Li, Zongji; Seidel, Tobias; Leib, Dominik; Bortz, Michael; Heese, Raoul

Efficient solution of the number partitioning problem on a quantum annealer: a hybrid quantum-classical decomposition approach

Journal of Heuristics

Provided in Cooperation with:
Springer Nature

Suggested Citation: Li, Zongji; Seidel, Tobias; Leib, Dominik; Bortz, Michael; Heese, Raoul (2025) : Efficient solution of the number partitioning problem on a quantum annealer: a hybrid quantum-classical decomposition approach, Journal of Heuristics, ISSN 1572-9397, Springer US, New York, NY, Vol. 31, Iss. 2,
https://doi.org/10.1007/s10732-025-09556-3

This Version is available at:
https://hdl.handle.net/10419/323325

Journal of Heuristics (2025) 31:21
https://doi.org/10.1007/s10732-025-09556-3

Efficient solution of the number partitioning problem on a quantum annealer: a hybrid quantum-classical decomposition approach

Zongji Li[1] · Tobias Seidel[1] · Dominik Leib[1] · Michael Bortz[1] · Raoul Heese[1]

Received: 22 February 2024 / Revised: 7 February 2025 / Accepted: 22 February 2025 /
Published online: 17 April 2025

Abstract
Current quantum computers can only solve optimization problems of a very limited size. For larger problems, decomposition methods are required in which the original problem is broken down into several smaller sub-problems. These are then solved on the quantum computer and their solutions are recombined into a final solution for the original problem. Often, these decomposition methods do not take the specific problem structure into account. In this paper, we present a tailored method using a divide-and-conquer strategy to solve the 2-way Number partitioning problem (NPP) with a large number of variables. The idea is to perform a specialized decomposition into smaller NPPs, which are solved on a quantum computer, and then recombine the results into another small auxiliary NPP. Solving this auxiliary problem yields an approximate solution of the original larger problem. We experimentally verify that our method allows to solve NPPs with over a thousand variables using the D-Wave Advantage quantum annealer (Advantage_system6.4).

Keywords Number partitioning problem · Quantum optimization · Quantum annealing · Decomposition approach

1 Introduction

The current D-Wave *Advantage* quantum annealer (D-Wave Systems Inc 2020, 2024a) can be used to solve optimization problems with binary variables and a quadratic objective function. For this purpose, each optimization variable is captured by one qubit and the dynamics of the system are controlled by the properties of the objective function such that the solution of a problem instance can be determined from the measurement of the final quantum state (Albash and Lidar 2018; Rajak et al. 2022;

✉ Tobias Seidel
tobias.seidel@itwm.fraunhofer.de

[1] Fraunhofer Institute for Industrial Mathematics ITWM, Fraunhofer-Platz 1, 67663 Kaiserslautern, Germany

Yulianti and Surendro 2022). However, there is no guarantee that a solution obtained in this way is actually an optimal solution, as quantum tunneling could result in a non-optimal result even on an ideal device. In addition, technical imperfections of the currently available Noisy Intermediate-Scale Quantum (NISQ) hardware (Preskill 2018) may lead to significant uncertainties. Therefore, quantum annealing is typically treated as a heuristic algorithm for which the solution is extracted from a large number of measurements.

The Advantage system (Advantage_system6.4) possesses 5000 qubits that can be used simultaneously to solve optimization problems. In practice, however, the number of optimization variables that can be concurrently encoded on such a device is typically much less than 5000. The reason is the limited connectivity of the hardware. For Quantum Annealing (QA), those qubits must interact with each other for which there is a correlation in the objective function. However, due to technical limitations, the hardware can only realize interactions between certain qubits. From a mathematical point of view, the connectivity of the hardware can be represented as a so-called "hardware graph", in which the qubits represent the nodes and the edges represent the possible interactions. The hardware graph of the Advantage_system6.4 used in this work has the Pegasus topology, which is an improvement in connectivity compared to the prior Chimera topology, but still far from being fully connected D-Wave Systems Inc (2021). Similarly, a problem instance can be represented as a "problem graph" in which the optimization variables represent the nodes and the edges represent non-vanishing correlations. A necessary classical pre-processing step before the actual QA process is to find feasible "minor embedding" of the problem graph in the hardware graph, which is by itself a NP-hard problem (Matoušek and Thomas 1992). However, efficient algorithms exist to find minor embeddings on D-Wave hardware topologies for practical applications (see Robertson and Seymour (1995) for a general algorithm and Cai et al. (2014); Boothby et al. (2016) for algorithms adapted to D-Wave hardware). For this purpose, the effective hardware connectivity can be increased by joining several qubits into a strongly correlated chain. This method is required to overcome the hardware limitations for sufficiently complex problems, but it is not always reliable and can therefore introduce additional uncertainties.

In practice, the difficulty of finding a minor embedding on a given hardware graph depends heavily on the nature of the problem. For problem graphs that are a sub-graph of the hardware graph, finding a minor embedding is trivial. On the other hand, it may be very difficult or even impossible to find a feasible embedding for a sufficiently large, densely connected graph (resulting from strongly correlated variables).

Hybrid quantum-classical decomposition methods offer a fallback option in such cases. They utilize both classical resources (for the decomposition of the original problem into smaller sub-problems with an easier to find minor embedding) and quantum resources (for the solution of the sub-problems with QA) in a possibly iterative way. Hybrid decomposition methods are an important cornerstone for the practical use of NISQ annealers. The hybrid decomposition solver *qbsolv* (QB) (Booth et al. 2017) is based on a heuristic general-purpose strategy. Aimed at improving this method, the work in Okada et al. (2019) further exploits the connectivity within problems to accommodate larger sub-problems. As also observed in Atobe et al. (2022), Osaba et al. (2021), Raymond et al. (2023), and Shaydulin et al. (2019), the current decom-

position methods are designed for compatibility with as many distinct problem species as possible and thus problem-agnostic, although Bass et al. (2021) states that QB is in general still the best choice for large problems as of 2021.

Problem dependent decomposition methods have been presented in Pelofske et al. (2019), Pelofske et al. (2021) focusing on graph problems. In Pelofske et al. (2022), Ottaviani and Amendola (2018), algorithms for the factorization of matrizes are presented.

In this paper, we propose a novel problem-oriented hybrid decomposition method, which is specifically designed to solve the optimization version of the NPP, which is also called *partition problem* and constitutes one of Karp's 21 NP-complete problems (Karp 1972). An NPP is defined as follows. Given a finite multiset

$$W := \{w_1, \ldots, w_n\} \tag{1}$$

of $n \geq 1$ elements $w_i \in \mathbb{N}$ for all $i \in [n] := \{1, \ldots, n\}$, the goal is to solve

$$\min_{A \subseteq W} E(W, A) \tag{2}$$

with the objective

$$E(W, A) := \left| \sum_{w \in A} w - \sum_{w' \in W \setminus A} w' \right|, \tag{3}$$

which represents the error (or energy) of the solution A to the NPP generated by W. Practical applications of NPP (or its extension to more than 2 sets known as multi-way NPP) include scheduling with multiple identical machines, as exemplified in Coffman et al. (1993) and Pinedo (2012).

As mentioned before, the NP-hardness of this problem has been established in Karp (1972). Furthermore, it has been demonstrated that both exact cover and knapsack problems are linearly reducible (in terms of problem size) to NPPs without any constraints in Karp (1972). This implies that solving the NPP also addresses these other NP-hard problems. Early exact algorithms with an exponential runtime are, for example, described in Horowitz and Sahni (1974); Schroeppel and Shamir (1981).

Classical heuristics for NPP include methods such as the Largest Differencing Method (LDM) introduced by Karmarkar and Karp (1983), and the Greedy Algorithm (GR) as described in Mertens (2005). The LDM is considered to perform well on an average basis as a polynomial-time heuristic, as analyzed in Boettcher and Mertens (2008) and Michiels et al. (2007). Based on LDM, Korf (1998) proposed an exact anytime algorithm for general l-way NPPs, which requires exponential time to obtain an optimal solution in the worst case. Alternatively, Krasecki et al. (2023) studies a hybrid approach using molecular computers for an instance of 5 elements.

In this study, we do not aim to solve the NPP on a classical device, but want to investigate to what extent quantum annealers can find a solution. Solving optimization problems with the help of quantum computers is a highly relevant research topic with

various opportunities and challenges. A more general discussion of this subject would go far beyond the scope of this paper. For a comprehensive review of this topic, we refer to Abbas et al. (2023).

To solve an NPP on a quantum annealer, we reformulate Eq. (2) as a Quadratic Unconstrained Binary Optimization (QUBO) (Kochenberger et al. 2014; Glover et al. 2022) of the form

$$\min_{x \in \{0,1\}^n} x^T Q x \tag{4}$$

with a coefficient matrix $Q \in \mathbb{Z}^{n \times n}$ with entries

$$Q_{ij} := \begin{cases} w_i w_j & \text{if } i \neq j \\ w_i(w_i - c) & \text{if } i = j \end{cases} \tag{5}$$

for $i, j \in [n]$ and $c := \sum_{w_i \in W} w_i$, as formulated, for example, in Glover et al. (2022). The n-dimensional vector x of binary optimization variables represents the solution $A = \{w_i \in W \mid x_i = 1\}$ of Eq. (3), where $x_i = x_i^2$ for $i \in [n]$ as a consequence of the binary domain. The equivalence of Eqs. (4) and (2) follows from the squared error

$$\left(\sum_{i=1}^{n} w_i x_i - \sum_{i=1}^{n} w_i(1 - x_i)\right)^2 = c^2 - 4c \sum_{i=1}^{n} w_i x_i + 4\left(\sum_{i=1}^{n} w_i x_i\right)^2 = c^2 + 4x^\top Q x,$$

which reduces to the objective from Eq. (4) when removing the constant term c^2 and the linear factor 4.

Assuming that no weight is vanishing (as defined in our problem statement), the coefficient matrix Q is fully dense, i.e., it is comprised entirely of non-vanishing elements. This makes it particularly challenging to solve the corresponding QUBO on a quantum annealer: for the complete embedding of the problem, every qubit has to be connected to every other qubit. Therefore, a decomposition is crucial for handling larger problem instances on a NISQ device.

In short, the main idea of the proposed decomposition method is to consider subsets of the multiset W. These subsets form sub-NPPs and can be chosen sufficiently small such that they can be solved on the D-Wave Advantage system. Solving these sub-problems leads to partial solutions, which, in another step, are used to formulate another auxiliary NPP. Subsequently, solving this auxiliary NPP allows us to merge the sub-solutions to an overall solution of the complete NPP. Numerical experiments show that the results are improved significantly compared to a general purpose decomposition method. This indicates that by focusing on specific problems and leveraging their unique properties, one can significantly enhance solution quality.

The remaining paper is structured as follows. We introduce our proposed hybrid decomposition method in Sect. 2. Subsequently, we investigate the proposed method in Sect. 3 on a theoretical level, where we examine the influence of our decomposition on so-called *perfect solutions*. To further support our conclusions, we show the results of the numerical experiments in Sect. 4, comparing LDM, GR, Simulated Annealing

(SA), QB, and different variants of our algorithm. We conclude the paper in Sect. 5 with a summary of our results and ideas for future research directions.

2 Method

We consider the NPP from Eq. (2) with the error function $E(W, A)$ and a given multiset W as introduced in Eqs. (3) and (1), respectively. The proposed method consists of three major steps: First, decompose the original NPP into multiple smaller sub-NPPs. Second, solve the sub-NPPs with QA. Third, recombine the solutions of the sub-NPPs into a solution for the original NPP by solving an auxiliary NPP with SA. This auxiliary NPP is in particular smaller than the original NPP. Both QA and SA utilize the formulation of a NPP as a QUBO from Eq. (5).

The key idea of the proposed method is that solving the sub-NPPs in the second step and also the auxiliary NPP in the third step is easier than solving the original NPP. Conceptually, our method differs from many other hybrid algorithms such as QB. Other hybrid algorithms often run a loop in which one tries to improve the solution step by step. This process terminates when no further progress is obtained or when the maximum number of iterations is reached. We partition the problem into sub-problems only once, solve them all, and then combine the solutions in a single step.

We describe the three steps of the proposed method in more detail in the following.

1. Decompose: Initially, we choose a number of sub-NPPs $m \in \mathbb{N}$ with $m \leq n$ and a corresponding decomposing vector $j \in [m]^n$ that determines their construction. Specifically, the multiset W of the original NPP, Eq. (1), is decomposed into m smaller multisets $\mathcal{W} := \{W_1, \ldots, W_m\}$ with $W_k := \{w_i \in W \mid j_i = k\}$ for all $k \in [m]$, where $\bigcup_{k=1}^{m} W_k = W$ and $\sum_{k=1}^{m} |W_k| = |W|$. Without loss of generality, we assume that $|W_k| \geq 1$ for all $k \in [m]$. Each of the multisets W_k in $\mathcal{W}$ is used to construct a sub-NPP defined by $\min_{A \subseteq W_k} E(W_k, A)$ that is smaller than the original NPP in terms of problem size.
   The choice of the decomposing vector j determines the decomposition and can therefore have a major influence on the performance of the method. Throughout this paper, we pursue a randomized and balanced decomposition strategy to keep the size of each sub-problem about equally large. Specifically, given n and m, we define a vector $j' \in [m]^n$ with elements

$$j'_i = \begin{cases} \lfloor (i-1)/K \rfloor + 1 & \text{if } i \leq Km \\ \lfloor (i - Km - 1)/K \rfloor + 1 & \text{otherwise} \end{cases} \tag{6}$$

   for $i \in [n]$, where $K := \lfloor n/m \rfloor$. We then construct j by shuffling j' in a uniformly random manner.
2. Solve sub-problems: We solve each sub-NPP independently, using an exchangeable solution method, which is called *SubSolver*. The only requirement for this solver is that it is able to provide a (not necessarily optimal) solution for any given NPP. By default, we use QA as the SubSolver. From each each sub-NPP, we obtain a solution A_k for all $k \in [m]$. The resulting partitions—which are not necessarily the

optimal partitions—are denoted by $A_k^1 := A_k$ and $A_k^2 := W_k \backslash A_k$, respectively, for all $k \in [m]$. Without loss of generality, we presume that $\sum_{w \in A_k^1} w \geq \sum_{w' \in A_k^2} w'$. We write $\mathcal{A} := \{A_1^1, \ldots, A_m^1\}$ to denote the full set of partitions.

3. Recombine: Finally, we merge all of the sub-NPP solutions from $\mathcal{A}$ to obtain a solution for the original NPP. A naive recombination approach would be to simply take the unions $\bigcup_{k=1}^m A_k^1$ and $\bigcup_{k=1}^m A_k^2$. However, this way, the errors accumulate. Instead, we propose here an alternative approach which allows us to recombine the sets in such a way that errors cancel out. For this purpose, we define an auxiliary NPP using the errors of the solutions of the sub-NPPs. Let $E_k := |\sum_{w \in A_k^1} w - \sum_{w \in A_k^2} w| = \sum_{w \in A_k^1} w - \sum_{w \in A_k^2} w$, where we omit the absolute value as we assumed the first sum to be larger. We construct another multiset $W' := \{E_k \mid k \in [m]\}$ and arrive at an auxiliary NPP defined by $\min_{A \subseteq W'} E(W', A)$. We solve this problem using another exchangeable solution method, which is called *RecombinationSolver*. The only requirement for this solver is that it is able to provide a (not necessarily optimal) solution for any given NPP in analogy to the SubSolver. By default, we use SA as the RecombinationSolver. While the resulting partition A' is again not necessarily the optimal solution, we expect that this auxiliary NPP is typically easier to solve than the original NPP because it contains fewer elements and the values of the elements are presumably smaller, as they are the error values of another NPP.
   An estimate $\hat{A}$ for the optimal partition A of the original NPP, Eq. (2), can then be reconstructed based on $\mathcal{A}$ and A' with

$$\hat{A} := \hat{A}(\mathcal{A}, W', A') := \bigcup_{k=1}^{m} A_k^{u(E_k, A')} \tag{7}$$

   with the abbreviation

$$u(E_k, A') := \begin{cases} 1 & \text{if } E_k \in A' \\ 2 & \text{if } E_k \notin A' \end{cases}.$$

   The corresponding objective reads $E' = |\sum_{k=1}^m E_k (-1)^{u(E_k, A')}| = E(W, \hat{A})$. However, it is not guaranteed that $\hat{A}$ is an optimal solution.

Our method is also summarized in Algorithm 1 and we refer to this algorithm in the following.

Although our default solver choices are QA as the SubSolver and SA as the RecombinationSolver, any other combination of (heuristic or deterministic) solvers can also be used. The motivation for our choice is that the sub-problem solutions can tolerate more noise, which can be compensated for the solution of the NPP for W'. Therefore, we can utilize the fast but noisier solution from QA. For the recombination, however, a more accurate solution is needed, as the error directly impacts our final solution. In Sect. 4, we investigate and compare other solver choices with numerical experiments.

Algorithm 1 Proposed algorithm to solve the NPP.

Require: W, j, SubSolver, RecombinationSolver
Ensure: partition A
1: Initialize an empty list $\mathcal{A}$
2: Initialize an empty list W'
3: $\mathcal{W} \leftarrow$ Decompose W according to j ▷ step 1
4: **for** each W_k in $\mathcal{W}$ **do** ▷ step 2
5: $A_k^1 \leftarrow$ Solve the NPP for W_k using the SubSolver ▷ step 2
6: Append A_k^1 to $\mathcal{A}$ ▷ step 2
7: $E_k \leftarrow$ Compute the energy of the solution using Equation (3) ▷ step 2
8: Append E_k to W' ▷ step 2
9: **end for**
10: $A' \leftarrow$ Solve the NPP for W' using the RecombinationSolver ▷ step 3
11: $\hat{A} \leftarrow$ Evaluate Equation (7) using $\mathcal{A}$, W', and A' ▷ step 3
12: **return** $\hat{A}$

Before we turn to the theoretical and numerical analysis of Algorithm 1, we provide a simple example to enable an intuitive understanding. Consider the NPP given by

$$W := \{1, 1, 3, 4, 5, 6\} \text{ and a decomposing vector } j = (1, 1, 1, 2, 2, 2)^\intercal$$

with $m = 2$. In the first step, we perform a decomposition into the multisets $W_1 := \{1, 1, 3\}$ and $W_2 := \{4, 5, 6\}$ according to j and solve an NPP for each of them using QA in the second step. Let us assume this approach yields the partitions

$$A_1^1 := \{3\},\ A_1^2 := \{1, 1\} \text{ for } W_1 \text{ as well as } A_2^1 := \{4, 5\},\ A_2^2 := \{6\} \text{ for } W_2$$

with the objectives $E_1 = 3 - 1 - 1 = 1$ and $E_2 = 4 + 5 - 6 = 3$, respectively. That is, we arrive at the set of sub-partitions $\mathcal{A} = \{A_1^1, A_2^1\}$. There are two distinct ways to merge these sub-partitions to obtain A for the original NPP, either

$$A_1 := A_1^1 \cup A_2^1 = \{3, 4, 5\} \text{ or } A_2 := A_1^1 \cup A_2^2 = \{3, 6\}$$

For the first choice, we find

$$E(W, A_1) = |3 + 4 + 5 - 1 - 1 - 6| = 4 = E_2 + E_1,$$

and for the second,

$$E(W, A_2) = |3 + 6 - 1 - 1 - 4 - 5| = 2 = E_2 - E_1.$$

While for A_1 the errors simply add up, we can use E_2 in the second possibility to make the overall error smaller.

These outcomes align with the potential results of $W' = \{E_1, E_2\}$ when treated as a NPP. We consequently solve this auxiliary NPP in the third step using SA. Let us assume that we obtain the solution $A' = \{E_2\}$ with the objective $E' = E(W, A_2)$. Hence, we find $\hat{A} = A_1^2 \cup A_2^1 = \{1, 1, 4, 5\}$ according to Eq. (7) as an estimate for the

partition A of the original NPP with the objective $E(W, \hat{A}) = E' = E_2 - E_1 = 2$. This is in fact an optimal solution.

3 Conservation of perfect solutions

In line 3 of Algorithm 1, we decompose the original NPP, Eq. (2), into $m \leq n$ sub-NPPs, where the parameter m can be chosen at will. This raises the question of whether there is an optimal choice for m. One of the core criteria for performance is that the auxiliary NPP in line 10 should be as simple to solve as possible, i.e., W' should contain few and small elements. From this observation, one could conclude to choose a small m. However, since we use QA to solve the sub-NPPs, finding a minor embedding becomes increasingly difficult with smaller m (implying larger sub-NPPs) and is ultimately limited by the hardware capacity. Accordingly, we expect that there is a practical sweet spot for m somewhere in-between too large and too small.

Before we study this question from a numerical perspective in Sect. 4, we provide some theoretical insights in the present section. To this end, we consider so-called *perfect solutions*. We differentiate between a "perfect" solution and an "optimal" solution in the following sense. A solution $A \subseteq W$ is called perfect, if and only if $E(W, A) \in \{0, 1\}$. Whereas A is optimal, if and only if no A' exists such that $E(W, A') < E(W, A)$. Note that a perfect solution is always optimal, but not necessarily vice versa. While a perfect solution does not exist for all instances of NPP, an optimal one always does. It is important to note that Algorithm 1 does not necessarily require perfect or optimal solutions of the sub-NPPs to perform effectively, as we discuss in Sect. 4. Instead, the elements in W' merely need to be sufficiently close to each other. The reason for studying perfect solutions instead of optimal ones is that their value is known a priori, which simplifies the investigation.

Analyzing the behavior of QA and SA is challenging since they are both probabilistic heuristics. Results from QA can additionally be influenced by hardware imperfections from a NISQ device. To simplify the theoretical analysis of our decomposition method, we therefore assume in the following that both QA and SA consistently yield optimal solutions. But even under this strong assumption, it is possible that we lose optimal or perfect solutions in the decomposition process. For example, let $W(n) := \{2^t \mid t \in [n]\}$. We observe that the optimal solution has always $E(W(n), \{2^n\}) = 1$, since $2^n - 1 = \frac{2^n-1}{2-1} = \sum_{t\in[n-1]} 2^t$. However, if we consider any decomposing vector j with $m \geq 2$ and at least two elements in each subset W_k, we always obtain a sub-optimal solution.

This means that it is easily possible to construct problems for which Algorithm 1 results in a sub-optimal solution even under perfect conditions. In order to provide a discussion beyond this worst-case scenario, we turn to a probabilistic analysis. For that purpose, we make two simplifying assumptions. First, the elements of W are drawn from a uniform distribution $U[\lambda]$ in $[\lambda]$ for a $\lambda \in \mathbb{N}$. Second, the sum of elements is even (i.e., $E = 0$ for a perfect solution). The expected number of perfect solutions

under these assumptions was estimated by Gent and Walsh (1998) to be

$$|Sol(W)| = \frac{2^n}{\lambda} \tag{8}$$

for a sufficiently large $n = |W|$ and λ, the maximum of the uniform distribution.

For reasons of notational simplicity, let us further assume that $m \mid n$, such that the size of each sub-problem is $|W_k| = \frac{n}{m}$ for the decomposing vector j sampled from Eq. (6). As each element of W is assumed to be drawn from a uniform distribution, the same holds true for the subset W_k. Then, we can apply Eq. (8) on W_k (if we assume $|W_k|$ is even). Consequently, for each $|W_k|$, the expected number of perfect solutions is $|Sol(W_k)| = \frac{2^{n/m}}{\lambda}$, which we transform into

$$m = \frac{n}{\log_2(\lambda \times |Sol(W_k)|)}. \tag{9}$$

The expected number of solutions only appears in the logarithm. This means that, provided λ does not grow exponentially with n, we can decompose the NPP into many sub-NPPs and still expect many perfect solutions.

For example, if we set $|Sol(W_k)| \geq 1$ as a condition, then Eq. (9) turns into

$$m \leq \frac{n}{\log_2(\lambda)}. \tag{10}$$

Exemplarily, in case of $n = 500$ and $\lambda = 5000$, we find $m \leq 40$.

Summarized, we observe that according to Inequality (10), our decomposition method begins to fail as $\lambda \to 2^n$, but, a suitable m is expected to exist for a constant $\lambda \in \mathbb{N}$ and sufficiently large n, such that our method performs well.

4 Numerical experiments

After these theoretical insights into our decomposition strategy within Algorithm 1, we proceed with a numerical study of the overall performance, where we concentrate on three main questions:

1. What is the influence of the sub-NPP size (i.e., the choice of m) on the solution quality?
2. What is the influence of the two solver choices (SubSolver and Recombination-Solver) on the solution quality?
3. How does the solution quality of our problem-oriented decomposition method compare with the solution quality of the problem-agnostic decomposition implemented in QB? We also compare the results with the classical NPP-heuristics LDM, GR, and SA.

Before proceeding with the presentation of our results, we explain the experimental setup.

We randomly generate a set of different NPPs, which serve as our benchmark instances. To this end, we generate 10 different instances for each problem size $n \in \{325, 500, 675, 850, 1025, 1200\}$. For each instance of a given size n, we draw the elements uniformly from the interval $[5n, 10n]$. The problem sizes and element spacing are chosen based on preliminary tests, as they provide a range of differently scaled instances of sufficient complexity for the purposes of the numerical study. All problem instances and solutions are available online (Li et al. 2024b). The number of instances for each n is limited by the limited access to D-Wave machines within this research endeavour. While the previous analysis suggests that an exponential growth leads to harder problems (and thus would present a more stringent test for the algorithms), we observe that the elements of the coefficient matrix Q, Eq. (5), contain the squares of the NPP values. As these values increase, the precision of their mapping onto the quantum device declines, leading to significantly worse results. Classical strategies are also included to highlight the non-triviality of these problems, as well as to showcase the performance difference between classical and hybrid strategies. We therefore adhere to linear growth. In total, we generate a collection of 60 different NPP instances.

We consider six candidate algorithms to solve those problem instances (either directly or as a sub solver):

1. Our proposed problem-specific quantum-classical hybrid strategy, Algorithm 1, with the randomized and balanced decomposition strategy, Eq. (6). Referred to as OUR in the following. Our algorithm is implemented by the authors in Python.
2. A modification of the problem-agnostic quantum-classical hybrid strategy QB as implemented in the *dwave-hybrid* Python package (D-Wave Systems Inc 2024b) under the name *SimplifiedQbsolv*. We exchanged the simple execution of one energy impact decomposition by default of *SimplifiedQbsolv* to a loop of the function with an *convergence* of 3 iterations, a variable *rolling_history*= 0.3 and a *max_subproblem_size*= 40 to run in parallel with the default Tabu Search racing branch. Exit *convergence* of the full routine is 1 and *max_iter* is set to 100 to force exit by convergence.
3. The classical strategy LDM with a standard implementation from the authors in Python.
4. The classical strategy GR with a standard implementation from the authors in Python.
5. The classical strategy SA as implemented in D-Wave Systems Inc (2023b). We use *SimulatedAnnealingSampler* from the *dwave-neal* python package (v0.6.0) with the standard settings (incl. geometric β-schedule and $num\_sweeps = 1000$), except for the parameter num_reads, which is set to 100.
6. The classical Tabu Search strategy (*tabu*) implemented by D-Wave in the function $TabuSampler$ from the Python package *dwave-tabu* (v0.5.0) with all standard settings.

Implementations for the algorithms are available online at Li et al. (2024a). We use the D-Wave Advantage quantum annealer (Advantage_system6.4) as a Quantum Processing Unit (QPU) for OUR and QB. As a classical host, we use a Ubuntu Virtual Machine based on 4 cores Intel Haswell processor. No additional fine-tuning of parameters (including, but not limited to, chain strength and manual embedding) nor any pre-

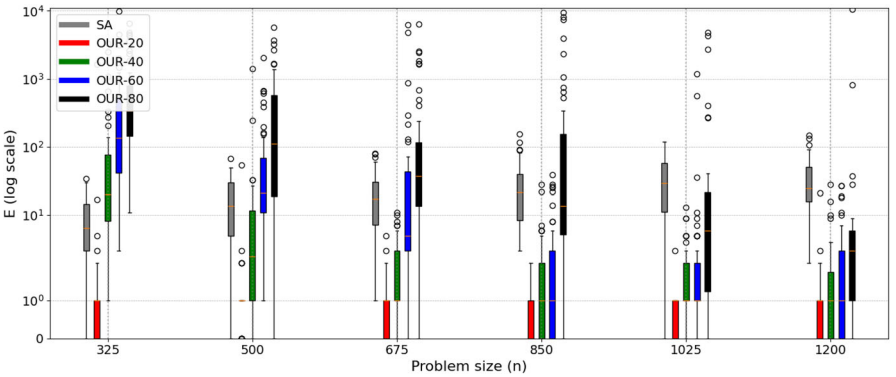

Fig. 1 Experiment 1. We solve 60 randomly generated NPP instances of different sizes n with SA and our proposed method, Algorithm 1, using the D-Wave Advantage quantum annealer as a QPU. We consider different sub-NPP sizes $m = \lfloor n/N \rfloor$ in our decomposition strategy, denoted by OUR-N accordingly. The solution error E, Eq. (3), is a measure for the solution quality and becomes 0 or 1 for a perfect solution. We run each algorithm five times on each instance to take their probabilistic nature into account. The resulting distributions of solution errors are shown as box plots

or post-processing techniques (such as local search) are applied in our experiments. This means that we use the standard settings. In specific, we use the *find_embedding* function provided by D-Wave's *minorminer* Python packages (v0.2.15) with all default parameters to find the embedding for QA, which is then used with *FixedEmbedding-Composite* from *dwave-system* (v1.26.0). The weights are scaled with the default method *auto_scale* from *dwave-system* (v1.26.0). The problems for QA are solved with the default chain strength tuning method of *uniform_torque_compensation*, and 100 shots each with the default anneal time of 20 microseconds. The post-processing method for chain breaking ("unembedding") is *majority_vote*, as in default.

Each problem instance is solved once with the deterministic algorithms LDM and GR. The probabilistic algorithms Algorithm 1 (incl. its variants), QB, and SA are executed 5 times on each instance, i.e., we perform a total of 60 optimization runs for each problem size n for each of these three algorithms.

In our first experiment, we investigate the influence of the sub-NPP size m on the quality of solutions represented by the error E, Eq. (3). For this purpose, we consider five different choices for m depending on the problem size n. Specifically, $m \in \{\lfloor n/20 \rfloor, \lfloor n/40 \rfloor, \lfloor n/60 \rfloor, \lfloor n/80 \rfloor\}$. We write OUR-$N$ with $N \in \{20, 40, 60, 80\}$ to denote the respective choice of $m = \lfloor n/N \rfloor$. For comparison, we also solve the problem instances with SA. The results are shown in Fig. 1.

We find that with OUR-20, a perfect solution is found in almost all cases, as indicated by the error $E \in \{0, 1\}$. This observation aligns with the findings discussed in the previous section. Since a perfect solution is always an optimal one, the error measure E not only represents the difference between the two sets but also serves as an indicator of the distance from the perfect solution. In cases where no perfect solution exists, this measure can be pessimistic. On the other hand, computing the actual optimal solution in every case can be computationally intensive. Since a perfect solution evidently

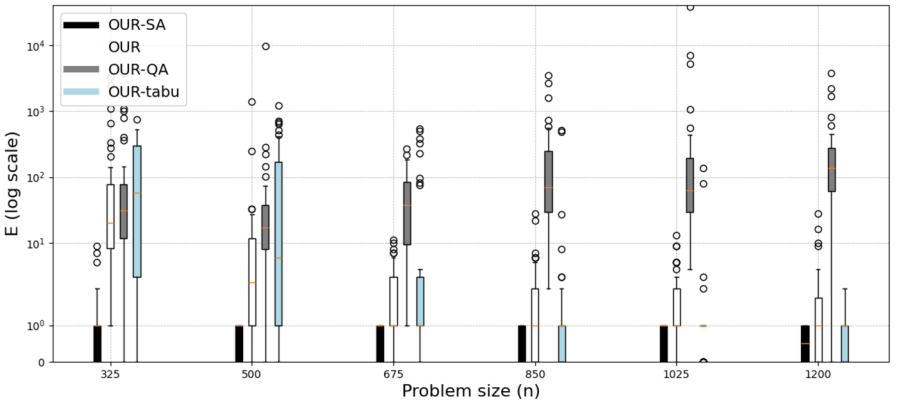

Fig. 2 Experiment 2. Comparing variants of our proposed method, Algorithm 1, that use both QA and SA (OUR), only QA (OUR-QA), only SA (OUR-SA), or only tabu to solve 60 randomly generated NPP instances of different sizes n. For QA, we use the D-Wave Advantage quantum annealer. In analogy to Fig. 1, we show box plots of the resulting distributions of solution errors E, Eq. (3)

exists for many instances, we report the error E instead of the distance to the optimal solution in this context.

For OUR-40, the median error is relatively high for $n \in \{325, 500\}$. In contrast, for $n \geq 675$, we again obtain a perfect solution in many cases. The performance of OUR-60 and OUR-80 is generally poorer, yet it shows an improving trend as the problem size increases. Across all variants, we observe that the solution improves as n increases. Conversely, the behaviour of SA differs. For small instances, SA outperforms OUR-40 to OUR-80. But the performance of SA does not improve as n increases. Consequently, all variants of the proposed algorithm perform better than SA for large problem sizes. This behaviour can be attributed to the characteristic of our decomposition method. The perfect solutions of the auxiliary NPP in the merging step are more likely to be conserved with the larger problems than the smaller ones, as discussed in Sect. 3.

The second question concerns the influence of the choice of SubSolver and RecombinationSolver in Algorithm 1. For this purpose, we consider different solver choices:

- OUR, where we consider our default choice of QA as the SubSolver and SA as the RecombinationSolver.
- OUR-QA, where we consider QA as both the SubSolver and the RecombinationSolver.
- OUR-SA, where we consider SA as both the SubSolver and the RecombinationSolver.
- OUR-tabu, where we conisder Tabu Search as both the SubSolver and the RecombinationSolver

For each variant, we choose $m = \lfloor n/40 \rfloor$ based on the results from the first experiment. Note that this is not the variant that performed best in the previous experiment. The motivation is to allow room for improvement in order to better compare the approaches and to examine the variant with a greater contribution from QA in the case of OUR. The results are shown in Fig. 2.

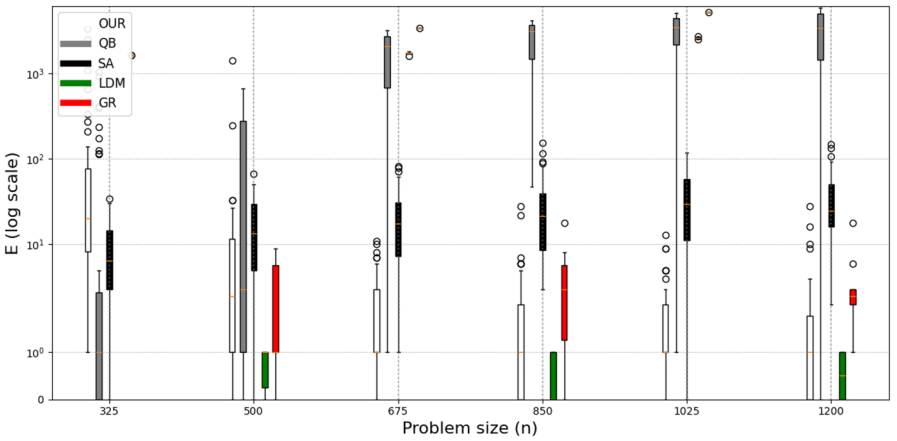

Fig. 3 Experiment 3. Benchmarking the solution quality of our proposed problem-oriented method (OUR) with a problem-agnostic decomposition strategy QB and the classical solvers SA, LDM, and GR for the solution of 60 randomly generated NPP instances of different sizes n. As a QPU, we use the D-Wave Advantage quantum annealer. In analogy to Figs. 1 and 2, we show box plots of the resulting distributions of solution errors E, Eq. (3)

From Fig. 2, it is apparent that OUR-SA outperforms the other three variants. This superior performance is to be attributed to the ability of SA to solve the sub-NPPs and the auxiliary NPP in the merging step with greater accuracy. First of all, an advantage arises since better solutions of sub-NPPs lead to smaller (and thus closer) values in the auxiliary NPP. The quality of the solution of the NPP in the merging step directly influences the overall solution. However, particularly for large instances, the approach can tolerate sub-optimal solutions in the sub-NPPs.

In a third experiment, we benchmark the solution quality and the runtime of our proposed algorithm (using both QA and SA) with QB, SA, LDM, and GR. The results for the solution quality are shown in Fig. 3, where we limit ourselves to OUR with $m = \lfloor n/40 \rfloor$ for the purpose of clarity. The runtime results are reported in Table 1, where we additionally include the results for the different variants of OUR.

According to Fig. 3, we observe that QB still finds acceptable solutions for the problem sizes $n = 325$ and $n = 500$, but its solution quality declines as n increases, indicating that our method outperforms QB for large instances. Additionally, both SA and GR consistently perform worse than OUR, where GR performs better than SA for instances with even problem size. Although LDM always finds perfect solutions for every instance with an even problem size, it performs poorly for others. For problems with an odd size, LDM and GR perform significantly worse than the hybrid OUR in Fig. 3. Furthermore, certain problems, such as $W = \{3000, 3000, 2000, 2000, 2000\}$, are known to be ill-behaved for LDM (Fischetti and Martello 1987). Our results suggest that Algorithm 1 is promising for NPPs where the magnitude of elements is linearly related to the problem size.

In Table 1, we report average runtimes (in seconds) over all runs for the corresponding instance size. For OUR, we provide both the runtime of the QPU as reported by the D-Wave cloud service (D-Wave Systems Inc 2023a) as well as the total end-to-end

runtime (including the QPU runtime and possible queueing times in the cloud service). We use the notation OUR-N for the choice $m = \lfloor n/N \rfloor$ in the same way as in the second experiment. For QB, SA, LDM, and GR, we only list the total end-to-end runtimes.

In addition, we provide a so-called "parallel" end-to-end runtime for OUR based on the presumption that the sub-NPPs from the second step of Algorithm 1 could be solved in parallel with QA (instead of a sequential solution as in our actual implementation). This would require the simultaneous execution of multiple QA instances, for example using multiple quantum annealers or one quantum annealer that has sufficient capacities for multitasking (Huang et al. 2023). Instead of a practical implementation, we are considering this concept here only theoretically and define the parallel end-to-end runtime as

$$t_{\text{para}} = \max_{k \in [m]} t_{2,k} + t_1 + t_3 \tag{11}$$

where t_i with $i \in [3]$ denotes the runtime of the respective step in Algorithm 1 and $t_{2,k}$ denotes the runtime of the QPU within the kth iteration of the second step in Algorithm 1. Similarly, we list a "parallel QPU" runtime, which is defined as

$$t_{\text{para,QPU}} = \max_{k \in [m]} t_{2,k},$$

We note here that the times depend on many factors, which are difficult to standardize.

Independent of this consideration, we find that GR has by far the shortest execution time, with LDM coming in second. Even if we ignore the classical overhead (e.g., for the minor embedding), and only compare with the QPU times, the classical heuristics still outperform OUR by orders of magnitude. However, we find that OUR-20 and OUR-40 have certain time advantages over SA and QB as well in terms of total times, especially for larger n.

Nevertheless, we note that there is plenty of room for improvements in these results, especially for the solution quality. For instance, employing better embeddings specifically tailored for the NPP, such as the one introduced in Lucas (2019), might enhance the results from this study. In contrast, we use the standard *minorminer* package provided by D-Wave with the default settings as specified earlier in this study. Additionally, fine-tuning other parameters could also be beneficial, as demonstrated by Asproni et al. (2020) in their work with QB. However, we aim to provide a first insight into the potential benefits of a problem-oriented approach in this paper without such detailed refinements.

5 Conclusions

In this manuscript, we propose a hybrid quantum-classical decomposition algorithm to solve large NPPs on the D-Wave Advantage quantum annealer. We investigate its performance compared to alternative classic and hybrid algorithms. Based on

Table 1 Experiment 3. Benchmarking the end-to-end runtimes of different variants of our proposed method (OUR) with QB, SA, LDM, and GR for the solution of 60 randomly generated NPP instances of different sizes n. For OUR, we additionally report the QPU runtimes and the hypothetical parallel runtimes based on Eq. (11), respectively. All times are averages (in seconds) over all runs for the corresponding instance size, rounded to four digits. As in Fig. 1, we write OUR-N for the choice $m = \lfloor n/N \rfloor$. As a QPU, we use the D-Wave Advantage quantum annealer

Algorithm \ n		325	500	675	850	1025	1200
OUR-QA	QPU	0.23	0.34	0.46	0.60	0.70	0.85
	total	34.89	42.98	40.90	42.00	45.81	45.98
	QPU (parallel)	0.03	0.03	0.03	0.03	0.03	0.03
	total (parallel)	29.33	33.32	29.76	27.10	28.60	23.87
OUR-SA	total	2.77	4.29	5.72	7.03	8.56	10.08
	total (parallel)	0.45	0.51	0.53	0.54	0.58	0.61
OUR-tabu	total	0.57	0.81	1.13	1.35	1.65	1.92
	total (parallel)	0.12	0.12	0.13	0.12	0.13	0.13
OUR-20	QPU	0.42	0.66	0.87	1.13	1.37	1.58
	total	15.74	18.75	23.94	29.95	35.85	41.84
	QPU (parallel)	0.03	0.03	0.03	0.03	0.03	0.03
	total (parallel)	3.75	2.98	3.54	3.61	4.04	3.85
OUR-40	QPU	0.23	0.34	0.46	0.61	0.70	0.84
	total	36.68	42.26	51.28	42.85	43.79	45.72
	QPU (parallel)	0.03	0.03	0.03	0.03	0.03	0.03
	total (parallel)	29.66	32.62	37.83	25.83	25.49	24.46
OUR-60	QPU	0.16	0.24	0.34	0.43	0.52	0.61
	total	166.21	135.07	136.31	125.22	124.86	139.96
	QPU (parallel)	0.03	0.03	0.03	0.03	0.03	0.03

Table 1 continued

Algorithm \ n		325	500	675	850	1025	1200
	total (parallel)	154.44	121.08	121.88	107.23	106.15	114.65
OUR-80	QPU	0.13	0.19	0.26	0.32	0.39	0.48
	total	384.25	409.91	412.51	394.34	411.09	318.30
	QPU (parallel)	0.03	0.03	0.03	0.03	0.03	0.03
	total (parallel)	373.93	390.39	386.90	375.91	384.94	293.16
QB	QPU	0.04	0.04	0.04	0.04	0.04	0.04
	total	77.60	74.89	101.95	97.24	113.82	158.38
SA		15.76	36.90	75.64	129.77	184.81	268.13
LDM		1.96e−3	3.45e−3	5.64e−3	7.37e−3	0.01	0.01
GR		1.58e−4	2.07e−4	3.95e−4	4.12e−4	4.76e−4	6.50e−4

numerical experiments, our proposed algorithm shows better solution quality than the problem-agnostic hybrid solver QB from D-Wave. This demonstrates the benefits of a special-purpose decomposition method over a general-purpose method.

Compared to well-established classical methods, our algorithm is competitive in terms of solution quality. In particular, for instances with an odd problem size, our proposed method finds significantly better solutions. A runtime advantage of our algorithm compared to classical heuristics is not observable from the results in this study. Although our implementation can be improved to achieve shorter times even with current quantum hardware, the QPU runtime alone is longer in the case of parallel execution than the runtime of the classical heuristics.

This study focuses on the 2-way NPP. However, by introducing additional quadratic constraints to the auxiliary NPP in the last step of our algorithm (quadratic constraints can be included in a QUBO as described in Alidaee et al. (2005)), our method can also be generalized for the solution of multi-way NPPs. Another generalization can be made in terms of problem size. For example, multiple layers of sub-problems can be introduced. Furthermore, this study only examined an uniformly random decomposing vector j. It might be beneficial to investigate other j, such as one which considers the magnitude of the elements. Moreover, the performance of our proposed algorithm for other NP-hard problems can also be investigated by reducing these problems to NPPs.

Throughout the paper, we choose a randomized and balanced decomposition into sub-NPPs. However, alternative decomposition strategies (e.g., evenly distributed element values) can also be used. As a further alternative, different solvers can be employed as SubSolver or RecombinationSolver beyond the presented choices. For example, as a quantum alternative to QA, QUBOs can also be solved on gate-based quantum devices using the Quantum Approximate Optimization Algorithm (QAOA) (Hadfield et al. 2019). Such variations could potentially increase the performance of the algorithm, at least for some types of instances.

In conclusion, we have demonstrated with our proposed method that quantum optimization can benefit from problem-specific decomposition strategies based on expert knowledge. Furthermore, the current research landscape paints a very positive picture. We anticipate that the improvement of quantum hardware will likely enhance the performance of our method both in terms of runtime and solution quality.

Acknowledgements This work was partially funded by the German Federal Ministry of Education and Research (Bundesministerium für Bildung und Forschung) within the project *Rymax One*, and by *The Quantum Initiative Rhineland-Palatinate QUIP*.

Author Contributions All authors contributed in the conceptualization and writing of the presented research.

Funding Open Access funding enabled and organized by Projekt DEAL. This work was partially funded by the German Federal Ministry of Education and Research (Bundesministerium für Bildung und Forschung) within the project *Rymax One*, and by *The Quantum Initiative Rhineland-Palatinate QUIP*.

Data availability Data from numerical experiments are available online at Li et al. (2024b).

Code availability All code is available at Li et al. (2024a).

Declarations

Conflict of interest The authors have no Conflict of interest to declare that are relevant to the content of this article.

Research involving Human Participants and/or Animals Not applicable.

Informed consent Not applicable.

References

Abbas, A., Ambainis, A., Augustino, B., Bärtschi, A., Buhrman, H., Coffrin, C., Zoufal, C.: Quantum optimization: Potential, challenges, and the path forward. *arXiv*, https://doi.org/10.48550/arXiv.2312.02279 (2023)

Albash, T., Lidar, D.A.: Adiabatic quantum computation. Rev. Modern Phys. (2018). https://doi.org/10.1103/RevModPhys.90.015002

Alidaee, B., Glover, F., Kochenberger, G.A., Rego, C.: A new modeling and solution approach for the number partitioning problem. J. Appl. Math. Decision Sci. **2005**(2), 113–121 (2005). https://doi.org/10.1155/JAMDS.2005.113

Asproni, L., Caputo, D., Silva, B., Fazzi, G., Magagnini, M.: Accuracy and minor embedding in subqubo decomposition with fully connected large problems: a case study about the number partitioning problem. Quant. Mach. Intell. (2020). https://doi.org/10.1007/s42484-020-00014-w

Atobe, Y., Tawada, M., Togawa, N.: Hybrid annealing method based on subQUBO model extraction with multiple solution instances. IEEE Trans. Comput. **71**(10), 2606–2619 (2021). https://doi.org/10.1109/TC.2021.3138629

Bass, G., Henderson, M., Heath, J., Dulny, J.: Optimizing the optimizer: decomposition techniques for quantum annealing. Quant. Mach. Intell. (2021). https://doi.org/10.1007/s42484-021-00039-9

Boettcher, S., Mertens, S.: Analysis of the Karmarkar-Karp differencing algorithm. Euro. Phys. J. B **65**(1), 131–140 (2008). https://doi.org/10.1140/epjb/e2008-00320-9

Booth, M., Reinhardt, S.P., Roy, A.: Partitioning optimization problems for hybrid classical/quantum execution (Tech. Rep. No. 14-1006A-A). D-Wave Systems Inc. https://www.dwavesys.com/media/jhlpvult/partitioning_qubos_for_quantum_acceleration-2.pdf (2017)

Boothby, T., King, A.D., Roy, A.: Fast clique minor generation in Chimera qubit connectivity graphs. Quant. Inf. Process. **15**(1), 495–508 (2016). https://doi.org/10.1007/s11128-015-1150-6

Cai, J., Macready, W.G., Roy, A.: A practical heuristic for finding graph minors. *arXiv*, https://doi.org/10.48550/arXiv.1406.2741 (2014)

Coffman, E.G., Johnson, D.S., Lueker, G.S., Shor, P.W.: Probabilistic analysis of packing and related partitioning problems. Stat. Sci. (1993). https://doi.org/10.1214/ss/1177011082

D-Wave Systems Inc. D-Wave hybrid solver service: An overview. https://www.dwavesys.com/media/4bnpi53x/14-1039a-b_d-wave_hybrid_solver_service_an_overview.pdf (Accessed 2022-11-08) (2020)

D-Wave Systems Inc. The advantage system: Performance update. Retrieved from https://www.dwavequantum.com/media/kjtlcemb/14-1054a-a_advantage_system_performance_update.pdf (Accessed 2025-04-02) (2021)

D-Wave Systems Inc. Operation and timing - D-Wave system documentation. https://docs.dwavesys.com/docs/latest/c_qpu_timing.html (Accessed 2024-10-15) (2023a)

D-Wave Systems Inc. Simulated annealing sampler - dwave-neal 0.5.9 documentation. https://docs.ocean.dwavesys.com/projects/neal/en/latest/reference/sampler.html (Accessed 2023-09-16) (2023b)
D-Wave Systems Inc. D-Wave advantage. https://www.dwavesys.com/solutions-and-products/systems (Accessed 2024-02-04) (2024a)
D-Wave Systems Inc. D-Wave hybrid. https://github.com/dwavesystems/dwave-hybrid (Accessed 2024-02-12) (2024b)
Fischetti, M., Martello, S.: Worst-case analysis of the differencing method for the partition problem. Math. Program. **37**(1), 117–120 (1987). https://doi.org/10.1007/BF02591687
Gent, I.P., Walsh, T.: Analysis of Heuristics for number partitioning. Comput. Intell. **14**(3), 430–451 (1998). https://doi.org/10.1111/0824-7935.00069
Glover, F., Kochenberger, G., Hennig, R., Du, Yu.: Quantum bridge analytics I: a tutorial on formulating and using QUBO models. Ann. Oper. Res. **314**(1), 141–183 (2022). https://doi.org/10.1007/s10479-022-04634-2
Hadfield, S., Wang, Z., O'Gorman, B., Rieffel, E.G., Venturelli, D., Biswas, R.: From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. Algorithms **12**(2), 34 (2019). https://doi.org/10.3390/a12020034
Horowitz, E., Sahni, S.: Computing partitions with applications to the knapsack problem. J. ACM **10**(1145/321812), 321823 (1974)
Huang, T., Zhu, Y., Goh, R., Luo, T.: When quantum annealing meets multitasking: potentials, challenges and opportunities. Array **17**, 100282 (2023). https://doi.org/10.1016/j.array.2023.100282
Karmarkar, N., Karp, R.M.: The differencing method of set partitioning (Tech. Rep. No. UCB/CSD-83-113). EECS Department, University of California, Berkeley. http://www2.eecs.berkeley.edu/Pubs/TechRpts/1983/6353.html (1983)
Karp, R.M.: Reducibility among Combinatorial Problems. In: Miller, R.E., Thatcher, J.W., Bohlinger, J.D. (eds.) Complexity of Computer Computations, pp. 85–103. Springer US, Boston, MA (1972). https://doi.org/10.1007/978-1-4684-2001-2_9
Kochenberger, G., Hao, J.-K., Glover, F., Lewis, M., Lü, Z., Wang, H., Wang, Y.: The unconstrained binary quadratic programming problem: a survey. J. Combinat. Optimiz. **28**(1), 58–81 (2014). https://doi.org/10.1007/s10878-014-9734-0
Korf, R.E.: A complete anytime algorithm for number partitioning. Artif. Intell. **106**(2), 181–203 (1998). https://doi.org/10.1016/S0004-3702(98)00086-1
Krasecki, V.K., Sharma, A., Cavell, A.C., Forman, C., Guo, S.Y., Jensen, E.T., Smith, M.A., Czerwinski, R., Friederich, P., Hickman, R.J., Gianneschi, N., Aspuru-Guzik, A., Cronin, L., Goldsmith, R.H.: The role of experimental noise in a hybrid classical-molecular computer to solve combinatorial optimization problems. ACS Central Sci. **9**(7), 1453–1465 (2023). https://doi.org/10.1021/acscentsci.3c00515
Li, Z., Seidel, T., Leib, D., Bortz, M., Heese, R. Code for "efficient solution of the number partitioning problem on a quantum annealer: A hybrid quantum-classical decomposition approach". Retrieved from https://github.com/Quantum-NPP/Quantum-NPP (Published 2025-02-07) (2024a)
Li, Z., Seidel, T., Leib, D., Bortz, M., Heese, R.: Data sets for "efficient solution of the number partitioning problem on a quantum annealer: A hybrid quantum-classical decomposition approach". https://doi.org/10.5281/zenodo.13929258 (Published 2024-10-14) (2024b)
Lucas, A.: Hard combinatorial problems and minor embeddings on lattice graphs. Quant. Inf. Process. (2019). https://doi.org/10.1007/s11128-019-2323-5
Matoušek, J., Thomas, R.: On the complexity of finding ISO- and other morphisms for partial k-trees. Discr. Math. **108**(1–3), 343–364 (1992). https://doi.org/10.1016/0012-365X(92)90687-B
Mertens, S.: The Easiest Hard Problem: Number Partitioning. Computational Complexity and Statistical Physics (2005). https://doi.org/10.1093/oso/9780195177374.003.0012
Michiels, W., Korst, J., Aarts, E., Leeuwen, J.: Performance ratios of the Karmarkar-Karp differencing method. J. Combinat. Optimiz. **13**(1), 19–32 (2006). https://doi.org/10.1007/s10878-006-9010-z
Okada, S., Ohzeki, M., Terabe, M., Taguchi, S.: Improving solutions by embedding larger subproblems in a D-Wave quantum annealer. Sci. Rep. (2019). https://doi.org/10.1038/s41598-018-38388-4
Osaba, E., Villar-Rodriguez, E., Oregi, I., Moreno-Fernandez-de Leceta, A.: Hybrid quantum computing - tabu search algorithm for partitioning problems: Preliminary study on the traveling salesman problem. IEEE Congress on Evolutionary Computation (2021). https://doi.org/10.1109/CEC45853.2021.9504923
Ottaviani, D., Amendola, A.: Low rank non-negative matrix factorization with d-wave 2000q. https://doi.org/10.48550/arXiv.1808.08721 (2018)

Pelofske, E., Hahn, G., Djidjev, H.: solving large maximum clique problems on a quantum annealer. In: Feld, S., Linnhoff-Popien, C. (eds.) Quantum Technology and Optimization Problems: First International Workshop, QTOP 2019, Munich, Germany, March 18, 2019, Proceedings, pp. 123–135. Springer International Publishing, Cham (2019). https://doi.org/10.1007/978-3-030-14082-3_11

Pelofske, E., Hahn, G., Djidjev, H.: Decomposition algorithms for Solving NP-hard problems on a quantum annealer. J. Signal Process. Syst. **93**(4), 405–420 (2021). https://doi.org/10.1007/s11265-020-01550-1

Pelofske, E., Hahn, G., O'Malley, D., Djidjev, H.N., Alexandrov, B.S.: Quantum annealing algorithms for Boolean tensor networks. Sci. Rep. (2022). https://doi.org/10.1038/s41598-022-12611-9

Pinedo, M.L.: Scheduling: Theory, Algorithms, and Systems. Springer US, Boston, MA (2012). https://doi.org/10.1007/978-1-4614-2361-4

Preskill, J.: Quantum computing in the NISQ era and beyond. Quantum **2**, 79 (2018). https://doi.org/10.22331/q-2018-08-06-79

Rajak, A., Suzuki, S., Dutta, A., Chakrabarti, B.K.: Quantum annealing: an overview. Philos. Trans. Royal Soc. A: Math., Phys. Eng. Sci. (2023). https://doi.org/10.1098/rsta.2021.0417

Raymond, J., Stevanovic, R., Bernoudy, W., Boothby, K., McGeoch, C.C., Berkley, A.J., Farré, P., Pasvolsky, J., King, A.D.: Hybrid quantum annealing for larger-than-QPU lattice-structured problems. ACM Trans. Quant. Comput. **4**(3), 1–30 (2023). https://doi.org/10.1145/3579368

Robertson, N., Seymour, P.D.: Graph minors .XIII. The disjoint paths problem. J. Combinat. Theory, Series B **63**(1), 65–110 (1995). https://doi.org/10.1006/jctb.1995.1006

Schroeppel, R., Shamir, A.: A $T = O(2^{n/2})$, $S = O(2^{n/4})$ algorithm for certain NP-complete problems. SIAM J. Comput. **10**(3), 456–464 (1981). https://doi.org/10.1137/0210033

Shaydulin, R., Ushijima-Mwesigwa, H., Negre, C.F.A., Safro, I., Mniszewski, S.M., Alexeev, Y.: A hybrid approach for solving optimization problems on small quantum computers. Computer **52**(6), 18–26 (2019). https://doi.org/10.1109/MC.2019.2908942

Yulianti, L.P., Surendro, K.: Implementation of quantum annealing: a systematic review. IEEE Access **10**, 73156–73177 (2022). https://doi.org/10.1109/ACCESS.2022.3188117