J Heuristics
DOI 10.1007/s10732-011-9175-1

Quadratic bottleneck knapsack problems

Ruonan Zhang · Abraham P. Punnen

Received: 1 March 2011 / Accepted: 16 May 2011
© Springer Science+Business Media, LLC 2011

Abstract In this paper we study the quadratic bottleneck knapsack problem (QBKP) from an algorithmic point of view. QBKP is shown to be NP-hard and it does not admit polynomial time ϵ-approximation algorithms for any $\epsilon > 0$ (unless $P = NP$). We then provide exact and heuristic algorithms to solve the problem and also identify polynomially solvable special cases. Results of extensive computational experiments are reported which show that our algorithms can solve QBKP of reasonably large size and produce good quality solutions very quickly. Several variations of QBKP are also discussed.

Keywords Quadratic · Knapsack problem · Bottleneck · Quadratic threshold · Semi-greedy

1 Introduction

Let $E = \{1, 2, \ldots, n\}$ be a finite set and for each $(i, j) \in E \times E$, a cost q_{ij} is prescribed. Then the *unconstrained 0-1 quadratic programming problem* (*UQP*) is to Minimize $\sum_{i=1}^{n} \sum_{j=1}^{n} q_{ij} x_i x_j$ where $x_j = 0$ or 1 for $j \in E$. Several well studied optimization problems of practical interest can be formulated as an UQP and the problem is known to be NP-hard. For any $X = (x_1, \ldots, x_n) \in \{0, 1\}^n$, let $S(X) = \{j : x_j = 1\}$. The bottleneck version of UQP, called the *bottleneck unconstrained 0-1 quadratic*

R. Zhang · A.P. Punnen (✉)
Department of Mathematics, Simon Fraser University Surrey, Central City, 250-13450 102nd AV, Surrey, British Columbia, V3T 0A3, Canada
e-mail: apunnen@sfu.ca

R. Zhang
e-mail: ruonanz@sfu.ca

Published online: 01 June 2011

programming problem (*BUQP*), can be stated as

$$\begin{aligned} &\text{Minimize} && \max\{q_{ij} : (i, j) \in S(X) \times S(X), S(X) \neq \emptyset\} \\ &\text{Subject to} && X \in \{0, 1\}^n. \end{aligned}$$

Unlike UQP, it is not difficult to see that BUQP is solvable in polynomial time. Interestingly, by introducing an additional linear constraint to BUQP, the complexity status is completely changed. We call the resulting problem *the quadratic bottleneck knapsack problem* (*QBKP*). Let $w_j \geq 0$ be a prescribed weight of element $j \in E$ and $c > 0$ be a given capacity. Then QBKP can be formally defined as

$$\begin{aligned} &\text{Minimize} && \max\{q_{ij} : (i, j) \in S(X) \times S(X)\} \\ &\text{Subject to} && \sum_{j=1}^{n} w_j x_j \geq c, \\ & && x_j = 0 \text{ or } 1 \quad \text{for } j = 1, \ldots, n, \end{aligned}$$

where $X = (x_1, \ldots, x_n)$. Also for QBKP we assume $q_{ij} \geq 0$ without loss of generality.

In QBKP, if the objective function is replaced by $\sum_{i=1}^{n} \sum_{j=1}^{n} q_{ij} x_i x_j$, we get *the quadratic knapsack problem* (*QKP*). Most of the applications of QKP have a natural interpretation in the context of QBKP since the QBKP objective function can be viewed as a measure of the "worst-case" behavior while the QKP objective function measures the "average" behavior. For instance we want to make a selection among n types of investments for a portfolio. Let q_{ij} be a measure of the combined risk for choosing both investments i and j. For example investment i with hedging j reduces the combined risk. Clearly, q_{ij} is zero when i and j are independent. Let w_j be the expected return on investment j. Then the objective is to minimize the maximum risk while the total expected return is no less than a prescribed threshold c. This problem can be formulated as a QBKP.

In this paper we study BUQP, QBKP and several variations of QBKP. The unconstrained problem BUQP is observed to be solvable in polynomial time, whereas QBKP is shown to be NP-hard and it does not admit polynomial time ϵ-approximation algorithms. When $q_{ij} = a_i b_j$ for all i and j or $q_{ij} = a_i + b_j$ for all i and j we show that QBKP can be solved in $O(n^2)$ time. In addition, if Q is symmetric then the complexity of these special cases reduces to $O(n)$. Also, we show that if the maximum weight independent set problem on an associated support graph G can be solved in polynomial time, then QBKP can be solved in polynomial time. Thus when G is series-parallel, bipartite, or the more general perfect graph, the corresponding QBKP can be solved n polynomial time. When the cost matrix Q is doubly graded, QBKP can be solved in $O(n)$ time, but verifying if Q is doubly graded takes $O(n^2)$ time. For the general QBKP, we give an exact algorithm which solves an $O(\log n)$ sequence of maximum weight independent set problems (Garey and Johnson 1979) or equivalently, knapsack problems with conflict pairs (Pferschy and Schauer 2009) or unconstrained quadratic 0-1 problems. Solving these subproblems by a heuristic, the exact algorithm reduces to a heuristic algorithm for QBKP. Further, we develop semi-greedy based heuristics for QBKP. Computational experiments on randomly generated problems show that our heuristic and exact algorithms

can solve large size problems in reasonable running time producing good quality solutions. We also discuss solution algorithms for several variations of QBKP.

The paper is organized as follows. In Sect. 2 we discuss the computational complexity of BUQP and QBKP. Section 3 deals with polynomially solvable special cases of QBKP and lower bounding schemes. In Sect. 4 we provide exact and heuristic algorithms for QBKP. Results of extensive computational experiments are presented in Sect. 5. Section 6 deals with variations of QBKP followed by concluding remarks in Sect. 7.

2 Computational complexity

Let us first consider the BUQP. Unlike its UQP counterpart, BUQP has a very simple solution. Choose $(i, j) \in E \times E$ such that $\max\{q_{ij}, q_{ji}, q_{ii}, q_{jj}\}$ is minimized. Let (r, s) be a pair for which this minimum is attained. Then

$$x_j^* = \begin{cases} 1 & \text{if } j = r \text{ or } j = s, \\ 0 & \text{otherwise} \end{cases}$$

is an optimal solution to BUQP. It may be noted that r could be equal to s. This simplicity in the solution approach however does not translate into QBKP although the bottleneck knapsack problem (BKP) and BUQP admits linear time solution procedures.

Theorem 1 *QBKP is NP-hard even if all w_j's are 1 and q_{ij}'s are 0 or 1 only. Further, unless $P = NP$, there does not exist a polynomial time ϵ-approximation algorithm for QBKP for any $\epsilon \geq 0$.*

Proof Let us first establish the NP-hardness when q_{ij} takes values 0 or 1 only. We reduce the maximum weight independent set problem to QBKP. Let $G = (V, E)$ be a graph on which the maximum independent set problem is defined and $V = \{1, 2, \ldots, n\}$. Construct an instance of QBKP as follows: choose $w_j = 1$ for $j = 1, \ldots, n$ and

$$q_{ij} = \begin{cases} 0 & \text{if } i \text{ is not adjacent to } j \text{ in } G \text{ or } i = j, \\ 1 & \text{otherwise.} \end{cases}$$

Then the resulting QBKP has optimal objective function value 0 if and only if G has an independent set of size greater than or equal to c. Since the maximum independent set problem is NP-hard, QBKP is also NP-hard. To establish the non-approximability result, we again use a reduction from the maximum independent set problem. Define

$$q_{ij} = \begin{cases} 1 & \text{if } i \text{ is not adjacent to } j \text{ in } G \text{ or } i = j, \\ 1 + 2\epsilon & \text{otherwise} \end{cases}$$

and choose $w_j = 1$ for $j = 1, \ldots, n$. Suppose that there is a polynomial time approximation algorithm α for QBKP which produces an ϵ-optimal solution X^* and let X^0 be an optimal solution for the constructed instance of QBKP. If there exists an independent set in G of size greater than or equal to c, then

$f(X^0) = \max\{q_{ij} : (i, j) \in S(X^0) \times S(X^0)\} = 1$. Let $f(X^*) = \max\{q_{ij} : (i, j) \in S(X^*) \times S(X^*)\}$. Also we know that $\frac{f(X^*)}{f(X^0)} \leq 1 + \epsilon$. This happens precisely when $f(X^*) = 1$ and hence G has an independent set of size c. If $f(X^0) = 1 + 2\epsilon$ then G has no independent of size of at least c and in this case $f(X^*)$ must also be $1 + 2\epsilon$. Thus whenever α is polynomial for any $\epsilon > 0$ we can solve the maximum independent set problem on G in polynomial time. Thus unless $P = NP$, there cannot be a polynomial time ϵ-approximation algorithm for QBKP. □

3 Polynomially solvable special cases and an exact algorithm

3.1 Polynomially solvable cases

By assuming special structures for the matrix Q, QBKP can be solved in polynomial time. We first consider the case where

$$q_{ij} = a_i \cdot b_j, \quad a_i \geq 0, \quad b_j \geq 0, \quad 1 \leq i, j \leq n. \tag{1}$$

It may be noted that even with this restricted cost matrix Q, the sum version QKP is NP-hard. Binary quadratic problems with this type of q_{ij}'s are investigated by many researchers (Aneja et al. 1984). Interestingly, if Eq. 1 is satisfied, we show that QBKP can be solved in polynomial time.

Let $X = (x_1, x_2, \ldots, x_n)$ be any feasible solution to QBKP. Suppose Eq. 1 is satisfied. Then $\max\{q_{ij} : (i, j) \in S(X) \times S(X)\} = \max\{a_i b_j : (i, j) \in S(X) \times S(X)\} = \max\{a_i : i \in S(X)\} \cdot \max\{b_i : i \in S(X)\}$. Thus QBKP reduces to the following *multiplicative bottleneck knapsack problem* (MBKP)

$$\begin{aligned} \text{Minimize} \quad & \max\{a_i \ : \ i \in S(X)\} \cdot \max\{b_i \ : \ i \in S(X)\} \\ \text{Subject to} \quad & \sum_{j=1}^{n} w_j x_j \geq c, \\ & x_j = 0 \text{ or } 1 \quad \text{for } j = 1, \ldots, n. \end{aligned}$$

MBKP can be solved as a sequence of n bottleneck knapsack problems by a simple modification of the algorithm for solving max+sum combinatorial optimization problem (Duin 1991; Punnen 1994). Since a bottleneck knapsack problem can be solved in $O(n)$ time using the binary search version of the threshold algorithm along with a variable reduction strategy, MBKP can be solved in $O(n^2)$ time.

Note that a zero matrix satisfies Eq. 1. Further a matrix Q satisfies condition Eq. 1 if and only if $-Q$ satisfies Eq. 1. Thus to test if a matrix Q satisfies Eq. 1, we can focus our attention simply to nonzero matrices with at least one positive element.

Lemma 2 *Let $Q = (q_{ij})_{n\times n}$ be a nonzero matrix with at least one positive element, say q_{kl}, and $\hat{Q} = (\hat{q}_{ij})_{n\times n}$ be defined as $\hat{q}_{ij} = q_{ij} - \alpha_i \beta_j$, where $\alpha_i = \frac{q_{il}}{\sqrt{q_{kl}}}$, $\beta_j = \frac{q_{kj}}{\sqrt{q_{kl}}}$. Then Q satisfies Eq. 1 if and only if $\hat{q}_{ij} = 0$ for $i, j = 1, \ldots, n$.*

Proof Suppose Q satisfies Eq. 1. Then there exists $a_i,\ b_i,\ i = 1, \ldots, n$ such that $q_{ij} = a_i b_j$ for all i and j. Since $q_{kl} > 0$ and from Eq. 1, $a_k, b_l \neq 0$. Now

$$\hat{q}_{ij} = q_{ij} - \alpha_i \beta_j = q_{ij} - \frac{q_{il}}{\sqrt{q_{kl}}} \frac{q_{kj}}{\sqrt{q_{kl}}} = q_{ij} - \frac{q_{il} q_{kj}}{q_{kl}} = a_i b_j - \frac{a_i b_l a_k b_j}{a_k b_l} = 0.$$

Conversely, suppose $\hat{q}_{ij} = q_{ij} - \alpha_i \beta_j = 0$, then clearly $q_{ij} = \alpha_i \beta_j$ holds and hence Q satisfies Eq. 1. □

Corollary 3 *If Q is a nonnegative symmetric matrix that satisfies Eq. 1, then there exists $\alpha_i, i = 1, \ldots, n$ such that $q_{ij} = \alpha_i \alpha_j$ for $i, j = 1, \ldots, n$.*

Proof The proof is trivial if Q is a zero matrix. Suppose Q is a symmetric, nonnegative and nonzero matrix that satisfies Eq. 1, then there exists $r \in \{1, \ldots, n\}$ such that $q_{rr} > 0$. Choosing $(k, l) = (r, r)$ in Lemma 2 we have $\alpha_i = \beta_i$ and hence $q_{ij} = \alpha_i \alpha_j$ for $i, j = 1, \ldots, n$. □

From Corollary 3, when the cost matrix is symmetric, nonnegative and satisfies Eq. 1, MBKP is equivalent to minimizing $\max\{\alpha_i \ : \ i \in S(X)\}$ with the same constraints. Therefore, MBKP and hence QBKP, can be solved in $O(n)$ time by solving one bottleneck knapsack problem.

Let us now consider another polynomially solvable case. Suppose

$$q_{ij} = a_i + b_j \quad \forall 1 \leq i, j \leq n. \tag{2}$$

For any feasible solution X of QBKP, $\max\{q_{ij} : (i, j) \in S(X) \times S(X)\} = \max_{i \in S(X)} a_i + \max_{i \in S(X)} b_i$. Thus QBKP reduces to the following *max+max knapsack problem* (MPMKP)

$$\begin{aligned} \text{Minimize} \quad & \max_{i \in S(X)} a_i + \max_{i \in S(X)} b_i \\ \text{Subject to} \quad & \sum_{j=1}^{n} w_j x_j \geq c, \\ & x_j = 0 \text{ or } 1 \quad \text{for } j = 1, \ldots, n. \end{aligned}$$

MPMKP can also be solved in $O(n^2)$ time as a sequence of $O(n)$ bottleneck knapsack problems by simple modifications of the algorithm for MBKP. This special case in a more general context was considered in Punnen (2011). Similar to the discussions of the previous case, when Q is symmetric and satisfies Eq. 2, there exist α_i, $i = 1, \ldots, n$ such that $q_{ij} = \alpha_i + \alpha_j$ for $i, j = 1, \ldots, n$. Hence this special case of QBKP can be solved in $O(n)$ time.

Before discussing additional polynomially solvable cases, let us consider an exact algorithm to solve QBKP as a sequence of $O(\log n)$ maximum weight independent set problems. This algorithm is a variation of the quadratic threshold algorithm discussed in Punnen (2011) for solving general quadratic bottleneck combinatorial optimization problems. Let β be a trial objective function value. First we examine how to test if there exists a feasible solution to QBKP with objective function

value no more than β. Consider the following graph $\bar{G} = (\bar{V}, \bar{E})$ with node set $\bar{V} = \{i : i \in \{1, 2, \ldots, n\},\ q_{ii} \leq \beta\}$ and edge set $\bar{E} = \{(i, j) : i, j \in \bar{V},\ q_{ij} > \beta\}$. Let w_j be the weight of node $j \in \bar{V}$. Then we may formulate the maximum weight independent set problem as follows.

$$\begin{aligned} \text{MWIP}: \quad \text{Maximize} \quad & \sum_{j \in \bar{V}} w_j x_j \\ \text{Subject to} \quad & x_i + x_j \leq 1 \quad \forall (i, j) \in \bar{E}, \\ & x_j = 0 \text{ or } 1 \quad \text{for } j \in \bar{V}. \end{aligned}$$

Let $X^0 = (x_2^0, x_2^0, \ldots, x_n^0)$ be an optimal solution to MWIP with optimal objective function value w^0. Define $X^* = (x_1^*, x_2^*, \ldots, x_n^*)$ as

$$x_j^* = \begin{cases} x_j^0 & \text{if } j \in \bar{V}, \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Note that $\sum_{j \in \bar{V}} w_j x_j^0 = \sum_{j=1}^n w_j x_j^* = w^0$.

Theorem 4 *There exists a feasible solution to QBKP with objective function value no more than β if and only if $w^0 \geq c$. Further, X^* is such a feasible solution if $w^0 \geq c$.*

Proof If $w^0 \geq c$, then X^* is a feasible solution to the QBKP. For $(i, j) \in \bar{E}$, either $x_i^* = 0$ or $x_j^* = 0$. Thus $x_i^* x_j^* = 0$ for $\forall (i, j) \in \bar{E}$ and hence the QBKP objective function value of X^* is no more than β. If $w^0 < c$, then X^* is not a feasible solution to QBKP. Suppose there exists a solution $\tilde{X} = (\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_n)$ of QBKP with objective function value no more than β. Then for any $(i, j) \in \bar{E}$, either $\tilde{x}_i = 0$ or $\tilde{x}_j = 0$. Thus $\tilde{X}$ determines an independent set of G and feasibility of $\tilde{X}$ for QBKP implies $\sum_{j=1}^n w_j \tilde{x}_j \geq c$. Thus $\tilde{X}$ is a better solution to MWIP, contradicting the optimality of X^0. This completes the proof. □

Based on Theorem 4, we have the following algorithm to solve QBKP.

Theorem 5 *The quadratic threshold algorithm solves QBKP in $O(\phi(n) \log n)$ time, where $\phi(n)$ is the complexity of the maximum weight independent set problem.*

The proof of this theorem can be easily established using Theorem 4.

3.2 Lower bounds and related solvable cases

Let us now consider algorithms to identify a lower bound for the optimal objective function value of QBKP.

Algorithm 1: The Quadratic Threshold Algorithm

Construct an ascending arrangement $z_1 < z_2 < \cdots < z_p$ of distinct q_{ij}'s;
$\ell = 1$; $u = p$;
while $u - \ell > 0$ **do**
  $k = \lfloor \frac{(l+u)}{2} \rfloor$;
  Solve MWIP on $G = (\bar{V}, \bar{E})$, where $\bar{V} = \{i : i \in \{1, \ldots, n\},\ q_{ii} \leq z_k\}$, $\bar{E} = \{(i, j) : i, j \in \bar{V},\ q_{ij} > z_k\}$, let X^0 be the solution to MWIP with objective function value w^0 and X^* is the solution obtained from X^0 as given by Eq. 3.
  if $w^0 \geq c$ **then**
    $u = k$;
  else
    $\ell = k + 1$;
  end if
end while
Output z_l as the optimal objective function value and the corresponding optimal solution.

Consider the following bottleneck knapsack problem:

$$\begin{aligned} \text{BKP}(i, Q): \quad \text{Minimize} \quad & \max\{q_{ij} \ : \ j \in S(X)\} \\ \text{Subject to} \quad & \sum_{j=1}^{n} w_j x_j \geq c, \\ & x_j = 0 \text{ or } 1 \quad \text{for } j = 1, \ldots, n. \end{aligned}$$

Let z^i be the optimal objective function value of BKP(i, Q). Now consider the bottleneck knapsack problem

$$\begin{aligned} \text{BKP}(Q): \quad \text{Minimize} \quad & \max\{z^j \ : \ j \in S(X)\} \\ \text{Subject to} \quad & \sum_{j=1}^{n} w_j x_j \geq c, \\ & x_j = 0 \text{ or } 1 \quad \text{for } j = 1, \ldots, n. \end{aligned}$$

Let $\tilde{L}$ be the optimal objective function value of BKP(Q).

Lemma 6 *$\tilde{L}$ is a lower bound for the optimal objective function value of QBKP.*

Proof Let X be any feasible solution to QBKP. Note that X is also a feasible solution of BKP(Q). Then $\max\{q_{ij} : (i, j) \in S(X) \times S(X)\} = \max_{i \in S(X)} \max_{j \in S(X)} q_{ij} \geq \max_{i \in S(X)} z^i \geq \tilde{L}$. □

Clearly, if we can find a feasible solution X for QBKP with objective function value $\tilde{L}$, it is indeed optimal. This is possible when the cost matrix Q satisfies additional properties.

Let $Q = (q_{ij})$ be an $n \times n$ matrix. Then Q is said to be *row graded* if $q_{i1} \leq q_{i2} \leq \cdots \leq q_{in}$ for all $i = 1, \ldots, n$. Further, Q is said to be *doubly graded* if Q and Q^T are row graded. Recall that any BKP can be solved by the greedy algorithm. An optimal solution $X^0 = (x_1^0, x_2^0, \ldots, x_n^0)$ to a BKP is called *a least-index optimal solution* if X^0 is optimal and $r = \max\{i : x_i^0 = 1\} \leq \max\{i : x_i = 1\}$ for any optimal solution X for BKP. The value r is called *the critical index of BKP*.

Lemma 7 *Let X^0 be a least-index optimal solution to BKP$(1, Q)$ with critical index r. If Q is row graded and $q_{1r} \leq q_{2r} \leq \cdots \leq q_{nr}$, then X^0 is an optimal solution to QBKP with cost matrix Q and the corresponding optimal objective function value is q_{rr}.*

Proof Since Q is row graded, X^0 is an optimal solution to BKP(i, Q) for all $i = 1, \ldots, n$ with corresponding optimal objective function value $z^i = q_{ir}$. Since $q_{1r} \leq q_{2r} \leq \cdots \leq q_{nr}$, X^0 is also an optimal solution to BKP(Q) with optimal objective function value q_{rr}. By Lemma 6, q_{rr} is a lower bound for the optimal objective function value of QBKP. The QBKP objective function value of X^0 can be verified to be q_{rr} and the proof follows. □

Note that all doubly graded matrices satisfy the conditions of Lemma 7. Further, QBKP with cost matrices Q and Q^T are equivalent. Thus if the conditions of Lemma 7 are satisfied for Q^T, then also QBKP can be solved in polynomial time. Lemma 7 can be extended to a larger class of cost matrices where the rows (and hence the columns) can be relabeled to obtain a matrix that satisfies conditions of the lemma.

Let L be a lower bound on the optimal objective function value of QBKP. Consider the graph $\hat{G} = (\hat{V}, \hat{E})$ where $\hat{V} = \{1, 2, \ldots, n\}$ and $(i, j) \in \hat{E}$ if and only if $q_{ij} \geq L$. We call $\hat{G}$ the *L-support graph* of Q. Based on Theorem 4, if the maximum weight independent set problem on all subgraphs of the L-support graph of Q can be solved in polynomial time and L can be identified in polynomial time, then QBKP can be solved in polynomial time. Note that if $\hat{G}$ is series-parallel, bipartite, or the more general perfect graph, the maximum weight independent set problem can be solved in polynomial time (Courcelle et al. 2000; Grötschel et al. 1984, 1988; Takamizawa et al. 1982).

Observation 8 *For an instance of QBKP with cost matrix $Q = (q_{ij})_{n\times n}$, there exists an equivalent cost matrix $Q' = (q'_{ij})_{n\times n}$ such that Q' is symmetric.*

Proof Let $q'_{ij} = \max\{q_{ij}, q_{ji}\}$. Then it can be verified that $q'_{ij} = q'_{ji}$ for $i, j = 1, \ldots, n$ and any optimal solution to QBKP with costs q'_{ij} is also an optimal solution to QBKP with costs q_{ij}. □

Thus hereafter we assume without loss of generality that Q is symmetric and we store only the elements of Q on or above the diagonal to represent Q.

4 Heuristic algorithms

Our first heuristic algorithm is a variation of the exact algorithm (the quadratic threshold algorithm) discussed in Sect. 3. Here, instead of solving the maximum weight independent set problem optimally, we solve it using a heuristic algorithm and the heuristic solution produced by this algorithm is used to determine the binary search direction. The resulting algorithm is called *approximate quadratic threshold heuristic* or AQT-heuristic. The quality and running time of this heuristic depends primarily on the quality and running time of the heuristic for the maximum weight independent set problem we use. In the section on computational results, we discuss this with additional details.

4.1 Semi-greedy algorithm

Let us now consider another heuristic using the semi-greedy strategy discussed in Hart and Shogan (1987), Resende (2003). Semi-greedy algorithms are originally developed to solve vehicle routing problems. Such algorithms are also used in various implementations of GRASP algorithms (Festa and Resende 2002). However, to the best of our knowledge, semi-greedy algorithms are not tested for any quadratic bottleneck problems. We are interested in semi-greedy algorithms because these are simple heuristics that provide reasonable solutions very quickly and the inherent randomization allows diversification in the construction process. Further, standard improvement-type algorithms seem not very efficient for quadratic bottleneck problems, as discussed later in this section. The algorithm builds a solution X by selecting elements of $S(X)$ one at a time. The algorithm maintains a subset $S(X)$ of $\{1, 2, \ldots, n\}$ in each iteration. Let $\bar{S} = \{1, 2, \ldots, n\} - S(X)$. For each $j \in \bar{S}$ let $p_j = \max\{q_{ij} : i \in S(X)\}$. Given a parameter γ, generate a random integer r from $[1, \gamma]$ and choose a subset I of $\bar{S}$ such that $j \in I$ implies p_j is no larger than the rth smallest element of the set $\{p_j : j \in \bar{S}\}$ for $r \leq \gamma$ (counting multiplicity). The algorithm then selects a j from I randomly and update $S(X)$ as $S(X) \cup \{j\}$. The process is continued until a feasible solution $S(X)$ is reached. This whole process is further repeated for a fixed number of times and we output the overall best solution obtained. A formal description of the algorithm is given in Algorithm 2.

When $\gamma = 1$ the above algorithm reduces to a greedy algorithm. The objective function value of the solution obtained by the semi-greedy algorithm can be used as an upper bound to speed up the quadratic threshold algorithm and the AQT-heuristic.

4.2 Improving a solution

Let X^* be any solution to the QBKP and z^* be its objective function value. Clearly X^* is a feasible solution to the integer program

$$
\begin{aligned}
\text{(IP)} \quad \text{Maximize} \quad & 0X \\
\text{Subject to} \quad & \sum_{j=1}^{n} w_j x_j \geq c, \\
& x_i + x_j \leq 1 \quad \text{if } q_{ij} > z^*, \\
& x_j = 0 \text{ or } 1.
\end{aligned}
$$

Algorithm 2: The Semi-Greedy Algorithm

γ_iter $= 0$;
Let $z^* = \infty$, $S^* = \emptyset$;
while γ_iter $<$ max_γ_iter **do**
  Choose a value of γ.
  iter $= 0$;
  while iter $<$ max-iter **do**
    Randomly choose $i^0 \in E$, let $S(X) = \{i^0\}$, $\bar{S} = \{1, \ldots, n\} \setminus S(X)$;
    while $\sum_{i \in S(X)} w_i < c$ **do**
      For each $j \in \bar{S}$ let $p_j = \max\{q_{ij} : i \in S(X)\}$;
      Generate a random integer $r \in [1, \min\{\gamma, |\bar{S}|\}]$;
      Let p_{j^*} be the r-th smallest element in $\{p_j : j \in \bar{S}\}$;
      $S(X) = S(X) \cup \{j^*\}$, $\bar{S} = \bar{S} \setminus \{j^*\}$;
    end while
    Let $z^0 = \max\{q_{ij} : (i, j) \in S(X) \times S(X)\}$;
    if $z^0 < z^*$ **then**
      $z^* = z^0$;
      $S^* = S(X)$;
    end if
    iter $=$ iter $+ 1$;
  end while
  γ_iter $= \gamma$_iter$+1$;
end while
Output S^* and z^*;

Let $z_1 < z_2 < \cdots < z_p$ be an ascending arrangement of distinct q_{ij} values and let $z^* = z_r$ for some $r \leq p$. Let $E_t = \{(i, j) : q_{ij} = z_t\}$. Then X^* can be improved if and only if the problem IP with the additional constraints

$$x_i + x_j \leq 1 \quad \text{for } (i, j) \in E_{r-1} \tag{4}$$

is feasible. Obviously, finding an improved solution is NP-hard. Since the modified IP is obtained from the original IP by adding the cutting planes described by Eq. 4, the resulting improvement scheme is called a cutting plane algorithm. If the modified IP is solved by an exact algorithm, the resulting cutting plane algorithm is the same as using a top-down sequential search in the quadratic threshold algorithm. If the modified IP is solved using a heuristic, the cutting plane algorithm can be viewed as the top-down sequential search version of the AQT-heuristic. In practice, it may be better to explore possible improvements by more than one step (say k steps, for small values of k). In this case we are adding cutting planes

$$x_i + x_j \leq 1 \quad \text{for } (i, j) \in E_{k,r} = \bigcup_{t=1}^{k} E_{r-t}.$$

Algorithm 3: The Cutting Plane Algorithm

1: Solve the QBKP using the semi-greedy algorithm. Let X^* be the solution produced and z^* be the optimal objective function value.
2: Construct an ascending arrangement $z_1 < z_2 < \cdots < z_p$ of distinct q_{ij}'s such that $q_{ij} \leq z^*$;
3: Let $\delta = p$;
4: Consider the IP

$$\begin{aligned} \text{Maximize} \quad & 0X \\ \text{Subject to} \quad & \sum_{j=1}^{n} w_j x_j \geq c, \\ & x_i + x_j \leq 1 \quad \text{if } q_{ij} > z^*, \\ & x_j = 0 \text{ or } 1. \end{aligned}$$

Let X^* be an optimal solution to the IP.
5: Choose the value of the parameter k;
6: **repeat**
7: Introduce the cutting planes

$$x_i + x_j \leq 1 \quad \text{for } (i, j) \in E_{k,\delta} = \bigcup_{t=1}^{k} E_{\delta - t}. \tag{5}$$

Designate IP as IP together with constraints 5 and re-solve the IP with these additional constraints. Let X^0 be the resulting solution if IP is feasible.
8: **if** IP is feasible **then**
9: $X^* = X^0$;
10: $\delta = \delta - k$;
11: **end if**;
12: **until** IP is infeasible or $\delta < k$;
13: Output X^*;

As soon as the modified IP becomes infeasible, the algorithm is terminated. The modified IP can be solved in many ways. For example by solving a MWIP (Garey and Johnson 1979) or a knapsack problem with conflict pairs (Pferschy and Schauer 2009). This cutting plane approach could work better if we have a good quality solution to start with. A formal description of the cutting plane algorithm is given in Algorithm 3.

5 Computational results

The lower bound algorithm, heuristics and the exact algorithm were coded in C++ and tested on a Linux workstation with Intel Xeon E5410 CPU (2.33 GHz) and

8 GB RAM running Red Hat Enterprise Linux (kernel 2.6.18). There are no standard benchmark problems available for QBKP. Thus we have generated the test instances randomly. The test instances are generated using two parameters n and Δ, where n is the number of variables and Δ is the percentage of the number of q_{ij}'s values that are non-zero. When Δ is small, the matrix Q is sparse. We selected $n = 100, 200, 300, \ldots, 700$ and $\Delta = 50, 75$ and 100 for our experiments. Note that for $n = 700$ and $\Delta = 100$, there are 245350 q_{ij} values and hence the problems considered are of reasonably large size. For each problem size (n, Δ), 5 different problems (q_{ij} values) were generated and solved using the quadratic threshold algorithm where the MWIP is solved using CPLEX. Some of these problems were solved quickly and produced optimal solutions whereas some took more time while the remaining were not solved even after running for a couple of days. From each problem size (n, Δ) we have selected one of the 'hard' problems (problems that took more time to solve by the quadratic threshold algorithm and problems that could not be solved) to include in our test bed and named the problems using the convention "QBKPxxx-y", where xxx represents n and $y = 1, 2, 3$ respectively represent $\Delta = 50,\ 75,\ 100$.

For semi-greedy heuristics, all experiments are conducted using the following values of the parameters: max_γ_iter $= 3$, $\gamma = 1, 3, 5$ and max_iter $= 5$.

Let us now discuss the implementations of the quadratic threshold algorithm (QT_algorithm) and the AQT_heuristic. Let $z_1 < z_2 < \cdots < z_p$ be the distinct values of q_{ij}. Let U be the objective function value of the semi-greedy heuristic and L be the lower bound obtained as discussed in Sect. 3. The values of l and u are selected such that $z_l = L$ and $z_u = U$. Depending on the way the MWIP is solved, we have tested two variations of the AQT_heuristic, called "AQT_KPC" and "AQT_UQP".

In AQP_KPC, we solve the MWIP using CPLEX 12.1.0 with an added inequality $\sum_{j=1}^{n} w_j x_j \geq c$. Thus we essentially solved *the knapsack problem with conflict pairs* (*KPC*)

$$
\begin{aligned}
\text{(KPC)} \quad \text{Maximize} \quad & \sum_{j=1}^{n} w_j x_j \\
\text{Subject to} \quad & \sum_{j=1}^{n} w_j x_j \geq c, \\
& x_i + x_j \leq 1 \quad \text{if } q_{ij} > z_k, \\
& x_j = 0 \text{ or } 1.
\end{aligned}
$$

We used a time limit of 60 seconds for each call of CPLEX and if no feasible solution is obtained, we heuristically considered the KPC to be infeasible. CPLEX is terminated as soon as a feasible solution is identified.

In AQP_UQP variation, we formulated the MWIP as an UQP with costs d_{ij}, where

$$
d_{ij} = \begin{cases} w_j & \text{if } i = j, \\ -M & \text{if } i \neq j \text{ and } q_{ij} > z_k, \\ 0 & \text{if } i \neq j \text{ and } q_{ij} \leq z_k, \end{cases}
$$

where M is a large number. If the resulting UQP has a solution with objective function value no less than c, then the MWIP has a feasible solution with weight greater than or

equal to c. We solved the UQP using a heuristic algorithm reported in Kochenberger et al. (2011) which is found to be very effective in solving several related optimization problems (Glover et al. 2010; Wang et al. 2010).

In the cutting plane algorithm, we again used CPLEX to solve the IP with a cpu time limit of 60 seconds. To set a proper value for the parameter k, we have tested $k = 1, 3, 5, 10, 20$ and selected $k = 10$ in the final experiments.

Table 1 summarizes our main experimental results. Here n is the number of variables and p is the number of distinct costs. The remaining columns respectively record information about the lower bound, the optimal objective function value, and test results for the 5 heuristics considered. Note that since our problem has a bottleneck objective function, the exact values of the q_{ij}'s are not important but their orders are. Therefore, we reported the positions instead of the exact values. For example, "0" means the objective function value returned equals the smallest q_{ij}. The column "cpu" under AQT_KPC is the time for completing the "while loop" in the AQT_heuristic, so the total cpu time for AQT_KPC should actually be reported as this time plus the cpu time of calculating p, lb and running time for the semi-greedy heuristic. But these extra time components are negligible. For AQT_UQP and the cutting plane algorithm, similar information is reported.

For the problems with $n \geq 400$, it took days to run the exact algorithm and hence we aborted these experiments. From Table 1 we can see that the lower bound algorithm takes very little time but does not consistently provide high quality bounds. The Greedy and Semi-greedy algorithms can obtain a heuristic solution quite fast but the quality of these solutions is much worse than the quality provided by the other three heuristics. AQT_UQP outperforms other heuristics by returning solutions with better objective function values and less cpu time. AQT_KPC can sometimes provided better solutions than the cutting plane algorithm but the latter is faster in 15 out of 21 problems.

6 Variations of QBKP

Let us now consider some variations of QBKP. Our first variation is *the quadratic bottleneck 0-1 programming problem* (*QB0-1*), which can be formulated as follows

$$\begin{array}{lll} \text{QB0-1:} & \text{Minimize} & \max\{q_{ij} : (i, j) \in S(X) \times S(X)\} \\ & \text{Subject to} & AX \geq b, \\ & & X \in \{0, 1\}^n, \end{array}$$

where A is a $m \times n$ matrix and $b \in \mathcal{R}^m$. Note that QBKP is a special case of QB0-1 when $m = 1$. The quadratic threshold algorithm can be modified to solve QB0-1, where instead of solving a MWIP in each iteration, we try to find if there exists a solution $X = (x_1, x_2, \ldots, x_n)$ such that $AX \geq b$ and $x_i + x_j \leq 1$ for $(i, j) \in \bar{E}$. Also, we can have heuristics similar to the AQT variations.

Table 1 Experimental results (numbers in bold represent the cases where the heuristics produced an optimal solution)

Problem name	n	p		opt	lb		Greedy		Semi-greedy		AQT_KPC		AQT_UQP		Cutting plane	
		Value	cpu	Position	Position	cpu	Position	cpu	Position	cpu	Position	cpu	Position	cpu	Position	cpu
QBKP100-1	100	2211	0.02	198	0	0	1195	0	825	0	**198**	6.69	**198**	306.94	205	7.02
QBKP100-2	100	3082	0	1653	0	0.01	2583	0	2036	0	**1653**	6.48	**1653**	493.18	1656	5.41
QBKP100-3	100	3851	0	1866	240	0	2924	0	2806	0	**1866**	7.09	**1866**	184.05	**1866**	7.1
QBKP200-1	200	8579	0.2	5749	0	0.04	6895	0	6788	0	**5749**	1034.07	5782	507.49	5758	287.72
QBKP200-2	200	12127	0.05	7883	0	0.03	10715	0	9867	0	**7883**	974.81	**7883**	507.71	7887	339.75
QBKP200-3	200	14960	0	11591	1631	0.03	13237	0	12965	0	**11591**	1582.95	11593	455.71	11595	468.58
QBKP300-1	300	16313	0.98	14540	0	0.13	15345	0	15344	0.01	14564	2450.36	14586	600.49	14584	842.52
QBKP300-2	300	21055	0.26	14140	0	0.09	17818	0	17087	0.01	14148	2914.92	14143	516.73	14167	1387.79
QBKP300-3	300	24434	0.01	21949	4040	0.08	23416	0	23037	0.01	**21949**	2199.25	22083	602.52	21987	1022.62
QBKP400-1	400	23158	0.11	11451	0	0.31	17640	0	15897	0.01	11885	2015.23	11614	512.83	15397	3248.35
QBKP400-2	400	27465	0.8	–	0	0.22	25522	0	25152	0.02	22514	2412.11	22352	673.25	22412	1858.6
QBKP400-3	400	29872	0.02	–	2819	0.17	27683	0.01	27525	0.02	25312	1921.81	25122	497.65	25285	1579.05
QBKP500-1	500	27956	7.55	–	0	0.59	26560	0	26319	0.03	24881	1910.58	24824	527.7	24729	2324.64
QBKP500-2	500	30903	1.99	–	0	0.41	22965	0	20949	0.01	17591	1006.99	12311	663.28	20759	3084.39
QBKP500-3	500	32004	0.03	–	267	0.32	16858	0	14750	0.01	13813	1397.63	8177	534.08	14705	600.02
QBKP600-1	600	30688	15.62	–	0	1.02	28987	0	28965	0.04	27440	2272.2	27212	586.01	27365	753.47
QBKP600-2	600	32233	3.93	–	0	0.68	31079	0.01	31078	0.05	29930	2968.35	29873	656.28	29938	2530.15
QBKP600-3	600	32631	0.06	–	5289	0.55	31776	0	31594	0.04	30932	2549.76	30766	776.06	31004	1725.54
QBKP700-1	700	32011	29.96	–	0	1.61	29943	0	29643	0.05	27748	2840.4	27416	584.8	28083	3043.85
QBKP700-2	700	32654	7.42	–	0	1.09	31178	0.01	31033	0.05	30314	3310.29	29657	564.9	30173	3012.9
QBKP700-3	700	32754	0.08	–	3856	0.85	31578	0	31511	0.04	30631	2840.77	30272	581.24	30981	3229.07

The maxmin version of QBKP can be defined as

$$\begin{array}{lll} \text{QBKP1:} & \text{Maximize} & \min\{q_{ij} : (i, j) \in S(X) \times S(X)\} \\ & \text{Subject to} & \sum_{j=1}^{n} w_j x_j \geq c, \\ & & x_j = 0 \text{ or } 1. \end{array}$$

The problems QBKP and QBKP1 are equivalent. To see this, consider an instance of QBKP1 with costs q_{ij} and weights w_j. Let $F_{\geq} = \{X = (x_1, \ldots, x_n) \in \{0, 1\}^n : \sum_{j=1}^{n} w_j x_j \geq c\}$. Now

$$\begin{aligned} &\max_{X \in F_{\geq}} \min\{q_{ij} : (i, j) \in S(X) \times S(X)\} \\ &\quad = - \min_{X \in F_{\geq}} \max\{-q_{ij} : (i, j) \in S(X) \times S(X)\}. \end{aligned} \tag{6}$$

Let M be a large number. Since $-q_{ij}$ and $M - q_{ij}$ have the same ordering, an optimal solution of $\min_{X \in F_{\geq}} \max\{-q_{ij} : (i, j) \in S(X) \times S(X)\}$ and $\min_{X \in F_{\geq}} \max\{(M - q_{ij}) : (i, j) \in S(X) \times S(X)\}$ are the same. Thus, the algorithms derived in the previous sections can be used to solve QBKP1. However, the reduction from QBKP1 to QBKP discussed above does not preserve performance ratios of approximation algorithms. Theorem 1 extends to QBKP1 as summarized below.

Theorem 9 *QBKP*1 *is NP-hard even if all* w_j*'s are* 1 *and* q_{ij}*'s are* 0 *or* 1 *only. Further, unless* $P = NP$*, there does not exist a polynomial time* ϵ*-approximation algorithm for QBKP*1 *for any* $\epsilon \geq 0$.

The proof of this theorem can be constructed following the proof of Theorem 1 and hence is omitted. By reversing the direction of the constraint of QBKP and QBKP1 we get two more variations of QBKP which are stated below.

$$\begin{array}{lll} \text{QBKP2:} & \text{Minimize} & \max\{q_{ij} : (i, j) \in S(X) \times S(X)\} \\ & \text{Subject to} & \sum_{j=1}^{n} w_j x_j \leq c, \\ & & x_j = 0 \text{ or } 1. \end{array}$$

$$\begin{array}{lll} \text{QBKP3:} & \text{Maximize} & \min\{q_{ij} : (i, j) \in S(X) \times S(X)\} \\ & \text{Subject to} & \sum_{j=1}^{n} w_j x_j \leq c, \\ & & x_j = 0 \text{ or } 1. \end{array}$$

When $q_{ij} \geq 0$, QBKP2 is trivial since $S(X) = \emptyset$ gives an optimal solution. However, its "max-sum" counterpart is NP-hard and studied by various authors (Billionnet and Calmels 1996; Chailou et al. 1989; Gallo et al. 1980; Hammer and Rader 1997; Helmberg et al. 2000; Michelon and Veilleux 1996; Rader and Woeginger 2002). Let us now consider QBKP3. Clearly there exists an optimal solution to QBKP3 where

Algorithm 4: QBKP3

Let $S^* = \emptyset,\ \bar{S} = E \times E,$ and $z^* = -\infty$;
while $\bar{S} \neq \emptyset$ **do**
  Take $(r, t) \in \bar{S}$;
  if $r = t,\ q_{rr} > z^*$ and $w_r \leq c$ **then**
    $S^* = \{r\}$;
    $z^* = q_{rr}$;
  end if
  if $r \neq t,\ \min\{q_{rr}, q_{tt}, q_{rt}, q_{tr}\} > z^*$ and $w_r + w_t \leq c$ **then**
    $S^* = \{r, t\}$;
    $z^* = \min\{q_{rr}, q_{tt}, q_{rt}, q_{tr}\}$;
  end if
  $\bar{S} = \bar{S} \setminus \{(r, t) \cup (t, r)\}$;
end while
Output S^* and z^*.

at most two x_j's are one and all other x_j's have value zero. Such an optimal solution can be identified in $O(n^2)$ time by a simple sequential search algorithm as discussed in Algorithm 4.

7 Conclusions

We have shown that the quadratic bottleneck knapsack problem introduced in this paper is NP-hard and does not admit polynomial time ϵ-approximation algorithms. Accordingly we have developed efficient exact and heuristic algorithms for the QBKP problem. Several polynomially solvable special cases are also identified. Our experimental results on randomly generated problems show that the heuristics can compute good quality solutions for large size problems in reasonable running time. Several variations of QBKP are also discussed. An interesting theoretical question is to identify more special cases that are solvable in polynomial time or can be approximated with constant performance ratio in polynomial time.

Acknowledgements This work was supported by an NSERC discovery grant awarded to Abraham P. Punnen.

We are thankful to Dr. Zhipeng Lü for providing us a copy of his computer code for solving UQP which was used in our experiments. Comments of the anonymous referees improved the presentation of the paper.

References

Aneja, Y.P., Aggarwal, V., Nair, K.P.K.: On a class of quadratic programs. Eur. J. Oper. Res. **18**, 62–70 (1984)

Billionnet, A., Calmels, F.: Linear programming for the 0-1 quadratic knapsack problem. Eur. J. Oper. Res. **92**, 310–325 (1996)

Cai, Y., Wang, J., Yin, J., Zhou, Y.: Memetic clonal selection algorithm with EDA vaccination for unconstrained binary quadratic programming problems. Expert Syst. Appl. **38**, 7817–7827 (2011)

Chailou, P., Hansen, P., Mahieu, Y.: Best network flow bound for the quadratic knapsack problem. In: Simeone (ed.) Combinatorial Optimization. Lecture Notes in Mathematics, vol. 403, pp. 225–235. Springer, Berlin (1989)

Courcelle, B., Makowsky, J., Rotics, U.: Linear time solvable optimization problems on graphs of bounded clique-width. Theory Comput. Syst. **33**, 125–150 (2000)

Duin, C.W., Volgenant, A.: Minimum deviation and balanced optimization: a unified approach. Oper. Res. Lett. **10**, 43–48 (1991)

Festa, P., Resende, M.: GRASP: An annotated bibliography. In: Ribeiro, C., Hansen, P. (eds.) Essays and Surveys in Metaheuristics, pp. 325–367. Kluwer Academic, Dordrecht (2002)

Gallo, G., Hammer, P., Simeone, B.: Quadratic knapsack problems. Math. Program. **12**, 132–149 (1980)

Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-completeness. Freeman, New York (1979)

Glover, F., Lü, Z., Hao, J.K.: Diversification-driven tabu search for unconstrained binary quadratic problems. 4OR-Q. J. Oper. Res. **8**, 239–253 (2010)

Grötschel, M., Lovász, L., Schrijver, A.: Geometric Algorithms Combinatorial Optimization, pp. 273–303. Springer, Berlin (1988)

Grötschel, M., Lovász, L., Schrijver, A.: Polynomial algorithms for perfect graphs. Math. Stud. **88**, 325–356 (1984)

Hammer, P., Rader, D. Jr.: Efficient methods for solving quadratic 0-1 knapsack problems. INFOR **35**, 170–182 (1997)

Hart, J.P., Shogan, A.W.: Semi-greedy heuristics: an empirical study. Oper. Res. Lett. **6**, 107–114 (1987)

Helmberg, C., Rendl, F., Weismantel, R.: A semidefinite programming approach to the quadratic knapsack problem. J. Comb. Optim. **4**, 197–215 (2000)

Kochenberger, G., Hao, J.K., Lü, Z., Wang, H., Glover, F.: Solving large scale max cut problems via tabu search. Technical report (2011)

Michelon, P., Veilleux, L.: Lagrangian methods for the 0-1 quadratic knapsack problem. Eur. J. Oper. Res. **95**, 671–682 (1996)

Pferschy, U., Schauer, J.: The knapsack problem with conflict graphs. J. Graph Algorithms Appl. **13**, 233–249 (2009)

Punnen, A.P.: On combined minmax-minsum optimization. Comput. Oper. Res. **21**, 707–716 (1994)

Punnen, A.P., Zhang, R.: Quadratic bottleneck problems. Naval Res. Logist. **58**, 153–164 (2011)

Rader, D., Jr., Woeginger, W.: The quadratic 0-1 knapsack problem with series-parallel support. Oper. Res. Lett. **30**, 159–166 (2002)

Resende, M., Ribeiro, C.: Greedy randomized adaptive search procedures. In: Glover, F., Kochenberger, G. (eds.) Handbook of Metaheuristics, pp. 219–249. Kluwer Academic, Dordrecht (2003)

Takamizawa, K., Nishizeki, T., Saito, N.: Linear-time computability of combinatorial problems on series-parallel graphs. J. ACM **29**, 623–641 (1982)

Wang, H., Kochenberger, G., Xu, Y.: A note on optimal solutions to quadratic knapsack problems. Int. J. Math. Modell. Numer. Optim. **1**, 344–351 (2010)