Les Cahiers du GERAD ISSN: 0711–2440

A novel dynamic programming heuristic for the quadratic knapsack problem

M. E. Fennich, F. Djeumou Fomeni, L. C. Coelho

G–2023–23

June 2023

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

A novel dynamic programming heuristic for the quadratic knapsack problem

M. Eliass Fennich [a, b]

Franklin Djeumou Fomeni [a, c]

Leandro C. Coelho [a, c, d]

[a] *GERAD, Montréal (Qc), Canada, H3T 1J4*

[b] *Department of Operations and Decision Systems, Université Laval, Québec (Qc), Canada, G1V 0A6*

[c] *Department of Analytics, Operations and Information Technology, UQÀM, Montréal (Qc), Canada, H3T 2A7*

[d] *Canada Research Chair in Integrated Logistics, Université Laval, Québec (Qc), Canada, G1V 0A6*

mohamed-eliass.fennich.1@ulaval.ca
djeumou_fomeni.franklin@uqam.ca
leandro.coelho@fsa.ulaval.ca

June 2023
Les Cahiers du GERAD
G–2023–23

Abstract : The Quadratic Knapsack Problem (QKP) is a combinatorial optimization problem that has attracted much attention over the past four decades. In this problem, one seeks to maximize a quadratic objective function of binary variables subject to a single linear knapsack constraint. This problem is interesting from both the practical and the theoretical points of view. In fact, applications of the QKP can be found in finance, logistics, and telecommunications, among others. It often appears as a subproblem to other combinatorial optimization problems. The QKP is known to be *NP*-hard in the strong sense. This paper proposes a novel idea that improves the regular value function found in the literature of dynamic programming (DP) for the QKP. We propose to consider an item with its contribution to both the items already selected in a given packing as well as an estimate of its potential contribution with respect to items yet to be considered. We also propose a propagation and a novel local search procedure to further improve the quality of the obtained results. The computational experiments show that our algorithm can dominate the performance of the existing deterministic heuristics in terms of the quality of the solutions for the standard QKP instances. Moreover, our novel algorithm significantly outperforms these heuristics on newer and more challenging QKP instances. It finds optimal or near-optimal solutions to the most challenging class of QKP instances, yielding improvements of up to 99% with respect to solutions that can be found with the existing algorithms.

Keywords : Dynamic programming, Binary Quadratic Problems, Quadratic Knapsack Problem

Acknowledgements: This work was partly supported by the Canadian Natural Sciences and Engineering Research Council under grants 2021–03307 and 2019–00094. This support is greatly appreciated. We also thank the Digital Research Alliance of Canada for providing high-performance computing facilities.

1 Introduction

Binary Quadratic Problems (BQP) find several applications in operations management (Padberg and Rijal 2012), theoretical computing (Furini and Traversi 2019, Hu and Sotirov 2021), and in several other areas. The idea behind this type of problem is to consider, in the choice of the components of the solution, not only the value of each element but also the value of their pairwise interactions. Among the most studied problems in this category is the Quadratic Knapsack Problem (QKP), for which one has to account for a knapsack capacity constraint. The QKP is defined with a set $N = \{1, 2, \ldots, n\}$ of items that can be added to a knapsack of integral capacity c. Each item i is characterized by an integer weight ω_i. A symmetric square matrix $P = (p_{ij})_{1 \leq i \leq n, 1 \leq j \leq n}$ of size n represents the profits, where p_{ii} indicates the individual value if item i is selected, and the additional profit $p_{ij} + p_{ji}$ if both items i and j are selected simultaneously. This problem can formally be expressed in the following form, using a binary variable x_i for each item $i \in N$ to indicate whether the item appears in the solution or not:

$$\begin{aligned} \max \quad & \sum_{i \in N} \sum_{j \in N} p_{ij} x_i x_j \\ \text{s.t.} \quad & \sum_{i \in N} \omega_i x_i \leq c, \\ & x_i \in \{0, 1\}, i \in N. \end{aligned} \tag{1}$$

The QKP was first introduced by Gallo et al. (1980) as a natural model for various problems in operations research, statistics, and combinatorics. They presented three examples of problems that could be modeled as QKPs. The first example was in telecommunication and concerned the location of satellites with a restricted budget. The second was concerned with hydrology for measuring rainfall in a geographical region, where the aim is to minimize redundancy when collecting data while keeping the level of variability beyond a certain threshold. Finally, the third example came from combinatorics, where one could use a QKP formulation to check whether a graph possesses a clique or not. They proposed a branch-and-bound algorithm for solving the QKP and used the idea of upper planes to derive an initial feasible solution. Their algorithm could only solve QKP instances of very small size.

The QKP is well-known as a strong NP-hard problem, as shown by Caprara et al. (1999). Several exact approaches have been developed to solve this problem; see Pisinger (2007), Cacchiani et al. (2022). Some of the most promising among these algorithms are limited either by the size of the problems that can be solved (Billionnet and Calmels 1996, Billionnet and Soutif 2004, Caprara et al. 1999) or by the time needed to solve these instances (Fomeni et al. 2022). On the other hand, several heuristic and metaheuristic methods have been developed to quickly find feasible solutions to the QKP. Some of these methods have produced good-quality solutions within a short computational time for standard QKP instances that are usually generated following a four-decade-old scheme initially proposed by Gallo et al. (1980). A recent study on the asymptotic behavior of the QKP by Schauer (2016) showed that these standard instances might sometimes be relatively easy, even for naively built heuristic algorithms. They went on to present a set of problem instances, which proved to be very challenging to many existing state-of-the-art QKP heuristic algorithms.

It is easy to remark that the QKP is a generalization of the linear knapsack problem (KP). Indeed, if all the pairwise profits (p_{ij}, with $i \neq j$) are equal to zero, then the QKP becomes a KP. One of the most efficient *exact* algorithms for the KP is the dynamic programming (DP) algorithm (Bellman 1957, Pferschy 1999), which runs in $\mathcal{O}(nc)$. It has been shown that the direct extension of this algorithm to the case of the QKP does not lead to an exact algorithm (Fomeni and Letchford 2014). Nevertheless, the idea of DP has been used to develop some efficient deterministic heuristic algorithms for the QKP. Firstly, Fomeni and Letchford (2014) presented a DP heuristic algorithm in which the value function of each stage is calculated by taking into account the profit of the current packing as well as the contribution of the new item with respect to the items that have already been selected. Recently,

Fomeni (2023) presented another deterministic heuristic algorithm for the QKP, which implements the idea of DP in the space of the lifted quadratic variables.

The aim of this paper is to propose a deterministic heuristic algorithm for the QKP using the idea of the DP algorithm. The innovative part of our proposed algorithm is that at each stage of the DP, we do not calculate only the profit contribution of an item with respect to the existing stage's partial solution; instead, we calculate the overall value function of the item. This overall value function is measured by considering the total profit of the existing stage's partial solution, the quadratic profit contribution of the current item with the items already selected, as well as an estimate of both the individual and the quadratic contribution of the best packing of items (among the ones not yet considered) that can accompany the current item in the residual space of the knapsack. We also propose an enhancement procedure in which we propagate the information of the states of the DP algorithm to subsequent states so that we can improve the quality of the stored information. Additionally, we propose a novel local search procedure to improve the resulting solution. This local search procedure is an improvement of the fill-up-and-exchange local search (Gallo et al. 1980), which has so far been used by many researchers. We test our algorithm on both standard and challenging sets of QKP instances. The computational results show that the proposed algorithm can dominate the performance of the existing deterministic algorithms in terms of the quality of the solutions found for the standard QKP instances. The results of the proposed algorithm for the Hidden clique instances, which has been very challenging for all the existing heuristics so far, are particularly impressive both for the quality of the solution and the computational time. Indeed, our algorithm can find solutions within 0.01% of optimality for these instances.

The remainder of this paper is structured as follows. Section 2 reviews the relevant literature on the QKP. In Section 3, we explain the motivation and the proposed formulation of the value function. Section 4 presents a novel DP heuristic for the QKP, which uses the proposed value function. Section 5 presents the details of the propagation procedure as well as the remove-and-fill-up local search procedure. In Section 6, we present the results of the computational experiments used to assess the efficiency of our algorithms. Finally, in Section 7 we present some concluding remarks.

2 Literature review

In this literature review, we will limit our discussion to certain key concepts from the literature which are vital to the understanding of the main contribution of this paper. We refer the reader to the books of Kellerer et al. (2004) and Martello and Toth (1990) for a complete understanding of the knapsack problem, to the survey on the QKP by Pisinger (2007) as well as to the recent study by Cacchiani et al. (2022) for an up-to-date overview of the most significant works on the QKP.

2.1 DP heuristic algorithms for the QKP

It is well-known that the DP algorithm is an exact solution approach for the linear KP. This result relies on the applicability of Bellman's principle of optimality (Bellman 1957). In 2014, Fomeni and Letchford (2014) showed that there is no analog of Bellman's principle of optimality to the case of the QKP (except for some problems that are intermediate in generality between the KP and the QKP, e.g., Rader and Woeginger (2002), Kellerer and Strusevich (2010)). They used the idea of the DP to yield a heuristic algorithm for the QKP. Their idea requires redefining the state value function of each stage $(k, r), k = 1, \ldots, n$ and $r = 0, \ldots, c$, as the profit of the *best packing found* by the heuristic that uses a selection of the first k items and whose total weight is equal to r. This profit is calculated by taking into account the individual profit of the item k and the pairwise profit of the item k with each item that is already present in the current stage solution. This algorithm runs in $\mathcal{O}(n^2c)$ time.

Another idea of DP for the QKP was recently presented by Fomeni (2023), which is an adaptation of the regular DP algorithm implemented in the space of the lifted QKP variables. The selection

or not of an item i in the definition of the QKP is represented by the decision variable x_i, whereas, the selection or not of a pair of items $\{i, j\}$ can be encoded in a variable $y_{ij} = x_i x_j$. The variables y_{ij} are usually referred to as the lifted space variables of the QKP. Thus the idea of implementing the DP in the space of the lifted QKP variable of Fomeni (2023) consists of expanding the stages of the DP algorithm to also consider the selection of pairs of items encoded in the variables y_{ij}. This allows capturing the pairwise profit of selecting some items to some extent. It also resulted in better solutions compared to the original DP algorithm of Fomeni and Letchford (2014) but at the expense of increasing the complexity of the algorithm to $\mathcal{O}(n^3 c)$. Also, in this lifted space DP algorithm, the calculation of the stage value function only considers the contribution of the current item or pair of items to the existing partial solution that has already been found.

2.2 Other heuristics and metaheuristics for the QKP

Several other deterministic heuristic algorithms can be found in the literature of the QKP; we refer the interested readers to Pisinger (2007) and Cacchiani et al. (2022) for an overview of these methods. Particularly relevant to our work is the heuristics by Gallo et al. (1980), who proposed the idea of upper planes to find an upper bound to the profit contribution of each item and then solved a linear KP, wherein the profit of each item is its corresponding upper plane. The solution to such a KP is clearly feasible for the QKP. They were also the first to introduce the idea of the local search procedure called *fill-up-and-exchange* to the context of the QKP. This procedure consists of either adding one item to the knapsack or exchanging one item in the knapsack for one item outside. Then Chaillou et al. (1983) presented a greedy heuristic, which starts by sorting all items in non-decreasing order of their loss-to-weight ratio δ_i/w_i, where δ_i represents the decrease in the profit that would be incurred if item i was removed, then placing all the items in the knapsack, and finally removing them iteratively (following the ordering initially established) until feasibility is achieved. Elsewhere, Billionnet and Calmels (1996) presented a hybrid method, in which the method of Chaillou et al. (1983) is used to form an initial solution, and then the fill-up-and-exchange procedure of Gallo et al. (1980) is used to improve that solution.

On the other hand, there have been a couple of probabilistic metaheuristic algorithms for the QKP. For example, Chen and Hao (2017) proposed a heuristic that combines variable reduction techniques and tabu search, obtaining the best results in the literature for a set of standard instances. Moreover, multiple studies have been done on metaheuristics for the QKP (Zhen et al. 2013, Dahmani et al. 2020).

Most of these heuristics and metaheuristics algorithms have been tested only on instances generated with the process proposed by Gallo et al. (1980). As demonstrated in an asymptotic study by Schauer (2016), these instances are the easiest to solve. Thus, our research also considers the newly proposed instances from Schauer (2016).

2.3 Exact solution methods for the QKP

Over the past four decades, several solution algorithms have been developed for the QKP. We refer the interested reader to the surveys by Kellerer et al. (2004) and by Pisinger (2007) as well as to the recent study by Cacchiani et al. (2022) for a good overview of some of these approaches. Among the most promising exact algorithms, we find the method developed by Caprara et al. (1999). Their idea is based on Lagrangian relaxation, subgradient optimization, and branch-and-bound. It can quickly solve instances with up to 400 items when the profit matrix is fully dense and instances with up to 200 items for low-density profit matrices. There are also some algorithms by Billionet and Soutif (2004b,a), based on integer programming linearization and Lagrangian decomposition. These algorithms produce the opposite effect of Caprara's algorithm in the sense that they perform well for low-density QKP instances and have less interesting results on high-density QKP instances. More recently, a cut-and-branch algorithm was introduced by Fomeni et al. (2022), which reported results for instances with up

to 800 items but with large computational times. It is clear that these exact algorithms are all limited by either the size of the problem instances that can be solved or the amount of time needed to solve them.

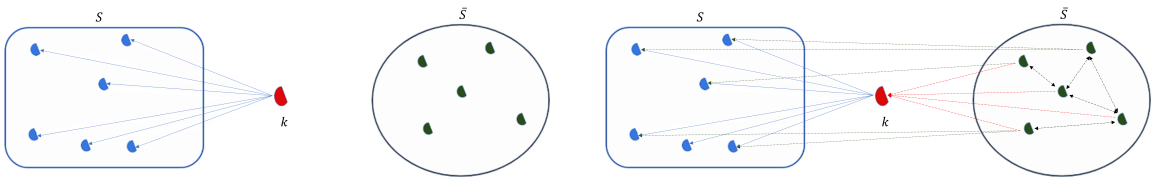
3 Novel value function calculation

One of the main contributions of this paper lies in the operation of evaluating an item k at a given stage k of the DP algorithm. This operation plays an important role in the quality of the final solution that will be obtained by the DP algorithm (Fomeni and Letchford 2014, Fomeni 2023). For instance, in the DP of Fomeni and Letchford (2014), at each stage k, by defining the set $S \subseteq \{1, \ldots, k-1\}$ to be the set of items already included in the partial solution at that stage, they considered adding an item k to the partial packing S from the stage $k-1$ based on its *attractiveness*, which is computed using the value function $g_k(S) = p_{kk} + 2\sum_{i \in S} p_{ik}$. Thus, the attractiveness of item k is based on how much it can improve the partial packing S. However, this approach clearly undermines the item's pairwise profit contribution with respect to the items yet to be considered. Indeed, even if an item is not attractive at some point due to the small quadratic gain generated with a given packing, it could become very attractive when other items are considered.

In our proposed approach, we compute the attractiveness of an item k by also taking into account an estimate of its potential future quadratic profit contribution. In other words, item k is considered along with both the partial packing S as well as the remaining items in $\overline{S} = \{k+1, k+2, \ldots, n\}$ that can best contribute to the overall profit. More precisely, we propose to compute the attractiveness of the item k by taking into account the following:

- ***i)*** the profit of the existing packing S;
- ***ii)*** the individual profit contribution of item k;
- ***iii)*** the pairwise profit contribution between item k and the items in S;
- ***iv)*** the pairwise potential profit contribution between item k and the items in $\overline{S}$ that may accompany k in the final solution;
- ***v)*** the pairwise potential profit contribution between the items in S and the items in $\overline{S}$;
- ***vi)*** the individual potential profit contribution of the items in $\overline{S}$;
- ***vii)*** the pairwise potential profit contribution between the items in $\overline{S}$.

It should be noted that the profit calculation used in previous DP heuristics (Fomeni and Letchford 2014, Fomeni 2023) consider only elements ***i)***, ***ii)***, and ***iii)***. Therefore, we consider all potential profits, linear and quadratic, obtained from the items not yet considered by S. We visually illustrate these profits in Figure 1, where Figure 1a represents how the value of k is computed in the literature, and Figure 1b represents our new approach for this calculation.



(a) The attractiveness of item k in existing DP algorithms **(b) The attractiveness of item k in our proposed algorithm**

Figure 1: Illustration of the attractiveness of item k

We denote $\omega(S)$, the total weight of the packing S. Computing the overall value of k concerning S requires identifying the best subset of items in $\overline{S}$ to be considered along k with respect to the remaining

knapsack capacity. This can be optimally obtained by solving the following QKP:

$$\begin{aligned} Q(k,S):\Pi(k,S) = \max \quad & \sum_{i\in\overline{S}}\left(p_{ii}+2p_{ik}+\sum_{j\in S}2p_{ij}\right)x_i + \sum_{\substack{i,j\in\overline{S}\\ i\neq j}} p_{ij}x_ix_j \\ \text{s.t.} \quad & \sum_{i\in\overline{S}}\omega_i x_i \leq c-\omega(S)-\omega_k, \\ & x_i\in\{0,1\}\,\forall i\in\overline{S}. \end{aligned} \tag{2}$$

Model $Q(k,S)$ determines the best subset of $\overline{S}$ maximizing the value of considering item k in the packing S. The objective function maximizes the individual profit of each potential item, their quadratic profit with item k, their quadratic profits with the elements already in S, and their quadratic profit with the other potential items in $\overline{S}$. This is subject to selecting the items whose weight fit in the remaining available capacity of the knapsack. This means that the objective function allows capturing the hidden gain from adding item k to the packing S. Thus, the profit from adding k into S is formally computed in the value function $\tilde{g}_k(S)$:

$$\tilde{g}_k(S) = g_k(S) + \Pi(k,S). \tag{3}$$

It should be noted that the computation of the future profit in equation (3) is not straightforward. Indeed, it can be seen that the calculation of $\tilde{g}_k(S)$ requires the value $\Pi(k,S)$ obtained from solving model $Q(k,S)$, which is a QKP of size $n^{'} = n-k$ and a capacity $c^{'} = c-\omega(S)-\omega_k$. In this model, the linear profit is the accumulation of the profits of items in $\overline{S}$ and their interaction with the items in $S\cup\{k\}$. A possible way to solve this problem could be to take advantage of the size reduction from the original problem and use some state-of-the-art algorithms such as Caprara et al. (1999), Fomeni et al. (2022), or even recursively use the DP approach. This will lead to an exact solution of $Q(k,S)$ and potentially the optimal value of the initial QKP if the items in $S\cup\{k\}$ are part of an optimal solution. However, because of the multiple calls to either algorithm, this will obviously lead to a large computational effort.

To overcome this challenge, we are only interested in heuristics for solving QKPs that ensure a good approximation of the value of $\Pi(k,S)$ within a reasonable time. While we lose the ability to prove an optimal solution, it remains a valid evaluation of the attractiveness of item k for S. We have conducted some preliminary experiments with several existing heuristic algorithms, and the results showed that the heuristic algorithm of Chaillou et al. (1983) provides a much better trade-off between solution quality and runtime for this purpose.

4 Novel DP Heuristic

In the previous section, we presented a novel idea to compute the attractiveness of an item k with respect to an existing packing S and the remaining items not yet in S. In this section, we show how to use this calculation in a DP algorithm. Fomeni and Letchford (2014) model the state space of a QKP solution for DP using sets $S(k,r)$ that, for each pair $k\in\{0,\ldots,n\}$ and $r\in\{0,\ldots,c\}$, encode the best-found packing for the first k items and a capacity r, and a matrix $f(k,r)$ that encodes their value. Hence, due to the weight codification, whenever an item k is considered for the set $S(k-1,r)$, then the set $S(k,r+\omega_k)$ is updated, and $f(k,r+\omega_k)$ is computed as follows:

$$f(k,r+\omega_k) = \begin{cases} \max\{f(k-1,r+\omega_k), f(k-1,r)+g_k(S(k-1,r))\} & \text{if} \quad r+\omega_k\leq c \\ f(k-1,r) & \text{otherwise.} \end{cases} \tag{4}$$

Considering our proposed approach to compute the attractiveness value of k with respect to the set $S(k-1,r)$, we replace $g_k(S(k-1,r))$ by $\tilde{g}_k(S(k-1,r))$ in equation (4), obtaining a better picture

of the potential total profit of adding item k to $S(k-1,r)$. Our method approximates the value of the solution of $Q(k,S(k-1,r))$, where the selected items are added to $S(k-1,r)\cup\{k\}$. Thus, let us define $X(k,S(k-1,r))$ as the set of items returned from solving $Q(k,S(k-1,r))$. Our proposed algorithm stores the value of the computed solution in the appropriate state, i.e., $r=\omega(S(k-1,r))+\omega(X(k,S(k-1,r)))+\omega_k$. Furthermore, the total weight of the solution always maximizes the knapsack usage, as by solving $Q(k,S(k-1,r))$, we try to fill the remaining capacity. Hence we know that $c-\min_{i\in N}\omega_i<\omega(S(k-1,r))+\omega(X(k,S(k-1,r)))+\omega_k\leq c$. We describe the novel approach in Algorithm 1. As it is seen in the proposed algorithm, the new computed value function is stored in

Algorithm 1 Novel DP algorithm for QKP

Initialize $f(0,0)=0$ and $f(k,r)=-\infty$ for $k=1,\dots,n$ and $r=0,\dots,c$
Initialize $S(k,r)=\emptyset$ for all $k=1,\dots,n$ and $r=0,\dots,c$
for $k=1,\dots,n$ **do**
  for $r=0,\dots,c$ **do**
    if $f(k-1,r)>f(k,r)$ **then**
      Set $f(k,r)=f(k-1,r)$
      Set $S(k,r)=S(k-1,r)$
    end if
    if $r+\omega_k\leq c$ **then**
      Let β_1 be the profit of $S(k-1,r)\cup\{k\}$
      if $\beta_1>f(k,r+\omega_k)$ **then**
        Set $f(k,r+\omega_k)=\beta_1$
        Set $S(k,r+\omega_k)=S(k-1,r)\cup\{k\}$
      end if
      if $r+\omega_k\leq c-\min\limits_{i\in N}\omega_i$ **then**
        Compute $X(k,r)$ by solving $Q(k,S)$ using the heuristic of Chaillou et al. (1983)
        Let β_2 be the profit of $S(k-1,r)\cup\{k\}\cup X(k,r)$
        if $\beta_2>f(k,r+\omega_k+\sum\limits_{i\in X(k,r)}\omega_i)$ **then**
          Set $f(k,r+\omega_k+\sum\limits_{i\in X(k,r)}\omega_i)=\beta_2$
          Set $S(k,r+\omega_k+\sum\limits_{i\in X(k,r)}\omega_i)=S(k-1,r)\cup\{k\}\cup X(k,r)$
        end if
      end if
    end if
  end for
end for
return $\max_{0\leq r\leq c} f(n,r)$

the corresponding state to the sum of all the weights of items in $X(k,S(k-1,r))$, plus the weight of item k, and the packing $S(k-1,r)$. In Fomeni and Letchford (2014), the authors store only the regular value of k in the corresponding state to its weight plus the weight of $S(k-1,r)$, $r=\omega(S)+\omega_k$. Our new approach generalizes the evaluation of k from equation (4). This means that our novel DP provides an upper bound to the DP algorithm of Fomeni and Letchford (2014), as it will lead to the same solution quality in the worst case. The computation of β_1, the value of k using the regular approach, is done in linear time as $\beta_1=f(k-1,r)+p_{kk}+\sum_{i\in S(k-1,r)}2p_{ik}$. On the other hand, the computation of β_2, the value of k using the novel approach, is done in quadratic time using $\beta_2=\beta_1+\sum_{i\in X(k,S(k-1,r))}p_{ii}+\sum_{j\in X(k,S(k-1,r)),j>i}2p_{ij}$. Hence, the overall time complexity of our novel DP heuristic is $\mathcal{O}(n^3c)$ compared to $\mathcal{O}(n^2c)$ for the DP heuristic of Fomeni and Letchford (2014).

5 Enhancements

Since the proposed value function is computed heuristically, it may not lead to an optimal solution. Hence, similarly to previous studies on DP, we propose two enhancements to improve the efficiency of our approach. We present a propagation heuristic in Section 5.1 that maximizes the useful information in DP. Then, we introduce a novel local search heuristic to further improve the final solution in Section 5.2.

5.1 Propagation heuristic for DP

The DP algorithm of Fomeni and Letchford (2014) is designed to explore, at each iteration, states with smaller weights and tries to improve them until the capacity constraint reaches its limit. As explained in Section 4, our proposed DP keeps this process and adds to it, also at each iteration, the computation of a solution that maximizes the knapsack total weight. This might leave some states between the two weights unexplored if the algorithm had not considered a leading combination yet. Therefore, we consider an enhancement idea where we look into the backward propagation of the improvement from states of higher weights into states of smaller weights. This operation can be performed at each iteration of the novel DP after the update in line 20 of Algorithm 1.

The idea of the propagation is that, starting from a right-side state representing a solution with a big weight, we try to deduce a solution with a lesser total weight if one exists. This amounts to removing one or more items from a given solution. The removed item is selected by choosing the least profit-to-weight ratio contribution to the solution. Then according to the resulting total weight, we update the appropriate state with the computed solution if it provides an improvement. The details of this procedure can be found in Algorithm 2.

Algorithm 2 Propagation procedure

1: **for** $v = \omega(X(k, S(k-1,r))) + \omega_k + r, \ldots, r$ **do**
2: Let q be the item with the least profit-to-weight ratio in $S(k,v)$
3: Let β_3 be the profit of $S(k,v) \setminus \{q\}$
4: **if** $\beta_3 > f(k, v-\omega_q)$ **then**
5: Set $f(k, v-\omega_q) = \beta_3$
6: Set $S(k, v-\omega_q) = S(k,v) \setminus \{q\}$
7: **end if**
8: **end for**

The computation of the item q with the smallest profit-to-ratio can be performed in $\mathcal{O}(n^2)$ time and in $\mathcal{O}(n)$ time for its resulting knapsack profit β_3 with the following formulas.

$$q = \arg\min_i \left\{ \frac{1}{\omega_i} \left(p_{ii} + 2 \sum_{j \in S(k,v)\setminus\{i\}} p_{ij} \right) \mid i \in S(k,v) \right\}$$

$$\beta_3 = f(k,v) - p_{qq} - 2 \sum_{j \in S(k,v)\setminus\{q\}} p_{qj}.$$

The motivation of this idea is to take advantage of the computed solutions from $X(k, S(k-1,r))$ and deduce solutions for different states from it. Our proposed DP fills states with large weights at the early stages of Algorithm 1. Thus, applying the propagation procedure increases the amount of explored states and further improves the ones already explored. However, if the set $X(k, S(k-1,r))$ is not considered, the propagation will not affect the result of the DP algorithm, as there is no initial solution to start the procedure. For this reason, Algorithm 2 needs to start at a weight $v = \omega(X(k, S(k-1,r))) + \omega_k + r$ because no change had happened at states of higher weights if they exist.

It is worth mentioning that neither the novel DP heuristic nor the propagation heuristic affects the DP space complexity. However, due to maximizing the number of explored states, the heuristics pushes DP toward its worst-case scenario. On the other hand, the time complexity of the propagation is $\mathcal{O}(n^2 c)$. We thus have a time complexity of $\mathcal{O}(n^3 c^2)$ for the novel DP with propagation.

5.2 Local search

Local search procedures have been used to improve the quality of deterministic heuristic algorithms for the QKP. This can be traced back to the fill-up-and-exchange procedure introduced by Gallo et al.

(1980). This local search procedure aims to process a solution through two stages. In the first stage, one tries to sequentially add the left-over items to the solution until the residual capacity can no longer accommodate any item. In the second stage, one tries to exchange each item in the solution with another item that is left out but can improve the total profit. Although this procedure has been used in many algorithms for the QKP, it is limited because it can only evaluate one item at a time, making it very dependent on the variable ordering.

In this study, we propose a new local search procedure, called *remove-and-fill-up* and which can reduce the impact of this issue, thus allowing a new level of improvement. This idea combines the fill-up and the exchange steps into a single procedure. We define S as the set of items in the initial solution and $\overline{S}$ as the set of items not in the solution. We denote z the profit of S, and $\overline{S}_\kappa$ the items in $\overline{S}$ with weights lesser or equal to κ, i.e., $\overline{S}_\kappa = \{i \in \overline{S} | \omega_i \leq \kappa\}$, for a given $\kappa \in \{0, 1, \ldots, c\}$.

In the remove-and-fill-up local search, we consider removing an item k from set S, which leaves us with a residual capacity of $\kappa = c - \sum_{i \in S \setminus \{k\}} \omega_i$. Thus we can define a new QKP, that we denote $Q_{\overline{S}_\kappa}$, for all the items in $\overline{S}$ with weights lesser than the residual capacity. In $Q_{\overline{S}_\kappa}$, we consider the interaction between those items and the solution $S \setminus \{k\}$ in the linear profit and their interaction with each other in the quadratic profit. The problem $Q_{\overline{S}_\kappa}$ is formally formulated as follows:

$$\begin{aligned} Q_{\overline{S}_\kappa} : \gamma = \max \quad & \sum_{i \in \overline{S}_\kappa} \left(p_{ii} + \sum_{j \in S \setminus \{k\}} 2p_{ij} \right) x_i + \sum_{\substack{i,j \in \overline{S}_\kappa \\ i \neq j}} p_{ij} x_i x_j \\ \text{s.t.} \quad & \sum_{i \in \overline{S}_\kappa} \omega_i x_i \leq \kappa, \\ & x_i \in \{0, 1\} \quad \forall i \in \overline{S}_\kappa. \end{aligned} \tag{5}$$

Taking advantage of the significant reduction in the size of $Q_{\overline{S}_\kappa}$, we can use multiple state-of-the-art algorithms to solve this problem exactly. However, we found that a good approximation heuristic is enough to reach the desired result; hence we again use the heuristic of Chaillou et al. (1983) for this matter. We then exchange the items in $X_{\overline{S}_\kappa}$, i.e., the solution of $Q_{\overline{S}_\kappa}$, with k in S if that improves the final solution. For that, we denote $\alpha = p_{kk} + 2\sum_{i \in S \setminus \{k\}} p_{ik}$ the profit contribution of item k to S. This approach is illustrated in Algorithm 3.

Algorithm 3 Remove-and-fill-up procedure

```
for k ∈ S do
    Let κ ⟵ c − Σ_{i∈S\{k}} ω_i
    Compute P the profit matrix of Q_{S̄_κ}
    Solve Q_{S̄_κ} and obtain its value γ
    if γ > α then
        z = z + γ − α
        S = S \ {k} ∪ X_{S̄_κ}
    end if
end for
```

It should be noted that the remove-and-fill-up procedure is dependent on the variable ordering in S if the $Q_{\overline{S}_\kappa}$ is solved exactly. In contrast, the fill-up-and-exchange procedure depends separately on the order of variables in S and $\overline{S}$.

6 Computational experiments and results

This section presents the results and analyses of the computational experiments conducted to test the efficiency of our proposed algorithms. All the discussed algorithms are coded in C++ using a single core and executed on machines equipped with 2 × Intel E5-2683 v4 Broadwell @ 2.1GHz with 32 GB

of RAM. We report results for both standard instances introduced in Gallo et al. (1980) and some new ones recently introduced by Schauer (2016), which we discuss in more detail in the following section. Furthermore, we used the cut-and-branch of Fomeni et al. (2022) to compute optimal solutions or upper bounds for our instances.

In each of the experiments reported in this section, we firstly generated the instances, then solved them using the cut-and-branch algorithm of Fomeni et al. (2022) for an exact solution or an upper bound, and subsequently, solved them with our heuristic algorithms. It should be noted that a time limit of 5 hours was set on the exact algorithm. Therefore, the gaps reported in this section are calculated as $\left(\frac{\text{Optimal Objective Value} - \text{Heuristic Objective Value}}{\text{Optimal Objective Value}}\right) \times 100$ when an optimal solution could be found within the time limit, or as $\left(\frac{\text{Best Upper Bound} - \text{Heuristic Objective Value}}{\text{Best Upper Bound}}\right) \times 100$ when an optimal solution could not be found.

6.1 Test instances description

We now present the various benchmark instances that have been used to assess the performance of our algorithm. These benchmarks include both standard instances, which are more than four decades old, and some challenging instances recently proposed by Schauer (2016).

6.1.1 Standard QKP instances.

The standard QKP instances have been around since the problem was first introduced by Gallo et al. (1980). Most of the existing QKP algorithms only report results for this category of instances, which are generated as follows. For a given value of n, each weight w_i, $i = 1, \ldots, n$, is an integer uniformly distributed between 1 and 100. The knapsack capacity c is an integer uniformly distributed between 50 and $\sum_{i=1}^{n} w_i$. Finally, for a given choice of *density parameter* $\Delta\%$, each profit term p_{ij}, $i, j = 1, \ldots, n$, is set to zero with probability $(100 - \Delta)\%$, and set to an integer, uniformly distributed between 1 and 100, with probability $\Delta\%$.

In our computational tests, we created a total of 240 instances corresponding to five random instances for each combination of $n \in \{50, 100, \ldots, 550, 600\}$ and $\Delta \in \{25, 50, 75, 100\}$. We also used some of the benchmark instances from Chen and Hao (2017), leading to a total of 340 instances for this experiment. Note that the latter instances are also generated using the same procedure.

6.1.2 Dispersion problem instances.

The dispersion problem consists of locating q facilities at n possible locations while maximizing the sum of the pairwise distances between facilities. The QKP formulation of this problem is as follows (Pisinger et al. 2007):

$$\begin{aligned} \max \quad & \sum_{i=1}^{n} \sum_{j=1}^{n} \lceil d_{ij} \rceil x_i x_j \\ \text{s.t.} \quad & \sum_{i=1}^{n} x_i = q, \\ & x \in \{0,1\}^n, \end{aligned}$$

where d_{ij} is the distance between locations i and j. Several variants of this problem, which differ by the distribution of d_{ij}, are proposed in Pisinger et al. (2007):

- *GEO* (*Geometrical problems*): the n locations are randomly located in a 100×100 square, and d_{ij} is the Euclidean distance between these locations.
- *WGEO* (*Weighted geometrical problems*): the locations are again randomly located in a square, but each location i is assigned a weight α_i in the interval $[5, \ldots, 10]$. The distance d_{ij} is then $\alpha_i \alpha_j$ times the Euclidean distance between locations i and j.

- *EXPO* (*Exponential problems*): for each pair $\{i, j\}$ of locations, d_{ij} is randomly drawn from a negative exponential distribution with mean 50.
- *RAN* (*Random problems*): for each pair $\{i, j\}$, d_{ij} is uniformly distributed in $[1, \ldots, 100]$.

For all these instances, $d_{ii} = 0$ for $i = 1, \ldots, n$, and the number of facilities q is randomly chosen in $[2, \ldots, n-2]$.

Pisinger et al. (2007) also presented a knapsack-like version of the four above-mentioned problem types. These are obtained by generating a weight w_i (randomly in $\{1, \ldots, 100\}$) for each location i, setting q to $\lfloor \frac{1}{2} \sum_{i=1}^{n} w_i \rfloor$, and changing the knapsack constraint accordingly. These instances are denoted *KP-*{*EXPO, GEO, WGEO, RAN*}. We generated ten instances for each combination of problem type and $n \in \{25, 50, 100, 200, 400\}$, for a total of 400 instances.

6.1.3 Densest subgraph instances.

The densest subgraph problem was also formulated as a QKP by Pisinger et al. (2007). Given a graph $G = (V, E)$, this problem amounts to finding a set of nodes $U \subseteq V$ of cardinality q (i.e., $w_i = 1$ for all $i = 1, \ldots, n$) for which the induced subgraph contains the maximum number of edges. Variants of the problem are obtained by changing the density of G. We experimented with the following settings:

- $DSUB25$: $d_{ij} = 1$ with probability 25%, $d_{ij} = 0$ otherwise.
- $DSUB50$: $d_{ij} = 1$ with probability 50%, $d_{ij} = 0$ otherwise.
- $DSUB75$: $d_{ij} = 1$ with probability 75%, $d_{ij} = 0$ otherwise.
- $DSUB90$: $d_{ij} = 1$ with probability 90%, $d_{ij} = 0$ otherwise.

Again, $d_{ii} = 0, i = 1, \ldots, n$ and the number of nodes q is randomly chosen in $[2, \ldots, n-2]$. We generated 10 instances for each combination of problem type and $n \in \{25, 50, 100, 200, 400\}$, for a total of 200 instances.

6.1.4 Hidden clique instances.

The final set of test instances considered is the so-called "hidden clique" (HC) instances. These instances were recently introduced in the context of the QKP by Schauer (2016), who showed that they are extremely challenging for existing QKP heuristic algorithms. In fact, this class of problems on its own makes up a completely different field of research (Alon et al. 1998, 2011). For a given n, one generates a random (so-called Erdős-Rényi) graph, in which each edge is present with probability 1/2. One then "hides" a clique in it by selecting a random set of $\lfloor \sqrt{n} \rfloor$ nodes and adding edges, where necessary, so that those nodes form a clique. The knapsack capacity is then set to $\lfloor \sqrt{n} \rfloor$. The weight of each vertex is 1, its linear profit is 0, and the quadratic profit is 1 whenever an edge is present in the graph and 0 otherwise. The optimal solution value is then almost surely $\frac{1}{2} \lfloor \sqrt{n} \rfloor \left(\lfloor \sqrt{n} \rfloor - 1 \right)$. We generated 10 HC instances for each value of $n \in \{50, 100, 150, \ldots, 1000\}$, for a total of 200 instances.

6.2 Results for standard instances

In this section, we present the results of two sets of experiments based on the standard QKP instances of Section 6.1.1. In the first set of these experiments, we generated 240 instances corresponding to five random instances for each combination of $n \in \{50, 100, \ldots, 550, 600\}$ and $\Delta \in \{25, 50, 75, 100\}$. In the second set, we used 100 benchmark standard instances from Chen and Hao (2017).

6.2.1 Results for the generic random instances.

The experiments reported in this section only include the novel DP algorithm without propagation. Indeed, we initially conducted tests with the propagation procedure and observed that it increased the

computational time beyond an acceptable threshold. This is due to the magnitude of the knapsack capacity of these instances and its contribution to the time complexity of the algorithm. Hence, we only include the fill-up-and-exchange and the remove-and-fill-up local searches in this set of experiments, besides the results of the Lifted DP algorithm of Fomeni (2023).

The results for this set of experiments are presented in Table 1, wherein we report the computation times and the optimality gap for the Lifted DP of Fomeni (2023), our novel DP, and the addition of the fill-up-and-exchange (FE) of Gallo et al. (1980), and our proposed remove-and-fill-up (RF). These results show that the novel DP algorithm implemented alone produces solutions that are usually within 0.5% of optimality. These gaps are slightly improved when the local search procedure is implemented along the novel DP algorithm. However, when looking at the performances of the novel DP algorithm with the local search procedure compared to the Lifted DP algorithm of Fomeni (2023), it appears that the Lifted DP is dominant in both the quality of the solution and the computational time. It should be noted, though, that the novel DP algorithm is implemented here without the propagation enhancement, which has proved to be a very important component of the overall proposed algorithm, as seen in the next experiments.

6.2.2 Results for the instances from Chen and Hao (2017).

In this section, we evaluate the performance of our algorithms on a second set of standard QKP instances. These are some of the benchmark instances used in Chen and Hao (2017). It should be noted that these instances originate from Billionet and Soutif (2004a,b) and are available at http://cedric.cnam.fr/~soutif/QKP/QKP.html. The results of this test are reported in Table 2. For these instances, we provide the results of our novel DP algorithm with both the propagation enhancement and the local search procedures. For comparison purposes, we also report the results of the Lifted DP algorithm of Fomeni (2023) as well as the results of the Iterative Hyperplane Exploration (**IHEA**) of Chen and Hao (2017) for these instances. Because the codes for the metaheuristic algorithm of Chen and Hao (2017) are not publicly available, we use the reported results from their tests. Their experiments were conducted on a computer with an AMD Opteron 4184 processor (2.8 GHz and 2 GB RAM) running Ubuntu 12.04 and using the C++ programming language.

The results in Table 2 show that the novel DP implemented without any enhancement is already able to find an optimal solution for all these 100 instances, thus matching the performance of the IHEA metaheuristic algorithm and outperforming the Lifted DP algorithm in terms of the quality of the solutions. However, the IHEA remains the fastest of the three algorithms. We highlight that the IHEA algorithm of Chen and Hao (2017) is a probabilistic metaheuristic algorithm, meaning it needs to be run many times (100 times for the reported results). Additionally, this algorithm has only been tested on standard instances, which according to Schauer (2016), are easy instances. Therefore, a better benchmark for our proposed algorithm is the Lifted DP algorithm of Fomeni (2023), which appears to be dominated in terms of optimality gap.

6.3 Results for the dispersion and densest subgraph instances

This section presents the results on the dispersion and densest subgraph instances, explained in Sections 6.1.2 and 6.1.3. The results are shown in Table 3, wherein we report the average performance (time and gap) of 10 instances for the novel DP algorithm when executed alone ("Novel DP"), and when combined with the propagation algorithm ("Novel DP + prop"), the propagation algorithm with the fill-up-and-exchange local search ("Novel DP + prop + FE"), as well as with the propagation algorithm with the remove-and-fill-up local search ("Novel DP + prop + RF"). Furthermore, we also present the results of Fomeni (2023), which have reported the best lower bounds so far for these QKP instances. It should be noted that for most of these instances, the cut-and-branch algorithm of Fomeni et al. (2022) could not find optimal solutions within the time limit. Therefore, most of the gaps reported in Table 3 are calculated with respect to the best upper bound instead of the optimal solution.

Table 1: Results for the set of generic standard QKP instances

size	density	Lifted DP (Fomeni 2023)		Novel DP		Novel DP + FE		Novel DP + RF	
		time (s)	gap (%)	time (s)	gap (%)	time (s)	gap (%)	time (s)	gap (%)
50	25	0.46	**0.00**	0.39	0.23	0.39	0.05	0.39	0.13
	50	0.59	**0.00**	0.42	0.25	0.42	0.07	0.42	0.18
	75	0.45	**0.00**	0.38	0.25	0.38	0.10	0.38	0.05
	100	0.39	**0.00**	0.28	0.32	0.28	0.19	0.28	0.12
100	25	7.78	**0.02**	5.52	0.13	5.52	0.12	5.52	0.10
	50	10.55	**0.01**	6.59	0.50	6.59	0.28	6.59	0.17
	75	5.68	**0.00**	3.94	0.03	3.94	0.03	3.94	0.03
	100	12.22	**0.01**	4.67	0.43	4.67	0.33	4.67	0.19
150	25	43.52	**0.00**	33.50	0.42	33.51	0.14	33.50	0.14
	50	33.03	**0.02**	20.88	0.12	20.88	0.07	20.88	0.04
	75	34.31	**0.01**	22.37	0.32	22.37	0.20	22.37	0.17
	100	48.15	**0.00**	28.53	0.18	28.53	0.13	28.53	0.08
200	25	94.52	**0.01**	64.99	0.31	64.99	0.14	64.99	0.18
	50	48.64	**0.11**	34.04	0.58	34.05	0.47	34.04	0.30
	75	62.17	**0.01**	46.36	0.42	46.36	0.33	46.36	0.10
	100	87.86	**0.00**	58.47	0.43	58.47	0.30	58.47	0.19
250	25	215.53	**0.01**	138.53	0.05	138.53	0.05	138.53	0.04
	50	425.38	**0.01**	218.34	0.22	218.34	0.16	218.34	0.10
	75	258.67	**0.03**	195.67	0.34	195.67	0.31	195.67	0.20
	100	179.26	**0.00**	107.30	0.37	107.30	0.34	107.30	0.14
300	25	606.72	**0.01**	401.21	0.03	401.21	0.01	401.21	**0.01**
	50	692.33	0.07	382.27	0.35	382.27	0.28	382.27	0.18
	75	502.88	**0.00**	460.62	0.08	460.62	0.06	460.62	0.04
	100	543.19	**0.05**	294.00	0.26	294.00	0.22	294.00	0.10
350	25	1 518.79	**0.01**	1 260.52	0.03	1 260.52	0.03	1 260.52	0.02
	50	855.79	**0.67**	611.90	0.71	611.91	0.69	611.90	**0.67**
	75	875.77	**0.00**	711.30	0.19	711.31	0.14	711.30	0.10
	100	1 418.87	**0.00**	987.31	0.15	987.31	0.13	987.31	0.06
400	25	1 779.88	**0.04**	2 156.47	0.07	2 156.47	0.07	2 156.47	**0.04**
	50	1 394.20	**0.00**	1 402.81	0.02	1 402.81	0.01	1 402.81	0.01
	75	1 069.97	**0.03**	1 070.57	0.40	1 070.57	0.26	1 070.57	0.15
	100	3 733.76	**0.00**	2 465.88	0.18	2 466.39	0.14	2 465.88	0.06
450	25	3 333.48	**0.07**	3 709.63	0.23	3 709.64	0.23	3 709.63	0.22
	50	2 601.72	**0.03**	3 338.39	0.15	3 338.40	0.12	3 338.39	0.12
	75	2 408.65	**0.01**	2 318.80	0.18	2 318.80	0.11	2 318.80	0.08
	100	6 029.76	**0.00**	4 577.66	0.10	4 577.67	0.08	4 577.66	0.05
500	25	4 947.04	**0.01**	5 685.99	0.03	5 685.99	0.02	5 685.99	**0.01**
	50	5 743.59	**0.00**	8 118.02	0.14	8 118.03	0.12	8 118.02	0.10
	75	4 182.74	**0.02**	5 629.25	0.22	5 629.27	0.19	5 629.25	0.14
	100	6 309.54	**0.05**	5 968.89	0.20	5 968.89	0.16	5 968.89	0.10
550	25	2 818.55	**0.07**	3 469.67	0.20	3 469.67	0.20	3 469.67	0.20
	50	3 705.73	**0.04**	3 959.15	0.33	3 959.16	0.22	3 959.15	0.19
	75	8 186.22	**0.01**	10 044.29	0.08	10 044.29	0.05	10 044.29	0.04
	100	8518.85	**0.05**	12 187.29	0.22	12 187.31	0.20	12 187.29	0.13
600	25	11 858.35	**0.01**	19 388.93	0.05	19 388.95	0.03	19 388.93	**0.01**
	50	4 707.52	**1.66**	6 466.16	1.81	6 466.19	1.75	6 466.16	1.71
	75	9 326.10	**2.10**	12 649.39	2.21	12 649.44	2.19	12 649.39	2.16
	100	9 318.56	**0.03**	15 602.34	0.18	15 602.35	0.15	15 602.34	0.12
Avg		2 303.29	0.11	2 839.79	0.31	2 839.80	0.24	2 839.79	0.20

Table 2: Results for a set of benchmark standard instances

		Lifted DP (Fomeni 2023)		IHEA Chen and Hao (2017)		Novel DP		Novel DP + prop		Novel DP + prop + FE		Novel DP + prop + RF	
size	density	time (s)	gap (%)	time (s)	gap (%)	time (s)	gap (%)	time (s)	gap (%)	time (s)	gap (%)	time (s)	gap (%)
100	25	1.589	0.000	0.325	0.000	11.287	0.000	1 034.81	0.000	1 034.81	0.000	1 035.67	0.000
	50	2.447	0.000	0.253	0.000	9.905	0.000	844.93	0.000	844.93	0.000	845.75	0.000
	75	2.157	0.000	0.334	0.000	10.187	0.000	1 004.38	0.000	1 004.38	0.000	1 004.50	0.000
	100	1.432	0.000	0.248	0.000	10.967	0.000	1 479.89	0.000	1 479.89	0.000	1 480.38	0.000
200	25	29.674	0.000	0.714	0.000	04.582	0.000	2 105.84	0.000	2 105.84	0.000	2 106.68	0.000
	50	26.368	0.000	0.827	0.000	148.146	0.000	9 149.65	0.000	9 149.65	0.000	9 149.86	0.000
	75	31.344	0.001	0.946	0.000	102.347	0.000	3 485.98	0.000	3 485.98	0.000	3 486.37	0.000
	100	27.588	0.000	0.722	0.000	120.597	0.000	2 135.98	0.000	2 135.98	0.000	2 136.68	0.000
300	25	125.104	0.005	1.122	0.000	684.109	0.000	2102.27	0.000	2102.27	0.000	2102.88	0.000
	50	258.679	0.000	1.156	0.000	895.265	0.000	7 632.98	0.000	7 632.98	0.000	7 633.73	0.000
Avg		50.638	0.001	0.665	0.000	199.739	0.000	3 097.671	0.000	3 097.671	0.000	3 098.250	0.000

From the results in Table 3, one can first notice that the best results of the novel DP are obtained with the propagation and the remove-and-fill-up heuristics. One can also notice that the propagation procedure highly impacts the computational time for these instances. Furthermore, the algorithm cannot report results for some of the knapsack-like versions and the weighted geometrical case of the dispersion problem instances when their size increases. This can be explained by the fact that these instances have a large knapsack capacity and that the novel value calculation and the propagation heuristic push the DP procedure towards its worst case in terms of both the computational time complexity and memory usage, considering that its computational time complexity is $\mathcal{O}(n^3c^2)$. Nevertheless, it is more important to highlight that the novel DP algorithm alone has been able to outperform the Lifted DP algorithm in terms of the quality of the solutions found, with a reasonable increase in computational times. When the novel DP algorithm is combined with its proposed enhanced procedures, it is able to find optimal solutions for more than 50% of the instances.

6.4 Results for the Hidden clique instances

This section presents the results of running our proposed algorithms on the Hidden clique instances. Similarly to the instances in the previous section, we report the average performance of 10 instances for each configuration and present the results of the Lifted DP of Fomeni (2023). It should be noted that for these instances, the optimal solution is known in advance when they are generated. Therefore an exact solution algorithm is not required in this set of experiments. It should be highlighted that Hidden clique instances have so far proved to be the most challenging QKP (Schauer 2016, Fomeni 2023, Fomeni et al. 2022).

The results of this set of experiments are reported in Table 4. One can note that the optimality gap from the Lifted DP algorithm of Fomeni (2023) varies between 7% and 19%. On the other hand, our novel DP algorithm finds gaps between 0% and 5%, which is a significant improvement from the results that could be obtained with existing algorithms. Moreover, applying the propagation enhancement to the novel DP algorithm provides solutions that are within 0.013% of optimality. The improvement is even more interesting with the remove-and-fill-up local search, with gaps that are consistently within 0.0035% of optimality. Moreover, the results of our newly proposed DP algorithm take, on average, only 1 minute and a half more than Lifted DP. It should be noted that the computational times for these instances are low because of the small value of the knapsack capacity. Note also that the propagation procedure has helped decrease the runtime of the algorithm by almost 25%.

Table 3: Results for the dispersion and densest subgraph instances

instance	size	Lifted DP (Fomeni 2023)		Novel DP		Novel DP + prop		Novel DP + prop + FE		Novel DP + prop + RF	
		time (s)	gap (%)	time (s)	gap (%)	time (s)	gap (%)	time (s)	gap (%)	time (s)	gap (%)
DSUB25	25	0.00	2.17	0.00	0.86	0.00	0.86	0.00	**0.63**	0.00	**0.63**
	50	0.03	0.33	0.06	0.00	0.10	0.00	0.10	**0.00**	0.10	**0.00**
	100	0.26	10.19	0.61	**7.57**	1.33	8.17	1.33	7.88	1.36	7.88
	200	2.24	7.49	7.97	5.85	21.06	5.80	21.06	**5.68**	21.21	**5.68**
	400	76.07	29.50	336.14	28.52	1 391.04	**28.22**	1 391.05	**28.22**	1 392.17	**28.22**
DSUB50	25	0.00	1.01	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**
	50	0.01	0.91	0.03	**0.03**	0.05	**0.03**	0.05	**0.03**	0.05	**0.03**
	100	0.18	8.01	0.31	6.32	0.97	6.24	0.97	6.15	0.99	**6.07**
	200	3.30	9.52	9.14	9.23	36.44	9.28	36.44	**9.22**	36.59	**9.22**
	400	126.93	0.18	721.67	**0.16**	1 906.02	0.17	1 906.02	0.17	1 906.36	0.17
DSUB75	25	0.00	1.55	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**
	50	0.01	0.34	0.02	**0.01**	0.04	**0.01**	0.04	**0.01**	0.04	**0.01**
	100	0.17	3.76	0.24	3.25	0.54	**3.17**	0.54	**3.17**	0.57	**3.17**
	200	4.45	2.01	6.38	**1.41**	24.37	**1.41**	24.37	**1.41**	24.60	**1.41**
	400	55.79	0.39	255.10	**0.00**	805.91	**0.00**	805.92	**0.00**	806.71	**0.00**
DSUB90	25	0.00	0.12	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**
	50	0.02	0.12	0.02	**0.00**	0.06	**0.00**	0.06	**0.00**	0.06	**0.00**
	100	0.25	2.83	0.39	2.51	1.03	2.51	1.03	**2.50**	1.05	**2.50**
	200	3.95	0.02	8.61	**0.00**	37.20	**0.00**	37.20	**0.00**	37.34	**0.00**
	400	83.26	**0.00**	211.05	**0.00**	1 000.25	**0.00**	1 000.26	**0.00**	1 001.45	**0.00**
EXPO	25	0.00	0.22	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**
	50	0.06	0.37	0.03	0.15	0.07	0.03	0.07	**0.01**	0.08	**0.01**
	100	0.87	4.36	0.38	3.78	1.18	3.73	1.18	3.73	1.20	**3.72**
	200	5.32	10.62	6.26	9.90	49.77	**9.89**	49.77	**9.89**	49.85	**9.89**
	400	155.28	0.10	245.42	**0.00**	2 806.93	**0.00**	2 806.93	**0.00**	2 806.95	**0.00**
GEO	25	0.00	0.01	0.00	0.19	0.00	0.19	0.00	**0.00**	0.00	**0.00**
	50	0.01	**0.00**	0.02	**0.00**	0.06	**0.00**	0.06	**0.00**	0.07	**0.00**
	100	0.22	0.54	0.39	**0.53**	1.78	**0.53**	1.78	**0.53**	1.81	**0.53**
	200	4.62	3.42	6.66	3.42	56.17	3.42	56.17	3.42	56.32	3.42
	400	97.61	**5.53**	178.29	**5.53**	3 099.32	**5.53**	3 099.32	**5.53**	3 099.57	**5.53**
WGEO	25	0.00	**0.00**	0.01	0.03	0.04	**0.00**	0.04	**0.00**	0.04	**0.00**
	50	0.08	**0.00**	0.13	**0.00**	1.16	**0.00**	1.16	**0.00**	1.16	**0.00**
	100	1.32	**0.00**	1.92	**0.00**	33.80	**0.00**	33.80	**0.00**	33.82	**0.00**
	200	22.15	**0.00**	31.45	**0.00**	1 145.32	**0.00**	1 150.00	**0.00**	1 145.38	**0.00**
	400	450.55	**0.00**	784.22	**0.00**	-	-	-	-	-	-
RAN	25	0.00	0.34	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**
	50	0.02	0.39	0.03	0.15	0.08	**0.12**	0.08	**0.12**	0.09	**0.12**
	100	0.21	7.42	0.30	6.91	0.88	6.87	0.88	6.82	0.90	**6.81**
	200	5.66	8.56	7.11	8.37	53.25	8.34	53.25	**8.33**	53.40	**8.33**
	400	44.57	18.31	82.49	17.07	831.74	**16.87**	831.75	**16.87**	832.77	**16.87**
KP-EXPO	25	0.01	**0.00**	0.09	**0.00**	2.66	**0.00**	2.66	**0.00**	2.67	**0.00**
	50	0.10	0.13	1.15	**0.00**	85.06	**0.00**	85.06	**0.00**	85.06	**0.00**
	100	1.04	0.14	19.18	**0.00**	2 600.13	**0.00**	2 600.13	**0.00**	2 600.14	**0.00**
	200	27.34	0.01	278.53	**0.00**	-	-	-	-	-	-
	400	357.08	**0.05**	-	-	-	-	-	-	-	-
KP-GEO	25	0.13	**0.00**	0.08	**0.00**	2.51	**0.00**	2.51	**0.00**	2.52	**0.00**
	50	2.50	**0.00**	1.20	**0.00**	113.40	**0.00**	113.40	**0.00**	113.41	**0.00**
	100	39.38	**0.00**	18.67	**0.00**	2 216.92	**0.00**	2 216.92	**0.00**	2 216.95	**0.00**
	200	135.71	**0.00**	258.89	**0.00**	-	-	-	-	-	-
	400	1 528.66	**0.00**	-	-	-	-	-	-	-	-
KP-RAN	25	0.15	**0.00**	0.08	**0.00**	2.35	**0.00**	2.35	**0.00**	2.35	**0.00**
	50	2.62	**0.00**	1.09	**0.00**	83.81	**0.00**	83.81	**0.00**	83.81	**0.00**
	100	45.26	**0.00**	20.25	**0.00**	2 163.02	**0.00**	2 163.02	**0.00**	2 163.03	**0.00**
	200	154.31	**0.00**	309.25	**0.00**	-	-	-	-	-	-
	400	1 648.22	**0.00**	-	-	-	-	-	-	-	-
KP-WGEO	25	0.14	**0.00**	0.08	**0.00**	2.56	**0.00**	2.56	**0.00**	2.56	**0.00**
	50	2.48	**0.00**	1.13	**0.00**	67.02	**0.00**	67.02	**0.00**	67.02	**0.00**
	100	40.48	**0.00**	18.92	**0.00**	2 692.84	**0.00**	2 692.84	**0.00**	2 692.89	**0.00**
	200	174.61	**0.00**	283.14	**0.00**	-	-	-	-	-	-
	400	1 876.93	**0.00**	-	-	-	-	-	-	-	-

Table 4: Results for the Hidden clique instances

size	Lifted DP (Fomeni 2023)		Novel DP		Novel DP + prop		Novel DP + prop + FE		Novel DP + prop + RF	
	time (s)	gap (%)	time (s)	gap (%)	time (s)	gap (%)	time (s)	gap (%)	time (s)	gap (%)
50	0.01	7.14	0.01	**0.00**	0.01	**0.00**	0.01	**0.00**	0.01	**0.00**
100	0.08	7.56	0.08	0.44	0.07	**0.01**	0.07	**0.01**	0.07	**0.01**
150	0.26	10.30	0.30	2.27	0.28	0.02	0.28	**0.01**	0.28	**0.01**
200	0.82	10.99	0.84	4.62	0.84	**0.02**	0.84	**0.02**	0.85	**0.02**
250	2.07	9.62	1.90	4.38	1.87	0.03	1.87	**0.02**	1.88	**0.02**
300	4.63	12.06	4.42	4.41	4.35	0.01	4.35	**0.00**	4.36	**0.00**
350	8.06	13.14	9.00	3.07	8.43	0.02	8.44	**0.01**	8.46	**0.01**
400	10.34	14.00	17.54	3.68	15.57	0.01	15.57	**0.00**	15.60	**0.00**
450	19.71	11.90	30.19	4.38	26.56	0.01	26.57	0.01	26.61	**0.00**
500	25.79	12.64	46.34	4.94	41.89	0.01	41.89	**0.00**	41.95	**0.00**
550	39.24	14.19	71.57	3.60	62.14	0.01	62.15	**0.00**	62.24	**0.00**
600	52.18	16.49	101.82	4.13	93.59	**0.00**	93.60	**0.00**	93.68	**0.00**
650	78.69	14.40	137.97	4.30	134.81	0.02	134.82	0.01	134.92	**0.00**
700	107.05	17.20	192.39	5.48	178.41	0.01	178.43	**0.00**	178.56	**0.00**
750	138.22	17.86	251.95	**0.00**	252.51	0.01	252.53	0.01	252.67	**0.00**
800	173.89	17.49	335.52	4.44	324.92	0.02	324.94	0.01	325.12	**0.00**
850	193.06	17.96	432.91	2.44	356.57	0.02	356.60	0.01	356.80	**0.00**
900	324.48	18.64	693.01	4.85	440.62	0.01	440.64	**0.00**	440.95	**0.00**
950	293.00	16.76	700.23	2.34	531.25	0.01	531.28	**0.00**	531.59	**0.00**
1000	467.35	19.05	828.22	4.39	659.68	0.01	659.72	**0.00**	660.03	**0.00**
Avg	96.9465	13.9695	192.8105	3.408	156.7185	0.013	156.73	0.006	156.8315	**0.0035**

7 Conclusion

In this paper, we have proposed a novel DP heuristic for the QKP. We have first introduced a novel way of computing the attractiveness of an item at each stage of the DP. In our approach, we generalize the profit contribution of an item with respect to the existing stage's partial solution by calculating the overall attractiveness of the item by considering the total profit of the existing stage's partial solution, the pairwise profit contribution of the current item with the items already selected, as well as an estimate of both the individual and the quadratic contribution of the best packing of items (among the ones not yet considered) that can accompany the current item in the residual space of the knapsack. Moreover, we have proposed an enhancement procedure in which we propagate the information of the states of the DP algorithm to subsequent states so that we can improve the quality of the stored information. Finally, we have also proposed a novel local search procedure to improve the resulting solution. This local search procedure is an improvement of the so-called fill-up-and-exchange.

We have tested this algorithm on both standard and challenging sets of QKP instances. The computational results show that the proposed algorithm outperforms the existing deterministic algorithms in terms of the quality of the solutions found for the standard QKP instances. The results of the proposed algorithm for the set of Hidden clique instances, which is particularly challenging, are good for both the quality of the solution and the computational time. Indeed, our algorithm can find solutions within 0.0035% of optimality for these instances, which represents more than 99% improvement over the previous best-known results.

References

N. Alon, M. Krivelevich, and B. Sudakov. Finding a large hidden clique in a random graph. Random Structures and Algorithms, 13:457–466, 1998.

N. Alon, S. Arora, R. Manokaran, D. Moshkovitz, and O. Weinstein. Inapproximabilty of densest k-subgraph from average case hardness. Technical report, Computer Science Department, Princeton University, 2011.

R Bellman. Dynamic programming. Princeton University Press, 1957.

A. Billionet and E. Soutif. An exact method based on lagrangian decomposition for the 0–1 quadratic knapsack problem. European Journal of Operational Research, 157:565–575, 2004a.

A. Billionet and E. Soutif. Using a mixed integer programming tool for solving the 0–1 quadratic knapsack problem. INFORMS Journal on Computing, 16:188–197, 2004b.

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

A. Billionnet and E. Soutif. An exact method based on lagrangian decomposition for the 0–1 quadratic knapsack problem. European Journal of Operational Research, 157(3):565–575, 2004.

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

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

P. Chaillou, P. Hansen, and Y. Mahieu. Best network flow bound for the quadratic knapsack problem. Presented at the International Workshop on Network Flow Optimization (NETFLOW), Pisa, Italy, 1983.

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

I. Dahmani, M. Hifi, T. Saadi, and Y. Labib. A swarm optimization-based search algorithm for the quadratic knapsack problem with conflict graphs. Expert Systems with Applications, 148, 2020.

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

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

F. D. Fomeni, K. Kaparis, and A. Letchford. A cut-and-branch algorithm for the quadratic knapsack problem. Discrete Optimization, 44, 2022.

F. Furini and E. Traversi. Theoretical and computational study of several linearization techniques for binary quadratic problems. Annals of Operations Research, 279(1):387–411, 2019.

G. Gallo, P.L. Hammer, and B. Simeone. Quadratic knapsack problems. In Mathematical Programming Studies, volume 12, pages 132–149, 1980.

H. Hu and R. Sotirov. The linearization problem of a binary quadratic problem and its applications. Annals of Operations Research, 307(1):229–249, 2021.

H. Kellerer and V.A. Strusevich. Fully polynomial approximation schemes for a symmetric quadratic knapsack problem and its scheduling applications. Algorithmica, 57:769–795, 2010.

H. Kellerer, U. Pferschy, and D. Pisinger. Knapsack Problems. Springer, Berlin, 2004.

S. Martello and P. Toth. Knapsack Problems: Algorithms and Computer Implementations. Wiley, Chichester, 1990.

M. W. Padberg and M. P. Rijal. Location, Scheduling, Design and Integer Programming, volume 3. Springer Science & Business Media, 2012.

U. Pferschy. Dynamic programming revisited: Improving knapsack algorithms. Computing, 63(4), 1999.

D. Pisinger. The quadratic knapsack problem—a survey. Discrete applied mathematics, 155(5):623–648, 2007.

D. Pisinger, A.B. Rasmussen, and R. Sandvik. Solution of large quadratic knapsack problems through aggressive reduction. INFORMS Journal on Computing, 19:280–290, 2007.

D. J. Rader and G. J. Woeginger. The quadratic 0–1 knapsack problem with series-parallel support. Operations Research Letters, 30:159–166, 2002.

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

Y. Zhen, Guoqing W. and C. Feng. An effective GRASP and tabu search for the 0–1 quadratic knapsack problem. Computers & Operations Research, 40(5):1176–1185, 2013.