European Journal of Operational Research 326 (2025) 1–12

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/eor

Invited Review

The quadratic knapsack problem

Laura Galli [a], Silvano Martello [b,*], Paolo Toth [b]

[a] Dipartimento di Matematica, Alma Mater Studiorum Università di Bologna, Italy
[b] DEI "Guglielmo Marconi", Alma Mater Studiorum Università di Bologna, Italy

ARTICLE INFO

Keywords:
Quadratic knapsack problem
Linearization
Upper bounds
Exact solution
Approximation
Heuristics
Metaheuristics
Computational results

ABSTRACT

The quadratic knapsack problem is a relevant $\mathcal{NP}$-hard combinatorial optimization problem, inspired, since the Seventies, by a number of real-world applications. After its formal definition in 1980, it was subject to intensive research, especially in the last two decades. No recent review on this problem appeared in the literature after a well-known survey, published in 2007 but updated to 2003. The purpose of this work is to provide a thorough overview of classical and recent results on the quadratic knapsack problem. We examine mathematical models, linearizations and reformulations. We review upper bounds, exact algorithms, heuristic and metaheuristic approaches, and provide a comparison of their computational performance.

1. Introduction

In the *Quadratic Knapsack Problem* (QKP) we are given a container, having positive integer capacity, and a set of items, a subset of which has to be optimally inserted into the container. Each item has a non-negative integer profit and a positive integer weight. Each pair of items additionally generates a non-negative integer profit if both items are selected, i.e., inserted into the knapsack. The QKP aims at selecting a subset of items, whose weight does not exceed the container capacity, such that the total profit it produces is a maximum.

Real-world applications and the birth of the problem

The QKP was inspired by a number of real-world applications explored since the Seventies in different fields, the first one being that studied by Rhys (1970) on the installation of freight handling terminals: "*The cost of a terminal is fixed. The existence of any particular pair of terminals permits a service to be operated between those terminals, and each service is associated with a net revenue. The problem is to find the optimal selection of terminals, given all relevant net revenues and costs*". Similar applications were examined by Laughhunn (1970) (portfolio selection from a set of risky, interrelated investments subject to a budget constraint), Witzgall (1975) (selection of sites for satellite stations to handle traffic of electronic messages), and Gallo, Hammer, and Simeone (1980) (selection of locations for railway stations, correlation of pluviometers in hydrological observations).

These applications inspired, in 1980, the first formal definition of the problem (see below).

More recent applications

Other optimization problems sharing the combinatorial structure of the QKP arise in contexts dealing with graph partitioning problems. Johnson, Mehrotra, and Nemhauser (1993) studied a graph problem representing a compiler design application in which a set of modules must be partitioned into clusters which are restricted in their total storage, and the communication costs between modules within the same cluster are negligible while different clusters imply substantial costs due to memory swap operations. Ferreira, Martin, de Souza, Weismantel, and Wolsey (1996) considered a node capacitated graph partitioning problem arising in VLSI design when large graphs need to be decomposed into smaller graphs of tractable size.

The QKP can model recent applications arising in wind farm layout problems (see, e.g., Fischetti and Pisinger (2019) and Cazzaro and Pisinger (2022)). It also arises as a subproblem in more complex optimization contexts like, e.g., the solution of column generation subproblems (see Johnson et al. (1993) and Thomadsen and Larsen (2007)) and the extension to the case of multiple knapsacks (see Section 3.1). In addition, the QKP is a generalized version of a number of graph theory problems (clique of fixed order k, densest k-subgraph problem, and others, see Billionnet, Faye, and Soutif (1999), Pisinger, Rasmussen, and Sandvik (2007), and Section 5).

This survey

In 1980, Gallo et al. (1980) formally defined the QKP as a 0–1 quadratic problem, and proposed a branch-and-bound algorithm for its exact solution. The problem did not attract extensive research over

* Corresponding author.
E-mail addresses: l.galli@unibo.it (L. Galli), silvano.martello@unibo.it (S. Martello), paolo.toth@unibo.it (P. Toth).

https://doi.org/10.1016/j.ejor.2024.12.032
Received 16 May 2024; Accepted 19 December 2024
Available online 26 December 2024
0377-2217/© 2025 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

the next two decades. The monograph (Kellerer, Pferschy, & Pisinger, 2004) on knapsack problems, published in 2004 by Kellerer, Pisinger, and Pferschy, includes a chapter on the QKP, partially based on the survey dedicated by Pisinger (2007) to the QKP (appeared in 2007 but updated to 2003). Both sources list just a dozen of specific studies on the problem, appeared up to 1999, mainly devoted to upper bounds, linear reformulations, relaxations, and computational experiments.

In subsequent years, the QKP attracted a considerable number of researchers, resulting in tens of results, ranging from exact algorithms and convex quadratic relaxations to heuristics and metaheuristics.

Apart from the previously mentioned contributions by Kellerer et al. (2004) and Pisinger (2007), we are not aware of specific surveys dedicated to the general QKP. Some contributions can be found in reviews dedicated to more general research areas. The survey on nonlinear knapsack problems by Bretthauer and Shetty (2002) briefly treats it in a section on nonseparable problems. Kellerer and Strusevich (2012) reviewed a special case, known as the *symmetric quadratic knapsack problem*. The recent survey by Cacchiani, Iori, Locatelli, and Martello (2022a, 2022b) includes a section on the QKP and some of its variants.

The purpose of the present survey is to review classical and recent results on the QKP, with special emphasis on the results appeared in the twenty years that have passed after the review by Pisinger (2007), and including results on heuristic and metaheuristic approaches (only marginally treated in Pisinger (2007)). In Section 2 we introduce our notation, and discuss the main mathematical models of the problem. In Section 3 we review linearizations and reformulations of the problem. Upper bounds and exact algorithms are treated in Section 4. The following sections are devoted to non-exact methods: approximation algorithms (Section 5), heuristics and metaheuristics (Section 6). The computational performance of the reviewed methods is examined in Section 7. Conclusions and directions for future investigations follow in Section 8.

As a rule of thumb, this review considers results published in peer reviewed journals, excluding, with very few exceptions, contributions appeared in conference proceedings.

2. Mathematical models and complexity

Let us introduce the formal notation we will adopt in the following. We are given a container (*knapsack*), having positive integer *capacity* c, and n *items*, each having a non-negative integer (individual) *profit* p_i and a positive integer *weight* w_i $(i = 1, \dots, n)$. Each pair of distinct items i, j $(i \neq j)$ generates a non-negative integer *pairwise profit* p_{ij} (with $p_{ji} = p_{ij}$) if both items are selected. The objective is to select a subset of items such that their overall weight does not exceed the capacity and the total profit they produce is maximized. In the following, we assume, without loss of generality, that $\sum_{i=1}^{n} w_i > c$.

Let $N = \{1, \dots, n\}$. By introducing n binary variables x_i $(i \in N)$, taking the value 1 if and only if item i is selected, the QKP is defined by the *0-1 Quadratic Program*

$$\text{(01QP)} \quad \max \quad \sum_{i=1}^{n} p_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} p_{ij} x_i x_j \tag{1}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} w_i x_i \leq c \tag{2}$$

$$x_i \in \{0, 1\} \qquad (i \in N). \tag{3}$$

The objective function (1) is composed by a *linear term* (the sum of individual profits) and a *quadratic term* (the sum of pairwise profits), while inequality (2) expresses the classical capacity constraint of knapsack problems. The problem is a generalization of the classical linear *0-1 Knapsack Problem* (KP), which arises when $p_{ij} = 0$ $(i, j = 1, \dots, n;\ i \neq n)$.

In the seminal paper that formalized the problem, Gallo et al. (1980) adopted an alternative, elegant formulation of the QKP, namely:

$$\max \quad x^T Q\, x \tag{4}$$

$$\text{s.t.} \quad x \in X, \tag{5}$$

where Q is a non-negative square matrix of order n and X denotes the feasible region defined by (2) and (3). The two models are equivalent, as it is easily seen by assuming

- $q_{ii} = p_i$ $(i \in N)$;
- $q_{ij} + q_{ji} = p_{ij}$ $(i, j \in N; i \neq j)$,

and remembering that $x_i^2 = x_i$ for $x_i \in \{0, 1\}$.

Model (4)–(5) allows one to prove that the QKP is strongly $\mathcal{NP}$-hard, by reduction from the strongly $\mathcal{NP}$-complete problem (see Garey and Johnson (1978))

> Clique: *Given a simple undirected graph $G = (V, E)$ and a positive integer $k \leq |V|$, establish whether G contains a complete subgraph of cardinality k or more.*

It is easily seen that this problem is a special case of the QKP: given an instance of Clique, we can define an instance of the QKP in which

- Q is the adjacency matrix of G;
- $w_i = 1$ $(i = 1, \dots, n)$;
- $c = k$.

The requested clique can only exist if the optimal value of the QKP is at least $k(k-1)$.

3. Linearizations and reformulations

In this section we review linear formulations and reformulations for the 01QP model (1)–(3) of the QKP, that can be directly used as input to *Integer Linear Programming* (ILP) or *Mixed Integer Linear Programming* (MILP) solvers.

3.1. Classical linear formulation

The first idea for transforming an algebraic function in binary variables into a linear function was presented in the early Sixties by Fortet (1960). In the Seventies, the approach was independently re-discovered by Glover and Woolsey (1973, 1974), who specialized it to general polynomial functions. In order to apply this method to the QKP, let us introduce $O(n^2)$ *2-index* binary variables $\hat{y}_{ij}$, representing, for each item pair (i, j), the product $x_i x_j$ (which takes the value 1 if and only if both items i and j are selected):

$$\hat{y}_{ij} = x_i x_j \text{ for } i \in N \setminus \{n\}, j \in N\ (j > i). \tag{6}$$

The *Fortet-Glover-Woolsey* (FGW) formulation of the QKP is the ILP model

$$\text{(FGW)} \quad \max \quad \sum_{i=1}^{n} p_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} p_{ij} \hat{y}_{ij} \tag{7}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} w_i x_i \leq c \tag{8}$$

$$\hat{y}_{ij} \leq x_i \qquad (i \in N \setminus \{n\}, j \in N(j > i)) \tag{9}$$

$$\hat{y}_{ij} \leq x_j \qquad (i \in N \setminus \{n\}, j \in N(j > i)) \tag{10}$$

$$\hat{y}_{ij} \geq x_i + x_j - 1 \qquad (i \in N \setminus \{n\}, j \in N(j > i)) \tag{11}$$

$$\hat{y}_{ij} \in \{0, 1\} \qquad (i \in N \setminus \{n\}, j \in N(j > i)) \tag{12}$$

$$x_i \in \{0, 1\} \qquad (i \in N). \tag{13}$$

Constraints (9) and (10) ensure that $\hat{y}_{ij}$ takes the value 0 when at least one of the two associated variables is 0. Constraints (11) force $\hat{y}_{ij}$ to take the value 1 when both associated variables are 1.

As shown in Galli, Martello, Rey, and Toth (2021) for the *Quadratic Multiple Knapsack Problem* (QMKP) (the generalization of the QKP in which there is a set of different knapsacks instead of just one),

- variables $\hat{y}_{ij}$ for which $p_{ij} = 0$ can be omitted, and
- constraints (11) and (12) are redundant, even for the *Linear Programming* (LP) relaxation of model FGW.

All $\hat{y}_{ij}$ variables and both constraints are however used in some of the linearizations examined in the next sections.

3.2. Improved linear formulation

Billionnet and Soutif (2004b) proposed a valid linearization of model (1)–(3) by adapting to the QKP the linearization method proposed by Glover and Woolsey (1973) for general 0–1 quadratic programming problems. Their method, which adds $O(n)$ continuous variables instead of $O(n^2)$ binary variables as in model (FGW), requires

- a feasible solution $[\widetilde{x}_i]$ to the QKP and its value V;
- the computation, for each item $i \in N \setminus \{n\}$, of its "contribution", when $x_i = 1$, to the corresponding quadratic term in the objective function. Such value is represented by the sum of variable z_i and a value ℓ_i in the MILP model

$$\text{(BS)} \quad \max \quad \sum_{i=1}^{n} p_i x_i + \sum_{i=1}^{n-1} z_i + \sum_{i=1}^{n-1} \ell_i x_i \tag{14}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} w_i x_i \le c \tag{15}$$

$$z_i \le (\varphi_i - \ell_i) x_i \qquad (i \in N \setminus \{n\}) \tag{16}$$

$$z_i \le \sum_{j=i+1}^{n} p_{ij} x_j - \ell_i \qquad (i \in N \setminus \{n\}) \tag{17}$$

$$x_i \in \{0,1\} \qquad (i \in N), \tag{18}$$

where, for each $i \in N \setminus \{n\}$,

- (16) forces z_i to take the value zero when $x_i = 0$;
- (17) provides an upper bound on $z_i + \ell_i$ when $x_i = 1$;
- as proved in Billionnet and Soutif (2004b), φ_i is provided by the optimal solution value of the 0–1 linear *knapsack problem* (KP)

$$\varphi_i = \max \quad \sum_{j=i+1}^{n} p_{ij} \hat{x}_j \tag{19}$$

$$\text{s.t.} \quad \sum_{j=i+1}^{n} w_j \hat{x}_j \le c - w_i \tag{20}$$

$$\hat{x}_j \in \{0,1\} \qquad (j \in N (j > i)) \tag{21}$$

- and ℓ_i is induced by the feasible solution $[\widetilde{x}_i]$ as

$$\ell_i = \begin{cases} L_i & \text{if} \quad \widetilde{x}_i = 1; \\ L'_i & \text{if} \quad \widetilde{x}_i = 0, \end{cases}$$

with L_i and L'_i given by the optimal solution values of the two linear programs

$$L_i = \min \quad \sum_{j=i+1}^{n} p_{ij} \hat{x}_j \tag{22}$$

$$\text{s.t.} \quad \sum_{j=1}^{n} w_j \hat{x}_j \le c \tag{23}$$

$$\hat{z}_j \le \varphi_j \hat{x}_j \qquad (j \in N \setminus \{n\}) \tag{24}$$

$$\hat{z}_j \le \sum_{k=j+1}^{n} p_{jk} \hat{x}_k \qquad (j \in N \setminus \{n\}) \tag{25}$$

$$\sum_{j=1}^{n} p_j \hat{x}_j + \sum_{j=1}^{n-1} \hat{z}_j \ge V \tag{26}$$

$$0 \le \hat{x}_j \le 1 \qquad (j \in N), \tag{27}$$

and

$$L'_i = \min \quad \sum_{j=i+1}^{n} p_{ij} \hat{x}_j \tag{28}$$

$$\text{s.t.} \quad (23), (24), (25), (26), (27)$$

$$\hat{x}_i = 0. \tag{29}$$

As shown in Billionnet and Soutif (2004b), the LP relaxations of FGW and BS do not dominate each other.

3.3. Reformulation linearization techniques

In the mid-Eighties, Adams and Sherali (1986) proposed a new linearization technique for 0–1 quadratic programming problems. The method, known as the *Reformulation Linearization Technique* (RLT), was later extended to general 0–1 problems in Sherali and Adams (1990).

For 0–1 quadratic programming problems like the QKP, the RLT consists of adding quadratic constraints to the original ILP formulation, with the objective of strengthening the corresponding LP relaxation. Given a (linear) constraint with $\bar{n}$ binary variables, the new constraints are obtained by multiplying it by each original binary variable, thus producing $\bar{n}$ additional quadratic constraints. All the resulting quadratic constraints are then linearized by introducing auxiliary binary variables to represent the products of the original ones, together with appropriate linking constraints.

The Fortet-Glover-Woolsey model (7)–(13) of the QKP already includes the auxiliary binary variables ($\hat{y}_{ij}$) and the corresponding linking constraints (9)–(11), so we can produce an RLT model by just adding the *RLT constraints* (obtained by multiplying the original capacity constraint (8) by the original variables x_i ($i \in N$)). By observing that, for a binary variable x_i, we have $x_i^2 = x_i$, one obtains the model

(RLT1) (7)–(13)

$$\sum_{j=1}^{i-1} w_j \hat{y}_{ji} + \sum_{j=i+1}^{n} w_j \hat{y}_{ij} \le (c - w_i) x_i \qquad (i \in N). \tag{30}$$

A different RLT model, proposed in Caprara, Pisinger, and Toth (1999), can be obtained as follows. Let us define $O(n^2)$ binary variables y_{ij}, similarly to variables $\hat{y}_{ij}$ of Section 3.1 but by considering *all* ordered pairs (i, j) with $i, j \in N$ and $i \ne j$, i.e.,

$$y_{ij} = x_i x_j \text{ for } i \in N, j \in N \setminus \{i\}. \tag{31}$$

The resulting RLT model for the QKP is then

$$\text{(RLT2)} \quad \max \quad \sum_{i=1}^{n} p_i x_i + \frac{1}{2} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \ne i}}^{n} p_{ij} y_{ij} \tag{32}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} w_i x_i \le c \tag{33}$$

$$y_{ij} \le x_i \qquad (i \in N, j \in N \setminus \{i\}) \tag{34}$$

$$y_{ij} = y_{ji} \qquad (i \in N \setminus \{n\}, j \in N (j > i)) \tag{35}$$

$$\sum_{j \in N \setminus \{i\}} w_j y_{ij} \le (c - w_i)\, x_i \quad (i \in N) \tag{36}$$

$$y_{ij} \in \{0,1\} \qquad (i \in N, j \in N \setminus \{i\}) \tag{37}$$

$$x_i \in \{0,1\} \qquad (i \in N). \tag{38}$$

By observing that each pair $\{i, j\}$ appears twice in (32), the objective function maps (7) in the new variable space. Constraints (34) and

(36) are equivalent to (9) and (30), respectively. Constraints (34) and (35) imply (10), while, as previously mentioned, constraints (11) are redundant. In the next section we show that RLT2 can be particularly useful when relaxations of the model are used to produce upper bounds.

As shown in Galli et al. (2021) for the QMKP, the LP relaxations of RLT1 and RLT2 are stronger than that of FGW.

3.4. A non-polynomial linearization

Rodrigues, Quadri, Michelon, and Gueye (2012) proposed an alternative way to linearize the QKP, christened the *t-linearization*. Instead of introducing a new binary variable for each quadratic term (as in the previous approaches), they added to the classical ILP model (1)–(3) a single *real* variable t to represent the quadratic term in the objective function, i.e.,

$$\text{(t-QKP)} \quad \max \quad \sum_{i=1}^{n} p_i x_i + t \tag{39}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} w_i x_i \le c \tag{40}$$

$$t \le \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} p_{ij} x_i x_j \tag{41}$$

$$x_i \in \{0,1\} \qquad (i \in N). \tag{42}$$

The additional quadratic constraint (41) is then replaced by a set of linear constraints obtained by the characterization of the corresponding convex *quadric hull* (see Padberg (1989)). Let Π^n be the set of all permutations of the integers $1,\dots,n$. The linear formulation is thus

$$\text{(RQMG)} \quad \max \quad \sum_{i=1}^{n} p_i x_i + t \tag{43}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} w_i x_i \le c \tag{44}$$

$$t \le \sum_{i=2}^{n} \sum_{j=1}^{i-1} p_{\pi(j)\pi(i)} x_{\pi(i)} \quad (\pi \in \Pi^n) \tag{45}$$

$$x_i \in \{0,1\} \qquad (i \in N), \tag{46}$$

where $\pi(k)$ denotes the kth element in permutation $\pi \in \Pi^n$. In spite of the introduction of $n!$ additional constraints, a partial enumeration technique allows the computation of the corresponding LP relaxation upper bound and the definition of an exact solution algorithm (see Section 4.12) whose practical effectiveness is however difficult to establish (see Section 7).

4. Upper bounds and exact algorithms

In this section we review, in chronological order, the main methods proposed for computing upper bounds for the QKP and for determining an optimal solution to the problem.

4.1. Gallo, Hammer and Simeone (1980)

The upper bounds proposed by Gallo et al. (1980) are based on the computation of *upper planes*, defined as linear functions $\sum_{i=1}^{n} \pi_i x_i$ not smaller than (1) for any solution $[x_i]$ satisfying (2) and (3). Four appropriate definitions of $[\pi_i]$ are proposed in Gallo et al. (1980): the reported computational experiments show that the best trade-off between quality of the bound and computational effort is obtained by defining π_i $(i \in N)$ as the optimal solution value of the continuous KP

$$\pi_i = p_i + \max \left\{ \frac{1}{2} \sum_{j \in N \setminus \{i\}} p_{ij} \bar{x}_j \;:\; \sum_{j \in N \setminus \{i\}} w_j \bar{x}_j \le c, \quad 0 \le \bar{x}_j \le 1 \quad (j \in N \setminus \{i\}) \right\} \tag{47}$$

(which can be computed in $O(n)$ time) and obtaining the associated upper bound UB_{GHS} as the solution of the continuous KP

$$UB_{\text{GHS}} = \max \left\{ \sum_{i \in N} \pi_i x_i \;:\; \sum_{i \in N} w_i x_i \le c, \quad 0 \le x_i \le 1 \quad (i \in N) \right\}. \tag{48}$$

Upper bound UB_{GHS}, which globally requires $O(n^2)$ time, was embedded in a binary branch-and-bound scheme. The resulting algorithm was tested on a set of benchmark instances generated according to characteristics that are, still nowadays, the most common methods for generating instances to test QKP algorithms (see Section 7).

4.2. Chaillou, Hansen, and Mahieu (1989)

Chaillou, Hansen, and Mahieu (1989) adopted a classical Lagrangian relaxation of capacity constraint (2):

$$UB_{\text{CHM}}(\lambda) = \max \quad \sum_{i=1}^{n} p_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} p_{ij} x_i x_j + \lambda \left(c - \sum_{i=1}^{n} w_i x_i\right)$$

$$= \lambda c + \max \left(\sum_{i=1}^{n} \tilde{p}_i(\lambda) x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} p_{ij} x_i x_j \right) \tag{49}$$

$$\text{s.t.} \quad x_i \in \{0,1\} \qquad (i \in N), \tag{50}$$

where $\tilde{p}_i(\lambda) = p_i - \lambda w_i$ $(i \in N)$ and $\lambda \ge 0$.

Problem (49)–(50) was solved with a polynomial-time algorithm based on the solution of an associated maximum flow problem. The optimal Lagrangian multiplier λ^* was obtained through a simple binary search requiring, in the worst case, $n-1$ iterations, and hence producing the upper bound with overall time complexity $O(n^4)$ (if the $O(n^3)$ max-flow algorithm by Karzanov (1974) is used). The resulting upper bound $UB_{\text{CHM}}(\lambda^*)$ was used to limit the search space in a binary branch-and-bound scheme.

4.3. Billionnet and Calmels (1996)

Clearly, the optimal solution value of the LP relaxation of one of the models presented in Section 3 provides a valid upper bound for the QKP.

Billionnet and Calmels (1996) proposed to consider the LP relaxation of model RLT1 of Section 3.3, strengthened by the addition of triangle inequalities of the form

$$x_i + x_j + x_k - \hat{y}_{ij} - \hat{y}_{jk} - \hat{y}_{ki} \le 1 \qquad (i,j,k \in N, i < j < k). \tag{51}$$

The corresponding upper bound can be computed in polynomial time by relaxing the model to an LP with $O(n^2)$ variables and $O(n^3)$ constraints. The practical efficiency of the method was improved by initially disregarding constraints (9)–(11) and (51), and adding, at each iteration, the disregarded constraints violated by the solution of the current relaxed problem. (In order to limit the number of LP solutions, in the computational experiments the iterations were halted as soon as a prefixed optimality gap had been reached.)

A reduction procedure, a heuristic approach, and a binary branch-and-bound algorithm based on the computation of the proposed upper bound are also given in Billionnet and Calmels (1996).

4.4. Michelon and Veilleux (1996)

Michelon and Veilleux (1996) adopted a Lagrangian decomposition. By using copy variables $y_i = x_i$ $(i \in N)$ in the capacity constraint (2) and relaxing in a Lagrangian fashion constraints $y_i = x_i$ with multipliers λ_i $(i \in N)$, the computation of the resulting upper bound $UB_{\text{MV}}(\lambda)$ decomposes into two subproblems:

$$z'(\lambda) = \max \quad \sum_{i=1}^{n} \tilde{p}_i(\lambda) x_i + \frac{1}{2} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} p_{ij} x_i x_j \tag{52}$$

$$\text{s.t.} \quad x_i \in \{0,1\} \quad (i \in N), \tag{53}$$

where $\tilde{p}_i(\lambda) = p_i - \lambda_i$ $(i \in N)$, and

$$z''(\lambda) = \max \quad \sum_{i=1}^{n} \lambda_i y_i \tag{54}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} w_i y_i \leq c \tag{55}$$

$$y_i \in \{0,1\} \quad (i \in N), \tag{56}$$

so $UB_{\text{MV}}(\lambda) = z'(\lambda) + z''(\lambda)$. Subproblem (52)–(53) can be solved through an $O(n^4)$ algorithm similar to that adopted by Chaillou et al. (1989) for problem (49)–(50), while subproblem (54)–(56) is a KP and hence it can only be solved in pseudo-polynomial time $O(nc)$. The overall bound $UB_{\text{MV}}(\hat{\lambda})$ is obtained using subgradient optimization to determine near-optimal values $[\hat{\lambda}]$ for the n Lagrangian multipliers $[\lambda]$. It is shown in Pisinger (2007) that, if the subgradient process starts with $\hat{\lambda}_i = \lambda^*$ for all $i \in N$, then $UB_{\text{MV}(\hat{\lambda})} \leq UB_{\text{CHM}}(\lambda^*)$.

Upper bound $UB_{\text{MV}(\hat{\lambda})}$ was tested by embedding it in the branch-and-bound algorithm of Chaillou et al. (1989).

4.5. Hammer and Rader (1997)

Hammer and Rader (1997) presented a branch-and-bound algorithm in which the upper bounds are obtained from Lagrangian relaxation, using the $O(n^4)$ technique, based on iterated max-flow solutions, developed by Chaillou et al. (1989) (Section 4.2). A three-step procedure is applied at each decision node to reduce the current instance through variable fixation. The steps are based on:

- the previously mentioned Lagrangian relaxation (Chaillou et al., 1989);
- *order relations*, derived from sufficient conditions establishing that a relation of the form $x_i \leq x_j$ must hold in some optimal solution;
- *constraint pairing*, obtained from the method developed by Hammer, Padberg, and Peled (1975) for general integer programs.

4.6. Caprara, Pisinger, and Toth (1999)

Caprara et al. (1999) proposed an upper bound based on the Lagrangian relaxation of constraints (35) in the LP relaxation of the RLT2 model of Section 3.3, namely

$$UB_{\text{CPT}}(\lambda) = \max \quad \sum_{i=1}^{n} p_i x_i + \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \hat{p}_{ij}(\lambda) y_{ij} \tag{57}$$

$$\text{s.t.} \quad (33), (34), (36)$$

$$y_{ij} \geq 0 \qquad (i \in N, j \in N \setminus \{i\}) \tag{58}$$

$$x_i \leq 1 \qquad (i \in N), \tag{59}$$

where: (i) $\hat{p}_{ij}(\lambda) = \frac{1}{2} p_{ij} + \lambda_{ij}$ is the Lagrangian profit associated with variable y_{ij}, (ii) $[\lambda_{ij}]$ are the Lagrangian multipliers associated with the relaxed constraints (35), and, for notational convenience, (iii) $\lambda_{ij} = -\lambda_{ji}$ for $j > i$.

By observing that variables $[y_{ij}]$ only appear in (57), (34), (36), and (58), the upper bound $UB_{\text{CPT}}(\lambda)$ can be computed in $O(n^2)$ time. Indeed, if variable x_i $(i \in N)$ is fixed to a value $\bar{x}_i$, the associated variables y_{ij} $(j \in N \setminus \{i\})$ can be obtained, in $O(n)$ time, as follows:

if $\bar{x}_i = 0$ **then** $y_{ij} = 0$ and the contribution of item i to $UB_{\text{CPT}}(\lambda)$ is nul

else $(\bar{x}_i = 1)$ y_{ij} and the possible contribution $\pi_i(\lambda)$ of item i to $UB_{\text{CPT}}(\lambda)$ are defined by the optimal solution of the continuous KP

$$\pi_i(\lambda) = p_i + \max \left\{ \sum_{j \in N \setminus \{i\}} \hat{p}_{ij}(\lambda) y_{ij} : \sum_{j \in N \setminus \{i\}} w_j y_{ij} \leq c - w_i, 0 \leq y_{ij} \leq 1 \, (j \in N \setminus \{i\}) \right\} \tag{60}$$

and the value of $UB_{\text{CPT}}(\lambda)$ is obtained by solving, in $O(n)$ time, the continuous KP (48) with π_i replaced by $\pi_i(\lambda)$. By comparing (47) and (60) we can observe that, for any $i \in N$, $\pi_i(0) \leq \pi_i$ and hence $UB_{\text{CPT}}(\lambda) \leq UB_{\text{CPT}}(0) \leq UB_{\text{GHS}}$.

Near optimal Lagrangian multipliers $[\hat{\lambda}_{ij}]$ can then be obtained through subgradient optimization over the array $[\lambda_{ij}]$ of $O(n^2)$ elements. Heuristic algorithms, a reduction procedure, and a binary branch-and-bound algorithm based on $UB_{\text{CPT}}(\hat{\lambda})$ are also proposed in Caprara et al. (1999).

4.7. Billionnet, Faye, and Soutif (1999)

Billionnet et al. (1999) proposed to compute an upper bound through Lagrangian decomposition. They decomposed the QKP into m independent subproblems by partitioning the item set N into m *clusters* $I_1, \ldots, I_m$. For each cluster k $(k \in M = \{1, \ldots, m\})$, they introduced new (copy) variables $y_j^k = x_j$ $(j \in N \setminus I_k)$ and added a redundant capacity constraint as follows. Let $\alpha(i)$ denote the index of the cluster containing item i. The QKP can then be written as

$$\max \quad \sum_{k=1}^{m} \left(\sum_{i \in I_k} p_i x_i + \frac{1}{2} \sum_{i \in I_k} \sum_{j \in I_k \setminus \{i\}} p_{ij} x_i x_j + \frac{1}{2} \sum_{i \in I_k} \sum_{j \in N \setminus I_k} p_{ij} x_i x_j \right)$$

$$= \sum_{k=1}^{m} \left(\sum_{i \in I_k} p_i x_i + \frac{1}{2} \sum_{i \in I_k} \sum_{j \in I_k \setminus \{i\}} p_{ij} x_i x_j + \frac{1}{2} \sum_{i \in I_k} \sum_{j \in N \setminus I_k} p_{ij} x_i y_j^k \right) \tag{61}$$

$$\text{s.t.} \quad y_j^k = x_j \qquad (k \in M, j \in N \setminus I_k) \tag{62}$$

$$x_i y_j^{\alpha(i)} = x_j y_i^{\alpha(j)} \qquad (i = 1, \ldots n-1, \, j = i+1, \ldots n, \, \alpha(i) \neq \alpha(j)) \tag{63}$$

$$\sum_{i \in I_k} w_i x_i + \sum_{j \in N \setminus I_k} w_j y_j^k \leq c \quad (k \in M) \tag{64}$$

$$x_i \in \{0,1\} \qquad (i \in I_k, \, k \in M) \tag{65}$$

$$y_j^k \in \{0,1\} \qquad (j \in N \setminus I_k, \, k \in M), \tag{66}$$

where (62) and (63) link the new variables to the original ones, while constraints (64) introduce relaxed copies of the capacity constraint. By relaxing in a Lagrangian fashion constraints (62) with multipliers λ_j^k $(j \in N \setminus I_k, \, k \in M)$, and constraints (63) with multipliers μ_{ij} $(i = 1, \ldots n-1, \, j = i+1, \ldots n, \, \alpha(i) \neq \alpha(j))$, by algebraic manipulation the problem can be decomposed into m independent QKPs of the form, for each $k \in M$,

$$\max \quad \sum_{i \in I_k} \left(\left(p_i + \sum_{h \in M \setminus \{k\}} \lambda_i^h \right) x_i + \sum_{j \in I_k \setminus \{i\}} \frac{1}{2} p_{ij} x_i x_j \right) - \sum_{j \in N \setminus I_k} \lambda_j^k y_j^k$$

$$+ \sum_{i \in I_k} \left(\sum_{\substack{j \in N \setminus I_k \\ j > i}} \left(\frac{1}{2} p_{ij} + \mu_{ij} \right) x_i y_j^k + \sum_{\substack{j \in N \setminus I_k \\ j < i}} \left(\frac{1}{2} p_{ij} - \mu_{ji} \right) x_i y_j^k \right) \tag{67}$$

$$\text{s.t.} \quad \sum_{i \in I_k} w_i x_i + \sum_{j \in N \setminus I_k} w_j y_j^k \leq c \tag{68}$$

$$x_i \in \{0,1\} \qquad (i \in I_k) \tag{69}$$

$$y_j^k \in \{0,1\} \qquad (j \in N \setminus I_k). \tag{70}$$

If the m clusters are defined by choosing a small cardinality (e.g., $|I_k| = 5$ for each $k \in M$, as adopted by the authors (Billionnet et al., 1999)), it is possible to enumerate, for each cluster I_k $(k \in M)$, all solutions produced by the associated variables x_i $(i \in I_k)$. For each $k \in M$ and for each feasible solution $\bar{x}_i$ $(i \in I_k)$, the corresponding problem (67)–(70) is defined by the linear KP

$$\max \quad \beta^k(\lambda) + \sum_{j \in N \setminus I_k} p_j^k(\lambda, \mu) y_j^k \tag{71}$$

$$\text{s.t.} \quad \sum_{j \in N \setminus I_k} w_j y_j^k \leq c - \gamma^k \tag{72}$$

$$y_j^k \in \{0,1\} \qquad (j \in N \setminus I_k), \tag{73}$$

where the coefficients

- $\beta^k(\lambda) = \sum_{i \in I_k} \left(\left(p_i + \sum_{h \in M \setminus \{k\}} \lambda_i^h \right) \bar{x}_i + \sum_{j \in I_k \setminus \{i\}} \frac{1}{2} p_{ij} \bar{x}_i \bar{x}_j \right)$;
- $p_j^k(\lambda, \mu) = -\lambda_j^k + \sum_{\substack{i \in I_k \\ j > i}} \left(\frac{1}{2} p_{ij} + \mu_{ij} \right) \bar{x}_i + \sum_{\substack{i \in I_k \\ j < i}} \left(\frac{1}{2} p_{ij} - \mu_{ji} \right) \bar{x}_i$ $(j \in N \setminus I_k)$;
- $\gamma^k = \sum_{i \in I_k} w_i \bar{x}_i$

can be computed in $O(n)$ time (due to the constant bound on the cardinality of I_k).

Let $\pi^k(\lambda, \mu)$ be the optimal solution value of the continuous relaxation of (71)–(73), which can be computed in $O(n)$ time, leading to an $O(n^2)$ time complexity for the computation of the upper bound produced by the $m = \frac{n}{5}$ clusters adopted in Billionnet et al. (1999). Good multipliers $[\hat{\lambda}_j^k]$ and $[\hat{\mu}_{ij}]$ can then be obtained through subgradient optimization over the two arrays $[\lambda_j^k]$ and $[\mu_{ij}]$ (each of $O(n^2)$ elements), producing an overall upper bound $UB_{\text{BFS}} = \sum_{k \in M} \pi^k(\tilde{\lambda}, \tilde{\mu})$.

4.8. Semidefinite programming relaxations(2000-)

Starting from the year 2000, some *Semidefinite Programming* (SDP) relaxations of the QKP have been adopted by various authors for producing upper bounds and heuristic solutions. While for the bounds and algorithms reviewed in the other sections, extensive results of computational experiments have been provided, very little is known about the practical usefulness of these approaches, and the few computational results known indicate that they are not competitive (although the recent progress in conic optimizers could suggest new computational testings). In the following, we briefly highlight the main results produced by these studies.

The first results in this area were presented by Helmberg, Rendl, and Weismantel (2000), who discussed possible improvements produced by cutting planes and combinatorial cuts (e.g., the triangle inequalities (51))for a number of basic SDP relaxations of the QKP. They concluded that the approach is theoretically interesting, but too time consuming to solve QKPs.

Zheng, Sun, Li, and Xu (2012) investigated the duality gap between the QKP and its Lagrangian dual or SDP relaxation, characterizing such gap by a distance measure from the set $\{0,1\}^n$ to polyhedral sets related to the saddle point conditions. Using tools from discrete geometry, they showed that the duality gap can be reduced by an amount proportional to the square of the distance.

Zhou, Chen, Wang, and Xing (2013) proposed a *conic* approximation method for the QKP and presented numerical comparisons with the direct SDP relaxation method. Such results were however obtained on small size instances ($n \leq 50$): for such instances, that can be solved to proven optimality in a few seconds by exact state-of-the-art algorithms, their method took thousands of seconds.

Very recently, Tang and Toh (2023) studied the low rank formulation of an SDP relaxation of the QKP. In order to solve it, they studied the geometric properties of its feasible region, which turns out to be an algebraic variety. They provided computational experiments showing the superiority of their approach with respect to two other SDP algorithms. In addition, they applied a rounding procedure to the solution of the relaxation to produce a feasible solution to the QKP.

4.9. Billionnet and Soutif (2004)

In Billionnet and Soutif (2004b), Billionnet and Soutif considered their improved linear formulation BS (see Section 3.2) and the classical linear formulation FGW (see Section 3.1). They used the ILP solver CPLEX both to compute, the upper bounds produced by the corresponding LP relaxations and to directly solve the QKP.

In a paper that appeared in the same year (Billionnet & Soutif, 2004a), they also proposed a binary depth-first branch-and-bound algorithm, based on variable fixing procedures and the computation of the Lagrangian decomposition upper bound by Billionnet et al. (1999) (see Section 4.7).

4.10. Pisinger, Rasmussen, and Sandvik (2007)

Pisinger et al. (2007) presented an exact solution framework (*aggressive reduction*) in which the main solution tool is a variable fixing preprocessing. Such reduction phase, based on implicit-enumeration techniques (and hence requiring non-polynomial time), attempts to fix a high number of variables at their optimal values using

- the computation of the upper bounds proposed in Caprara et al. (1999) (see Section 4.6) and Billionnet et al. (1999) (see Section 4.7), and
- the definition of a number of heuristic solutions found during the subgradient optimization phase executed to obtain good Lagrangian multipliers associated with the upper bounds.

The approach only switches to branch-and-bound (the algorithm by Caprara et al. (1999)) when the reduction phase can no longer fix any variable and subgradient optimization can no longer improve any of the bounds.

4.11. Létocart, Nagih, and Plateau (2012)

The computation of the bounds by Caprara et al. (1999) (see Section 4.6) and Billionnet et al. (1999) (see Section 4.7), requires the solution of a high number of continuous linear KPs to obtain good Lagrangian multipliers through subgradient optimization. Létocart, Nagih, and Plateau (2012) proposed to improve the efficiency of such computation through a reoptimization technique (based on that proposed by Thiongane, Nagih, and Plateau (2006) for the *0-1 bidimensional knapsack problem*) which stores, for each KP solution produced during the process, information that can accelerate the solution of the next KP.

4.12. Rodrigues, Quadri, Michelon, and Gueye (2012)

Rodrigues et al. (2012) used a constraint generation algorithm to implement an implicit enumeration method for the computation of the upper bound produced by the LP relaxation of their (non-polynomial) t-linearization (see Section 3.4). They presented a polynomial-time algorithm for the separation problem induced by each generation of an inequality in (45). The upper bound was embedded in a binary branch-and-bound algorithm which includes both an initialization phase producing a feasible solution (through the heuristic algorithm by Billionnet and Calmels (1996), see Section 6.1) and a classical reduction phase performed at each decision node.

4.13. Fampa, Lubke, Wang, and Wolkowicz (2020)

Fampa, Lubke, Wang, and Wolkowicz (2020) considered a parametric convex quadratic programming relaxation of the QKP and presented a primal–dual interior point method to optimize a perturbation of the quadratic function and to compute an upper bound. (The convergence of the algorithm is not provided in Fampa et al. (2020), although

convergence analysis for similar problems can be found in the literature.) An effective cutting plane algorithm is used to improve the upper bound and to produce heuristic solutions. They also proved that the same perturbation approach, when applied in the context of semidefinite programming relaxations of the QKP (see Section 4.8), cannot improve the upper bound given by the corresponding linear semidefinite programming relaxation.

4.14. Fomeni, Kaparis, and Letchford (2022)

Fomeni, Kaparis, and Letchford (2022) presented a sophisticated method, consisting of four families of cutting planes to be added to the FGW formulation (7)–(13) in order to produce a tight upper bound for the QKP:

- recall that the RLT inequalities (30) were obtained by multiplying the original capacity constraint (8) by the original variables x_i ($i \in N$). Fomeni et al. (2022) included, in addition, the inequalities obtained by multiplying (8) by the complement variables, $1 - x_i$ ($i \in N$), i.e.,
$$\sum_{j=1}^{i-1} w_j(x_j - \hat{y}_{ji}) + \sum_{j=i+1}^{n} w_j(x_j - \hat{y}_{ij}) \le c - cx_i \qquad (i \in N); \tag{74}$$
- the triangle inequalities applied to the QKP by Billionnet and Calmels (1996) (Section 4.3) and Helmberg et al. (2000) (Section 4.8);
- the cover-tree inequalities developed by Johnson et al. (1993) for a min-cut clustering problem;
- the inequalities obtained by applying the RLT to the classical lifted cover inequalities developed by Balas (1975) and Wolsey (1975) for the KP.

The cutting planes were embedded in a cut-and-branch framework which also includes separation procedures, primal heuristics, and reduction. The resulting outcome produces a binary LP formulation, which is then given in input to an ILP solver to obtain an upper bound (from the corresponding LP relaxation) and an optimal solution to the problem. As we will see in Section 7, the resulting upper bound is very tight, but it requires very high computing times.

5. Approximation algorithms

Very little is known on the approximation of the QKP, and obtaining developments in this field appears to be problematic. Few theoretical results have been obtained through the graph characterization of the problem induced by (4)–(5). Given a capacity c and a simple undirected graph $G = (V, E)$ with:

(i) profits p_{ij} associated with the edges $(i, j) \in E$;

(ii) weights w_i and profits p_{ii} associated with the vertices $v_i \in V$,

the QKP is equivalent to finding a collection of vertices with total weight at most c and total profit (i.e., the sum of all profits of vertices and edges in the induced subgraph) as large as possible.

Fomeni and Letchford (2014) observed that the results by Khot (2006), which rule out the existence of a *Polynomial Time Approximation Scheme* (PTAS) for a number of graph problems, imply that, even in the special case of unitary weights, the QKP cannot be approximated within a constant factor in polynomial time.

Pisinger et al. (2007) observed that the special case of G where the weights and profits are all one is equivalent to the *densest k-subgraph problem* (select k vertices so that their induced subgraph has a maximum number of edges) with $k = c$.

Using a result by Bhaskara, Charikar, Chlamtac, Feige, and Vijayaraghavan (2010) on the approximability of the densest k-subgraph problem, Taylor (2016) proved that there is an algorithm for the QKP that has, for any $\varepsilon > 0$, an approximation ratio within $O(n^{2/5+\varepsilon})$ and a running time within $O(n^{9/\varepsilon})$. To the best of our knowledge, this is the only known approximation result for the general QKP.

Few results have been obtained for special cases and generalizations. Rader and Woeginger (2002) considered different classes of the underlying graph:

- for the case of *edge series–parallel graphs*, they provided two pseudo-polynomial time *dynamic programming* (DP) algorithms and a *Fully Polynomial Time Approximation Scheme* (FPTAS);
- in contrast with this, they showed that, for the case of *vertex series–parallel graphs*, the problem is strongly $\mathcal{NP}$-hard;
- finally, they proved, via transformation from the *subset-sum problem*, that the generalization of the QKP in which single and pairwise profits can also take negative values cannot have a polynomial-time approximation algorithm with fixed approximation ratio unless $\mathcal{P} = \mathcal{NP}$.

Pferschy and Schauer (2016) gave a *Polynomial Time Approximation Scheme* (PTAS) for the case of graphs with bounded treewidth, and a PTAS for the case of planar graphs.

6. Heuristics and metaheuristics

The first heuristic for the QKP was included in the seminal paper by Gallo et al. (1980). In the following fifteen years, only one new heuristic was proposed, while others appeared in the next decade. The first metaheuristic algorithm was presented in the early Noughties by Glover, Kochenberger, Alidaee, and Amini (2002), giving rise to a stream of approaches of this kind.

6.1. Heuristic algorithms

Gallo et al. (1980) proposed a number of upper planes to be used within a branch-and-bound framework (see Section 4). Let $f(x)$ denote the objective function in (1), and $g(x)$ a linear function such that $g(x) \ge f(x)$ for all $x \in X$, where X denotes the feasible region defined by (2) and (3). Any optimal solution $\hat{x}$ to the (linear) KP corresponding to the relaxed problem $\{\max g(x) : x \in X\}$ provides simultaneously an upper bound $g(\hat{x})$ and a lower bound $f(\hat{x})$. As described in Section 4, Gallo et al. (1980) proposed four different ways to obtain suitable $g(x)$ functions. In order to quickly compute bounds, they solved the continuous relaxation of the corresponding KP, whose optimal solution has at most one binary variable, and hence a feasible QKP solution is immediately obtained by setting such variable to zero. The resulting solution was then improved through two simple procedures originally developed by Petersen (1974):

- *inserts*: given $\hat{x}_i = 0$, set $\hat{x}_i = 1$,
- *exchanges*: given $\hat{x}_i = 0$ and $\hat{x}_j = 1$, set $\hat{x}_i = 1$ and $\hat{x}_j = 0$,

which are performed provided the capacity constraint (2) is preserved and the overall profit is increased. The computational experiments provided in Gallo et al. (1980) show a satisfactory optimality gap for the resulting solutions.

The algorithm by Chaillou et al. (1989) generates an (infeasible) solution by inserting all items into the knapsack (i.e., by setting $x_i = 1$ $\forall\, i \in N$), and then drives it to feasibility by iterating the following steps until the solution becomes feasible:

(1) let $S = \{i \in N : x_i = 1\}$;

(2) for each $i \in S$, compute $\rho_i = \left(p_i + \frac{1}{2}\sum_{j \in S \setminus \{i\}} p_{ij}\right)/w_i$;

(3) let $i^* = \arg\min_{i \in S}\{\rho_i\}$ and set $x_{i^*} = 0$.

An efficient procedure is proposed for updating the values ρ_i ($i \in S$) at each execution of the three steps above.

Billionnet and Calmels (1996) proposed a hybrid approach which combines the previous methods. The first phase produces a feasible solution through the algorithm by Chaillou et al. (1989), while the

second phase is a sequence of local insertions and exchanges following the procedures by Petersen (1974) (as already done by Gallo et al. (1980)).

Hammer and Rader (1997) presented a greedy-type heuristic based on one of the upper planes used (Gallo et al., 1980) (see Section 4.1), obtained by replacing the objective function (1) with the linear function

$$\sum_{i=1}^{n} p_i^* x_i, \tag{75}$$

where, for each $i \in N$, $p_i^* = p_i + \frac{1}{2}\sum_{j \in N \setminus \{i\}} p_{ij}$. Hammer and Holzman (1992) had proved that (75) is the best linear approximation of (1) in the 2-norm. The algorithm iteratively sets to one the free variable associated with the item having highest $\rho_i = p_i^*/w_i$ ratio, sets to zero the free variables associated with weights that no longer fit in the residual capacity, and updates the ρ_i values accordingly. As in Gallo et al. (1980), the resulting solution is improved through the *inserts* and *exchanges* procedures by Petersen (1974).

Note that the algorithm proposed by Chaillou et al. (1989) is very similar to the one by Hammer and Rader (1997). Indeed, the former is a dual greedy that removes *least* promising items until the solution becomes feasible, while the latter is a primal greedy that keeps adding *most* promising items while possible. Both algorithms rely on the same "metric" ρ_i to define a *promising* item.

In their branch-and-bound algorithm for the QKP (see Section 4), 1u Caprara et al. (1999) used two different heuristics, at the root node and at subsequent decision nodes, respectively, both consisting of two phases (starting solution followed by local exchanges) and only differing in the first one. At the root node, the starting solution is obtained through the algorithm by Chaillou et al. (1989). At subsequent nodes a new solution is produced at each iteration of the subgradient procedure used to compute Lagrangian multipliers: whenever a new set of multipiers is obtained, the solution of the continuous relaxation of the current Lagrangian problem is rounded down to produce a starting feasible solution. In both situations, the second phase improves the starting solution through a series of insertions and exchanges.

Talaván and Yáñez (2006) presented a *neural network* approach for a generalization of the problem (obtained by adding a set of linear constraints) which includes as particular cases both the QKP and the *traveling salesman problem*. They presented computational experiments obtained on 25 instances from Reinelt (1991) TSPLIB, exhibiting a mean value of the ratio (tour length)/(optimal tour length) equal to 2.19, while no result was provided for the QKP.

Sipahioglu and Saraç (2009) developed a *Modified Subgradient Algorithm* (MSG, a method originally proposed by Gasimov (2002) for nonconvex minimization problems with equality constraints) to solve the QKP. They start by converting the problem into a continuous nonlinear problem through an additional constraint that guarantees a binary solution, and solve the dual problem via MSG by constructing an augmented Lagrangian function. The algorithm was computationally tested on twenty small-size instances, but no comparison was provided with other algorithms.

Wu, Yang, Bai, and Mammadov (2011) studied global optimality conditions (either necessary or sufficient) for the QKP. The necessary conditions were used to develop a local optimization method for the problem, while the sufficient ones lead to a global optimization algorithm. The efficiency of the proposed approach was only evaluated on some numerical examples.

Fomeni and Letchford (2014) modified the classical DP recursion for the linear knapsack problem to obtain a constructive heuristic for the QKP, enhanced by a simple local search. The algorithm, which has $O(n^2c)$ time complexity and $O(nc)$ space complexity, provided solutions within 0.05% of the optimum on a large set of randomly generated instances.

Cunha, Simonetti, and Lucena (2016) studied two linearizations of the QKP, yielding two Lagrangian heuristics: one derived from the exact method by Caprara et al. (1999) and one based on a *relax-and-cut* scheme (see Escudero, Guignard, and Malik (1994)). Computational comparisons with other heuristics from the literature proved their efficiency, especially on large instances with up to 1000 items.

Recently, Fomeni (2023) presented a combination of a DP approach and a deterministic local search, both adapted to the space of the lifted variables, and hence resulting in a time complexity increase from $O(n^2c)$ to $O(n^3c)$ (but no space complexity increase) with respect to the DP heuristic of Fomeni and Letchford (2014). Further improvements were obtained by Fennich, Fomeni, and Coelho (2024), who included, at each stage of the DP framework by Fomeni (2023), a more accurate computation of the profit contribution of each item and an effective local search procedure, especially aimed at the solution of a class of difficult QKP instances (see Sections 7.1 and 7.3).

6.2. Metaheuristic algorithms

As it frequently occurs for difficult combinatorial optimization problems, several metaheuristics have been presented in the last two decades. In this section we briefly review such approaches.

To the best of our knowledge, the first metaheuristic for the QKP was proposed in 2002 by Glover et al. (2002). They transformed the capacity constraint into an equality constraint through a classical slack variable, and added to the objective function a quadratic "penalization" term produced by the constraint violation. The problem can then be written as a general unconstrained quadratic problem of the form $\left\{\max x^T \hat{Q}\, x : x \in \{0,1\}\right\}$, where $\hat{Q}$ is an appropriately modified Q matrix. By adapting to the QKP the algorithm they had previously developed in Glover, Kochenberger, Alidaee, and Amini (1999) for binary quadratic programs, Glover et al. (2002) obtained a *Tabu search* algorithm, that was computationally evaluated on a set of QKP instances randomly generated according to characteristics similar to those adopted for testing exact QKP algorithms (see Section 7.1).

In 2005 Julstrom (2005) presented two simple greedy heuristics similar to the heuristic proposed by Hammer and Rader (1997). The heuristics were then embedded in a *genetic algorithm* which favors items with higher ρ_i values in the generation of chromosomes, and in the crossover and mutation operators. The effectiveness of the resulting procedure was computationally evaluated on 20 instances (with $n \leq 200$) of the Billionnet and Soutif (2004b) QKP library (see Section 7.1). For all the considered instances, the genetic algorithm was able to find the optimal solution.

A variety of metaheuristic paradigms was produced in the following years by various authors. Some of these approaches exhibited a computational performance better than that of previous approaches, namely

- *mini-swarm* algorithm (Xie & Liu, 2007);
- *artificial bee colony* (Pulikanti & Singh, 2009);
- GRASP and Tabu search (Yang, Wang, & Chu, 2013);
- Iterated hyperplane exploration (Chen & Hao, 2017),

while other paradigms did not consistently provide improvements over the state-of-the-art at the time in which they were published, e.g., *quantum evolutionary* (Patvardhan, Bansal, & Srivastav, 2015), *artificial fish swarm* (Azad, Rocha, & Fernandes, 2014), *migrating birds* (Lalla-Ruiz, Segredo, & Voß, 2019). As it will be detailed in Section 7.3, the best performance is currently obtained by the algorithm by Chen and Hao (2017). Their method, inspired by successful heuristics developed for the multidimensional knapsack problem, relies on searching over a cardinality constrained set of hyperplanes that concentrates on promising solutions of the feasible set. The resulting hyperplanes are then explored through variable fixing and Tabu search.

Worth is mentioning in addition that a study on the possibility of solving the QKP by modeling it as an Ising spin model, using an Ising machine, was recently presented by Parizy and Togawa (2021).

7. Computational experiments

It is not easy to summarize and compare the outcome of computational experiments that span a quarter of a century, especially when they have been obtained on different instances, on computers having different speed, and using different versions of MILP solvers.

7.1. Instances

The most classical instance generation, adopted by many authors, is the one originally proposed by Gallo et al. (1980):

- linear and quadratic profits p_i and p_{ij} ($= p_{ji}$) are nonzero with *density* d, i.e., they are uniformly random integers in $[1, 100]$ with probability d, and zero otherwise;
- four density values are considered: $d = 0.25$, $d = 0.50$, $d = 0.75$, $d = 1.00$;
- the weights w_i are uniformly random integers in the range $[1, 50]$;
- the capacity is a uniformly random integer in the range $[50, \sum_{i=1}^{n} w_i]$.

In the following, instances generated in this way will be referred to as *standard instances*. Most subsequent authors adopted this generation for testing their approaches. A subset of 100 instances ($n \leq 300$), used in the computational experiments performed by Billionnet and Soutif (2004a, 2004b), can be downloaded from https://cedric.cnam.fr/~soutif/QKP/. Worth is mentioning however that other authors did not use them, and preferred to generate the instances anew.

Two sets of larger instances (80 with $n = 1000$ and 2000, and 40 with $n = 5000$ and 6000), were generated by Yang et al. (2013) and by Chen and Hao (2017), respectively. These 120 instances can be downloaded from http://www.info.univ-angers.fr/pub/hao/QKP.html

Pisinger (2006) studied two problems which turn out to be special cases of the QKP:

- *p-dispersion-sum problem*: given n locations and a value $p \leq n$, select p locations such that the total distance sum among them is a maximum;
- *densest k-subgraph problem* (see Section 5).

The former problem is the QKP arising when $c = p$, the individual profits p_i take the value 0, the pairwise profits p_{ij} are given by the distance between locations i and j, all weights w_i take the value one, and the density is 1.00. The latter problem arises (for any density) when $c = k$, all nonzero (individual and pairwise) profits and all weights take the value one. Pisinger (2006) proposed a branch-and-bound framework for these problems, which turned out to be very effective on randomly generated instances. The same instances were later used by Pisinger et al. (2007) and by Fomeni et al. (2022) to test their QKP exact algorithms, not surprisingly requiring much larger CPU times than those required by the specialized algorithm presented in Pisinger (2006).

Another very special class of QKP instances (called "hidden clique" instances) was designed by Schauer (2016). Such instances are obtained by embedding a clique of size $\lfloor\sqrt{n}\rfloor$ in a random graph $G(n, \frac{1}{2})$ (a graph of n nodes in which every edge (i, j) exists with probability $\frac{1}{2}$). An hidden clique instance defines a QKP when $c = \lfloor\sqrt{n}\rfloor$, and all weights w_i have value one, all individual profits p_i have value zero, and every pairwise profit p_{ij} has value one if edge (i, j) exists and zero otherwise. For such instances, the optimal solution value is known (with overwhelming probability) to be equal to $\frac{1}{2}\lfloor\sqrt{n}\rfloor(\lfloor\sqrt{n}\rfloor - 1)$, but there is theoretical evidence that solving them is hard. Their solution was studied in Schauer (2016), Fomeni (2023), and Fennich et al. (2024).

7.2. Upper bounds and exact algorithms

In 1997, Hammer and Rader (1997) published the results of the first extensive computational testing of exact QKP algorithms. The testing was performed on a SPARCserver 1000, using standard instances (with a slightly different generation of the capacities) with n up to 100. The results showed the superiority of the method proposed in Hammer and Rader (1997) (Section 4.5) over the algorithms by Gallo et al. (1980) and Chaillou et al. (1989). Caprara et al. (1999) later made computationally experiments on standard instances with $n \leq 400$, using an HP 9000/735 125MHz workstation. The results indicated that their algorithm (Section 4.6) outperforms all previous exact algorithms. The corresponding code `quadknap` (implemented in C language) can be downloaded from http://hjemmesider.diku.dk/~pisinger/quadknap.c.

A thorough analysis of the computational performance of the upper bounds developed until 2003 was carried out by Pisinger (2007) and his students, (Rasmussen & Sandvik, 2003). They implemented in C language all such bounds, and tested them on a Linux-PC with a Pentium III (Coppermine) 930 MHz processor. The tests were performed on standard instances with n up to 200, executing 10 instances for each pair (n, d). The LP relaxations were solved with CPLEX 7.0, while the SeDuMi 1.05 package by Sturm (2001) was used for the semidefinite programming relaxations. Extensive computational experiments showed that, by considering both the *average percentage gap* with respect to the optimal solution value and the *average CPU time*, the best results are provided by

- the bound by Billionnet et al. (1999) (Section 4.7), which produces the smallest gaps (around 0.5%), with global average CPU times around 40 s;
- the bound by Caprara et al. (1999) (Section 4.6), which gives higher gaps (around 1.3%) in less than 2 s;
- the bound by Chaillou et al. (1989) (Section 4.2), which has even higher gaps (around 1.9%) but is much faster, requiring about 0.3 s.

The experiments additionally showed that the SDP relaxation by Helmberg et al. (2000) (Section 4.8) provides bounds with gaps around 0.8% and CPU times over 400 s, i.e., clearly dominated by those obtained by Billionnet et al. (1999).

Coming to the results appeared in the last two decades, the following computational results were presented:

- Billionnet and Soutif (2004b) (Section 4.9) performed computational experiments on a Pentium II 300 MHz computer, considering standard instances with values of n between 80 and 140. They used CPLEX 6.0 for solving their reformulations, both to generate upper bounds through LP relaxation and to produce an optimal solution. Their computational results show that their methods are dominated by those by Caprara et al. (1999) (Section 4.6), which produce upper bounds with smaller gaps and require, even by considering the different speeds of the computers, considerably smaller CPU times both for computing bounds and for producing exact solutions. The exact algorithm simultaneously published by the same authors (Billionnet & Soutif, 2004a), tested on the same computer with values of n up to 300, obtained instead good computational results, beating the algorithm by Caprara et al. (1999) on instances with $140 \leq n \leq 300$ for density 0.25 and with $180 \leq n \leq 300$ for density 0.50.
- the computational results obtained by Pisinger et al. (2007) (Section 4.10) on standard instances, using an Intel Pentium 4 running at 2.4 GHz, showed that their method obtains the best results for instances with $n > 300$ (and up to 1500);
- Létocart et al. (2012) (Section 4.11) tested their reoptimization technique on an Intel Xeon bi-processor dual core 3 GHz (using only one core), considering standard instances with n up to 600.

The results indicate that reoptimization reduces the computational effort required by the preprocessing phase of the methods by Caprara et al. (1999) (Section 4.6) and Billionnet et al. (1999) (Section 4.7) by 4.2% and 21.3%, respectively;

- Rodrigues et al. (2012) (Section 4.12) used a Pentium 4 2.66 GHz computer to evaluate the upper bound produced by the LP relaxation of their t-linearization and their exact branch-and-bound algorithm. They considered standard instances with $n = 100$ for the upper bound, and with n up to 400 for the exact algorithm. Concerning the upper bound, the reported results show that it is dominated by the one by Billionnet et al. (1999) (Section 4.7), for what concerns both gaps and CPU time. Concerning the exact algorithm, no evaluation can be assessed, since there are evident incongruences in the presented results: indeed, for $n = 100$, the authors report very small CPU times (equal, on average, to 0.73 s, see Table 7.3 in Rodrigues et al. (2012)), which are however two orders of magnitude *smaller* than those reported for the computation of the corresponding upper bound (equal, on average, to 79.47 s, see Table 7.2 in Rodrigues et al. (2012)). We contacted the authors, who were not able to provide an explanation.
- Fampa et al. (2020) (Section 4.13) evaluated their parametric convex quadratic programming relaxation on 10 standard instances (5 with $n = 50$ and 5 with $n = 100$, density not provided) on an Intel Core i5-4200U 2.3 GHz notebook. Their method produced upper bounds with very tight gaps (around 0.09%) but at the expenses of huge CPU times, about three orders of magnitude higher than those required by state-of-the-art algorithms to produce an optimal solution;
- Fomeni et al. (2022) (Section 4.14) made computational experiments on standard instances, with n up to 800, on a single cluster node of the supercomputer of *Compute Canada* with processor at 2.26 GHz (solving LPs with CPLEX 12.5). Their upper bounds are very tight but required CPU times always higher than those needed by state-of-the-art algorithms to find the optimal solution of the corresponding instances.

Overall recommendations

In conclusion, even by considering the differences in hardware and software, the following general considerations can be drawn:

- *Upper bounds*: No practical improvement has been produced in the last two decades. Without obviously considering upper bounds requiring CPU times larger that those required to provide the optimal solution, the ranking established by Pisinger (2007) in 2007 (see the beginning of the present section) remains valid;
- *Exact Algorithms*: For $n \leq 300$, the winners are
  - *Density 0.25*: Billionnet and Soutif (2004a) (Section 4.9);
  - *Density 0.50*: Caprara et al. (1999) (Section 4.6) for $n \leq 200$, Pisinger et al. (2007) (Section 4.10) for $n \geq 200$;
  - *Densities 0.75, 1.00*: Caprara et al. (1999) (Section 4.6),

  while, for $n \geq 300$ (and up to 1500) the winner is Pisinger et al. (2007) (Section 4.10).

These results show that, as it frequently occurs for difficult combinatorial optimization problems, the success of an enumerative approach is mainly due to an effective trade-off between the enumeration scheme and the bound computation. The branch-and-bound methods by Billionnet and Soutif (2004a) and Caprara et al. (1999) are quite similar to each other. The algorithm by Pisinger et al. (2007) is basically the same as that by Caprara et al. (1999) with the addition of powerful initial reduction techniques, which explains its effectiveness on large-size instances. The bound by Billionnet et al. (1999) (used in the algorithm by Billionnet and Soutif (2004a)) is more accurate than that by Caprara et al. (1999), but it requires higher computing times: the results in Table 2 of Pisinger (2007) show that, for densities higher than 0.25, the quality of the two bounds gets closer, which is probably the reason behind the different behavior of the two corresponding algorithms for different densities. Finally, the fact that no improvement was produced by the upper bounds proposed in the last two decades derives from the very high computing times needed to produce smaller optimality gaps.

7.3. Heuristics and metaheuristics

No computational experiment has been reported for the approximation algorithms reviewed in Section 5.

The experiments performed in the first twenty years provide a nice picture of the developments of computers and languages in that period. Gallo et al. (1980) reported good computational results obtained by the FORTRAN implementation of their heuristics of Section 6.1, executed for small-size ($n \leq 50$, density $d = 1.00$) standard instances on an IBM 370/168 computer. The algorithm by Chaillou et al. (1989) obtained, for instances with the same characteristics, better approximation at the expense of higher CPU times (although obtained on a slower IBM 360/65 computer). Further improvements were obtained by Billionnet and Calmels (1996) on a wider set of instances of the same type ($n \leq 30$ with all four densities, $n = 40$ with $d = 1.00$) with a C language implementation of their heuristics, run on an HP 9000. A slightly better performance was obtained by Hammer and Rader (1997) with a C code implementation of their algorithm, run on a SPARCstations 1+ and a SPARCserver 1000 with two processors. No computational comparison of their heuristics with other QKP methods was provided by Talaván and Yáñez (2006), Sipahioglu and Saraç (2009), and Wu et al. (2011).

In 2016, Cunha et al. (2016) performed, on a Intel Xeon X5472 3.00 GHz, a thorough comparison of the best performing heuristics, using standard instances with n between 100 and 1000. The heuristic proposed in Cunha et al. (2016) was coded in C/C++, while the original codes of the algorithms by Caprara et al. (1999) and Fomeni and Letchford (2014) were provided by the authors. The tested instances were uploaded to the web page https://www.cos.ufrj.br/~jossian/instance, which appears to be currently unavailable. For $n \leq 500$ (the maximum size that the algorithm in Caprara et al. (1999) can handle), the "sequence" Billionnet and Calmels (1996), Caprara et al. (1999), Fomeni and Letchford (2014), Cunha et al. (2016) shows increasing improvement of the solution quality at the expense of increasing CPU time. For instances with $600 \leq n \leq 1000$ the best approaches (both in terms of quality and CPU time) are that by Fomeni and Letchford (2014) for densities 0.25 and 0.50, and that by Cunha et al. (2016) for densities 0.75 and 1.00.

More recently, the DP heuristic by Fomeni (2023), coded in C language and run on a single cluster node of the supercomputer of *Compute Canada* with processor at 2.26 GHz, obtained, for $50 \leq n \leq 850$, better results than those by Fomeni and Letchford (2014), but in many cases it required very high CPU times (e.g., higher than those needed by the best exact algorithms to obtain the optimal solution for instances with density 0.75 and 1.00). The modified DP heuristic by Fennich et al. (2024) exhibited sightly worse results with respect to the original DP heuristic by Fomeni (2023) for the standard instances, but provided good results for the solution of the difficult "theoretical" *hidden clique* instances (see Section 7.1).

Coming to the metaheuristics of Section 6.2, the best results appear to be those obtained by the restricted hyperplane exploration proposed by Chen and Hao (2017). The algorithm was implemented in C++ and compared, on an AMD Opteron4184 2.8 GHz, with the best heuristic and metaheuristic approaches on 220 standard instances subdivided into three groups. (All results are available at https://info.univ-angers.fr/pub/hao/QKPResults.zip .) The algorithm obtained the optimal solutions for all instances with n up to 300, the best known results for the instances with $1000 \leq n \leq 2000$, and the only known results for larger instances with n up to 6000 (for which it exhibited

an average percentage gap not greater than 1.359 between lower and upper bound). In all cases, the CPU time was very reasonable.

Overall recommendation

The algorithm by Chen and Hao (2017) is the best current choice for the heuristic solution of the QKP.

8. Conclusions

Defined for the first time in 1980, the QKP is a strongly $\mathcal{NP}$-hard problem, very difficult to solve in practice, which is still attracting considerable interest in the combinatorial optimization community. Over one third of our references appeared after 2010, almost half of which after 2020.

In spite of these efforts, there is room for future investigations. Very little is known on its approximation. While effective heuristic algorithms have been produced, no major improvement on its optimal solution has been produced after 2007. The availability of computer implementations of efficient exact algorithms, and the existence of standardized benchmark instances could stimulate new research in this direction. The recent progress in conic optimizers could suggest new computational tests on semidefinite programming relaxations, which did not provide, so far, competitive practical results. Also the impact of emerging technologies, like *quantum computing* (see, e.g., Glover, Kochenberger, and Du (2019) and Tasseff et al. (2024)) or *generative artificial intelligence*, could deserve attention for future solution approaches.

CRediT authorship contribution statement

Laura Galli: Methodology, Investigation, Formal analysis, Conceptualization. **Silvano Martello:** Methodology, Investigation, Formal analysis, Conceptualization. **Paolo Toth:** Methodology, Investigation, Formal analysis, Conceptualization.

Acknowledgment

This research was supported by the Air Force Office of Scientific Research under Grants no. FA8655-20-1-7019 and FA8655-20-1-7012.

References

Adams, W. P., & Sherali, H. D. (1986). A tight linearization and an algorithm for zero-one quadratic programming problems. *Management Science*, *32*(10), 1274–1290.

Azad, A. K., Rocha, A. M. C., & Fernandes, E. M. (2014). A simplified binary artificial fish swarm algorithm for 0–1 quadratic knapsack problems. *Journal of Computational and Applied Mathematics*, *259*, 897–904.

Balas, E. (1975). Facets of the knapsack polytope. *Mathematical Programming*, *8*, 146–164.

Bhaskara, A., Charikar, M., Chlamtac, E., Feige, U., & Vijayaraghavan, A. (2010). Detecting high log-densities: an $O(n^{1/4})$ approximation for densest k-subgraph. In *Proceedings of the forty-second ACM symposium on theory of computing* (pp. 201–210).

Billionnet, A., & Calmels, F. (1996). Linear programming for the 0–1 quadratic knapsack problem. *European Journal of Operational Research*, *92*(2), 310–325.

Billionnet, A., Faye, A., & Soutif, É. (1999). A new upper bound for the 0-1 quadratic knapsack problem. *European Journal of Operational Research*, *112*(3), 664–672.

Billionnet, A., & Soutif, É. (2004a). An exact method based on Lagrangian decomposition for the 0–1 quadratic knapsack problem. *European Journal of Operational Research*, *157*(3), 565–575.

Billionnet, A., & Soutif, É. (2004b). Using a mixed integer programming tool for solving the 0–1 quadratic knapsack problem. *INFORMS Journal on Computing*, *16*(2), 188–197.

Bretthauer, K. M., & Shetty, B. (2002). The nonlinear knapsack problem – algorithms and applications. *European Journal of Operational Research*, *138*(3), 459–472.

Cacchiani, V., Iori, M., Locatelli, A., & Martello, S. (2022a). Knapsack problems-an overview of recent advances. Part I: Single knapsack problems. *Computers & Operations Research*, *143*, Article 105692.

Cacchiani, V., Iori, M., Locatelli, A., & Martello, S. (2022b). Knapsack problems-an overview of recent advances. Part II: Multiple, multidimensional, and quadratic knapsack problems. *Computers & Operations Research*, *143*, Article 105693.

Caprara, A., Pisinger, D., & Toth, P. (1999). Exact solution of the quadratic knapsack problem. *INFORMS Journal on Computing*, *11*(2), 125–137.

Cazzaro, D., & Pisinger, D. (2022). Variable neighborhood search for large offshore wind farm layout optimization. *Computers & Operations Research*, *138*, Article 105588.

Chaillou, P., Hansen, P., & Mahieu, Y. (1989). Best network flow bounds for the quadratic knapsack problem. In B. Simeone (Ed.), *Lecture notes in mathematics, Combinatorial optimization* (1403), (pp. 225–235). Springer.

Chen, Y., & Hao, J. K. (2017). An iterated hyperplane exploration approach for the quadratic knapsack problem. *Computers & Operations Research*, *77*, 226–239.

Cunha, J. O., Simonetti, L., & Lucena, A. (2016). Lagrangian heuristics for the quadratic knapsack problem. *Computational Optimization and Applications*, *63*(1), 97–120.

Escudero, L. F., Guignard, M., & Malik, K. (1994). A Lagrangian relax-and-cut approach for the sequential ordering problem with precedence relationships. *Annals of Operations Research*, *50*, 219–237.

Fampa, M., Lubke, D., Wang, F., & Wolkowicz, H. (2020). Parametric convex quadratic relaxation of the quadratic knapsack problem. *European Journal of Operational Research*, *281*(1), 36–49.

Fennich, M. E., Fomeni, F. D., & Coelho, L. C. (2024). A novel dynamic programming heuristic for the quadratic knapsack problem. *European Journal of Operational Research*, *319*(1), 102–120.

Ferreira, C. E., Martin, A., de Souza, C. C., Weismantel, R., & Wolsey, L. A. (1996). Formulations and valid inequalities for the node capacitated graph partitioning problem. *Mathematical Programming*, *74*, 247–266.

Fischetti, M., & Pisinger, D. (2019). Mathematical optimization and algorithms for offshore wind farm design: An overview. *Business & Information Systems Engineering*, *61*, 469–485.

Fomeni, F. D. (2023). A lifted-space dynamic programming algorithm for the quadratic knapsack problem. *Discrete Applied Mathematics*, *335*, 52–68.

Fomeni, F. D., Kaparis, K., & Letchford, A. N. (2022). A cut-and-branch algorithm for the quadratic knapsack problem. *Discrete Optimization*, *44*, Article 100579.

Fomeni, F. D., & Letchford, A. N. (2014). A dynamic programming heuristic for the quadratic knapsack problem. *INFORMS Journal on Computing*, *26*(1), 173–182.

Fortet, R. (1960). L'algebre de boole et ses applications en recherche opérationnelle. *Trabajos de Estadistica*, *11*(2), 111–118.

Galli, L., Martello, S., Rey, C., & Toth, P. (2021). Polynomial-size formulations and relaxations for the quadratic multiple knapsack problem. *European Journal of Operational Research*, *291*(3), 871–882.

Gallo, G., Hammer, P. L., & Simeone, B. (1980). Quadratic knapsack problems. *Mathematical Programming Studies*, *12*, 132–149.

Garey, M. R., & Johnson, D. S. (1978). "Strong" NP-completeness results: Motivation, examples, and implications. *Journal of the ACM*, *25*, 499–508.

Gasimov, R. N. (2002). Augmented Lagrangian duality and nondifferentiable optimization methods in nonconvex programming. *Journal of Global Optimization*, *24*, 187–203.

Glover, F., Kochenberger, G., Alidaee, B., & Amini, M. (1999). Tabu search with critical event memory: an enhanced application for binary quadratic programs. In S. Voß, S. Martello, I. H. Osman, & C. Roucairol (Eds.), *Meta-heuristics: advances and trends in local search paradigms for optimization* (pp. 93–109). Springer.

Glover, F., Kochenberger, G., Alidaee, B., & Amini, M. (2002). Solving quadratic knapsack problems by reformulation and tabu search: Single constraint case. In *Combinatorial and Global Optimization* (pp. 111–121). World Scientific.

Glover, F., Kochenberger, G., & Du, Y. (2019). Quantum bridge analytics I: a tutorial on formulating and using QUBO models. *4OR. A Quarterly Journal of Operations Research*, *17*(4), 335–371.

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

Glover, F., & Woolsey, E. (1974). Converting the 0-1 polynomial programming problem to a 0-1 linear program. *Operations Research*, *22*(1), 180–182.

Hammer, P. L., & Holzman, R. (1992). Approximations of pseudo-Boolean functions; applications to game theory. *Zeitschrift für Operations Research*, *36*(1), 3–21.

Hammer, P., Padberg, M., & Peled, U. (1975). Contraint pairing in integer programming. *INFOR. Information Systems and Operational Research*, *13*, 68–81.

Hammer, P., & Rader, D. J., Jr. (1997). Efficient methods for solving quadratic 0–1 knapsack problems. *INFOR. Information Systems and Operational Research*, *35*(3), 170–182.

Helmberg, C., Rendl, F., & Weismantel, R. (2000). A semidefinite programming approach to the quadratic knapsack problem. *Journal of Combinatorial Optimization*, *4*, 197–215.

Johnson, E. L., Mehrotra, A., & Nemhauser, G. L. (1993). Min-cut clustering. *Mathematical Programming*, *62*(1–3), 133–151.

Julstrom, B. A. (2005). Greedy, genetic, and greedy genetic algorithms for the quadratic knapsack problem. In *Proceedings of the 7th annual conference on genetic and evolutionary computation* (pp. 607–614).

Karzanov, A. V. (1974). Determining the maximum flow in a network by the method of preflows. *Soviet Mathematics—Doklady*, *15*, 434–437.

Kellerer, H., Pferschy, U., & Pisinger, D. (2004). Knapsack problems. In *Springer nature book archives millennium*, Springer.

Kellerer, H., & Strusevich, V. A. (2012). The symmetric quadratic knapsack problem: Approximation and scheduling applications. *4OR. A Quarterly Journal of Operations Research*, *10*(2), 111–161.

Khot, S. (2006). Ruling out PTAS for graph min-bisection, dense k-subgraph, and bipartite clique. *SIAM Journal on Computing*, *36*(4), 1025–1071.

Lalla-Ruiz, E., Segredo, E., & Voß, S. (2019). A cooperative learning approach for the quadratic knapsack problem. In *Learning and intelligent optimization: 12th international conference, LION 12, Kalamata, Greece, June 10–15, 2018, revised selected papers 12* (pp. 31–35).

Laughhunn, D. J. (1970). Quadratic binary programming with application to capital-budgeting problems. *Operations Research*, *18*, 454–461.

Létocart, L., Nagih, A., & Plateau, G. (2012). Reoptimization in Lagrangian methods for the 0-1 quadratic knapsack problem. *Computers & Operations Research*, *39*(1), 12–18.

Michelon, P., & Veilleux, L. (1996). Lagrangean methods for the 0–1 quadratic knapsack problem. *European Journal of Operational Research*, *92*(2), 326–341.

Padberg, M. (1989). The boolean quadric polytope: some characteristics, facets and relatives. *Mathematical Programming*, *45*, 139–172.

Parizy, M., & Togawa, N. (2021). Analysis and acceleration of the quadratic knapsack problem on an Ising machine. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, *104*(11), 1526–1535.

Patvardhan, C., Bansal, S., & Srivastav, A. (2015). Solving the 0–1 quadratic knapsack problem with a competitive quantum inspired evolutionary algorithm. *Journal of Computational and Applied Mathematics*, *285*, 86–99.

Petersen, C. C. (1974). A capital budgeting heuristic algorithm using exchange operations. *AIIE Transactions*, *6*(2), 143–150.

Pferschy, U., & Schauer, J. (2016). Approximation of the quadratic knapsack problem. *INFORMS Journal on Computing*, *28*(2), 308–318.

Pisinger, D. (2006). Upper bounds and exact algorithms for *p*-dispersion problems. *Computers & Operations Research*, *33*(5), 1380–1398.

Pisinger, D. (2007). The quadratic knapsack problem - a survey. *Discrete Applied Mathematics*, *155*, 623–648.

Pisinger, W. D., Rasmussen, A. B., & Sandvik, R. (2007). Solution of large quadratic knapsack problems through aggressive reduction. *INFORMS Journal on Computing*, *19*(2), 280–290.

Pulikanti, S., & Singh, A. (2009). An artificial bee colony algorithm for the quadratic knapsack problem. In *International conference on neural information processing* (pp. 196–205).

Rader, D. J., Jr., & Woeginger, G. J. (2002). The quadratic 0–1 knapsack problem with series–parallel support. *Operations Research Letters*, *30*, 159–166.

Rasmussen, A. B., & Sandvik, R. (2003). *Kvaliteten af grænsevæ rdier for det kvadratiske knapsack problem*: *Technical report*, DIKU, University of Copenhagen.

Reinelt, G. (1991). TSPLIB—A traveling salesman problem library. *ORSA Journal on Computing*, *3*(4), 376–384.

Rhys, J. (1970). A selection problem of shared fixed costs and network flows. *Management Science*, *17*, 200–207.

Rodrigues, C. D., Quadri, D., Michelon, P., & Gueye, S. (2012). 0-1 quadratic knapsack problems: an exact approach based on a t-linearization. *SIAM Journal on Optimization*, *22*(4), 1449–1468.

Schauer, J. (2016). Asymptotic behavior of the quadratic knapsack problem. *European Journal of Operational Research*, *255*(2), 357–363.

Sherali, H. D., & Adams, W. P. (1990). A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM Journal on Discrete Mathematics*, *3*(3), 411–430.

Sipahioglu, A., & Saraç, T. (2009). The performance of the modified subgradient algorithm on solving the 0–1 quadratic knapsack problem. *Informatica*, *20*(2), 293–304.

Sturm, J. F. (2001). SeDuMi: Matlab toolbox for solving optimization problems over symmetric cones. Web: currently maintained at https://coral.ise.lehigh.edu/.

Talaván, P. M., & Yáñez, J. (2006). The generalized quadratic knapsack problem. A neuronal network approach. *Neural Networks*, *19*(4), 416–428.

Tang, T., & Toh, K.-C. (2023). A feasible method for solving an SDP relaxation of the quadratic knapsack problem. *Mathematics of Operations Research*.

Tasseff, B., Albash, T., Morrell, Z., Vuffray, M., Lokhov, A. Y., Misra, S., et al. (2024). On the emerging potential of quantum annealing hardware for combinatorial optimization. *Journal of Heuristics*, *30*(5), 325–358.

Taylor, R. (2016). Approximation of the quadratic knapsack problem. *Operations Research Letters*, *44*(4), 495–497.

Thiongane, B., Nagih, A., & Plateau, G. (2006). Lagrangean heuristics combined with reoptimization for the 0-1 bidimensional knapsack problem. *Discrete Applied Mathematics*, *154*(15), 2200–2211.

Thomadsen, T., & Larsen, J. (2007). A hub location problem with fully interconnected backbone and access networks. *Computers & Operations Research*, *34*(8), 2520–2531.

Witzgall, C. (1975). *Mathematical methods of site selection for Electronic Message Systems (EMS)*: *Technical report 76*, NASA STI/Recon.

Wolsey, L. A. (1975). Faces for a linear inequality in 0-1 variables. *Mathematical Programming*, *8*(1), 165–178.

Wu, Z. Y., Yang, Y. J., Bai, F. S., & Mammadov, M. (2011). Global optimality conditions and optimization methods for quadratic knapsack problems. *Journal of Optimization Theory and Applications*, *151*, 241–259.

Xie, X. F., & Liu, J. G. (2007). A mini-swarm for the quadratic knapsack problem. In *2007 IEEE swarm intelligence symposium* (pp. 190–197).

Yang, Z., Wang, G., & Chu, F. (2013). An effective GRASP and tabu search for the 0-1 quadratic knapsack problem. *Computers & Operations Research*, *40*(5), 1176–1185.

Zheng, X. J., Sun, X. L., Li, D., & Xu, Y. F. (2012). On reduction of duality gap in quadratic knapsack problems. *Journal of Global Optimization*, *54*, 325–339.

Zhou, J., Chen, D., Wang, Z., & Xing, W. (2013). A conic approximation method for the 0-1 quadratic knapsack problem. *Journal of Industrial & Management Optimization*, *9*(3), 531–547.