180 (1990)
11. Balas, E.: Duality in discrete programming II: the quadratic case. Manag. Sci. **16**(1), 14–32 (1969)
12. Bank, B., Hansel, R.: Stability of mixed-integer quadratic programming problems. Math. Program. Study **21**, 1–17 (1984)
13. Blair, C.E., Jeroslow, R.G.: The value function of an integer program. Math. Program. **23**(1), 237–273 (1982)
14. Bretthauer, K.M., Shetty, B.: The nonlinear resource allocation problem. Operations Research **43**(4), 670–683 (1995)
15. Caprara, A., Pisinger, D., Toth, P.: Exact solution of the quadratic knapsack problem. INFORMS J. Comput. **11**(2), 125–137 (1999)
16. Carøe, C.C., Tind, J.: L-shaped decomposition of two-stage stochastic programs with integer recourse. Math. Program. **83**(1–3), 451–464 (1998)
17. Cook, W., Gerards, A.M.H., Schrijver, A., Tardos, E.: Sensitivity results in integer linear programming. Math. Program. **34**(3), 251–264 (1986)
18. Daniel, J.W.: Stability of the solution of definite quadratic programs. Math. Program. **5**(1), 41–53 (1973)
19. De Loera, J.A., Haws, D., Hemmecke, R., Huggins, P., Strumfels, B., Yoshida, R.: Short rational functions for toric algebra and applications. J. Symb. Comput. **38**(2), 959–973 (2004)
20. Dua, V., Papalexandri, K.P., Pistikopoulos, E.N.: Global optimization issues in multiparametric continuous and mixed-integer optimization problems. J. Glob. Optim. **30**(1), 59–89 (2004)
21. Erenguc, S.S., Benson, H.P.: An algorithm for indefinite quadratic integer programming. INFORMS J. Comput. **11**(2), 125–137 (1999)
22. Gallo, G., Hammer, P.L., Simeone, B.: Quadratic knapsack problems. Math. Program. Study **12**, 132–149 (1980)
23. Gilmore, P.C., Gomory, R.E.: The theory and computation of knapsack functions. Oper. Res. **14**(6), 1045–1074 (1966)
24. Glover, F.: Improved linear integer programming formulations of nonlinear integer problems. Manag. Sci. **22**(4), 445–460 (1975)

25. Granot, F., Skorin-Kapov, J.: Some proximity and sensitivity results in quadratic integer programming. Math. Program. **47**(2), 259–268 (1990)
26. Klein Haneveld, W.K., van der Vlerk, M.H.: Stochastic integer programming: general models and algorithms. Ann. Oper. Res. **85**(1), 39–57 (1999)
27. Helmberg, C., Rendl, F.: Solving quadratic (0,1)-problems by semidefinite programs and cutting planes. Math. Program. **82**(3), 291–315 (1998)
28. Helmberg, C., Rendl, F., Weismantel, R.: A semidefinite programming approach to the quadratic knapsack problem. J. Comb. Optim. **4**(2), 197–215 (2000)
29. Horst, R., Tuy, H.: Global Optimization: Deterministic Approaches, 3rd edn. Springer, Berlin (1996)
30. Huang, H.-X., Pardalos, P.M., Prokopyev, O.A.: Lower bound improvement and forcing rule for quadratic binary programming. Comput. Optim. Appl. **33**(2–3), 187–208 (2006)
31. Johnson, E.L.: Integer Programming: Facets, Subadditivity, and Duality for Group and Semi-Group Problems. SIAM Publications, Philadelphia (1980)
32. Johnson, E.L.: Subadditive lifting methods for partitioning and knapsack problems. J. Algorithms **1**(1), 75–96 (1980)
33. Johnson, E.L.: Characterization of facets for multiple right-hand choice linear programs. Math. Program. Study **14**, 112–142 (1981)
34. Kalvenes, J., Kennington, J., Olinick, E.V.: Base station location and service assignment in W-CDMA networks. INFORMS J. Comput. **18**(3), 366–376 (2006)
35. Kong, N., Schaefer, A.J., Hunsaker, B.: Two-stage integer programs with stochastic right-hand sides: a superadditive dual approach. Math. Program. **108**(2), 275–296 (2006)
36. Lawler, E.L.: The quadratic assignment problem. Manag. Sci. **9**(4), 586–599 (1963)
37. Lazimy, R.: Mixed-integer quadratic programming. Math. Program. **22**(1), 332–349 (1982)
38. Lazimy, R.: Improved algorithm for mixed-integer quadratic programs and a computational study. Math. Program. **32**(1), 110–113 (1985)
39. Mao, J.C.T., Wallingford, B.A.: An extension of Lawler and Bell's method of discrete optimization with examples from capital budgeting. Manag. Sci. **15**(2), 851–860 (1969)
40. McBride, R.D., Yormark, J.S.: An implicit enumeration algorithm for quadratic integer programming. Manag. Sci. **26**(3), 282–296 (1980)
41. McBride, R.D., Yormark, J.S.: Finding all solutions for a class of parametric quadratic integer programming problems. Manag. Sci. **26**(8), 784–795 (1980)
42. Nemhauser, G.L., Wolsey, L.A.: Integer and Combinatorial Optimization. Wiley, NewYork (1988)
43. Ntaimo, L., Sen, S.: The million-variable 'march' for stochastic combinatorial optimization. J. Glob. Optim. **32**(3), 385–400 (2004)
44. Oral, M., Kettani, O.: A linearization procedure for quadratic and cubic mixed-integer problems. Oper. Res. **40**(S1), 109–116 (1990)
45. Oral, M., Kettani, O.: Reformulating nonlinear combinatorial optimization problems for higher computational efficiency. Eur. J. Oper. Res. **58**(2), 236–249 (1992)
46. Pardalos, P.M., Rodgers, G.P.: Computational aspects of a branch and bound algorithm for quadratic zero–one programming. Computing **45**(2), 131–144 (1990)
47. Picard, J.-C., Ratliff, H.D.: A cut approach to a class of quadratic integer programming problems. Networks **10**(4), 363–370 (1980)
48. Quadri, D., Soutif, E., Tolla, P.: Exact solution method to solve large scale integer quadratic multidimensional knapsack problems. J. Comb. Optim. **17**(2), 157–167 (2009)
49. Schultz, R.: On structure and stability in stochastic programs with random technology matrix and complete integer recourse. Math. Program. **70**(1), 73–89 (1995)
50. Schultz, R.: Stochastic programming with integer variables. Math. Program. **97**(1–2), 285–309 (2003)
51. Schultz, R., Stougie, L., van der Vlerk, M.H.: Solving stochastic programs with integer recourse by enumeration: A framework using Gröbner basis reductions. Math. Program. **83**(1–3), 229–252 (1998)
52. Smith, J.C., Schaefer, A., Yen, J.W.: A stochastic integer programming approach to solving a synchronous optical network ring design problem. Networks **44**(1), 12–26 (2004)
53. Snyder, L.V., Daskin, M.S., Teo, C.-P.: The stochastic location model with risk pooling. Eur. J. Oper. Res. **179**(3), 1221–1238 (2007)
54. SQIPRANDOMRHS. Test instances for two-stage stochastic quadratic integer programming. Available from http://www.engr.pitt.edu/industrial/faculty-staff/prokopyev/downloads/sqiprandomrhs/index.html (2010). Accessed 05 May 2010
55. Stougie, L.: Design and analysis of algorithms for stochastic integer programming, 1987. Ph.D. dissertation, Center for Mathematics and Computer Science, Amsterdam

56. Thoai, N.V.: Global optimization techniques for solving the general quadratic integer programming problem. Comput. Optim. Appl. **10**(2), 149–163 (1998)
57. Watters, L.G.: Reduction of integer polynomial problems to zero-one linear programming problems. Oper. Res. **15**(6), 1171–1174 (1967)
58. Wets, R.J.-B.: Stochastic programs with fixed recourse: the equivalent deterministic problem. SIAM Rev. **16**, 309–339 (1974)
59. Wolsey, L.A.: Integer programming duality: price functions and sensitivity analysis. Math. Program. **20**(2), 173–195 (1981)