Quantum Computing Techniques for Multi-Knapsack Problems

Abhishek Awasthi¹, Francesco Bär², Joseph Doetsch³, Hans Ehm⁴, Marvin Erdmann⁵, Maximilian Hess⁴, Johannes Klepsch⁵, Peter A. Limacher², Andre Luckow⁵, Christoph Niedermeier⁶, Lilly Palackal⁴, Ruben Pfeiffer⁴, Philipp Ross⁵, Hila Safi⁶, Janik Schönmeier-Kromer², Oliver von Sicard⁶, Yannick Wenger³, Karen Wintersperger⁶ ⋈, and Sheir Yarkoni⁷

```
    BASF SE, Ludwigshafen am Rhein, Germany,
    SAP SE, Walldorf, Germany,
    Lufthansa Industry Solutions AS GmbH, Raunheim, Germany,
    Infineon Technologies AG, Neubiberg, Germany,
    BMW AG, Munich, Germany,
    Siemens AG, Munich, Germany,
    Volkswagen AG, Munich, Germany.
    This paper was developed within the Quantum Technology and Application Consortium (QUTAC).*
```

Abstract. Optimization problems are ubiquitous in various industrial settings, and multi-knapsack optimization is one recurrent task faced daily by several industries. The advent of quantum computing has opened a new paradigm for computationally intensive tasks, with promises of delivering better and faster solutions for specific classes of problems. This work presents a comprehensive study of quantum computing approaches for multi-knapsack problems, by investigating some of the most prominent and state-of-the-art quantum algorithms using different quantum software and hardware tools. The performance of the quantum approaches is compared for varying hyperparameters. We consider several gate-based quantum algorithms, such as QAOA and VQE, as well as quantum annealing, and present an exhaustive study of the solutions and the estimation of runtimes. Additionally, we analyze the impact of warm-starting QAOA to understand the reasons for the better performance of this approach. We discuss the implications of our results in view of utilizing quantum optimization for industrial applications in the future. In addition to the high demand for better quantum hardware, our results also emphasize the necessity of more and better quantum optimization algorithms, especially for multi-knapsack

Keywords – Knapsack problem, QAOA, VQE, Quantum Annealing

 $^{^{\}star} \boxtimes \mathrm{K.}$ Wintersperger: karen.wintersperger@siemens.com, QUTAC: info@qutac.de

This is an Author Accepted Manuscript version of the following chapter: Awasthi et al., Quantum Computing Techniques for Multi-Knapsack Problems, published in Intelligent Computing, edited by Kohei Arai, 2023, Springer Cham reproduced with permission of Springer Cham. The final authenticated version is available online at: https://doi.org/10.1007/978-3-031-37963-5_19

1 Introduction

The knapsack problem deals with a set of items, each having a value and a weight, which are assigned to a knapsack with a certain capacity. The task is to maximize the total value of items placed in the knapsack while respecting the capacity of the knapsack. This optimization problem is also referred to as a one-dimensional 0/1 knapsack problem and is known to be NP-hard [17]. Generalizing the problem to M knapsacks (multi-knapsack problem) and valuing items differently in each knapsack complicates the problem further [25]. However, these more delicate versions of the knapsack problem can be used as models for many real-world use cases. The scale of many industrial applications, combined with the multi-objective nature of the M-dimensional 0/1 knapsack problem, poses a significant challenge to classical solution approaches [25]. Therefore, new computational paradigms such as quantum computing are explored, with the goal of finding a speed-up over traditional approaches [6, 32]. Quantum computing allows solving certain types of complex problems significantly faster than classical devices [11, 31]. Due to its broad applicability, the knapsack problem and its variants have been well studied, and several classical approaches have been proposed to find solutions to this class of problems. Surveys on heuristic algorithms for multiple categories of knapsack problems can be found in [20] and [35].

The field of research dedicated to quantum computing solutions for knapsack problems is considerably younger and smaller. Two techniques that are built upon the quantum approximate optimization algorithm (QAOA) are introduced in [4]: one of them "warm-starts" the quantum optimization algorithm by seeding it with an initial solution using a greedy classical method, and the other uses special mixing Hamiltonians to improve the exploration of the solution space. Results indicate that both approaches outperform similarly shallow classical heuristics in one-dimensional knapsack problem instances. Reformulations of this version of the knapsack problem are evaluated in [28]. For the multi-knapsack problem, two quantum-inspired evolutionary algorithms (QIEA) are presented in [23], solving knapsack problem instances with more than 10,000 items. Besides these promising findings of the potential of quantum computing for the knapsack use case, there are studies indicating challenges and even an inability of quantum optimization techniques to outperform classical methods, at least in the era of near-term noisy intermediate-scale quantum (NISQ) devices. In [26], it is shown that a D-Wave 2000Q quantum annealer could not provide optimal solutions for many small-scale knapsack problems due to the limitations of the hardware. A general theory on the limitations of optimization algorithms on NISQ devices is presented in [10]. Since most of the optimization algorithms for NP-hard problems are heuristic in nature, frameworks such as QUARK [9] are essential to obtain a thorough comparison of the performances of classical and quantum algorithms on various hardware backends.

This work presents a comprehensive benchmark of different quantum algorithms for a use case relevant to many industries. We study an M-dimensional 0/1 knapsack problem and carry out an extensive comparison of

results obtained via QAOA, warm-start QAOA, VQE, Quantum Annealing as well as the iterative heuristic solver with Simulated Annealing. Much work about benchmarking quantum algorithms for optimization problems focuses mainly on a single type of algorithm (circuit model or adiabatic model). However, to identify the most promising quantum algorithm, holistic benchmarking of several quantum approaches needs to be carried out. We study the suitability of quantum algorithms for the multi-knapsack problem with the help of key performance indicators (KPIs) relevant to industrial applications. Since we are interested in the performance of the algorithms, we benchmark the gate-based algorithms on noiseless simulators (while quantum annealing has been carried out on quantum hardware). Nonetheless, in contrast to most of the studies published in this field, we present a practical estimation of the runtimes on quantum hardware for the QAOA and VQE algorithms. Additionally, we compare the warm-start QAOA described in [7] as well as another new variant of warm-start QAOA with the standard algorithms mentioned above, which is another novel contribution of this work to the best of our knowledge.

The remainder of the paper is structured as follows: Section 2 provides the business motivation and possible use cases. The modeling of the multi-knapsack problem is described in Section 3, while Section 4 gives a brief overview of the quantum algorithms being used, *i.e.*, QAOA, VQE and Quantum Annealing. The information about the problem instances and the definitions of the KPIs used in this work are provided in Section 5. We then present our benchmarking results for the studied algorithms, along with the runtimes using a quantum annealer and an estimation of the runtimes of the gate-based algorithms on a real quantum device. We conclude our work with Section 6 and provide an analysis of the results obtained as well as an outlook. The complete python code for our implementations is available at https://github.com/QutacQuantum/Knapsack.

2 Business Motivation

The Quantum Technology and Application Consortium (QUTAC) and its members focus on industry use cases for quantum computing applications [27]. The general knapsack problem has many applications in decision-making processes along the entire value chain, such as optimizing portfolios [18], business operations and supply chains. A prominent example of a multi-knapsack problem in many industries, including the automotive, semiconductor and chemical industry, is the optimization of complex supply chains, since products are usually processed in a global manufacturing network rather than in a single factory. Hence, the need for planning and communication between the manufacturing sites emerges to realize an optimal global manufacturing process. Optimization techniques are crucial to addressing common supply chain challenges and increasing the responsiveness to disruptions, e.g., optimizing freight and warehouse capacities, labor planning and carbon emissions, thereby also enhancing the overall sustainability [13]. For the semiconductor industry alone, where supply chain management is particularly complex and dynamic,

optimization plays a major role in the context of integrated supply chain planning, daily. One major goal is to solve the demand-capacity matching, which corresponds to maximizing the available product units relative to the units promised to the customer. This task consists of several computationally hard optimization problems, since more than a million order confirmations and reorder confirmations have to be regularly calculated. Therefore, high-quality solutions need to be delivered in a reasonable time. In the following, we will restrict ourselves to a simplified model in which we assume that the demand, capacity, and costs are given. Note that this is a strong assumption, as determining the inputs of the model is a computationally hard task itself. The simplified demand-capacity match can be encoded as a multi-knapsack problem.

3 Modeling

In this section, we present a formal description of the multi-knapsack optimization problem, along with the mathematical formulation of the QUBO model. Given N items and M knapsacks, the objective is to assign as many valuable items as possible to each of the knapsacks while not exceeding the capacity of any knapsack. The problem can be stated as follows. Let $j \in \{0,1,\ldots,N-1\}$, then $w_j \in \mathbb{N}_0$ denotes the weight of item j and $v_{i,j} \in \mathbb{N}_0$ denotes the value of item j in knapsack $i \in \{0,1,\ldots,M-1\}$. The capacity of a knapsack i is denoted by $c_i \in \mathbb{N}_0$. We define a decision variable $x_{i,j}$, such that $x_{i,j} = 1$ if and only if item j is assigned to knapsack i, and 0 otherwise. We can now formulate the corresponding QUBO model, including the problem constraints and objective term.

- Any item j can be assigned to at most one knapsack. It is possible that an item is not assigned to any knapsack.

$$H_{\text{single}} = \sum_{j=0}^{N-1} \left(\sum_{i=0}^{M-1} x_{i,j} \right) \cdot \left(\sum_{i=0}^{M-1} x_{i,j} - 1 \right) . \tag{1}$$

– Ensure that no knapsack's capacity is exceeded. This is achieved by introducing slack bits $y_{i,b}$ with binary expansion, based on the work of Lucas [22]. The filling of knapsack $i \in \{0, \ldots, M-1\}$, which can be smaller than its capacity, is thereby expressed as $c_i - \sum_{b=0}^{\lfloor \log_2 c_i \rfloor} 2^b \cdot y_{i,b}$. Using this formulation, no fillings larger than the knapsack capacity can be encoded, also when c_i is not a power of two. If the sum corresponds to a number larger than c_i , the actual filling becomes negative and thus leads to an even larger penalty in the Hamiltonian. The capacity term of the Hamiltonian reads

$$H_{\text{capacity}} = \sum_{i=0}^{M-1} \left[\left(\sum_{j=0}^{N-1} w_j \cdot x_{i,j} \right) + \left(\sum_{b=0}^{\lfloor \log_2 c_i \rfloor} 2^b \cdot y_{i,b} \right) - c_i \right]^2. \tag{2}$$

- The objective term is formulated such that our original maximization objective function is converted to a minimization problem, as shown below.

$$H_{\text{obj}} = -\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} v_{i,j} \cdot x_{i,j} . \tag{3}$$

With the above QUBO terms, we can now formulate the complete QUBO for multi-knapsack optimization problems as H, where

$$H = A \cdot H_{\text{single}} + B \cdot H_{\text{capacity}} + C \cdot H_{\text{obj}}.$$
 (4)

The coefficients A, B > 0 are the penalty weights, C > 0 is the objective weight. The minimization of H results is an optimal solution to the multi-knapsack problem. We have to choose A, B, C such that $\frac{A}{C} > \max_{i,j}(v_{i,j})$ and $\frac{B}{C} > \max_{i,j}(v_{i,j})$. This ensures that any value gained by breaking a constraint is offset by an even larger penalty. Without loss of generality, we therefore choose C = 1 and set $A = B = 2 \cdot \max_{i,j}(v_{i,j})$.

4 Quantum Optimization Algorithms

This work provides a comparison for the multi-knapsack problem between several quantum algorithms. Since most of these algorithms are well-known and explained in detail in the literature, we provide only a short overview below.

4.1 Quantum Approximate Optimization Algorithm (QAOA)

The QAOA is a popular variational algorithm inspired by the adiabatic theorem, devised to produce approximate solutions for combinatorial optimization problems [8]. For brevity, we outline the basics of the algorithm below; for an in-depth explanation of the algorithm we refer the reader to [38]. The QAOA algorithm optimizes any Hamiltonian \mathcal{C} by constructing a predefined parameterized quantum circuit and optimizing the circuit parameters by utilizing classical iterative algorithms.

Concretely, the QAOA algorithm requires a quantum circuit to sample a quantum state $|\psi_{\gamma,\beta}\rangle = \left(\prod_{l=p}^1 U(\mathcal{B},\beta_l) \cdot U(\mathcal{C},\gamma_l)\right) \cdot |s\rangle$, where $|s\rangle$ is the uniform superposition state, $U(\mathcal{B},\beta_l) = e^{-i\beta_l\mathcal{B}}$ is the unitary operator resulting from a mixing Hamiltonian and $U(\mathcal{C},\gamma_l)$ is an operator for the problem Hamiltonian \mathcal{C} . The mixing Hamiltonian is defined as the sum of Pauli-X (σ^x) observables acting on all the n qubits, $\mathcal{B} = \sum_{j=1}^n \sigma_j^x$. The optimization task is to maximize/minimize $\langle \psi_{\gamma,\beta} | \mathcal{C} | \psi_{\gamma,\beta} \rangle$, the expectation value of $|\psi_{\gamma,\beta}\rangle$ given the problem Hamiltonian \mathcal{C} . For the implementation of QAOA there are a few hyperparameters like the initialization of γ , β and the number of layers p that need to be specified. We provide the analysis of these aspects in Section 5. A schematic diagram of the QAOA circuit is provided in Figure 1.

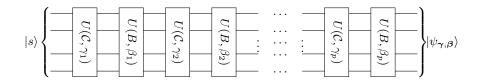


Fig. 1: Schematic diagram of the QAOA circuit with p layers.

4.1.1 Warm-start QAOA: Warm-start QAOA (WS-QAOA) is a variant of QAOA developed by Egger *et al.* [7]. The difference essentially lies in the initial state and the mixing Hamiltonian, which are defined based on the solution of the relaxed QUBO (which admits variables in [0, 1] instead of $\{0, 1\}$). Suppose there are in total L variables in the original QUBO and c_l^* be the optimal solution value of the lth variable in the relaxed QUBO, the initialization of the WS-QAOA circuit is done with $|\phi^*\rangle$, such that

$$|\phi^*\rangle = \bigotimes_{l=0}^{L-1} \hat{R}_Y(\theta_l) |0\rangle^{\otimes L} = \bigotimes_{l=0}^{L-1} \left(\sqrt{1 - c_l^*} |0\rangle + \sqrt{c_l^*} |1\rangle\right) , \qquad (5)$$

where $\hat{R}_Y(\theta_l)$ is a rotation gate on the lth qubit parameterized by angle $\theta_l=2 \arcsin\left(\sqrt{c_l^*}\right)$. As we can see, this initialization state ensures that the probability of measuring qubit l in state $|1\rangle$ is c_l^* . Another difference to the standard QAOA in WS-QAOA is the mixer Hamiltonian. Instead of the Pauli-X Hamiltonian, WS-QAOA utilizes a Hamiltonian whose ground state is $|\phi^*\rangle$ with eigenvalue -n. Formally, the mixer Hamiltonian in WS-QAOA is $\hat{H}_M^{ws}=\sum_{l=0}^{L-1}\hat{H}_{M,l}^{ws}$ such that

$$\hat{H}_{M,l}^{ws} = \begin{pmatrix} 2c_l^* - 1 & -2\sqrt{c_l^* (1 - c_l^*)} \\ -2\sqrt{c_l^* (1 - c_l^*)} & 1 - 2c_l^* \end{pmatrix} . \tag{6}$$

The mixer operator is $\exp(-i\beta \hat{H}_{M,l}^{ws})$, which is implemented in WS-QAOA using single qubit rotation gates $\hat{R}_Y(\theta_l)\hat{R}_Z(-2\beta)\hat{R}_Y(-\theta_l)$. The implementation of the WS-QAOA in this work is identical to the the warm-start QAOA described by Egger *et al.* [7].

4.1.2 Warm-started Standard QAOA: As we will see in our results and the study presented by Egger *et al.* [7], the warm-start QAOA definitely performs much better than the standard QAOA. To clearly understand the effect of warm-starting, *i.e.*, initializing the QAOA (WS-QAOA) circuit with the relaxed solution of a QUBO, we propose a variant of WS-QAOA, in that the circuit is initialized with the relaxed QUBO solution, but the mixer Hamiltonian is unchanged to the standard QAOA. We call this approach of QAOA the warm-started standard QAOA, and for brevity we refer to it as WS-Init-QAOA. The WS-Init-QAOA can be formally expressed as a variational quantum circuit which samples $|\psi'_{\gamma,\beta}\rangle = \prod_{l=p}^{1} U(\mathcal{B}, \beta_l) \cdot U(\mathcal{C}, \gamma_l) \cdot |\phi^*\rangle$,

where $|\phi^*\rangle$ is the relaxed QUBO solution, explained in Section 4.1.1. $U(\mathcal{B}, \beta_l) = e^{-i\beta_l \mathcal{B}}$ is the parameterized unitary operator resulting from the Pauli-X mixing Hamiltonian and $U(\mathcal{C}, \gamma_l)$ is an operator for the problem Hamiltonian \mathcal{C} . Note that the only difference to the WS-QAOA is the mixing Hamiltonian.

4.2 Variational Quantum Eigensolver

Like QAOA, the variational quantum eigensolver (VQE) [24] consists of a parameterized quantum circuit $U(\theta)$, where $\theta \in [0, 2\pi]^n$ are angles for single-qubit rotation gates. Analogous to QAOA, these parameters are tuned to minimize the expectation value $\langle \psi(\theta) | \mathcal{C} | \psi(\theta) \rangle$, where $\psi(\theta) := U(\theta) | 0 \rangle$ is the trial state prepared by the circuit $U(\theta)$ and \mathcal{C} is the Hamiltonian encoding the optimization problem. The VQE offers more freedom in the choice of the circuit. QAOA can be seen as a special case of VQE, namely if we choose a QAOA circuit as $U(\theta)$. A common requirement for the circuit $U(\theta)$ is hardware efficiency [16], meaning that the circuits consist of parameterized one-qubit rotations and two-qubit entangling gates and are kept relatively shallow in order to deal with short coherence times on NISQ devices. The drawback of using generic ansatz circuits is that a larger number of parameters (compared to the QAOA) is needed to guarantee expressivity [5] of the circuit, i.e., the circuit's ability to prepare the ground state of any choice of target Hamiltonian. For a full technical review of methods and best practices for VQE, we refer the reader to [34].

4.3 Quantum Annealing

Quantum annealing (QA) is a metaheuristic quantum optimization algorithm inspired by Adiabatic Quantum Optimization (AQO) and Adiabatic Quantum Computing (AQC). The algorithm starts by initializing a system of qubits in a simple-to-prepare optimum (known as the ground state or the initial Hamiltonian) that is slowly evolved to represent a combinatorial optimization problem expressed as an Ising model or QUBO (known as the final Hamiltonian) [15]. At the end of this process, the qubits' states represent a possible solution to the combinatorial optimization problem in the final Hamiltonian, with a non-zero probability of being a global optimum (i.e., the ground state of the final Hamiltonian). The specific evolution parameters which govern the path from initial to final Hamiltonian dictate the success of the QA algorithm, specifically the probability of measuring a ground state of the final Hamiltonian. For a full description of the physics and implementation of QA, we refer the reader to [12]; for a review of applications tested using QA, we refer the reader to [37].

Quantum annealing (or any QUBO solving method) can be expanded upon via the Iterative Heuristic Solver (IHS), originally developed by Rosenberg *et al.* [29]. The general idea of this metaheuristic procedure is to split a QUBO problem into several smaller subproblems and solve these iteratively instead of solving the entire problem at once. Suppose we aim to minimize $\mathbf{x}^T Q \mathbf{x}$, where

 $Q \in \mathbb{R}^{n \times n}$ is the QUBO matrix and $\mathbf{x} \in \{0, 1\}^n$ the binary solution vector. In the basic version of the IHS, we start with a randomly generated initial configuration of \mathbf{x} and repeatedly perform the following steps:

- 1. Randomly choose k variables from \mathbf{x} and fix the other n-k variables.
- 2. Optimize over the k chosen variables using the underlying QUBO solver.
- 3. Check for an improvement in solution quality over the previous iteration.

If no improvement is detected in the final step for several iterations, we assume convergence and output the final state of the vector \mathbf{x} as the optimized solution. It is to be noted that an iterative metaheuristic like this has a significantly larger runtime by design than other more direct approaches to solving a QUBO problem. However, the IHS avoids having to embed the full QUBO matrix of a problem, which is potentially large, on a quantum annealer. This increases the size of potentially solvable problems significantly, considering hardware limits, which makes it worthwhile to test the approach alongside our other presented methods.

5 Results

In this section, we first discuss different problem instances considered in this work for the multi-knapsack problem, along with the description of different measures to assess the performance of QAOA, VQE, and quantum annealing. Subsequently, we present detailed results obtained via benchmarking these algorithms over different scenarios. It must be noted that in this work we have executed the QAOA and VQE algorithms using state-vector simulations, while quantum annealing was carried out on real quantum devices. Additionally, we present the comparative performance of quantum annealing versus classical heuristic methods.

5.1 Problem Instances

To compare the performance of the various optimization methods and algorithms, 4 different instances of the knapsack problem are considered, increasing in problem size and complexity, which are listed in Table 1. The optimal solution was derived classically for each problem size using 0/1 integer programming.

Table 1: Benchmark Scenarios

Scenarios		2	3	4
Knapsacks	1	2	2	2
Items	8	4	5	6
${f Qubits}$	12	14	16	19
Optimal Sol. (v_{opt})	22	12	13	13
$\begin{array}{c} \textbf{Optimal Sol.} \ (v_{\textbf{opt}}) \\ \textbf{Prefactors} \ (A=B) \end{array}$	32	10	8	8

The maximal value of the optimal item distribution and the prefactors A and B of the penalty terms in the Hamiltonian in Equation (4) are given in the last two rows of Table 1. The parameters for the problem instances were initially chosen randomly and partly modified afterwards in order to ensure different levels of complexity.

5.2 Measures for Solution Quality

When solving the knapsack problem with a quantum program, each solvers' output is a probability distribution of quantum states represented as bitstrings, characterized by executing the program several times and taking measurements.

To inspect the validity of the solutions, the sum of all penalty terms in the QUBO is evaluated. If the total penalty value is equal to zero, the bitstring is considered a valid solution. Note that in this way, bitstrings encoding an item distribution with a total weight smaller than the knapsack capacity $(\sum_{j=0}^{N-1} w_j \cdot x_{i,j} \leq c_i)$ are considered invalid if the corresponding y_i^b slack bits are not correct (i.e., $H_{\text{capacity}} > 0$). The best solution \mathbf{x}_{\min} is defined as the bitstring with the lowest energy. This solution is characterized by the total value of the corresponding item distribution v_{tot} , divided by the value of the known optimal solution v_{opt} . This relative value, also known as the approximation ratio, is averaged over several runs N_{run} for all valid best solutions with the same parameters to obtain the mean closeness to optimum $C_{\text{opt}}(\mathbf{x}_{\min})$, such that

$$C_{\text{opt}}(\mathbf{x}_{\min}) = \frac{1}{N_{\text{run}}} \sum_{r=1}^{N_{\text{run}}} \frac{v_{\text{tot}}(\mathbf{x}_{\min,r})}{v_{\text{opt}}} \cdot 100.$$
 (7)

In practice, it is useful to find a distribution of items that is not necessarily optimal, but still has a high $C_{\rm opt}$ value. We consider all valid bitstrings ${\bf x}$ within a probability distribution which have a $C_{\rm opt}$ value above a certain threshold $C_{\rm lim}$, which we chose to be 90%. For each set of parameters, the amplitudes⁸ $a({\bf x})$ of all ${\bf x}$ with $C_{\rm opt}({\bf x}) \geq C_{\rm lim}$ are added and the sums are averaged over all runs. Thus, we define the overlap of the sampled solutions with 0.90-opt solution as $\langle {\bf O}_{90} \rangle$, where

$$\langle \mathbf{O}_{90} \rangle = \frac{1}{N_{\text{run}}} \sum_{r=1}^{N_{\text{run}}} \sum_{\mathbf{x} \mid C_{\text{opt}}(\mathbf{x}) \ge C_{\text{lim}}} a_r(\mathbf{x}) . \tag{8}$$

Note that it might be possible that the optimal solution has a high relative value, but the $\langle \mathbf{O}_{90} \rangle$ remains small, since there are no further solutions with a relative value above 90%.

5.3 Results for QAOA, WS-QAOA and WS-Init-QAOA

In this section, we discuss detailed results of the QAOA, WS-QAOA and WS-Init-QAOA, implemented using Qiskit [14]. For all results presented here, the quantum circuits were sampled with $n_{\rm samp}=10,000$ shots. The classical optimization of γ and β is carried out with the off-the-shelf optimizer class of SciPy Python library using the sequential least squares programming (SLSQP) algorithm [30], where the maximal number of iterations was limited to 10,000.

We carried out tests for the three QAOA based algorithms for the four different problem instances with a number of QAOA layers ranging from p=1

 $^{^8}$ The amplitude of a state ${\bf x}$ is defined as the square root of its probability in the sampled solution.

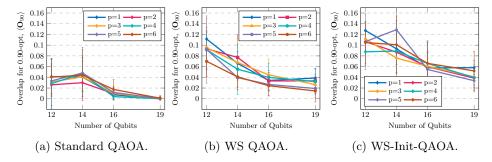


Fig. 2: Standard QAOA, Warm-start QAOA and WS-Init-QAOA overlaps with 0.90-opt results for different problem sizes and number of QAOA layers.

to p=6. For each problem instance, we randomly initialize the γ and β angles in the range $[-\pi,\pi]$. Additionally, to better assess the convergence of results, we repeat each test 20 times and report the average and the standard deviation values.

Figure 2 presents 0.90-opt overlap $\langle \mathbf{O}_{90} \rangle$ results for QAOA, WS-QAOA and WS-Init-QAOA against problem instances ranging from 12 to 19 qubits, for varying numbers of QAOA layers p, along with their standard deviation. The overlap values evidently decrease by increasing the problem sizes, reaching ≈ 0 for the last scenario with 19 qubits for standard QAOA. Increasing the number of layers seems to present none to minimal improvement in the solution values. This can be seen as an indicator that adding more layers does not increase the subspace of states explored by the QAOA-circuit. The overlap values obtained for WS-QAOA (Figure 2b) are much better than for QAOA. As discussed in Section 4.1.1, WS-QAOA differs from the standard QAOA in two aspects, the initial state $|\phi^*\rangle$ and the mixing Hamiltonian \hat{H}_M^{ws} whose ground state eigenvector is $|\phi^*\rangle$. Our results show that these two modifications certainly lead to a better performance over standard QAOA.

Remarkably though, WS-Init-QAOA (Figure 2c) seems to perform much better for all the instances, in comparison to WS-QAOA. This is an interesting result considering the mixer Hamiltonian for the WS-Init-QAOA brings the quantum states to their superposition position state (similar to standard QAOA). On the other hand, the mixer Hamiltonian for WS-QAOA is designed to bring the quantum states to the optimum continuous solution. Regardless, it is apparent that just improving the choice of initial state leads to even better results compared to modifying the mixer operator along with the initial value. We would like to emphasize this aspect and suggest that the good solutions obtained by WS-QAOA are mainly due to the classical pre-processing (i.e., the continuous solutions to the relaxed QUBO), and not due to the modified mixing Hamiltonian (\hat{H}_M^{ws}). As far as the closeness of the best solution to the optimum is concerned, both warm-start approaches outperform standard QAOA, as shown in Figure 3. While standard QAOA (Figure 3a) values drop below the 90%-mark already for 14 qubit problems, the WS-QAOA (Figure 3b), maintains 90% of the

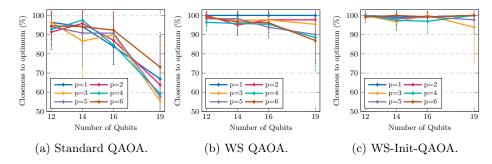


Fig. 3: Closeness to optimum of the best solutions obtained from standard QAOA, warm-start QAOA and WS-Init-QAOA for different problem sizes and number of QAOA layers.

optimal solution also for 19 qubits and the WS-Init-QAOA results remain even well above 90% (Figure 3c). The number of layers seems to have only a minor influence on the quality of results. Regarding the stability of the results, we notice that the standard deviation of both WS-QAOA results is smaller than in the standard QAOA. WS-Init-QAOA seems to again provide better values than the WS-QAOA.

In summary, while the number of QAOA layers does not have a considerable influence, using warm-start QAOA increases the quality of the results. Moreover, from the comparison of the two warm-start approaches we see that choosing a better initial state for the quantum circuit leads to significant improvements of the result quality. On the other hand, the additional modification of the mixing Hamiltonian incorporated in the WS-QAOA algorithm does not seem to have a distinct effect, since the WS-Init-QAOA approach, that just uses a different initial state, gives the best results. Other variations of QAOA including better initial rotation angles, as suggested in [38], need to be looked at to fully understand the true potential of QAOA for multi-knapsack problems.

5.4 Results from VQE

The ansatz circuit chosen for the VQE experiments is adopted from the work of Liu et al. [21] and consists of parameterized single-qubit rotations and two-qubit entangling gates without parameterization, schematically shown in Figure 4. We conducted the experiments with p=1 and p=2 layers, sampling 10,000 shots from the corresponding quantum circuits. All results are averaged over 20 runs, and the error bars in Figure 5 indicate the standard deviation. Parameters are randomly initialized in the range $[0, 2\pi]$, and the COBYLA optimizer is used for the classical parameter tuning. Compared to the QAOA results, the VQE reaches slightly worse approximation rates in general. However, when good solutions are found, they are sampled with higher probability compared to QAOA. The quality of the solutions improves when we take an ansatz circuit with 2 layers,

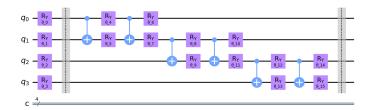


Fig. 4: A quantum circuit for 4-qubit VQE with a single layer.

although this results in almost twice as many parameters and thus a substantial calculation overhead on the parameter tuning side.

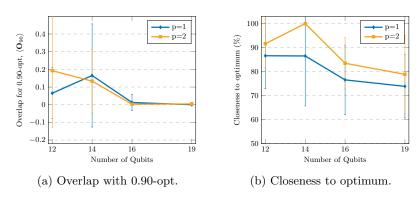


Fig. 5: Overlap and closeness results for VQE algorithm for the multi-knapsack problem for 1 and 2 layers circuit ansatz.

As can be seen from the high number of iterations needed for convergence of the classical optimizer in Table 3, VQE exhibits a longer runtime than QAOA, even when one accounts for the simulation time (see Section 5.5). These runtimes may serve as a motivation to look for strategies which reduce the number of classical iterations. Overall, the benefits of VQE, *i.e.*, hardware-adapted quantum circuits which suffer less from noise, can only become apparent when running the algorithms on actual quantum hardware. Thus, despite the worse approximation rates and longer run times, VQE deserves further exploration in the context of quantum optimization algorithms.

5.5 Runtime estimation on quantum devices for QAOA and VQE

As described in the previous sections, the results for QAOA and VQE are obtained by simulations of the quantum circuits. In order to assess the performance of each quantum algorithm and provide a meaningful comparison

with the results from quantum annealing, we estimate the runtimes for standard QAOA and VQE on a quantum processing unit (QPU). A simple model to describe the total runtime T of a variational quantum algorithm is explained in [36] and reads:

$$T = n_{\text{iter}} \cdot [n_{\text{samp}} \cdot (t_{\text{circ}} + t_{\text{meas}}) + t_{\text{opt}} + t_{\text{comm}}], \tag{9}$$

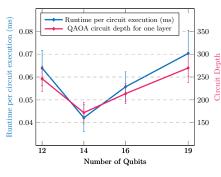
where $n_{\rm iter}$ denotes the number of iterations of the optimizer, $n_{\rm samp}$ is the number of samples taken to measure the quantum state and $t_{\rm circ}$, $t_{\rm meas}$, $t_{\rm opt}$ describe the times to execute all gates of the quantum circuit, measure all qubits and perform the classical optimization, respectively.

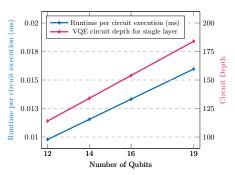
The communication time between the quantum and the classical computer used for optimization is represented by t_{comm} . To provide a rough estimate of the expected runtimes on a QPU, we assume that the times for classical optimization, measurement and communication remain in a similar range as for the simulation and just consider how t_{circ} is changed. The circuit execution time on a QPU \tilde{t}_{circ} can be calculated from the gate execution times and the structure

Table 2: Gate execution times from Qiskit backend FakeBrooklyn.

Gate	Exec. time (ns)
R_Z	0
S_X	35.56
X	35.56
CX	370 ± 80

of the circuit. As an example, we choose the IBM-Q Brooklyn device consisting of 65 qubits. The properties such as the topology and gate execution times are estimated using the FakeBrooklyn backend from the FakeProvider module of Qiskit. The corresponding execution times for the native gate set are averaged over all qubits and the mean values are presented in Table 2.





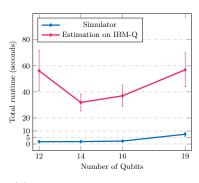
(a) QAOA runtime and circuit depth

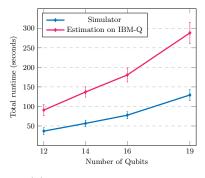
(b) VQE runtime and circuit depth

Fig. 6: Mean circuit execution times \tilde{t}_{circ} on an IBM-Q device estimated from the gate times and mean circuits depths for QAOA and VQE for p=1.

The quantum circuits for QAOA and VQE are then transpiled to the FakeBrooklyn backend using the standard Qiskit transpiler with optimization level 3 [14]. For each circuit, the execution time is derived from a schedule describing the execution of all gates on all qubits, taking into account parallel

execution of gates as well as constraints on the timings imposed by two-qubit gates. The QPU runtimes were estimated for QAOA and VQE with p=1, averaging over 20 compilation runs to account for the stochastic placement of SWAP gates within the standard Qiskit transpilation pass. The mean runtimes $\tilde{t}_{
m circ}$ for a single execution of the circuit and their standard deviation are shown in Figure 6, along with the corresponding mean circuit depths. As described in Section 4.2, the VQE algorithm needs less gates, resulting in a mean depth being smaller by about a factor of 2 compared to QAOA. Due to the different kinds of gates being used, the circuit runtimes are decreased even by a factor of 4. For VQE, the circuit depth and execution time grow linearly with the problem size, since the structure of the circuit is independent of the specific problem instance and just depends on the number of qubits. In contrast, the circuit depth and execution time of QAOA are larger for the first scenario than for scenario 2 and 3. The reason lies in the different problem structure of scenario 1, which contains only a single knapsack, but more items than the following scenarios which finally leads to a higher number of two-qubit gates in the problem Hamiltonian.





(a) Overall runtimes for QAOA

(b) Overall runtimes for VQE

Fig. 7: Mean overall runtimes on the simulator and estimated on an IBM-Q device for QAOA and VQE with p = 1. The error bars are computed by error propagation.

To estimate the overall runtime \tilde{T} on the QPU, we replace $t_{\rm circ}$ in Equation (9) with $\tilde{t}_{\rm circ}$, keeping the other contributions unchanged. The execution times and the overall runtimes on the simulator are obtained from a python profiler, averaging over 5 runs with random initial parameters.

Table 3: Mean number of optimizer iterations for QAOA and VQE for p = 1.

Num. qubits		
12	80 ± 20	700 ± 100
14	70 ± 10	900 ± 100
16	60 ± 10	700 ± 100 900 ± 100 1000 ± 100
19	80 ± 20	1400 ± 200

In Figure 7, the resulting runtimes for the simulator are compared with the estimate on the QPU. For QAOA, the runtimes on the QPU are about 30 times larger than on the simulator. Both runtimes show a similar dependence on the problem size and complexity, following the trend observed in Figure 6a for the circuit depth

and circuit execution time. In general, we observed no considerable difference in the

simulator runtimes and number of optimizer iterations n_{iter} between standard QAOA, WS-QAOA and WS-init-QAOA.

The overall runtimes for VQE on the simulator and QPU differ only by a factor of 2-3 and are both larger than for QAOA. While the QPU circuit runtimes $\tilde{t}_{\rm circ}$ are smaller for VQE, the mean circuit runtimes on the simulator lie in the same range as for QAOA with $t_{\rm circ}^{\rm vqe} \sim (2.5-4.2)\mu{\rm s}$ and $t_{\rm circ}^{\rm qaoa} \sim (0.8-7.1)\mu{\rm s}$. However, the VQE algorithm contains more parameters to optimize and thus it takes more iterations to find the optimal parameter values, as shown by the values in Table 3. This increases the runtimes on the simulator as well as the QPU estimate accordingly.

5.6 Results from Annealing

We tested both simulated and quantum annealing, along with the IHS with simulated annealing. For quantum annealing, we focused on two of the devices available on Amazon Braket: *D-Wave 2000Q* (2,048 qubits) and the larger *D-Wave Advantage 6.1* (5,760 qubits). For the IHS, we used simulated annealing with 50 iterations in a single run and set the number of optimization parameters to 12. All annealing algorithms were executed with 1,000 reads and repeated ten times to evaluate the algorithms' stability.

The $\langle \mathbf{O}_{90} \rangle$ results displayed in Figure 8a show a higher overlap for simulated annealing - between 0.25 ± 0.01 in scenarios with 19 qubits and 0.43 ± 0.01 with 14 qubits - compared to the quantum annealing options which perform similar and do not exceed 0.14 ± 0.05 with *D-Wave 2000Q* in scenarios with 14 qubits. For IHS, the 0.90-opt overlap cannot be computed in a meaningful way due to the nature of the algorithm.

Figure 8b displays the closeness of the found solution relative to the optimum. The simulated annealing and IHS approaches find optimal solutions for each of the tested knapsack instances. With quantum annealing, the optimal solutions were obtained only for Scenario 2 with 14 qubits. For the other scenarios, D-Wave Advantage 6.1 exhibits solution qualities of $95.4 \pm 7.4\%$ for the largest problem instance up to $98.2 \pm 3.8\%$ for Scenario 1 and outperforms *D-Wave* 2000Q yielding $88.5 \pm 11.0\%$ to $94.5 \pm 6.4\%$. Figure 8c displays the number of physical qubits needed to embed the theoretical qubits from the QUBO, compensating for the limited connectivity of the physical qubits. As expected, the D-Wave 2000Q requires more physical qubits to embed the same QUBO than the *D-Wave Advantage 6.1*, as its architecture has lower connectivity [3]. Since finding the embedding is done by a non-deterministic algorithm, the required number physical qubits on each device varies between runs of the same scenario. In scenarios with two knapsacks (Scenarios 2-4), the number of physical qubits grows proportionally to the number of logical qubits needed to model the respective problem. However, in Scenario 1 more physical qubits are required to embed the QUBO than in Scenario 2, even though less logical qubits are necessary to represent the problem with one knapsack and eight items

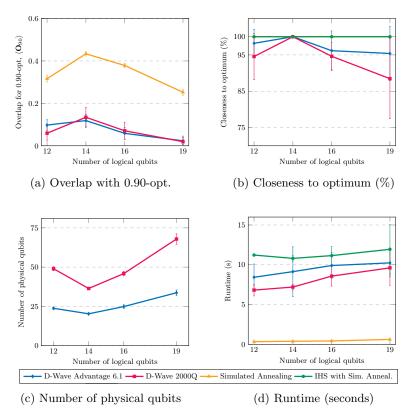


Fig. 8: Overlap, closeness to optimum, number of required physical qubits for annealing and runtime.

(see Figure 1). This shows the importance of considering the physical properties of the particular quantum hardware and not solely looking at the theoretical number of logical qubits.

The runtimes for various annealing approaches are compared in Figure 8d. For simulated annealing, the average runtimes are below 1s for all scenarios, while the runtimes of the IHS are the longest of all annealing approaches considered in this work, ranging from $10.8 \pm 0.3s$ to $11.9 \pm 3.1s$. With runtimes between $6.8 \pm 0.7s$ and $9.6 \pm 2.2s$ (D-Wave Advantage 6.1) and between $8.4 \pm 1.7s$ and $10.2 \pm 1.7s$ (D-Wave 2000Q), respectively, quantum annealing approaches are about a factor 3-7 faster than the QPU estimates for QAOA and about 10-30 times faster than the estimates for VQE presented in the previous section. The quantum annealing runtimes grow only moderately with the problem size compared to the pronounced increase observed especially with VQE. Note that in our analysis of quantum annealing runtimes, we also include the communication overhead between the user and the QPU as well as measurement times, which were not included in the estimation for QAOA and VQE. Thus, the runtimes

for the gate-based approaches are expected to increase even more in practice, also when adding more layers to the circuits. In general, using a different hardware platform will also influence the runtime, since the gate times depend on the physical realization of the QPU. The circuit runtime also has to be compared to the coherence time of the qubits: while the gate execution times on other platforms such as trapped ions or cold atoms are in general longer than for superconducting QPUs such as the IBM-Q device considered here, those platforms usually also feature longer coherence times. Moreover, realizations based on ions or atoms exhibit better connectivity between the qubits which in turn reduces the amount of gates needed to run the circuit on the QPU and thus also reduces the amount of noise introduced on NISQ-devices [36]. Eventually, the absolute values of the runtimes have to be judged along with the quality of the delivered results within the specific context of the application.

5.7 Comparison of the quantum algorithms

The result quality provided by the different quantum solvers is summarized in Figure 9, where the results for the gate-based approaches are averaged over different numbers of layers. While the result quality is decreasing with the problem size for all approaches, this effect is most pronounced for VQE and standard QAOA. Overall, the best results are delivered by WS-Init-QAOA and QA on the *D-Wave Advantage 6.1* device, which show quite comparable values for the 0.90-opt overlap and the closeness to optimum. As discussed in the previous section however, the runtimes for QA on a QPU are shorter than the corresponding estimates for QAOA.

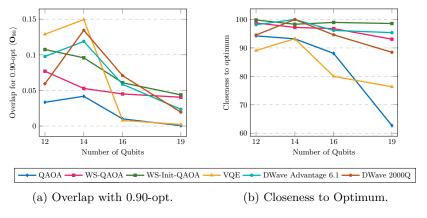


Fig. 9: Average overlap and closeness to optimum values for all the quantum approaches, over different problem sizes.

When comparing empirical results from different quantum algorithms, it is important to consider how the differences in implementation can affect both results and their interpretation. For quantum annealers, it is known that noise, embedding overhead, and high precision requirements are all detrimental to performance. Simulated QAOA and VQE do not suffer from these issues, but rather are limited by the quality of parameter settings with classical optimization and statistical sampling accuracy. Conversely, quantum annealing samples are known to primarily populate local minima due to classical effects after the freeze-out point [2], which is not the case for QAOA and VQE.

6 Conclusion and outlook

The aim of this work was to compare different approaches for solving the multi-knapsack problem in view of practical applications. We have defined appropriate measures for this comparison such as the 0.90-opt overlap and have also discussed the implementation of the considered quantum algorithms on real hardware as well as their limitations. The direct comparison of the most common gate-based algorithms with quantum annealing and simulated annealing provided in this work will help to better estimate the potential of these approaches for solving more complex optimization problems such as the knapsack problem in the future. Since practitioners might not have access to various types of quantum computing hardware, the suggested estimation of runtimes on real hardware derived from simulations of quantum circuits can be useful to carry out benchmarks of quantum algorithms.

Our results show that adapting a standard algorithm such as QAOA can considerably improve the quality of the delivered results. Comparable advantages might be achieved for VQE by finding similar initialization and warm-starting strategies as for QAOA. Deriving optimized starting angles for QAOA (or VQE) as described in [38] or tailoring the schedule of quantum annealing to minimize transfer to higher energy states [38] constitute other promising approaches. Moreover, we can conclude that variational gate-based approaches will profit from better optimization strategies for the circuit parameters to lower the number of iterations, especially in the case of VQE.

As a future work, it would be of great interest to study the QAOA and VQE algorithms incorporating the ideas from Koretsky et al. [19] and Braine et al. [1], which provide an alternative and qubit-efficient way of formulating inequality constraints in a QUBO model without requiring binary slack bits. These techniques seem to be promising since the convergence of slack bits in all quantum algorithms based on QUBOs is generally hard to achieve. Moreover, it would also be interesting to study and implement the qubit-efficient encoding of optimization problems especially with VQE [33].

Tackling realistic problems with millions of qubits, as described in Section 2, is out of scope for the currently available NISQ-devices. Considering the roadmaps of various quantum hardware vendors, however, quantum computers with up to 10,000 qubits might become available within 2-3 years. Thus, using strategies to find optimized problem formulations and algorithms with reduced number of qubits and quantum operations will be of key importance

to fit realistic problems onto those intermediate-sized quantum computers. In addition, matching the design of algorithms and quantum computing hardware is seen as another quite promising approach in the NISQ-era.

References

- Braine, L., Egger, D.J., Glick, J.R., Woerner, S.: Quantum algorithms for mixed binary optimization applied to transaction settlement. IEEE Transactions on Quantum Engineering 2, 1–8 (2019)
- 2. D-Wave Systems Inc.: Annealing Implementation and Controls (2022), https://docs.dwavesys.com/docs/latest/c_qpu_annealing.html
- D-Wave Systems Inc.: D-Wave QPU Architecture: Topologies (2022), https://docs.dwavesys.com/docs/latest/c_gs_4.html
- 4. van Dam, W., Eldefrawy, K., Genise, N., Parham, N.: Quantum optimization heuristics with an application to knapsack problems. arXiv preprint arXiv:2108.08805 (2021)
- 5. Du, Y., Tu, Z., Yuan, X., Tao, D.: Efficient measure for the expressivity of variational quantum algorithms. Physical Review Letters 128(8) (feb 2022)
- Ebadi, S., Keesling, A., Cain, M., Wang, T.T., Levine, H., Bluvstein, D., Semeghini, G., Omran, A., Liu, J.G., Samajdar, R., Luo, X.Z., Nash, B., Gao, X., Barak, B., Farhi, E., Sachdev, S., Gemelke, N., Zhou, L., Choi, S., Pichler, H., Wang, S.T., Greiner, M., Vuletić, V., Lukin, M.D.: Quantum optimization of maximum independent set using rydberg atom arrays. Science 376(6598), 1209–1215 (June 2022)
- Egger, D.J., Mareček, J., Woerner, S.: Warm-starting quantum optimization. Quantum 5, 479 (2021)
- 8. Farhi, E., Goldstone, J., Gutmann, S.: A quantum approximate optimization algorithm. arXiv preprint arXiv:1411.4028 (2014)
- Finžgar, J.R., Ross, P., Klepsch, J., Luckow, A.: QUARK: A Framework for Quantum Computing Application Benchmarking. arXiv preprint arXiv:2202.03028 (2022)
- França, D.S., García-Patrón, R.: Limitations of optimization algorithms on noisy quantum devices. Nature Physics 17(11), 1221–1227 (oct 2021)
- 11. Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing (STOC). p. 212–219. Association for Computing Machinery, New York, NY, USA (1996)
- 12. 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 (may 2020)
- 13. Henrich, J., Li, J., Mazuera, C., Perez, F.: Future-proofing the supply chain (2022), https://www.mckinsey.com/capabilities/operations/our-insights/future-proofing-the-supply-chain
- 14. IBM: QisKit SDK (2022), https://qiskit.org/
- Kadowaki, T., Nishimori, H.: Quantum annealing in the transverse ising model. Phys. Rev. E 58, 5355–5363 (Nov 1998)
- Kandala, A., Mezzacapo, A., Temme, K., Takita, M., Brink, M., Chow, J.M., Gambetta, J.M.: Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. Nature 549(7671), 242–246 (sep 2017)

- 17. Karp, R.M.: Reducibility among Combinatorial Problems, pp. 85–103. Springer US, Boston, MA (1972)
- Kellerer, H., Pferschy, U., Pisinger, D.: Knapsack Problems, p. 461. Springer, Berlin (2004)
- Koretsky, S., Gokhale, P., Baker, J.M., Viszlai, J., Zheng, H., Gurung, N., Burg, R., Paaso, E.A., Khodaei, A., Eskandarpour, R., Chong, F.T.: Adapting quantum approximation optimization algorithm (qaoa) for unit commitment. In: 2021 IEEE International Conference on Quantum Computing and Engineering (QCE). pp. 181–187 (2021)
- 20. Laabadi, S., Naimi, M., El Amri, H., Achchab, B.: The 0/1 multidimensional knapsack problem and its variants: A survey of practical models and heuristic approaches. American Journal of Operations Research 8, 395–439 (2018)
- Liu, X., Angone, A., Shaydulin, R., Safro, I., Alexeev, Y., Cincio, L.: Layer VQE: A
 variational approach for combinatorial optimization on noisy quantum computers.
 IEEE Transactions on Quantum Engineering 3, 1–20 (2022)
- 22. Lucas, A.: Ising formulations of many NP problems. Frontiers in Physics 2 (2014)
- 23. Patvardhan, C., Bansal, S., Srivastav, A.: Quantum-inspired evolutionary algorithm for difficult knapsack problems. Memetic Computing 7, 135–155 (2015)
- 24. Peruzzo, A., McClean, J., Shadbolt, P., Yung, M.H., Zhou, X.Q., Love, P.J., Aspuru-Guzik, A., O'Brien, J.L.: A variational eigenvalue solver on a photonic quantum processor. Nature Communications 5(1) (jul 2014)
- Pisinger, D.: Where are the hard knapsack problems? Computers & Operations Research 32(9), 2271–2284 (2005)
- 26. Pusey-Nazzaro, L., Date, P.: Adiabatic quantum optimization fails to solve the knapsack problem. arXiv preprint arXiv:2008.07456 (2020)
- 27. Quantum Technology and Application Consortium QUTACet al.: Industry quantum computing applications. EPJ Quantum Technology 8(25) (2021)
- 28. Quintero, R.A., Zuluaga, L.F.: Characterizing and Benchmarking QUBO Reformulations of the Knapsack Problem. Tech. rep., Technical Report. Department of Industrial and Systems Engineering, Lehigh University (2021)
- 29. Rosenberg, G., Vazifeh, M., Woods, B., Haber, E.: Building an iterative heuristic solver for a quantum annealer. Computational Optimization and Applications 65(3), 845–869 (apr 2016)
- 30. SciPy: SciPy documentation (2022), https://docs.scipy.org/doc/scipy/index.html
- 31. Shor, P.: Algorithms for quantum computation: discrete logarithms and factoring. In: Proceedings 35th Annual Symposium on Foundations of Computer Science (FOCS). pp. 124–134 (1994)
- 32. Streif, M., Yarkoni, S., Skolik, A., Neukart, F., Leib, M.: Beating classical heuristics for the binary paint shop problem with the quantum approximate optimization algorithm. Physical Review A 104(1) (July 2021)
- 33. Tan, B., Lemonde, M.A., Thanasilp, S., Tangpanitanon, J., Angelakis, D.G.: Qubit-efficient encoding schemes for binary optimisation problems. Quantum 5, 454 (May 2021)
- 34. Tilly, J., Chen, H., Cao, S., Picozzi, D., Setia, K., Li, Y., Grant, E., Wossnig, L., Rungger, I., Booth, G.H., Tennyson, J.: The variational quantum eigensolver: A review of methods and best practices. Physics Reports 986, 1–128 (2022)
- 35. Wilbaut, C., Hanafi, S., Salhi, S.: A survey of effective heuristics and their application to a variety of knapsack problems. IMA Journal of Management Mathematics 19(3), 227–244 (2008)

- 36. Wintersperger, K., Safi, H., Mauerer, W.: QPU-System Co-Design for Quantum HPC Accelerators. Proceedings of the 35th GI/ITG International Conference on Architecture of Computing Systems (2022)
- 37. Yarkoni, S., Raponi, E., Bäck, T., Schmitt, S.: Quantum annealing for industry applications: introduction and review. Rep. Prog. Phys. 85(10), 104001 (Sep 2022)
- 38. Zhou, L., Wang, S.T., Choi, S., Pichler, H., Lukin, M.D.: Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices. Physical Review X 10(2), 021067 (2020)