Translating Constraints into QUBOs for the Quadratic Knapsack Problem

Tariq Bontekoe[1], Frank Phillipson[1,2], and Ward van der Schoot[1]

[1] TNO dept. Applied Cryptography & Quantum Algorithms, The Netherlands
[2] Maastricht University, School of Business and Economics, The Netherlands
{frank.phillipson,ward.vanderschoot}@tno.nl

Abstract. One of the first fields where quantum computing will likely show its use is optimisation. Many optimisation problems naturally arise in a quadratic manner, such as the quadratic knapsack problem. The current state of quantum computers requires these problems to be formulated as a quadratic unconstrained binary optimisation problem, or QUBO. Constrained quadratic binary optimisation can be translated into QUBOs by translating the constraint. However, this translation can be made in several ways, which can have a large impact on the performance when solving the QUBO. We show six different formulations for the quadratic knapsack problem and compare their performance using simulated annealing. The best performance is obtained by a formulation that uses no auxiliary variables for modelling the inequality constraint.

Keywords: quadratic knapsack problem · quadratic unconstrained binary optimisation problem · quantum computing · simulated annealing

1 Introduction

Over the past decades quantum computing research has made an impressive growth. In less than 25 years, quantum computers have evolved from laboratory experiments to public-access devices which are already being used in practice [8, 19]. While there is currently no advantage in using quantum devices over their classical counterparts, their development suggests that this may soon be otherwise. Experts believe that quantum computers will first show its use in the fields of chemistry, machine learning and optimisation, as indicated by [21]. In this work, we will focus on the latter.

Within the field of optimisation, the *Knapsack Problem* is well-known. In short, the problem entails of selecting a subset of items with the highest total gain, such that the weight or costs of that subset is below a certain limit. In mathematical terms we can define it as follows. We have a list of N objects, labelled by indices i, where the weight of each object is given by w_i, and its value given by c_i. We have a knapsack which can only carry weight W. If x_i is a binary variable denoting whether (denoted by 1) or not (denoted by 0) object i is contained in the knapsack, the total weight and gain of the knapsack are: $\mathcal{W} = \sum_{i=1}^{N} w_i x_i$ and $\mathcal{C} = \sum_{i=1}^{N} c_i x_i$, respectively. We wish to maximise within

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36030-5_8

the limitations of our knapsack, which gives the optimisation problem $\max \mathcal{C}$ such that $\mathcal{W} \leq W$.

Over the years, various adaptations to the basic knapsack problem have been proposed. In [1] and [4], they give an overview of such adaptations, such as multiple knapsacks, special constraints, fractional items, multiple dimensions and many more. This work studies the adaptation called the Quadratic Knapsack Problem (QKP) as proposed by [9]. The knapsack problem above is extended by adding a value c_{ij} for every two nodes i and j. This value equals the extra gain we obtain if both object i and j are in the knapsack. Using the notation $c_{ii} = c_i$, the total gain in case of the QKP then equals: $\mathcal{C} = \sum_{i=1}^{N} \sum_{j=1}^{N} c_{ij} x_i x_j$, which is a problem of quadratic nature. The problem is known to be (strongly) NP-hard as shown in [12]. Applications of the QKP can be found in the field of telecommunication, logistics, production and more [20].

Numerous methods have been proposed to solve the QKP since its introduction in 1980. From early on, various methods based upon the Langrangian methods were proposed, for example by Billionnet et al. [2] and Caprara et al. [5]. These methods work by decomposing the QKP in smaller instances, after which a branch-and-bound algorithm is used to find the final solution. A similar approach is given by Pisinger et al. [20], which reduces the dimension of the problem by aggresive reduction instead of decomposition, after which a branch-and-bound algorithm is used to find the final solution as well. Other methods take a different approach in which they linearise the QKP, after which the method ends with a branch-and-bound algorithm again. For example, Billionnet and Soutif [3] proposed three ways of linearising the QKP using Mixed-Integer Programs, while Rodrigues et al. [23] obtained linearisation by replacing quadratic terms by linear constraints. More recently, Schauer [24] showed that a greedy algorithm achieves asymptotically the same results on the instances usually used in QKP works. The most recent work comes from Fomeni et al. [6], which presented a cut-and-branch algorithm which combines the concepts of cutting-planes and branch-and-bound.

With the rise of quantum computing, it is interesting to wonder how quantum devices perform at solving the QKP. Especially the quadratic nature of the QKP is of particular interest for quantum computing, as currently most quantum computing methods for solving optimisation problems require problems to be of quadratic nature. While we do not expect quantum computing to efficiently give best-case solutions for all NP-hard problems, it does seem to help in finding better approximate solutions and finding them faster [21], as shown in, for example, [13, 16, 19].

To apply quantum computing to quadratic optimisation problems, we usually require them to be formulated as a Quadratic Unconstrained Binary Optimisation problem (QUBO), or Ising problem. At the time of writing, there are two popular quantum techniques that can solve QUBOs, namely the so-called Quantum Approximate Optimisation Algorithm (QAOA) on gate-based quantum devices, as proposed by [7], and Quantum Annealing, which runs on a different form of quantum device, namely quantum annealers [11]. While these methods

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36030-5_8

solve such QUBOs in a generic, problem-independent way, the formulating of such QUBOs is different for each problem. It should be noted that there can be different ways of formulating a problem as a QUBO.

For many famous optimisation problems at least one QUBO formulation is known, examples of which can be found in [10, 14]. The second overview also shows a QUBO formulation for the QKP, which is used in the thesis from [15]. On the contrary, the number of optimisation problems for which multiple QUBO formulations are present in literature is much lower. Even if QUBO formulations solve the same problem, they can be different both mathematically and performance-wise. That is why for practical purposes, it can be interesting to consider different QUBO formulations for the same problem.

In this work, we list various QUBO formulations for the QKP, either from literature or constructed by the authors. In addition, we compare their performance in practice by using the classical method of Simulated Annealing, which is similar to quantum annealing. While different mathematically equivalent formulations for general QUBOs have been studied before [22], we take a different approach. We specifically consider how the translation of the constraint into the QUBO influences its performance. To the our best knowledge, we are the first to make a comparison of the performance of different QUBO formulations for the QKP based on different ways of incorporating the constraint into the QUBO. While our work focuses on the QKP, it has implications for other optimisation problems as well. Particularly, we show that it can be very beneficial to consider other or multiple ways of translating constraints into a QUBO.

We have structured our work as follows. We start with background information on quantum and simulated annealing, a definition of a QUBO and different QUBO formulations for the QKP in Section 2. Then, we describe our approach at comparing these formulations and list the corresponding results in Section 3. In Section 4 the conclusions and directions for further research are presented.

2 Background

In this section we explain how quantum and simulated annealing works and define the general QUBO formulation. Subsequently, we explain how to formulate the Quadratic Knapsack Problem as a QUBO and present five additional QUBO formulations of the QKP. All of these formulations use a different technique to include the weight constraint into the objective function.

2.1 Quantum and Simulated Annealing

Quantum annealing is a quantum computing optimisation process specifically suitable for finding minimal solutions of objective functions with many local minima. It is designed for objective functions which have a certain number of binary decision variables and it works by mapping each of the decision variables to one or multiple qubits on the quantum annealer. Each of the basis states of

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36030-5_8

the qubits then correspond to a possible assignment of the decision variables. QUBO problems are perfectly suitable for being solved in a quantum annealer.

A quantum annealer finds minima in the following way. It starts out in an equal superposition of all possible states, after which it lets the qubits evolve under a problem-specific Hamiltonian. This Hamiltonian yields a certain energy landscape, in which minima of the energy landscape corresponds to minimal solutions of the original objective function. By evolving this system for a suitable time, the quantum state ends up near a minimum of the energy landscape, after which a measurement likely results into a local minimum of the objective function with high probability. The shape of the energy landscape directly influences the ability of the system to evolve towards lower local minima. In general, a smoother energy landscape results in lower minima and hence has a better performance.

In this work we do not work with quantum annealing, but with simulated annealing instead. This which can be seen as the classical alternative of quantum annealing , but it does not simulate quantum annealing. Instead, it simulates the annealing process found in metallurgy. Still, simulated annealing is suitable for the same family of problems as quantum annealing. Just like quantum annealing, simulated annealing is a method which walks along the energy landscape and tries to find local minima. Where quantum annealing achieves this by running a quantum system under a specific Hamiltonian, simulated annealing uses a temperature parameter. At each time step, the simulated annealing solver chooses a candidate solution which directly neighbours the current solution. When the temperature parameter is higher, the system is more likely to accept worse solutions, allowing it to explore a wide range of solutions. By gradually decreasing the temperature, this becomes less likely because of which the system is likely to settle in one of the local minima at the end of the process.

2.2 QUBO Definition

A QUBO is an optimisation problem of the form $\min y = x^t Q x$, where $x \in \{0,1\}^n$ are binary decision variables and Q is an $n \times n$ coefficient matrix. The term y is called the *objective function*. Note that this expression contains linear as well as quadratic terms, as $x_i^2 = x_i$ for decision variables $x_i \in \{0,1\}$. Many NP-hard problems can be written as a QUBO [10].

While some NP-hard problems naturally arise in this form, most, including the QKP, contain constraints as well. If these constraints are linear, they can be transformed into a quadratic *penalty function* and added to the objective function. This should be done so that that minimising the penalty function corresponds to satisfying the constraints. In this way we can write quadratic problems with linear constraints as a QUBO as well. Linear inequality contraints can be included as a quadratic penalty function as well by considering them as many equality constraints together.

For example, consider a general quadratic objective function $x^t Q x$ with linear constraint $Ax = b$, corresponding to the problem,

$$\min y = x^t Q x, \text{ subject to } Ax = b. \tag{1}$$

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36030-5_8

We bring the constraint into the objective function by adding a penalty term:

$$\min y = x^t Q x + \lambda(Ax - b)^t(Ax - b) = x^t Q x + x^t R x + d = x^t P x, \tag{2}$$

where $P = Q + R$ and the matrix R and constant d follow from the matrix multiplication. Note that the term d can be neglected, as it is constant.

The term λ is called the *weight* of the penalty function, or the *penalty value*. This parameter controls both the importance of the constraint, as well as the performance of the resulting QUBO. On the one hand, larger values of λ correspond to a higher likelihood that the constraint is met in minimal solutions. On the other hand, larger values of λ also reduce the performance of the resulting QUBO in practice. The choice of suitable weight is a study in itself and depends largely on the application. While the theoretical value to enforce the constraint can be computed for each use case, this usually results in too large weights, which influence the energy landscape too much. Each time, a deliberate consideration has to be made between performance and whether the constraints are satisfied.

2.3 Original QUBO Formulation for the QKP

To model the QKP as a QUBO, we start with the formulation from Section 1:

$$\max \mathcal{C} = \sum_{i=1}^{N}\sum_{j=1}^{N} c_{ij}x_i x_j \quad \text{s.t.} \quad \mathcal{W} = \sum_{i=1}^{N} w_i x_i \leq W. \tag{3}$$

If we assume the constraint in Eq. (1) to be an equality first, we can create the following objective and penalty function respectively, with penalty weight λ:

$$\sum_{j=1}^{N}\sum_{i=1}^{N} c_{ij}x_i x_j \quad \text{and} \quad \lambda\left(W - \sum_{i=1}^{N} w_i x_i\right)^2. \tag{4}$$

To derive the inequality constraint we have to introduce auxiliary variables to fill up the inequality to an equality:

$$H_B = \lambda\left(W - \sum_{i=1}^{N} w_i x_i - \sum_{j=1}^{M} 2^{j-1} y_j\right)^2,$$

where y_j are $M = \lceil \log_2(W+1) \rceil$ auxiliary binary variables. This yields a binary sum which can yield all numbers up to $\mathcal{W}$. This results in the following QUBO:

$$-\sum_{i=1}^{N}\sum_{j=1}^{N} c_{ij}x_i x_j + \lambda\left(W - \sum_{i=1}^{N} w_i x_i - \sum_{k=1}^{M} 2^{k-1} y_k\right)^2. \tag{5}$$

This QUBO formulation was originally posed by [10].

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36030-5_8

2.4 Alternative QUBO Formulations for the QKP

We now list five alternative QUBO formulations for the QKP. These QUBO formulations are devised by the authors and are, to the best of our knowledge, not listed in literature yet. We call the formulation of Eq. (5) **Type 1**. All these formulations can easily be shown to be solve the same QKP problem. For each type, we will indicate what the difference is between the different formulations and mention in which way they encode the constraint.

Type 2:

$$-\sum_{i=1}^{N}\sum_{j=1}^{N} c_{ij}x_i x_j + \lambda\left(\left(W+1-2^{M-1}\right)y_M + \sum_{k=1}^{M-1} 2^{k-1}y_k - \sum_{i=1}^{N} w_i x_i\right)^2. \quad (6)$$

In contrast to Type 1, we use the slack variables to encode the remaining capacity instead of encoding the total weight of the items. To achieve this, an offset of $2^{M-1}-1$ is introduced and again a set of binary auxiliary variables. In this case again $M = \lceil \log_2(W+1) \rceil$ auxiliary binary variables are needed.

Type 3:

$$-\sum_{i=1}^{N}\sum_{j=1}^{N} c_{ij}x_i x_j + \lambda\left(W - \sum_{i=1}^{N} w_i x_i - \sum_{k=1}^{M}(k-1)\,y_k\right)^2. \quad (7)$$

This QUBO formulation is similar to Type 1, but in this definition a one-hot encoding is used instead of a binary one. Here, $M = \max_{i=1}^{n} w_i$ auxiliary variables are needed, as any solution with a difference between total weight and capacity larger than M is clearly suboptimal, since one can add any item without violating the capacity constraint. Note that this QUBO formulation only works when all c_{ij} are non-negative, when any of the c_{ij} are non-positive, one should set $M = W+1$. Also note that the one-hot encoding is not strictly enforced.

Type 4:

$$-\sum_{i=1}^{N}\sum_{j=1}^{N} c_{ij}x_i x_j + \lambda\left(\sum_{k=1}^{M}(W-k+1)\,y_k - \sum_{i=1}^{N} w_i x_i\right)^2. \quad (8)$$

This formulation is similar to Type 2, however we use a one-hot encoding instead of a binary encoding. Here also, $M = \max_{i=1}^{n} w_i$ auxiliary variables are needed, using the same trick. Again, this trick requires that all c_{ij} are non-negative and if not, we require $M = W+1$ auxiliary variables. Note that again the one-hot encoding is not strictly enforced.

Type 5:

$$-\sum_{i=1}^{N}\sum_{j=1}^{N} c_{ij}x_i x_j + \lambda\left(W - W_{\text{offset}} - \sum_{i=1}^{N} w_i x_i\right)^2. \quad (9)$$

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36030-5_8

This type requires no auxiliary variables. The goal of this QUBO is to get the total weight $\sum_{i=1}^{N} w_i x_i$ as close to the capacity W as possible. To achieve this, we introduce a variable W_{offset}, which measures the offset from the capacity. The QUBO then has the goal to get the capacity as close to the fictional capacity $W_{\text{fictional}} := W - W_{\text{offset}}$ as possible. Note that while this formulation does have the advantage that it does not have any auxiliary binary variables, this comes at the cost of having an extra free parameter W_{offset}.

Type 6:

$$-\sum_{i=1}^{N}\sum_{j=1}^{N} c_{ij}x_i x_j + \lambda_1 \left(W - \sum_{i=1}^{N} w_i x_i - \sum_{k=1}^{M} (k-1)\, y_k \right)^2 + \lambda_2 \left(\sum_{k=1}^{M} y_k - 1 \right)^2 . \tag{10}$$

This type is an extension to type 3. By adding an extra penalty term with weight λ_2 we enforce the one-hot encoding in type 3. The original penalty term has weight λ_1. Again, we need $M = \max_{i=1}^{n} w_i$ auxiliary variables if all c_{ij} are non-negative and $M = W + 1$ if not.

3 Benchmark of QUBO formulations

In this section, we compare the performance of the QUBO formulations defined in the previous section. First, we discuss the QKP instances used to asses the performance, and then reason which penalty values we used across the different formulations. Finally, we discuss how we implemented these instances using Simulated Annealing and present the performance of each QUBO formulation.

3.1 Problem Instances

We use the existing QKP instances from [3]. Each QKP instance is labelled as N_D_S, which corresponds to an instance with the following parameters: the QKP considers N objects, the matrix $C = (c_{ij})$ has density D, and the seed or index of the instance is denoted by S. For each of the following combinations of N and D, there are 10 different instances, namely $D = 25\%, 50\%$ for $N = 100, 200, 300$ and $D = 50\%, 100\%$ for $N = 100, 200$. For each instance, the cost values c_{ij} and weight values w_i lie in the interval $[0, 100]$.

3.2 Penalty Values

We now discuss the process of determining penalty values for our different formulations. To theoretically enforce the constraint, we require penalty values in the order of magnitude of 10,000. However, our experience suggests that these values do not work with annealing. As these values are much larger than the QUBO entries, this would influence the energy landscape too much and likely

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36030-5_8

result in a bad performance. Instead, we believe that penalty values between 1 and 10 will be more suitable. In addition, in [17], they suggest

$$\lambda = \text{strength} \cdot N \cdot D,$$

with a strength of 0.1. In our case this results in values from 2.5 to 20. That is why we have decided to focus on penalty values between 1 and 20.

To determine the optimal penalty value for each formulation, we use simulated annealing to determine an approximation of where the optimal penalty value should lie for each of the given QUBO formulations. For these experiments, we try all integers from 1 to 10 as well as 15 and 20 as potential penalty values. For QUBO Type 6 we take λ_1 and λ_2 to be equal.

From these initial experiments, it follows that the seed S has no significant influence on the optimal penalty value, which is therefore discarded in the final selection for the best penalty value. By considering 3 or 4 different seeds per (size, density) instance, we determined the computed optimal values for each (size, density) instance and QUBO type combination. Optimality is determined by using the penalty value that results in the highest area-under-the-curve value (see section 3.4 for an explanation). It turned out that the optimal penalty value is independent of the QUBO type. However, the different instances show quite a variety of optimal penalty values, ranging from 3 to 20. The resulting penalty values can be found in Table 6.

For Type 5, we chose the offset variable as follows. As we believe that local minima will likely lie close to the constraints, we choose offset variables which are close to 0. To fully gauge the performance of this somewhat special formulation, we consider it a total of six times, each with a different offset from 0 up to 5.

3.3 Approach

We now gauge the performance of the various QUBO formulations by finding solutions with simulated annealing (SA). Specifically, we use the D-Wave implementation[3] with its default parameters. For each QUBO formulation with corresponding penalty values, we run the following steps a total of 200 000 times:

1. Pick a random initial value for the simulated annealing algorithm.
2. Run simulated annealing for our QUBO formulation from this initial value.
3. Check whether the result satisfies the constraints. If so, we add it to our results. If not, we discard it.

Note that we repeat the above steps 200 000 times to limit the considerable amount of randomness involved in applying simulated annealing. We call each of the 200 000 different runs an *SA sample*. Note that due to discarding of the results not meeting the constraints, it is likely that for each QUBO formulation there are less than 200 000 SA samples considered in the results below.

[3] https://docs.ocean.dwavesys.com/projects/neal/en/latest/reference/sampler.html

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36030-5_8

3.4 Results

In this section we discuss the results for each QUBO instance. We consider three different metrics to measure the performance of the formulations, namely, the best solution value that is found; the number of times the optimal[4] value is found; and the area-under-the-curve (AUC) value of the curve that plots the highest solution value (y-axis) found within the number of SA samples (x-axis) so far. The y-axis is normalised against the optimal solution values of the respective instances as defined in [18]. More details can be found in below.

The first two metrics give an intuition on how likely it is that a given formulation returns an optimal solution. However, it does not say anything about the quality of other solutions. The AUC value tells us how many simulating annealing steps one needs to get a good solution. In particular, the AUC value will be high if a value close to the optimum is found within a reasonable number of steps. If either of those are not the case it will be notably lower.

Best solution. The first performance metric is the best solution value found after 200 000 SA samples. We gauge this best solution value by comparing it to the optimum as a percentage of this optimum. This allows us to compare the formulations over all problem instances. We also compare our best solutions with those found by the QUBO formulation solved with SA in [15] to see how our QUBO formulations perform against the solutions there. These results are summarised in Tables 1 and 2.

Surprisingly, some of the solutions found in [15] return higher objective values than the optimal value. It is suspected that these values belong to infeasible solutions, since the optimal values found in [18] are widely assumed to be correct. Despite this, it is interesting to see that, for most instances, the solutions of all our QUBO formulations seem to be better than the SA solutions of [15].

Additionally, we find that formulations of Type 1, 2, and 4 seem to perform worse. For Type 3, 5, and 6, we see similar performances, where one does not clearly outperform the other. Although, we do observe that Type 5 performs slightly better with higher offsets, especially for the larger instances.

Optimal solution fraction. This metric measures the number of times an optimal solution is found. The results are shown in Table 3. We see that for most instances - especially those with a higher number of objects - no optimal value is found for all of the QUBO formulations. However, when an optimal value is found by any QUBO type, then usually the Type 5 formulation also finds the optimal value. In addition, the Type 5 formulations usually find the optimal value most often, specifically the formulations with higher offset value.

Area under the curve The area-under-the-curve value represents the area under the curve of which the graph which depicts the best solution found over a certain number of SA samples. In this diagram, the x-axis denotes the number of SA samples that have been taken from 500 up to and including 20 000, and the y-axis denotes the best solution over all valid solutions given by this number

[4] Optimal meaning true optimal solution for each instance (from [18]), and thus not necessarily the best value we found.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36030-5_8

Table 1. Optimal objective value per instance, accompanied by the relative best found objective value as a percentage of the optimal value for the QUBO formulations over 200,000 SA samples. The highest and lowest percentage per problem instance are displayed in bold and underline respectively.

Problem instance	Optimal value	[15]	Type 1	Type 2	Type 3	Type 4	Type 5					Type 6
							Offset 0	*Offset 1*	*Offset 2*	*Offset 3*	*Offset 4*	
100_25_1	18558	93.6%	97.8%	97.4%	97.7%	84.3%	97.8%	97.7%	97.6%	97.5%	98.3%	**98.8%**
100_25_2	56525	75.5%	85.1%	88.4%	97.3%	98.1%	98.7%	98.8%	98.8%	**99.1%**	98.7%	97.1%
100_25_3	3752	92.1%	98.6%	98.6%	98.6%	0.0%	98.6%	98.6%	**99.1%**	**99.1%**	**99.1%**	98.6%
100_25_4	50382	72.6%	93.7%	91.6%	94.9%	97.4%	97.0%	97.4%	**98.1%**	97.7%	97.4%	95.6%
100_25_5	61494	66.7%	81.5%	79.9%	97.4%	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	97.4%
100_25_6	36360	62.1%	97.2%	98.4%	97.4%	97.2%	97.3%	97.6%	98.0%	**99.3%**	98.5%	97.5%
100_25_7	14657	86.9%	99.1%	99.3%	99.5%	0.0%	99.5%	98.6%	99.5%	99.3%	**99.7%**	98.9%
100_25_8	20452	92.1%	99.3%	97.9%	99.3%	85.5%	98.0%	98.7%	97.3%	97.8%	97.4%	**100.0%**
100_25_9	35438	81.1%	94.9%	96.4%	95.2%	94.6%	**97.5%**	94.8%	95.7%	95.8%	95.7%	94.3%
100_25_10	24930	89.1%	96.6%	96.1%	94.8%	95.5%	**97.1%**	96.2%	95.7%	96.0%	96.6%	96.3%
100_50_1	83742	87.4%	97.9%	97.2%	**99.0%**	97.7%	97.8%	97.8%	98.3%	98.1%	98.5%	98.4%
100_50_2	104856	64.2%	96.2%	96.8%	**99.6%**	99.0%	99.1%	99.5%	99.1%	99.0%	99.4%	99.4%
100_50_3	34006	96.9%	99.2%	**99.9%**	**99.9%**	0.0%	99.2%	99.5%	99.8%	99.8%	99.6%	99.5%
100_50_4	105996	60.2%	91.8%	90.1%	98.6%	98.8%	98.9%	99.2%	99.1%	**99.3%**	99.2%	99.0%
100_50_5	56464	83.3%	98.4%	98.2%	**99.7%**	98.6%	98.2%	99.2%	99.0%	98.8%	99.3%	98.7%
100_50_6	16083	97.0%	**100.0%**	**100.0%**	**100.0%**	0.0%	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**
100_50_7	52819	87.4%	96.6%	96.5%	97.3%	96.2%	97.1%	97.3%	97.1%	97.1%	97.9%	**98.3%**
100_50_8	54246	93.4%	97.9%	98.0%	98.1%	98.0%	97.0%	97.5%	97.5%	97.6%	98.4%	**98.5%**
100_50_9	68974	87.5%	96.9%	97.0%	97.2%	96.9%	96.3%	96.9%	97.3%	96.9%	97.1%	**97.3%**
100_50_10	88634	61.4%	96.8%	97.1%	98.2%	98.4%	98.7%	98.7%	98.9%	98.7%	98.9%	**99.2%**
100_75_1	189137	62.1%	83.0%	82.1%	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**
100_75_2	95074	95.5%	**98.9%**	97.5%	98.4%	97.5%	98.5%	97.9%	98.6%	97.9%	97.9%	98.4%
100_75_3	62098	95.0%	98.4%	97.5%	97.5%	95.3%	97.5%	**99.9%**	97.7%	98.4%	97.7%	99.7%
100_75_4	72245	95.7%	**99.7%**	98.7%	99.2%	98.3%	98.5%	98.6%	98.7%	98.8%	99.3%	99.2%
100_75_5	27616	99.0%	99.7%	99.7%	99.7%	0.0%	99.7%	99.9%	**100.0%**	**100.0%**	**100.0%**	99.7%
100_75_6	145273	65.9%	97.5%	97.5%	99.2%	99.2%	98.9%	99.4%	99.4%	**99.7%**	98.9%	99.3%
100_75_7	110979	96.1%	98.3%	98.1%	**98.9%**	98.1%	98.7%	97.9%	98.1%	98.0%	97.9%	98.4%
100_75_8	19570	97.7%	**100.0%**	99.8%	**100.0%**	0.0%	**100.0%**	99.7%	99.2%	99.2%	98.6%	99.4%
100_75_9	104341	87.1%	98.5%	97.7%	98.6%	97.5%	97.7%	98.0%	97.7%	97.6%	**99.0%**	98.8%
100_75_10	143740	98.9%	97.7%	96.7%	99.5%	99.2%	99.4%	99.5%	99.3%	99.4%	99.1%	**99.6%**
100_100_1	81978	**100.1%**	99.1%	100.0%	100.0%	91.8%	99.1%	97.5%	100.0%	99.8%	97.6%	99.9%
100_100_2	190424	86.5%	97.3%	97.2%	97.6%	**98.1%**	97.7%	98.1%	97.9%	97.9%	97.9%	97.6%
100_100_3	225434	76.7%	87.0%	91.9%	99.5%	99.8%	99.8%	99.9%	99.8%	99.5%	**100.0%**	99.7%
100_100_4	63028	**193.6%**	97.4%	97.7%	97.6%	0.0%	97.3%	97.6%	97.2%	99.7%	97.7%	98.0%
100_100_5	230076	72.9%	85.7%	87.6%	97.9%	99.6%	**99.9%**	99.8%	99.8%	99.7%	99.9%	98.8%
100_100_6	74358	99.0%	96.7%	95.9%	**99.7%**	91.8%	96.2%	96.4%	96.9%	97.0%	96.7%	97.4%
100_100_7	10330	**103.7%**	75.6%	73.6%	88.5%	0.0%	0.0%	97.1%	97.1%	98.6%	100.0%	88.5%
100_100_8	62582	**99.7%**	96.3%	96.4%	96.6%	74.9%	96.6%	96.2%	96.1%	97.0%	96.1%	96.6%
100_100_9	232754	76.5%	87.1%	84.0%	99.7%	100.0%	**100.0%**	100.0%	**100.0%**	**100.0%**	100.0%	99.6%
100_100_10	193262	77.6%	97.6%	96.8%	98.9%	99.0%	98.9%	99.1%	99.1%	99.3%	**99.3%**	99.0%
200_25_1	204441	74.8%	89.2%	91.3%	98.5%	98.9%	**98.9%**	98.8%	98.4%	98.7%	98.6%	98.6%
200_25_2	239573	53.8%	79.5%	81.8%	99.7%	99.8%	99.8%	99.8%	**99.9%**	99.9%	99.8%	99.8%
200_25_3	245463	54.4%	74.1%	77.2%	99.7%	99.8%	**100.0%**	100.0%	99.9%	100.0%	100.0%	99.7%
200_25_4	222361	60.8%	83.5%	80.1%	**99.2%**	98.8%	98.9%	98.9%	98.9%	99.1%	98.9%	99.1%
200_25_5	187324	65.9%	93.7%	93.2%	98.0%	98.3%	98.3%	98.8%	98.5%	98.4%	98.3%	**99.1%**
200_25_6	80351	79.6%	94.8%	94.3%	95.5%	95.1%	95.4%	95.7%	96.4%	96.6%	**96.6%**	96.1%
200_25_7	59036	73.3%	**98.6%**	98.1%	97.4%	0.0%	98.2%	98.6%	98.4%	98.4%	98.2%	98.0%
200_25_8	149433	65.5%	97.0%	97.4%	**98.3%**	97.9%	97.7%	98.1%	97.8%	98.1%	97.7%	98.3%
200_25_9	49366	67.9%	97.9%	97.4%	**99.2%**	0.0%	97.6%	98.4%	98.8%	98.4%	99.0%	97.8%
200_25_10	48459	76.6%	98.1%	99.0%	98.4%	0.0%	98.4%	98.5%	98.9%	**99.3%**	98.7%	98.4%
200_50_1	372097	84.9%	90.3%	91.7%	96.9%	96.7%	**97.1%**	96.8%	97.0%	96.8%	96.8%	96.5%
200_50_2	211130	75.2%	94.3%	94.5%	94.6%	94.1%	94.7%	94.0%	93.6%	**94.9%**	94.1%	94.7%
200_50_3	227185	73.4%	95.0%	94.3%	95.0%	95.5%	94.8%	**95.7%**	95.2%	95.6%	95.0%	95.5%
200_50_4	228572	70.2%	93.4%	94.1%	94.5%	94.1%	94.0%	94.8%	94.1%	**95.1%**	94.2%	94.5%
200_50_5	479651	77.3%	71.6%	73.3%	98.6%	99.8%	99.7%	99.8%	99.7%	99.8%	**99.9%**	99.2%
200_50_6	426777	76.5%	83.3%	78.0%	98.4%	98.5%	98.6%	98.6%	98.6%	98.6%	**98.7%**	98.7%
200_50_7	220890	86.6%	93.3%	93.6%	**94.5%**	93.1%	93.5%	93.5%	94.4%	93.6%	94.1%	94.0%
200_50_8	317952	49.7%	94.7%	94.2%	96.2%	96.2%	95.9%	96.1%	95.9%	96.0%	95.8%	**96.3%**
200_50_9	104936	91.2%	96.4%	95.1%	95.5%	0.0%	94.1%	95.2%	96.3%	**97.5%**	96.2%	96.1%
200_50_10	284751	81.5%	95.0%	95.1%	95.8%	95.5%	95.0%	95.7%	95.4%	95.5%	**95.9%**	95.7%
200_75_1	442894	59.5%	93.6%	93.9%	94.9%	94.2%	95.4%	94.7%	94.7%	**95.5%**	94.3%	95.0%
200_75_2	286643	77.6%	89.5%	89.2%	91.2%	90.0%	89.8%	91.3%	89.8%	89.8%	**91.3%**	91.2%
200_75_3	61924	**101.7%**	98.1%	97.7%	96.7%	0.0%	92.9%	99.8%	99.8%	100.0%	99.6%	96.2%
200_75_4	128351	**93.2%**	82.3%	86.4%	85.9%	0.0%	85.7%	85.0%	85.7%	83.8%	86.3%	86.6%
200_75_5	137885	93.5%	92.6%	93.8%	95.3%	0.0%	94.3%	95.6%	95.3%	**96.4%**	94.6%	95.4%
200_75_6	229631	73.1%	93.6%	94.5%	94.3%	85.8%	94.6%	93.4%	93.1%	94.8%	94.6%	**95.5%**
200_75_7	269887	80.5%	90.4%	92.2%	**93.1%**	91.3%	90.2%	92.4%	92.7%	91.6%	92.3%	91.9%
200_75_8	600858	68.7%	87.8%	86.3%	97.0%	97.3%	97.2%	97.3%	97.4%	98.0%	97.2%	**98.1%**
200_75_9	516771	84.0%	91.4%	92.4%	94.8%	95.0%	95.6%	95.1%	95.0%	95.1%	**95.7%**	95.5%
200_75_10	142694	94.7%	91.5%	92.7%	91.8%	0.0%	91.3%	92.5%	**95.2%**	92.3%	91.8%	93.7%

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36030-5_8

Table 2. Optimal objective value per instance, accompanied by the relative best found objective value as a percentage of the optimal value for the QUBO formulations over 200,000 SA samples. The highest and lowest percentage per problem instance are displayed in bold and underline respectively.

Problem instance	Optimal value	[15]	Type 1	Type 2	Type 3	Type 4	Type 5					Type 6
							Offset 0	*Offset 1*	*Offset 2*	*Offset 3*	*Offset 4*	
200_100_1	937149	79.1%	69.5%	68.4%	98.5%	99.7%	99.7%	99.7%	99.7%	99.8%	**99.8%**	98.7%
200_100_2	303058	86.8%	92.8%	93.6%	94.3%	0.0%	92.9%	**94.4%**	93.6%	92.6%	92.4%	92.7%
200_100_3	29367	**103.4%**	83.5%	83.4%	93.1%	0.0%	0.0%	100.0%	100.0%	100.0%	100.0%	92.7%
200_100_4	100838	**101.2%**	96.9%	96.4%	97.4%	0.0%	96.8%	96.5%	97.2%	97.5%	97.2%	97.8%
200_100_5	786635	94.4%	81.5%	85.2%	96.1%	**96.6%**	96.5%	96.6%	96.5%	96.5%	96.6%	96.3%
200_100_6	41171	**102.9%**	86.8%	86.8%	91.7%	0.0%	0.0%	100.0%	100.0%	100.0%	100.0%	91.7%
200_100_7	701094	87.2%	91.1%	90.3%	95.6%	95.1%	95.1%	95.0%	95.2%	95.6%	95.3%	**95.8%**
200_100_8	782443	78.2%	90.4%	86.2%	97.5%	97.5%	97.1%	97.5%	**97.5%**	97.4%	97.4%	97.2%
200_100_9	628992	87.1%	93.3%	94.4%	**95.3%**	95.1%	94.6%	94.8%	94.9%	95.1%	95.1%	95.1%
200_100_10	378442	86.5%	91.6%	92.0%	93.0%	91.5%	91.3%	91.4%	91.6%	91.9%	**93.4%**	93.2%
300_25_1	29140	89.7%	98.6%	98.8%	98.3%	0.0%	91.6%	99.1%	99.6%	99.2%	**99.7%**	98.1%
300_25_2	281990	68.2%	90.0%	88.4%	89.0%	89.0%	89.3%	89.5%	89.2%	89.0%	**90.3%**	89.3%
300_25_3	231075	86.7%	88.4%	87.6%	89.1%	89.4%	89.4%	**89.8%**	87.7%	88.5%	89.4%	88.7%
300_25_4	444759	77.8%	78.1%	78.3%	93.1%	94.7%	95.2%	94.6%	**95.2%**	94.8%	94.8%	93.9%
300_25_5	14988	**101.0%**	92.0%	92.8%	93.4%	0.0%	0.0%	0.0%	100.0%	100.0%	100.0%	93.6%
300_25_6	269782	76.0%	88.5%	87.7%	**89.3%**	87.7%	86.9%	88.3%	87.6%	88.0%	88.2%	87.6%
300_25_7	485263	85.6%	73.9%	74.2%	93.8%	96.3%	96.5%	96.5%	**96.7%**	96.4%	96.4%	94.6%
300_25_8	9343	**102.4%**	88.6%	88.7%	90.6%	0.0%	0.0%	0.0%	94.4%	100.0%	100.0%	90.3%
300_25_9	250761	58.5%	88.5%	87.3%	89.1%	88.5%	88.8%	90.1%	88.9%	88.9%	89.0%	**90.4%**
300_25_10	383377	71.2%	87.2%	85.9%	90.4%	91.5%	91.2%	90.8%	91.1%	**91.7%**	91.4%	90.5%
300_50_1	513379	80.3%	89.2%	89.2%	91.1%	89.5%	89.8%	90.2%	89.7%	90.3%	89.8%	**91.2%**
300_50_2	105543	75.2%	92.3%	90.5%	92.5%	0.0%	89.9%	91.4%	92.4%	**94.4%**	92.1%	91.8%
300_50_3	875788	76.8%	79.9%	78.8%	94.7%	95.7%	95.7%	95.0%	**95.7%**	95.3%	95.6%	95.0%
300_50_4	307124	74.4%	89.7%	89.9%	91.3%	0.0%	90.9%	**92.9%**	90.2%	90.8%	91.0%	91.4%
300_50_5	727820	84.5%	90.5%	89.7%	**94.3%**	93.3%	93.4%	94.0%	93.2%	94.0%	94.1%	93.8%
300_50_6	734053	75.4%	90.5%	89.9%	94.2%	93.7%	94.3%	93.6%	**94.5%**	94.2%	94.1%	93.9%
300_50_7	43595	**101.0%**	97.8%	99.6%	99.8%	0.0%	97.9%	99.8%	99.4%	99.7%	99.2%	98.9%
300_50_8	767977	80.0%	88.8%	89.6%	93.7%	94.3%	**94.5%**	94.5%	94.4%	94.3%	94.3%	93.8%
300_50_9	761351	58.8%	90.0%	90.3%	93.9%	94.1%	94.3%	94.3%	**94.9%**	94.5%	94.6%	94.0%
300_50_10	996070	81.5%	69.5%	71.6%	97.2%	**98.0%**	97.8%	97.7%	97.7%	97.8%	97.8%	96.4%

Table 3. Number of times the optimal value was obtained out of 200 000 SA samples per problem instance and QUBO formulation. The 85 instances for which no formulation found the optimal value are left out.

Problem instance	Type 1	Type 2	Type 3	Type 4	Type 5					Type 6
					Offset 0	*Offset 1*	*Offset 2*	*Offset 3*	*Offset 4*	
100_25_5	0	0	0	2	4	4	4	6	8	0
100_25_8	0	0	0	0	0	0	0	0	0	1
100_50_6	163	172	238	0	97	98	67	38	11	235
100_75_1	0	0	133	8756	12946	12716	12938	13575	14012	244
100_75_5	0	0	0	0	0	0	74	18	16	0
100_75_8	2	0	1	0	1	0	0	0	0	0
100_100_7	0	0	0	0	0	0	0	0	29609	0
100_100_9	0	0	0	0	2	0	1	1	0	0
200_25_3	0	0	0	0	1	0	0	0	0	0
200_75_3	0	0	0	0	0	0	0	1	0	0
200_100_3	0	0	0	0	0	313	154	35	28	0
200_100_6	0	0	0	0	0	153	139	121	204	0
200_100_10	0	0	0	0	0	0	0	0	0	0
300_25_5	0	0	0	0	0	0	58	57	73	0
300_25_8	0	0	0	0	0	0	0	4739	3384	0

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36030-5_8

of SA samples. All y-values are normalised against the optimal solution value for the respective problem instance. In other words, if the value on the y-axis equals 1.0000 for some number n of SA samples, we know that at least 1 of those n SA samples was an optimal solution. Therefore, the y-values lie in the range $[0, 1]$.

Since SA is a random algorithm, its behaviour can differ each run, influencing the AUC value significantly. To mitigate this, the curve used for computation of the AUC values is obtained by averaging this curve over 10 different runs of 200 000 SA samples. An AUC value of 1.0000 thus implies that for each of the 10 runs a valid solution with optimal value was already found in the first 500 SA samples.

The AUC values for each combination of instance and QUBO Type is given in Tables 4 and 5. We see that the Type 1, 2, and 4 (almost) never have the highest AUC value. Moreover, we observe that quite often they even have an AUC value that is significantly worse than that of the other formulations. Especially the Type 4 formulation fails to find any valid solutions at all for quite some instances, corresponding to an AUC value of 0.0000.

Furthermore, we see that QUBO Type 3 and 6 almost never have the worst AUC value and often have a relatively high AUC value. For most instances the AUC values of these two formulations are close together. Therefore, this metric does not give a clear indication which formulation performs best.

When looking at the Type 5 formulation we see that it quite often has the best AUC value, especially for offsets with value 3 and 4. Specifically, for the instances with 300 items we see that this formulation is performing relatively well when compared to the other QUBO formulations.

4 Conclusion

In this paper, we defined six different QUBO formulations for the Quadratic Knapsack Problem and compared their performance when solved by simulated annealing. Specifically, they were compared on how often an optimal value was found, what the best value found was and how many SA steps are required until the formulation delivers an optimal or good solution.

First of all, we saw that SA is in most cases able to find solutions that are close to optimal. For some QUBO formulations and instances optimal solutions were found. We also found that our QUBO formulations mostly performed better than those found in [15]. This shows that all QUBO formulations are capable of solving different instances of the QKP at a high level.

However, the performance of the 6 different formulations is still very different. First of all, we see that Type 1, 2 and 4 are outperformed by Type 3, 5 and 6. Of these last three, Type 3 and 6 perform similarly, while in turn Type 5 outperforms both of them. Generally, the Type 5 formulation with higher offsets (such as 3 and 4) shows the best performance. Note that all three metrics suggest this and that we can hence quite confidently draw these conclusions.

The better performance of the Type 5 formulation could be explained by the fact that it is quite different in nature than the other formulations. The other 5

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36030-5_8

Table 4. Area-under-the-curve values per problem instance and QUBO type, averaged over 10 runs of 200 000 samples each. The highest and lowest AUC values are depicted in bold and underline respectively.

Problem instance	Type 1	Type 2	Type 3	Type 4	Type 5					Type 6
					Offset 0	*Offset 1*	*Offset 2*	*Offset 3*	*Offset 4*	
100_25_1	0.9590	0.9600	**0.9676**	0.2935	0.9566	0.9575	0.9572	0.9596	0.9640	0.9649
100_25_2	0.8208	0.8089	0.9487	0.9756	0.9761	0.9784	0.9795	0.9781	**0.9796**	0.9516
100_25_3	0.9706	0.9759	0.9788	0.0000	0.8446	0.9853	**0.9907**	0.9907	0.9907	0.9816
100_25_4	0.8850	0.8764	0.9349	0.9627	0.9648	0.9645	0.9666	**0.9672**	0.9663	0.9391
100_25_5	0.7707	0.7643	0.9532	0.9961	0.9976	0.9970	0.9974	**0.9979**	0.9975	0.9584
100_25_6	0.9585	0.9627	0.9582	0.9605	0.9619	0.9621	0.9651	**0.9695**	0.9686	0.9637
100_25_7	0.9738	0.9747	0.9825	0.0000	0.9696	0.9711	0.9753	0.9755	**0.9826**	0.9773
100_25_8	0.9611	0.9609	**0.9730**	0.3519	0.9569	0.9611	0.9615	0.9589	0.9616	0.9719
100_25_9	0.9302	0.9307	0.9193	0.9280	0.9351	0.9343	0.9393	0.9379	**0.9405**	0.9251
100_25_10	0.9412	0.9409	0.9355	0.9365	0.9413	0.9406	0.9419	0.9417	**0.9435**	0.9394
100_50_1	0.9643	0.9631	**0.9780**	0.9711	0.9706	0.9706	0.9718	0.9711	0.9709	0.9752
100_50_2	0.9174	0.9267	0.9879	0.9844	0.9838	0.9873	0.9846	0.9839	0.9847	**0.9880**
100_50_3	0.9780	0.9801	0.9853	0.0000	0.9514	0.9828	0.9869	**0.9891**	0.9890	0.9862
100_50_4	0.8702	0.8736	0.9780	0.9810	0.9810	0.9814	0.9814	**0.9819**	0.9816	0.9796
100_50_5	0.9720	0.9699	**0.9804**	0.9707	0.9387	0.9728	0.9761	0.9766	0.9794	0.9740
100_50_6	0.9998	0.9998	0.9999	0.0000	0.9943	0.9997	0.9996	0.9989	0.9968	**0.9999**
100_50_7	0.9552	0.9550	0.9610	0.9478	0.9548	0.9568	0.9571	0.9554	0.9567	**0.9635**
100_50_8	0.9602	0.9642	0.9695	0.9601	0.9576	0.9582	0.9610	0.9615	0.9630	**0.9705**
100_50_9	0.9517	0.9511	**0.9611**	0.9526	0.9546	0.9574	0.9591	0.9553	0.9581	0.9601
100_50_10	0.9484	0.9486	0.9759	0.9753	0.9756	0.9759	0.9782	0.9771	**0.9783**	0.9751
100_75_1	0.7781	0.7785	0.9996	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9998
100_75_2	0.9666	0.9654	**0.9742**	0.9665	0.9684	0.9680	0.9688	0.9688	0.9686	0.9720
100_75_3	0.9626	0.9621	0.9661	0.9244	0.9503	**0.9673**	0.9667	0.9667	0.9671	0.9665
100_75_4	0.9719	0.9711	**0.9831**	0.9689	0.9705	0.9738	0.9710	0.9722	0.9734	0.9820
100_75_5	0.9913	0.9890	0.9953	0.0000	0.9801	0.9892	**0.9995**	0.9992	0.9989	0.9954
100_75_6	0.9576	0.9581	0.9820	0.9787	0.9796	0.9806	0.9806	0.9804	0.9791	**0.9849**
100_75_7	0.9710	0.9688	**0.9769**	0.9709	0.9704	0.9704	0.9718	0.9708	0.9704	0.9753
100_75_8	0.9790	0.9800	0.9816	0.0000	0.9800	0.9744	0.9757	0.9722	0.9742	**0.9823**
100_75_9	0.9659	0.9632	0.9712	0.9669	0.9668	0.9687	0.9677	0.9680	0.9691	**0.9727**
100_75_10	0.9530	0.9418	0.9869	0.9840	0.9838	0.9829	0.9838	0.9842	0.9842	**0.9874**
100_100_1	0.9680	0.9711	**0.9818**	0.7725	0.9409	0.9636	0.9724	0.9694	0.9691	0.9794
100_100_2	0.9244	0.9250	0.9704	0.9741	0.9732	0.9753	0.9742	0.9747	**0.9754**	0.9707
100_100_3	0.8171	0.8217	0.9806	0.9861	0.9886	0.9872	0.9874	0.9861	**0.9899**	0.9815
100_100_4	0.9646	0.9618	**0.9696**	0.0000	0.9594	0.9585	0.9626	0.9663	0.9660	0.9695
100_100_5	0.8220	0.8206	0.9749	0.9902	0.9928	0.9916	0.9928	**0.9931**	0.9921	0.9761
100_100_6	0.9424	0.9341	**0.9595**	0.7809	0.9336	0.9431	0.9472	0.9531	0.9543	0.9579
100_100_7	0.7007	0.7198	0.8540	0.0000	0.0000	0.9459	0.9679	0.9859	**1.0000**	0.8550
100_100_8	0.9366	0.9422	0.9527	0.0509	0.9325	0.9352	0.9410	0.9394	0.9432	**0.9539**
100_100_9	0.7982	0.7882	0.9872	0.9977	0.9984	0.9981	**0.9988**	0.9984	0.9984	0.9880
100_100_10	0.9332	0.9362	0.9752	0.9749	0.9774	0.9780	0.9769	0.9791	**0.9799**	0.9742
200_25_1	0.8532	0.8488	0.9797	0.9804	0.9805	0.9820	0.9808	0.9812	**0.9822**	0.9790
200_25_2	0.7707	0.7745	0.9946	0.9958	**0.9964**	0.9963	0.9960	0.9962	0.9959	0.9948
200_25_3	0.7158	0.7158	0.9939	0.9962	0.9971	**0.9973**	0.9965	0.9971	0.9972	0.9942
200_25_4	0.7709	0.7643	0.9870	0.9840	0.9853	0.9849	0.9861	0.9857	0.9854	**0.9876**
200_25_5	0.9060	0.8995	0.9769	0.9767	0.9778	0.9771	0.9778	**0.9786**	0.9781	0.9781
200_25_6	0.9274	0.9254	0.9394	0.9218	0.6330	0.9088	0.9389	0.9508	**0.9561**	0.9405
200_25_7	0.9543	0.9583	0.9627	0.0000	0.9209	0.9625	0.9650	**0.9693**	0.9688	0.9638
200_25_8	0.9613	0.9643	0.9726	0.9681	0.9689	0.9722	0.9708	0.9729	0.9724	**0.9737**
200_25_9	0.9572	0.9566	0.9625	0.0000	0.9518	0.9657	0.9727	0.9732	**0.9766**	0.9662
200_25_10	0.9570	0.9564	0.9672	0.0000	0.9054	0.9636	0.9680	0.9730	**0.9748**	0.9649
200_50_1	0.8774	0.8848	0.9588	0.9599	0.9603	0.9612	**0.9629**	0.9628	0.9613	0.9579
200_50_2	0.9210	0.9215	0.9355	0.9277	0.9221	0.9251	0.9262	0.9292	0.9295	**0.9357**
200_50_3	0.9274	0.9270	0.9376	0.9374	0.9192	0.9285	0.9373	0.9374	0.9389	**0.9389**
200_50_4	0.9198	0.9212	**0.9356**	0.9301	0.9229	0.9279	0.9306	0.9328	0.9323	0.9346
200_50_5	0.6817	0.6877	0.9825	0.9957	0.9949	0.9956	0.9956	0.9957	**0.9961**	0.9845
200_50_6	0.7690	0.7576	0.9792	0.9819	0.9825	0.9828	0.9826	0.9828	**0.9832**	0.9795
200_50_7	0.9169	0.9161	**0.9317**	0.9216	0.9166	0.9248	0.9264	0.9263	0.9277	0.9308
200_50_8	0.9295	0.9286	0.9519	0.9496	0.9500	0.9495	0.9489	0.9513	0.9510	**0.9527**
200_50_9	0.9350	0.9295	0.9406	0.0000	0.8862	0.9334	0.9380	**0.9502**	0.9475	0.9437
200_50_10	0.9369	0.9365	**0.9499**	0.9458	0.9409	0.9435	0.9453	0.9447	0.9467	0.9488

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36030-5_8

Table 5. Area-under-the-curve values per problem instance and QUBO type, averaged over 10 runs of 200 000 samples each. The highest and lowest AUC values are depicted in bold and underline respectively.

Problem instance	Type 1	Type 2	Type 3	Type 4	Type 5					Type 6
					Offset 0	*Offset 1*	*Offset 2*	*Offset 3*	*Offset 4*	
200_75_1	0.9202	0.9207	**0.9366**	0.9343	0.9304	0.9335	0.9359	0.9364	0.9360	0.9364
200_75_2	0.8790	0.8806	**0.8955**	0.8858	0.8772	0.8855	0.8854	0.8887	0.8922	0.8926
200_75_3	0.9206	0.9128	0.9470	0.0000	0.2275	0.9754	0.9898	**0.9925**	0.9898	0.9412
200_75_4	0.8020	0.8188	**0.8345**	0.0000	0.8098	0.8134	0.8223	0.8219	0.8284	0.8342
200_75_5	0.8961	0.8951	0.9268	0.0000	0.6068	0.9182	0.9234	**0.9305**	0.9301	0.9232
200_75_6	0.9134	0.9197	0.9230	0.2653	0.9000	0.9137	0.9219	0.9250	0.9238	**0.9277**
200_75_7	0.8911	0.8938	**0.9097**	0.8806	0.8765	0.9009	0.9048	0.9009	0.9053	0.9080
200_75_8	0.8310	0.8200	0.9622	0.9688	0.9674	0.9675	0.9690	**0.9693**	0.9684	0.9664
200_75_9	0.8944	0.8989	0.9417	0.9431	0.9415	0.9426	0.9428	0.9450	**0.9461**	0.9432
200_75_10	0.8788	0.8876	0.9042	0.0000	0.8682	0.8824	0.8980	0.8987	0.9014	**0.9074**
200_100_1	0.6648	0.6667	0.9687	0.9939	0.9951	0.9950	0.9950	**0.9958**	0.9953	0.9741
200_100_2	0.8924	0.8952	**0.9260**	0.0000	0.8112	0.8997	0.9121	0.9085	0.9100	0.9215
200_100_3	0.8065	0.8125	0.9010	0.0000	0.0000	**1.0000**	0.9999	0.9989	0.9990	0.8961
200_100_4	0.9355	0.9345	**0.9572**	0.0000	0.8877	0.9299	0.9466	0.9524	0.9532	0.9560
200_100_5	0.7928	0.7943	0.9492	0.9599	**0.9602**	0.9599	0.9593	0.9596	0.9596	0.9534
200_100_6	0.7891	0.8300	0.9119	0.0000	0.0000	0.9179	0.9974	0.9978	**0.9995**	0.9116
200_100_7	0.8763	0.8778	0.9442	0.9443	0.9398	0.9426	**0.9458**	0.9447	0.9456	0.9441
200_100_8	0.8491	0.8335	0.9608	0.9665	0.9623	0.9669	0.9674	0.9668	**0.9682**	0.9629
200_100_9	0.9186	0.9168	0.9420	**0.9443**	0.9388	0.9417	0.9418	0.9423	0.9436	0.9420
200_100_10	0.8979	0.8997	0.9169	0.9001	0.8896	0.8967	0.9027	0.9051	0.9105	**0.9173**
300_25_1	0.9236	0.9168	0.9428	0.0000	0.5000	0.9690	0.9829	0.9823	**0.9835**	0.9329
300_25_2	0.8722	0.8741	0.8736	0.8778	0.8766	0.8802	0.8797	0.8820	**0.8831**	0.8768
300_25_3	0.8647	0.8635	0.8716	0.8708	0.8657	0.8660	0.8680	0.8742	**0.8754**	0.8719
300_25_4	0.7521	0.7482	0.9180	0.9385	0.9393	0.9401	0.9406	0.9405	**0.9412**	0.9200
300_25_5	0.8345	0.8228	0.9212	0.0000	0.0000	0.0000	0.9989	0.9996	**0.9997**	0.9197
300_25_6	0.8610	0.8575	0.8656	0.8614	0.8570	0.8616	0.8656	**0.8675**	0.8664	0.8649
300_25_7	0.6921	0.6928	0.9283	0.9570	0.9576	0.9585	0.9588	0.9589	**0.9591**	0.9350
300_25_8	0.6386	0.5824	0.8923	0.0000	0.0000	0.0000	0.9232	**1.0000**	**1.0000**	0.8843
300_25_9	0.8649	0.8614	0.8741	0.8732	0.8630	0.8702	0.8726	0.8754	**0.8794**	0.8775
300_25_10	0.8376	0.8254	0.8914	0.9036	0.8984	0.9000	0.9034	0.9065	**0.9067**	0.8934
300_50_1	0.8784	0.8812	0.8936	0.8858	0.8790	0.8845	0.8850	0.8851	0.8868	**0.8937**
300_50_2	0.8682	0.8687	0.9036	0.0000	0.8336	0.8882	0.9000	**0.9072**	0.9050	0.8933
300_50_3	0.7516	0.7485	0.9368	0.9467	0.9464	0.9473	0.9472	0.9471	**0.9482**	0.9394
300_50_4	0.8768	0.8708	**0.8985**	0.0000	0.7979	0.8632	0.8811	0.8883	0.8914	0.8951
300_50_5	0.8849	0.8779	0.9268	0.9259	0.9232	0.9264	0.9265	**0.9286**	0.9283	0.9263
300_50_6	0.8706	0.8669	0.9279	0.9315	0.9306	0.9305	**0.9337**	0.9329	0.9330	0.9295
300_50_7	0.9329	0.9325	0.9478	0.0000	0.6653	0.9840	0.9856	0.9863	**0.9865**	0.9479
300_50_8	0.8501	0.8513	0.9297	0.9328	0.9332	**0.9351**	0.9341	0.9340	0.9346	0.9320
300_50_9	0.8711	0.8649	0.9306	0.9353	0.9329	0.9348	0.9364	0.9365	**0.9377**	0.9327
300_50_10	0.6713	0.6690	0.9558	0.9736	0.9735	0.9734	0.9738	**0.9744**	0.9735	0.9589

Table 6. Penalty values for each problem instance.

100_25	**100_50**	**100_75**	**100_100**	**200_25**	**200_50**	**200_75**	**200_100**	**300_25**	**300_50**
3	4	5	10	3	8	15	20	10	15

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36030-5_8

formulations require auxiliary variables to incorporate the capacity constraint, while Type 5 accomplishes this by introducing an offset variable. In general, more variables means a larger energy landscape, which makes finding a solution harder. In the case of the QKP, this indeed seems to be the case.

However, it should be mentioned that Type 5 also brings a disadvantage. While Type 5 manages to remove the need for auxiliary variables, it does introduce a new variable called the offset variable. For each use case, research needs to be performed to gauge what a suitable value for this offset variable is. This makes it less directly implementable than the other QUBO formulations.

Even though our conclusions are based on many different results, there are some comments which can be made about these results. Firstly, we should note that our results mainly consider one family of QKP problems. It will be interesting to see whether our results could be reproduced for different families of QKP problems. Secondly, we should mention that we only considered a small set of potential penalty values and offset variables. It could very well be that these values benefit one QUBO formulation more than the other. That is why we think that repeating our assessment for a wider variety of penalty values would allow for a fairer comparison between the different formulations.

While this work focuses on the QKP, it also has implications for other optimisation and QUBO problems. First of all, we show that the QUBO formulation significantly influences the performance of the QUBO. It is hence advisory for other QUBO problems to consider alternative formulations as well. In addition, we show that the introduction of an offset variable might be more beneficial than introducing auxiliary variables, a method which is currently not so widely used. This also opens the door to research new, different techniques for generating QUBOs to see whether they perform even better in practice.

Another avenue for future research would be to test our QUBO formulations on quantum annealing devices. As quantum annealing devices currently have limited resources and hence limited applicability, it is likely that performance varies significantly over the different QUBO formulations. It might even turn out that different QUBO formulations are preferred by quantum annealing devices than the ones preferred by the simulated annealing solver. Since quantum annealing devices are promising great applicability in the (near) future, choosing suitable QUBO formulations could become an important are of research to enhance to applicability of quantum annealers.

References

1. Assi, M., Haraty, R.A.: A survey of the knapsack problem. In: 2018 International Arab Conference on Information Technology (ACIT). pp. 1–6. IEEE (2018)
2. Billionnet, A., Éric Soutif: An exact method based on lagrangian decomposition for the 0–1 quadratic knapsack problem. EJOR **157**(3), 565–575 (2004)
3. Billionnet, A., Soutif, É.: Using a mixed integer programming tool for solving the 0–1 quadratic knapsack problem. INFORMS Journal on Computing **16**(2), 188–197 (2004)

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36030-5_8

4. Cacchiani, V., Iori, M., Locatelli, A., Martello, S.: Knapsack problems-an overview of recent advances. part ii: Multiple, multidimensional, and quadratic knapsack problems. Computers & Operations Research p. 105693 (2022)
5. Caprara, A., Pisinger, D., Toth, P.: Exact solution of the quadratic knapsack problem. INFORMS Journal on Computing **11**(2), 125–137 (1999)
6. Djeumou Fomeni, F., Kaparis, K., Letchford, A.N.: A cut-and-branch algorithm for the quadratic knapsack problem. Discrete Optimization **44**, 100579 (2022)
7. Farhi, E., Goldstone, J., Gutmann, S.: A quantum approximate optimization algorithm. arXiv preprint arXiv:1411.4028 (2014)
8. Feld, S., Roch, C., Gabor, T., Seidel, C., Neukart, F., Galter, I., Mauerer, W., Linnhoff-Popien, C.: A hybrid solution method for the capacitated vehicle routing problem using a quantum annealer. Frontiers in ICT **6**, 13 (2019)
9. Gallo, G., Hammer, P.L., Simeone, B.: Quadratic knapsack problems. In: Combinatorial optimization, pp. 132–149. Springer (1980)
10. Glover, F., et al.: Quantum bridge analytics I: a tutorial on formulating and using QUBO models. Annals of Operations Research pp. 1–43 (2022)
11. Hauke, P., Katzgraber, H.G., Lechner, W., Nishimori, H., Oliver, W.D.: Perspectives of quantum annealing: Methods and implementations. Reports on Progress in Physics **83**(5), 054401 (2020)
12. Kellerer, H., Pferschy, U., Pisinger, D.: Multidimensional knapsack problems. In: Knapsack problems, pp. 235–283. Springer (2004)
13. Van der Linde, S., et al.: Hybrid classical-quantum computing in geophysical inverse problems: The case of quantum annealing for residual statics estimation. In: Sixth EAGE High Performance Computing Workshop. vol. 2022, pp. 1–5. EAGE Publications BV (2022)
14. Lucas, A.: Ising formulations of many NP problems. Frontiers in physics p. 5 (2014)
15. Mohamed, M.: Quantum Annealing: Research and Applications. Master's thesis, University of Waterloo (2021)
16. Neukart, F., Compostella, G., Seidel, C., Von Dollen, D., Yarkoni, S., Parney, B.: Traffic flow optimization using a quantum annealer. Frontiers in ICT **4**, 29 (2017)
17. Parizy, M., Togawa, N.: 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 (2021)
18. Patvardhan, C., Bansal, S., Srivastav, A.: Solving the 0–1 quadratic knapsack problem with a competitive quantum inspired evolutionary algorithm. Journal of Computational and Applied Mathematics **285**, 86–99 (2015)
19. Phillipson, F., Bhatia, H.S.: Portfolio optimisation using the D-Wave quantum annealer. In: International Conference on Computational Science. pp. 45–59. Springer (2021)
20. Pisinger, D.: The quadratic knapsack problem—a survey. Discrete applied mathematics **155**(5), 623–648 (2007)
21. Preskill, J.: Quantum computing in the NISQ era and beyond. Quantum **2**, 79 (2018)
22. Punnen, A.P., Pandey, P., Friesen, M.: Representations of quadratic combinatorial optimization problems: A case study using quadratic set covering and quadratic knapsack problems. Computers & Operations Research **112**, 104769 (2019)
23. Rodrigues, C.D., Quadri, D., Michelon, P., Gueye, S.: 0-1 quadratic knapsack problems: An exact approach based on a t-linearization. SIAM Journal on Optimization **22**(4), 1449–1468 (2012)
24. Schauer, J.: Asymptotic behavior of the quadratic knapsack problem. European Journal of Operational Research **255**(2), 357–363 (2016)

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36030-5_8