technologies

MDPI

Article

Dual-Matrix Domain Wall: A Novel Technique for Generating Permutations by QUBO and Ising Models with Quadratic Sizes

Koji Nakano [1,*], Shunsuke Tsukiyama [1], Yasuaki Ito [1], Takashi Yazane [2], Junko Yano [2], Takumi Kato [2], Shiro Ozaki [2], Rie Mori [2] and Ryota Katsuki [2]

[1] Graduate School of Advanced Science and Engineering, Hiroshima University, Kagamiyama 1-4-1, Higashihiroshima 739-8527, Hiroshima, Japan
[2] Research and Development Headquarters, NTT DATA Group Corporation, Toyosu Center Bldg, Annex, 3-9, Toyosu 3-chome, Koto-ku 135-8671, Tokyo, Japan
* Correspondence: nakano@cs.hiroshima-u.ac.jp

Abstract: The Ising model is defined by an objective function using a quadratic formula of qubit variables. The problem of an Ising model aims to determine the qubit values of the variables that minimize the objective function, and many optimization problems can be reduced to this problem. In this paper, we focus on optimization problems related to permutations, where the goal is to find the optimal permutation out of the $n!$ possible permutations of n elements. To represent these problems as Ising models, a commonly employed approach is to use a kernel that applies one-hot encoding to find any one of the $n!$ permutations as the optimal solution. However, this kernel contains a large number of quadratic terms and high absolute coefficient values. The main contribution of this paper is the introduction of a novel permutation encoding technique called the dual-matrix domain wall, which significantly reduces the number of quadratic terms and the maximum absolute coefficient values in the kernel. Surprisingly, our dual-matrix domain-wall encoding reduces the quadratic term count and maximum absolute coefficient values from $n^3 - n^2$ and $2n - 4$ to $6n^2 - 12n + 4$ and 2, respectively. We also demonstrate the applicability of our encoding technique to partial permutations and Quadratic Unconstrained Binary Optimization (QUBO) models. Furthermore, we discuss a family of permutation problems that can be efficiently implemented using Ising/QUBO models with our dual-matrix domain-wall encoding.

Keywords: quantum computing; combinatorial optimization; traveling salesman problem; graph isomorphism problem

check for updates

Citation: Nakano, K.; Tsukiyama, S.; Ito, Y.; Yazane, T.; Yano, J.; Kato, T.; Ozaki, S.; Mori, R.; Katsuki, R. Dual-Matrix Domain Wall: A Novel Technique for Generating Permutations by QUBO and Ising Models with Quadratic Sizes. *Technologies* **2023**, *11*, 143. https://doi.org/10.3390/technologies11050143

Academic Editors: Dongran Song and Pedro Antonio Gutiérrez

Received: 4 August 2023
Revised: 1 September 2023
Accepted: 12 October 2023
Published: 17 October 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

A Binary Quadratic Model (BQM) [1] is defined by an objective function that includes a quadratic formula with multiple variables. The problem associated with a BQM is to find the values of these variables that minimize the resulting value of the quadratic formula. A BQM is referred to as *a Quadratic Unconstrained Binary Optimization (QUBO)* [2] model when the variables are restricted to binary values, i.e., they can only take *bit (or binary)* values in $\{0, 1\}$. On the other hand, if the variables can only take *qubit (or spin)* values in $\{-1, +1\}$, the model is called an Ising model. It is worth noting that QUBO and Ising models can be converted into each other interchangeably [1,3].

Since optimization problems such as the traveling salesman problem; scheduling problems; and various graph problems, including max cut, maximum independent set, and graph isomorphism, can be transformed into QUBO/Ising models [4], there has been significant research dedicated to finding efficient algorithms, hardware, and systems to solve them. However, solving optimization problems for QUBO/Ising models is known to be NP-hard. This means that unless $P = NP$, it is not possible to design a polynomial time algorithm using classical computers with digital circuit devices of polynomial size. In the quest for solutions, researchers have explored the potential of ideal quantum annealers based on quantum

mechanics, which could potentially find optimal solutions for large Ising models within a reasonable time frame [5]. Unfortunately, the current quantum annealers available are not yet powerful enough to tackle such problems effectively. The number of qubits is limited, and the presence of undesirable flux noise significantly reduces the probability of finding optimal solutions [6]. Hence, as an alternative to an ideal quantum annealer, BQM solvers on various non-quantum computing platforms, such as ASICs [7,8], FPGAs [9–12], GPUs [13–16], and optimal fibers [17,18], have been proposed. Further, D-Wave Systems released a hybrid BQM solver [19] that uses both a classical computer and a quantum annealer to find solutions for large BQMs with up to 1,000,000-node complete graphs.

In this paper, our primary focus is on the size and resolution of QUBO/Ising models, which are represented by the quadratic term count and the maximum absolute value of the coefficients, respectively. These metrics serve as important indicators for assessing the models. While it is not always the case, QUBO/Ising models with smaller sizes and resolutions are generally considered more desirable. This is particularly true when considering the limitations of quantum annealers, which have restricted size and resolution capabilities. Therefore, it becomes crucial to ensure that the Ising models embedded in quantum annealers possess small sizes and low resolutions. Additionally, even when these models are processed by classical digital computers, the required memory size to store QUBO/Ising models is proportional to the quadratic term count, and a higher resolution requires a larger word size of memory. With this perspective in mind, this paper places significant emphasis on the size and resolution of QUBO/Ising models. We provide precise evaluations of these metrics, offering valuable insights into their characteristics.

QUBO/Ising models that are converted from permutation-based combinatorial optimization problems for n elements should have a kernel capable of generating any one of the $n!$ possible permutations as the optimal solution. To achieve this, a common approach is to use the one-hot encoding of permutations, which involves a *bit*/*qubit* matrix of size $n \times n$. In this encoding, each row i ($0 \leq i \leq n-1$) of the matrix represents the i-th number as a one-hot vector, where exactly one element is set to $1/+1$, and its position corresponds to the number it represents. The reader should refer to Figure 1, which illustrates a 4×4 matrix representing the permutation $[1, 3, 2, 0]$. In order to generate any one of these matrices as the optimal solution, QUBO/Ising models require kernels that have optimal solutions if and only if each row and each column contains exactly one $1/+1$. For instance, QUBO models utilizing this kernel have been proposed for addressing Hamiltonian cycle/path problems and the graph isomorphism problem, as demonstrated in [4]. Similarly, models for the traveling salesman problem (TSP) and the quadratic assignment problem (QAP) were introduced in [20,21]. However, these kernels typically involve $n^3 - n^2$ quadratic terms. Additionally, the kernel of the Ising model features a non-constant large coefficient of $2n - 4$. Recently, a permutation generation technique using domain-wall encoding has been introduced [22]. This technique employs a matrix of size $n \times (n-1)$, where each row i ($0 \leq i \leq n-1$) stores the i-th number as a domain-wall vector [23–25]. By utilizing this technique, the number of quadratic terms is reduced to $\frac{1}{2}n^3 - \frac{3}{2}n$, which is half the number of quadratic terms in QUBO/Ising kernels obtained through conventional one-hot encoding. This technique still involves a cubic number of quadratic terms. Moreover, QUBO/Ising models using this domain-wall encoding require coefficients with large absolute values, namely $2n - 3/n - 1$, respectively.

The main contribution of this paper is to introduce a novel permutation encoding technique called *the dual-matrix domain wall*, which uses two matrices A and B to store dual permutations. This new technique can significantly reduce the number of quadratic terms and the maximum absolute coefficient values in the resulting Ising kernel. The QUBO/Ising kernel obtained by this technique has only $6n^2 - 12n + 4$ quadratic terms. Also, the maximum absolute value of the coefficients of the Ising kernel is just 2.

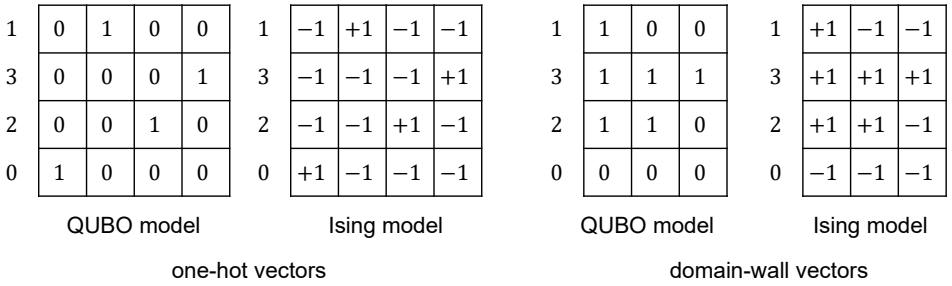

Figure 1. Representation of a permutation $[1, 3, 2, 0]$ by a 4×4 matrix with one-hot vectors and a 4×3 matrix with domain-wall vectors

We also discuss partial-permutation generation by QUBO/Ising kernels, which is a sequence of m numbers selected from n numbers $0, 1, \ldots, n-1$ without repetition. It is not difficult to apply one-hot encoding to generate a partial permutation [26,27]. For partial-permutation generation, one uses an $m \times n$ matrix such that each i-row ($0 \leq i \leq n-1$) stores the i-th selected number as a one-hot vector. QUBO/Ising kernels that use one-hot encoding to generate any such matrices need $\frac{1}{2}m^2n + \frac{1}{2}mn^2 - mn$ quadratic terms. We show that we can apply the dual-matrix domain-wall technique for partial-permutation generation, and the resulting QUBO/Ising models have only $6mn - 6m - 6n + 4$ quadratic terms.

Moreover, we introduce a generic problem called the particle placement problem (PPP), which aims to optimize the placement of m particles in n locations ($m \leq n$) with no collision. In other words, the problem is to find an optimal permutation of m numbers selected from n numbers. In this study, we demonstrate that the PPP can be efficiently reduced to a QUBO/Ising model using permutation-generating kernels. Additionally, we establish that several permutation-based combinatorial optimization problems, such as the quadratic assignment problem (QAP), the traveling salesman problem (TSP), the sub-graph isomorphism problem, and the maximum weight matching problem, can also be reduced to the PPP. Thus, these problems can be converted to equivalent QUBO/Ising models using permutation-generating kernels.

Finally, we conducted an evaluation of the cells required to embed Ising kernels on a D-Wave quantum annealer, specifically the Advantage 4.1. The objective was twofold: first, to assess the feasibility of utilizing the dual-matrix domain-wall technique on presently accessible quantum annealers, and second, to examine the influence of the number of quadratic terms in Ising models on the cell requirements.

This paper is organized as follows. In Section 2, we begin by providing a formal definition of QUBO and Ising models. We also establish the relationship between quantum annealers and Ising models. Additionally, we explore different encoding techniques such as one-hot, zero-one-hot, and domain-wall encoding for representing numbers in QUBO and Ising models. Section 3 reviews the conventional one-hot encoding technique used to represent permutations of n numbers. Furthermore, we introduce the all-different domain-wall encoding technique, which effectively reduces the number of quadratic terms by half [22]. However, both of these techniques require a cubic number of quadratic terms. In Section 4, we present a novel encoding technique called *dual-matrix domain-wall encoding*. This technique enables the generation of permutations using a QUBO/Ising kernel with only a quadratic number of quadratic terms. Section 5 generalizes the concept of permutations to partial permutations, which represent a partial permutation of m numbers selected from a set of n numbers without repetition. We demonstrate how QUBO/Ising kernels can generate partial permutations using both the conventional one-hot encoding technique and our dual-matrix domain-wall encoding technique. In Section 6, we extend the dual-matrix domain-wall technique by incorporating a matrix called a one-hot matrix that stores a partial permutation as a one-hot encoding. We introduce the particle placement problem (PPP) in Section 7, highlighting its ability to reduce many permutation-based combinatorial optimization problems. We evaluate the number of quadratic terms in the resulting QUBO/Ising models obtained through reduction via the PPP for each problem.

Section 8 presents an analysis of the number of cells required to embed Ising kernels for generating permutations in a D-Wave quantum annealer. Finally, Section 9 concludes our work.

2. Preliminaries

The main objective of this section is to introduce the concepts of QUBO and Ising models, as well as the fundamental aspects of their design. To begin with, we provide a formal definition of Ising and QUBO models. Subsequently, we establish the connections between Ising models and quantum annealers, offering valuable insights to the reader. Lastly, we showcase the applications of QUBO and Ising models in generating one-hot, zero-one-hot, and domain-wall vectors.

2.1. QUBO and Ising Models

A Binary Quadratic Model (BQM) [1] is defined as an objective function with a quadratic formula involving multiple variables. The goal of a BQM is to determine the variable values that minimize the resulting value of the quadratic formula.

A model is referred to as *a Quadratic Unconstrained Binary Optimization (QUBO) model* [2] if the variables can take *bit (or binary)* values of 0 or 1. Specifically, let $X = (x_i)$ $(0 \leq i \leq n-1)$ represent an n-bit vector of binary variables. A QUBO model with X can be defined using an upper triangular weight matrix $W = (W_{i,j})$ $(0 \leq i \leq j \leq n-1)$. The objective of the QUBO problem is to find the binary values of X that minimize the energy $E(X)$, which is defined by the following quadratic formula:

$$E(X) = \sum_{i=0}^{n-1} \sum_{j=i+1}^{n-1} W_{i,j} x_i x_j + \sum_{i=0}^{n-1} W_{i,i} x_i + C, \tag{1}$$

$$= \sum_{i=0}^{n-1} \sum_{j=i}^{n-1} W_{i,j} x_i x_j + C. \tag{2}$$

Here, C is a constant called the *offset*. Note that Equations (1) and (2) are equivalent because $x_i^2 = x_i$ always holds. While most papers use the energy $E(X)$ in Equation (2) with no offset C, this paper adopts the energy $E(X)$ defined by Equation (1) to distinguish linear and quadratic terms with coefficients $W_{i,i}$ $(0 \leq i \leq n-1)$ and $W_{i,j}$ $(0 \leq i < j \leq n-1)$, respectively.

A BQM is called *an Ising model* if variables take *qubit (or spin)* values of -1 or $+1$. An Ising model with an n-qubit vector $S = (s_i)$ $(0 \leq i \leq n-1)$ is defined by quadratic term coefficients $J = (J_{i,j})$ $(0 \leq i < j \leq n-1)$ called *interactions* and linear term coefficients $h = (h_i)$ $(0 \leq i \leq n-1)$ called *biases*. The Ising problem aims to find the qubit values of S that minimize *the Hamiltonian* $H(S)$, which is defined by the following quadratic formula:

$$H(S) = \sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} J_{i,j} s_i s_j + \sum_{i=0}^{n-1} h_i s_i + C. \tag{3}$$

We make the assumption that all coefficients in the linear and quadratic terms of QUBO/Ising models are integers, unless otherwise specified. Using integers with large absolute values can lead to discrete values below the effective resolution, so it is important to keep the maximum absolute value of all coefficients as small as possible. To achieve this, we design QUBO/Ising models that have no common factor in all coefficients. This allows us to reduce the coefficients by dividing the common factor without affecting the optimal solutions. It is worth mentioning that the offset C does not influence the optimal solution and can be disregarded. Consequently, it can take a non-integer value.

We define the number of non-zero elements in $W_{i,i}/h_i$ $(0 \leq i \leq n-1)$ as *the linear term count* and that in $W_{i,j}/J_{i,j}$ $(0 \leq i < j \leq n-1)$ as *the quadratic term count* of a QUBO/Ising model. Generally, having smaller term counts is advantageous as it helps reduce the

hardware resource usage of quantum computers when solving problems of QUBO/Ising models. In particular, the quadratic term count has a significant impact on the hardware resource usage of quantum annealers when solving Ising problems. By minimizing the quadratic term counts, we can effectively reduce the amount of resources needed for implementing and solving QUBO/Ising models on quantum hardware. This optimization is essential for improving the efficiency and performance of quantum computing systems.

It is easy to see that QUBO and Ising models can be equivalently converted to each other [3]. Both QUBO and Ising models, excluding the offset C, can be represented as weighted undirected graphs with n nodes $0, 1, \ldots, n-1$ such that $W_{i,i}/h_i$ is the weight of node i and $W_{i,j}/J_{i,j}$ is the weight of edge (i, j) of the graph. Figure 2 shows a graph corresponding to a QUBO model with the the following energy:

$$\begin{aligned} E(X) = {} & 2x_0x_1 - x_0x_3 - 2x_0x_4 - x_1x_2 + x_1x_5 + 3x_2x_5 - 2x_3x_4 - 2x_4x_5 \\ & - 2x_0 - x_1 + 2x_2 + 4x_5 \end{aligned} \tag{4}$$

The figure also illustrates the equivalent Ising model with the following Hamiltonian:

$$\begin{aligned} H(S) = {} & 2s_0s_1 - s_0s_3 - 2s_0s_4 - s_1s_2 + s_1s_5 + 3s_2s_5 - 2s_3s_4 - 2s_4s_5 \\ & - 5s_0 + 6s_2 - 3s_3 - 6s_4 + 10s_5 \end{aligned} \tag{5}$$

We omit terms with zero coefficients in the formulas and edges with zero weights in the graphs. As QUBO and Ising models are represented as graphs, we use graph theory terminologies such as the degree of a node (i.e., the number of edges connecting to a node) and the diameter (i.e., the largest shortest path over all pairs of nodes). The QUBO and Ising models have optimal solutions $X = [1, 0, 0, 1, 1, 0]$ with energy $E(X) = -7$ and $S = [+1, -1, -1, +1, +1, -1]$ with Hamiltonian $H(S) = -32$, respectively. These models are equivalent, because $4E(X) = H(S) + 4$ always holds for all X and S satisfying $s_i = 2x_i - 1$ for all i ($0 \leq i \leq n-1$).

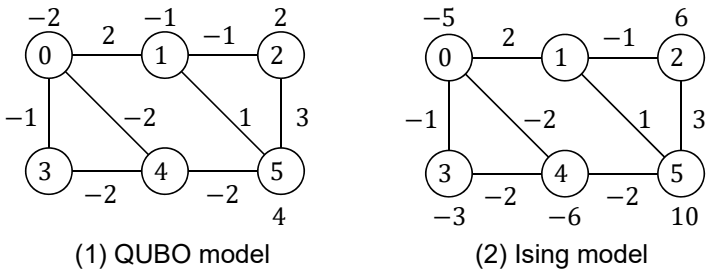

Figure 2. Equivalent QUBO and Ising models.

As many optimization problems can be reduced to QUBO/Ising models [4], there has been a significant effort by researchers to find effective algorithms, hardware, and systems to solve them. Digital computers and devices can directly operate $0/1$ bits, and users/developers can handle them more easily than $-1/+1$ qubits. As a result, QUBO models are more frequently used than Ising models, and QUBO solvers have been developed by many researchers. On the other hand, quantum annealers based on quantum mechanics can directly operate $-1/+1$ qubits in Ising models. Therefore, for solving QUBO problems on quantum annealers, they are converted to equivalent Ising models. By this preprocessing step, quantum annealers can be used as QUBO solvers. However, to maximize the performance of QUBO solvers, we may design Ising models without using QUBO–Ising conversion.

When a QUBO/Ising model for solving a specific permutation-based combinatorial optimization problem is designed, this involves creating a sub-model that generates any one of all possible permutations as the optimal solution. This sub-model is referred to as *the QUBO/Ising kernel*. The main focus of this paper is the design of QUBO/Ising kernels for generating permutations. For instance, we will present QUBO/Ising kernels

for generating an $n \times n$-matrix that stores one-hot vectors representing a permutation, as shown in Figure 1.

2.2. Quantum Annealers and Ising Models

D-Wave Systems developed a quantum annealer called D-Wave 2000Q [28], which was based on quantum mechanics. It served as a solver for Ising models using a 2048-node Chimera graph. Subsequently, D-Wave Systems released a more advanced quantum annealer known as D-Wave Advantage [29], which was capable of handling Ising models with a larger 5760-node Pegasus graph [30]. Specifically, the D-Wave Advantage quantum annealer comprises 5760 cells interconnected according to the topology of a 5760-node Pegasus graph. The cell biases and interaction strengths are programmable, allowing for the acquisition of optimal or approximate solutions to the corresponding Ising models through quantum annealing.

While a quantum annealer is designed to solve Ising models with a specific graph topology, it is possible to solve Ising models with different topologies through a process known as minor embedding. Minor embedding involves mapping the problem graph (with a different topology) onto the physical graph of the quantum annealer, effectively embedding the problem into the hardware. This allows the quantum annealer to solve Ising models that may not directly match its native graph topology. Minor embedding is a technique used to leverage the capabilities of quantum annealers for a broader range of Ising models. We explain the idea of minor embedding using Figure 3. Suppose that we need to solve an Ising problem with the six-node graph in Figure 2 on a quantum annealer with an eight-cell grid topology, as shown in Figure 3. We arrange node 0 (or qubit s_0) in Figure 2 to two cells 0 (or qubit s_0) and $0'$ (or qubit $s_{0'}$) in Figure 3. We add a quadratic term $-Ps_0s_{0'}$ so that optimal solutions satisfy $s_0 = s_{0'}$, where P is a constant number large enough to guarantee it. By virtue of this embedding, s_0 in Figure 2 can be simulated by s_0 and $s_{0'}$ combined in Figure 3. The reader should have no difficulty in confirming that the Ising model in Figure 2 can be solved by the quantum annealer in Figure 3.

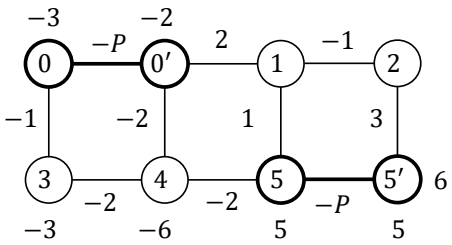

Figure 3. Example of minor embedding for the Ising model of Figure 2 to the quantum annealer.

More than two cells in the quantum annealer may be arranged to simulate a node of the Ising model. D-Wave Systems calls a set of cells arranged to form a qubit *a chain*, and the value of P the *chain strength* [31]. To obtain an optimal or approximate solution of an Ising model by the D-Wave Advantage quantum annealer, one needs to find its minor embedding in a 5760-node Pegasus graph. If an Ising model is a dense graph with a large degree, the chains will be large. For example, if an Ising model is a complete graph, only 177 nodes can be embedded in the D-Wave Advantage quantum annealer [29]. Hence, the size of an Ising model that can be solved by the D-Wave Advantage quantum annealer is limited. In particular, the quadratic term count of an Ising model impacts the resource usage of cells. Thus, it is quite important to minimize the quadratic term count when one designs Ising models.

The D-Wave Advantage quantum annealer [29] can take real numbers for the interaction $J_{i,j}$ and bias h_i of Ising models. The ranges of $J_{i,j}$ and h_i are limited to $[-1.0, +1.0]$ and $[-4.0, +4.0]$, respectively. Although they take any real number in these ranges, they are operated as analog values, and the effective resolution is limited. The resolution is only five to six bits, and two values with a difference less than $\frac{1}{2^6} = 0.015$ may not be distinguished

due to the flux noise. Recall that we assume the interaction $J_{i,j}$ and bias h_i of Ising models are integers. When we load such Ising models onto the D-Wave Advantage quantum annealer, these integer values are automatically reduced to fit in the ranges $[-1.0, +1.0]$ and $[-4.0, +4.0]$. For example, if $J_{i,j}$ takes integers in the range $[-100, +100]$, then they will be divided by 100 to fit in the range $[-1.0, +1.0]$, and the required resolution 0.01 is smaller than the effective resolution of the D-Wave Advantage quantum annealer. Hence, optimal solutions of the Ising model cannot be expected. Since the resolution is limited, the maximum absolute values of interactions and biases of Ising models should be as small as possible to obtain optimal solutions with a higher probability by quantum annealers.

QUBO/Ising models with larger diameters may be solvable using a heuristic algorithm in parallel. They may be split into many disjoint sub-models, and parallel divide-and-conquer techniques can be applied to such models. Heuristic algorithms to improve the solutions of sub-models can be executed in parallel, and better approximate solutions can be obtained more quickly. On the other hand, smaller-diameter models cannot be split into many disjoint sub-models, and it is difficult to apply parallel heuristic techniques. Thus, larger-diameter models are preferable to solve the problems in parallel.

2.3. QUBO/Ising Models for One-Hot/Zero-One-Hot/Domain-Wall Vectors

One-hot encoding has often been used to represent integers in QUBO and Ising models. A k-bit/qubit vector is *a one-hot vector* if it has exactly one $1/+1$, and the remaining $k-1$ bits/qubits take $0/-1$. It represents an integer i ($0 \leq i \leq k-1$) if and only if the i-th bit/qubit is $1/+1$. For technical reasons, we introduce *the zero-one-hot vector*, which can take a bit/qubit vector with all $0/-1$s in addition to one-hot vectors. Such vectors with all $0/-1$s represent a special value φ associated with "undefined" or "N/A". Table 1 shows four-bit/qubit one-hot/zero-one-hot vectors with corresponding integers or φ.

Table 1. One-hot/zero-one-hot/domain-wall vectors with $k = 4$ bits.

One-Hot			Zero-One-Hot			Domain-Wall		
	Bits	**Qubits**		**Bits**	**Qubits**		**Bits**	**Qubits**
0	[1,0,0,0]	[+1,−1,−1,−1]	0	[1,0,0,0]	[+1,−1,−1,−1]	0	[0,0,0,0]	[−1,−1,−1,−1]
1	[0,1,0,0]	[−1,+1,−1,−1]	1	[0,1,0,0]	[−1,+1,−1,−1]	1	[1,0,0,0]	[+1,−1,−1,−1]
2	[0,0,1,0]	[−1,−1,+1,−1]	2	[0,0,1,0]	[−1,−1,+1,−1]	2	[1,1,0,0]	[+1,+1,−1,−1]
3	[0,0,0,1]	[−1,−1,−1,+1]	3	[0,0,0,1]	[−1,−1,−1,+1]	3	[1,1,1,0]	[+1,+1,+1,−1]
			φ	[0,0,0,0]	[−1,−1,−1,−1]	4	[1,1,1,1]	[+1,+1,+1,+1]

Domain-wall encoding [22–25] has also been used to represent integers. A k-bit/qubit vector is *a domain-wall vector* if it consists of consecutive $1/+1$s following consecutive $0/-1$s. It represents the integer i ($0 \leq i \leq k$) if it contains i consecutive 1s. Therefore, it can represent $k+1$ integers ranging from 0 to k. Table 1 illustrates four-bit/qubit domain-wall vectors that represent integers from 0 to 4.

This sub-section presents QUBO/Ising models for one-hot/zero-one-hot/domain-wall vectors, aimed at helping the reader understand the fundamental concepts of using these models to represent numbers. These models are designed to achieve their optimal value if and only if the vectors X/S are one-hot/zero-one-hot/domain-wall vectors.

We will first design a QUBO model with k-bit variable $X = (x_i)$ ($0 \leq i \leq k-1$) that takes the minimum value of 0 if and only if it stores a k-bit one-hot vector. It is clear that X is a one-hot vector if and only if the sum of all bits is 1. Based on this property, we can design a QUBO model with an energy function $E_1(X)$ as follows:

$$E_1(X) = \left(1 - \sum_{i=0}^{k-1} x_i\right)^2 = 2\sum_{i=1}^{k-1} x_{i-1}x_i - \sum_{i=0}^{k-1} x_i + 1. \quad (6)$$

It is evident that $E_1(X)$ takes the minimum value of 0 if and only if X is a one-hot vector. Additionally, we can design a QUBO model that takes the minimum value if and

only if it stores a k-bit zero-one-hot vector. A k-bit vector X is a zero-one-hot vector if and only if the sum of all bits is either 0 or 1. Thus, the energy $E_{01}(X)$ defined below takes the minimum value of 0 if X is a zero-one-hot vector.

$$E_{01}(X) = \frac{1}{2}\sum_{i=0}^{k-1} x_i \left(1 - \sum_{i=0}^{k-1} x_i\right) = \sum_{i=1}^{k-1} x_{i-1}x_i. \tag{7}$$

QUBO models with $E_{01}(X)$ for zero-one-hot vectors have no linear terms, making them simpler than those with $E_1(X)$ for one-hot vectors. Since all quadratic terms in the expanded summation are 2, $E_{01}(X)$ has a coefficient of $\frac{1}{2}$. Our objective is to minimize the integer coefficients of both linear and quadratic terms. Consequently, we will employ such coefficients whenever possible throughout the remainder of this paper.

Similarly, we can design Ising models with $H_1(S)$ and $H_{01}(S)$ for a k-qubit variable $S = (s_i)$ $(0 \leq i \leq k-1)$ storing one-hot and zero-one-hot vectors, respectively. A k-qubit variable S stores a one-hot vector if and only if $\sum_{i=0}^{k-1} s_i = 1 \cdot (+1) + (k-1) \cdot (-1) = -(k-2)$. Thus, the following $H_1(S)$ takes the minimum value of 0 if and only if S is a one-hot vector:

$$H_1(S) = \frac{1}{2}\left((k-2) + \sum_{i=0}^{k-1} s_i\right)^2 = \sum_{i=1}^{k-1} s_{i-1}s_i + (k-2)\sum_{i=0}^{k-1} s_i + \frac{1}{2}k^2 - \frac{3}{2}k + 2. \tag{8}$$

Clearly, $H_1(S)$ takes the minimum value of 0 if and only if S is a one-hot vector. Additionally, S stores a zero-one-hot vector if and only if it has zero or one $+1$. Thus, the following Ising model with $H_{01}(S)$ takes the minimum value of 0 if and only if S is a zero-one-hot vector, which means that the sum of all qubits is $-k$ or $-(k-2)$:

$$H_{01}(S) = \frac{1}{2}\left(k + \sum_{i=0}^{k-1} s_i\right)\left((k-2) + \sum_{i=0}^{k-1} s_i\right) = \sum_{i=1}^{k-1} s_{i-1}s_i + (k-2)\sum_{i=0}^{k-1} s_i + \frac{1}{2}k^2 - \frac{1}{2}k. \tag{9}$$

We will now design a QUBO model for a k-bit variable $X = (x_i)$ $(0 \leq i \leq k-1)$ that stores a domain-wall vector. For technical reasons, we assume the existence of fixed *guard bits* $x_{-1} = 1$ and $x_k = 0$ for X. With these guard bits, if X stores a domain-wall vector, then $x_{i-1} - x_i \neq 0$ $(0 \leq i \leq k)$ holds for exactly one i. Otherwise, it holds for more than two i. Based on this fact, we have the following QUBO model:

$$E_d(X) = \frac{1}{2}\sum_{i=0}^{n-1}(x_{i-1} - x_i)^2 = -\sum_{i=1}^{k-1} x_{i-1}x_i + \sum_{i=1}^{k-1} x_i + \frac{1}{2}. \tag{10}$$

The energy function $E_d(X)$ takes the minimum value of $\frac{1}{2}$ if and only if $(x_{i-1} - x_i)^2 = 1$ for exactly one i and X stores a domain-wall vector. Similarly, we can design an Ising model for a k-qubit variable $S = (s_i)$ $(0 \leq i \leq k-1)$ that stores a domain-wall vector using the same approach. We also assume the existence of fixed *guard qubits* $s_{-1} = +1$ and $s_k = -1$. The following Ising model with $H_d(S)$ takes the minimum value of 2 if and only if S stores a domain-wall vector and $(s_{i-1} - s_i)^2 = 4$ for exactly one i:

$$H_d(S) = \frac{1}{2}\sum_{i=0}^{k}(s_{i-1} - s_i)^2 = -\sum_{i=1}^{k-1} s_{i-1}s_i - s_0 + s_{k-1} + (k+1) \tag{11}$$

Table 2 summarizes the features of QUBO/Ising models with k-bit/qubit one-hot/zero-one-hot/domain-wall vectors. We utilized SymPy [32], a Python library for symbolic mathematics, to expand the mathematical formulas and derive the features of the QUBO/Ising models presented throughout this paper. We observe that QUBO/Ising models for one-hot vectors are fully connected and consist of $\frac{1}{2}k^2 - \frac{1}{2}k$ quadratic terms. In contrast, models for domain-wall vectors have only $k-1$ quadratic terms. Furthermore, the linear term

coefficients of Ising models for one-hot and zero-one-hot vectors are $k-2$ and $k-1$, respectively. On the other hand, those for domain-wall vectors have only two linear terms with coefficients -1 and $+1$, respectively.

Table 2. QUBO/Ising models for k-bit/qubit one-hot/domain-wall vectors.

Encoding Type	One-Hot	Zero-One-Hot	Domain-Wall
QUBO models			
Quadratic formula	$E_1(X)$	$E_{01}(X)$	$E_d(X)$
Linear term count	k	0	$k-1$
Linear term coefficients	-1	-	$+1$
Quadratic term count	$\frac{1}{2}k^2-\frac{1}{2}k$	$\frac{1}{2}k^2-\frac{1}{2}k$	$k-1$
Quadratic term coefficients	$+2$	$+1$	-1
Diameter	1	1	$k-1$
Offset	1	0	$\frac{1}{2}$
Optimal energy	0	0	$\frac{1}{2}$
Ising models			
Quadratic formula	$H_1(S)$	$H_{01}(S)$	$H_d(S)$
Linear term count	k	k	2
Linear term coefficients	$k-2$	$k-1$	$-1,+1$
Quadratic term count	$\frac{1}{2}k^2-\frac{1}{2}k$	$\frac{1}{2}k^2-\frac{1}{2}k$	$k-1$
Quadratic term coefficients	$+1$	$+1$	-1
Diameter	1	1	$k-1$
Offset	$\frac{1}{2}k^2-\frac{3}{2}k+2$	$\frac{1}{2}k^2-\frac{1}{2}k$	$k+1$
Optimal Hamiltonian	0	0	2

3. QUBO/Ising Model Kernels for Generating Permutation Involving a Cubic Number of Quadratic Terms

A permutation of n numbers can be defined by a bijection $\pi : \{0,1,\ldots,n-1\} \to \{0,1,\ldots,n-1\}$, where the list $[\pi(0),\pi(1),\ldots,\pi(n-1)]$ represents one of the $n!$ possible permutations. This section initially describes a conventional permutation encoding method that employs one-hot vectors. This approach is widely used for solving permutation-based combinatorial optimization problems, not just in QUBO/Ising models but also in mixed-integer programming. We then explain a permutation encoding method using domain-wall vectors, which was introduced in [22] and reduces the number of quadratic terms by half.

3.1. Conventional Permutation Encoding by One-Hot Vectors

A permutation π is commonly represented by an $n \times n$ matrix of variables, where each row i ($0 \le i \le n-1$) stores $\pi(i)$ as a one-hot vector. This representation ensures that each row of the matrix is a one-hot vector. We will design n^2-bit QUBO/Ising kernels with $X = (x_{i,j})/S = (s_{i,j})$ ($0 \le i,j \le n-1$), which can generate permutations as the optimal solutions. For this purpose, we apply $E_1(X)/H_1(S)$ to all rows and columns to guarantee that they are one-hot vectors as follows:

$$E_1^{nn}(X) = \frac{1}{2}\sum_{i=0}^{n-1}\left(1-\sum_{j=0}^{n-1}x_{i,j}\right)^2 + \frac{1}{2}\sum_{j=0}^{n-1}\left(1-\sum_{i=0}^{n-1}x_{i,j}\right)^2 \tag{12}$$

$$H_1^{nn}(S) = \frac{1}{2}\sum_{i=0}^{n-1}\left((n-2)+\sum_{j=0}^{n-1}s_{i,j}\right)^2 + \frac{1}{2}\sum_{j=0}^{n-1}\left((n-2)+\sum_{i=0}^{n-1}s_{i,j}\right)^2 \tag{13}$$

We can obtain the quadratic formulas for QUBO and Ising models by expanding $E_1^{nn}(X)$ and $H_1^{nn}(S)$, respectively. Both formulas take the minimum value of 0 if and only if every row and every column is a one-hot vector. The features of QUBO/Ising models obtained by expanding $E_1^{nn}(X)/H_1^{nn}(S)$ can be found in Table 3. It is important to note that

both models have $n^3 - n^2$ quadratic terms, and the Ising model also includes linear terms with a coefficient of $2n - 4$.

Table 3. QUBO/Ising kernels for generating a permutation of n numbers.

Encoding Type	One-Hot	All-Different	Domain-Wall Dual-Matrix	Extended
Bit/qubit count	n^2	$n^2 - n$	$2n^2 - 2n$	$3n^2 - 2n$
QUBO models				
Quadratic formula	$E_1^{nn}(X)$	$E_a^{nn}(X)$	$E_d^{nn}(A, B)$	$E_e^{nn}(A, B, X)$
Linear term count	n^2	$n^2 - 2n$	$2n^2 - 4n$	$3n^2 - 6n + 2$
Linear term coefficients	-1	$-(2n-3), -(2n-6), \ldots, -2$	$+2$	$-1, +1, +2$
Quadratic term count	$n^3 - n^2$	$\frac{1}{2}n^3 - \frac{3}{2}n$	$6n^2 - 12n + 4$	$6n^2 - 8n$
Quadratic term coefficients	$+1$	$-1, +2$	$-2, -1, +1$	$-2, -1, +1$
Diameter	2	$n - 1$	$2n - 3$	$2n$
Offset	n	$\frac{1}{3}n^3 - \frac{1}{2}n^2 - 2n$	$2n - 1$	$2n$
Optimal energy	0	$\frac{1}{2}n$	n	n
Ising models				
Quadratic formula	$H_1^{nn}(S)$	$H_a^{nn}(S)$	$H_d^{nn}(A, B)$	$H_e^{nn}(A, B, S)$
Linear term count	n^2	$n^2 + n(n \bmod 2) - 2n$	$4n$	$n^2 + 4n - 4$
Linear term coefficients	$2n - 4$	$-(n-1), -(n-4), \ldots, +(n-1)$	$-2, +2$	$-2, +1, +2$
Quadratic term count	$n^3 - n^2$	$\frac{1}{2}n^3 - \frac{3}{2}n$	$6n^2 - 12n + 4$	$6n^2 - 8n$
Quadratic term coefficients	$+1$	$-1, +1$	$-2, -1, +1$	$-2, -1, +1$
Diameter	2	$n - 1$	$2n - 3$	$2n$
Offset	$n^3 - 3n^2 + 4n$	$\frac{1}{6}n^3 + n^2 - \frac{1}{6}n$	$4n^2 - 4$	$6n^2 - 4n$
Optimal Hamiltonian	0	$2n$	$4n$	$4n$

3.2. Permutation Encoding by All-Different Domain-Wall Encoding

We will now introduce *the all-different domain-wall technique*, which was presented in [22] and can be used to generate permutations with domain-wall vectors. This technique utilizes a matrix $X = (x_{i,j})$ of size $n \times (n-1)$, where each row i $(0 \leq i \leq n-1)$ stores the domain-wall vector representing $\pi(i)$ for a permutation π. Figure 1 illustrates an example with $n = 4$. It is important to note that a matrix X, where each row contains a domain-wall vector, represents a permutation if and only if all the domain-wall vectors are distinct. The all-different domain-wall technique leverages this property to design a QUBO kernel for generating a permutation, as follows:

$$E_a^{nn}(X) = \frac{1}{2}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}(x_{i,j-1} - x_{i,j})^2 + \sum_{j=0}^{n-2}\left((n-j-1) - \sum_{i=0}^{n-1} x_{i,j}\right)^2 \tag{14}$$

Here, fixed guard bits $x_{i,-1} = 1$ and $x_{i,n-1} = 0$ for all i $(0 \leq i \leq n-1)$ are used. The first summation term takes the minimum value of $\frac{1}{2}n$ if and only if every row is a domain-wall vector. Also, the the second summation term takes the minimum value of 0 if and only if the number of 1s in each column j $(0 \leq j \leq n-1)$ is $n - j - 1$. If this is the case, all rows of X store distinct domain-wall vectors. Therefore, X is a permutation if and only if it is an optimal solution that satisfies $E_a^{nn}(X) = \frac{1}{2}n$.

We can apply the same technique for designing an Ising kernel with an $n \times (n-1)$ matrix $S = (s_{i,j})$ that generates a permutation as a domain-wall vector. If all rows in S store distinct domain-wall vectors, then the number of $+1$s in each column j $(0 \leq j \leq n-1)$ is $(n-j-1)$, and the sum of all elements in it is $(n-j-1)(+1) + (j+1)\cdot(-1) = n - 2j - 2$. Based on this fact, we can design a desired Ising model as follows:

$$H_a^{nn}(X) = \frac{1}{2}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}(s_{i,j-1} - s_{i,j})^2 + \frac{1}{2}\sum_{j=0}^{n-2}\left((n-2j-2) - \sum_{i=0}^{n-1} s_{i,j}\right)^2 \quad (15)$$

Here, $s_{i,-1} = +1$ and $s_{i,n-1} = -1$ for all i ($0 \leq i \leq n-1$). The first summation term takes the minimum value of $\frac{1}{2} \cdot 4n = 2n$ if and only if every row is a domain-wall vector. Also, the second summation term takes 0 if and only if the number of $+1$s in each column j ($0 \leq j \leq n-1$) is $n-j-1$. If this is the case, all rows of S store distinct domain-wall vectors. Therefore, S stores domain-wall vectors representing a permutation if and only if it is an optimal solution that satisfies $H_a^{nn}(X) = 2n$.

Table 3 presents the features of QUBO and Ising models derived from the expansion of mathematical expressions $E_a^{nn}(X)$ and $H_a^{nn}(S)$. The number of quadratic terms in these models is halved compared to conventional one-hot encoding. However, these models include linear terms with significantly large absolute values. The maximum absolute values for the QUBO and Ising models are $2n-3$ and $n-1$, respectively.

4. QUBO/Ising Kernels with a Quadratic Number of Quadratic Terms Based on Our Dual-Matrix Domain-Wall Technique

Our new permutation encoding technique utilizes the inverse permutation. For a permutation π of n numbers, let π^{-1} denote the inverse, such that $\pi^{-1}(\pi(i)) = i$ for all i ($0 \leq i \leq n-1$). We refer to the permutations $[\pi(0), \pi(1), \ldots, \pi(n-1)]$ and $[\pi^{-1}(0), \pi^{-1}(1), \ldots, \pi^{-1}(n-1)]$ as *dual permutations*.

Recall that in conventional one-hot encoding, each row i ($0 \leq i \leq n-1$) of an $n \times n$ matrix stores $\pi(i)$ as a one-hot vector, as illustrated in Figure 1. It is easy to confirm that each column j ($0 \leq j \leq n-1$) also stores a one-hot vector representing the inverse $\pi^{-1}(j)$. Therefore, the row one-hot vectors and column one-hot vectors represent dual permutations. Figure 1 shows an example of a 4×4 matrix storing the row permutation $[1, 3, 2, 0]$ and the column permutation $[3, 0, 2, 1]$, which are dual permutations. Our dual-matrix domain-wall technique was inspired by this fact. It employs two matrices: matrix $A = (a_{i,j})$ ($0 \leq i \leq n-1$ and $0 \leq j \leq n-2$) of size $n \times (n-1)$ and matrix $B = (b_{i,j})$ ($0 \leq i \leq n-2$ and $0 \leq j \leq n-1$) of size $(n-1) \times n$. These matrices are used to store dual permutations, where the rows of A represent a permutation and the columns of B represent its inverse permutation. For a clearer understanding, the reader should refer to Figure 4, which provides an example for $n = 4$ using a 4×3 matrix A and a 3×4 matrix B.

For domain-wall encoding, fixed guard bits/qubits are attached to the leftmost and rightmost columns of matrix A, as well as the top and bottom rows of matrix B, as illustrated in Figure 4. Specifically, we set $a_{i,-1} = 1/+1$ and $a_{i,n-1} = 0/-1$ for all i ($0 \leq i \leq n-1$), and $b_{-1,j} = 1/+1$ and $b_{n-1,j} = 0/-1$ for all j ($0 \leq i \leq n-1$) for QUBO/Ising models. Our goal is to design QUBO/Ising models that have optimal solutions satisfying the following three conditions:

Row domain-wall condition: Each row of matrix A stores a domain-wall vector.

Column domain-wall condition: Each column of matrix B stores a domain-wall vector.

Dual-permutation condition: The row permutation of matrix A and the column permutation of matrix B are dual to each other.

An optimal solution of the designed QUBO/Ising models corresponds to any one of the possible $n!$ permutations.

We begin by introducing the dual-matrix domain-wall technique for QUBO models. We define matrices $\Delta A = (\Delta a_{i,j})$ and $\Delta B = (\Delta b_{i,j})$ of size $n \times n$ ($0 \leq i, j \leq n-1$) by computing the differences in each row of matrix A and each column of matrix B as follows:

$$\Delta a_{i,j} = a_{i,j-1} - a_{i,j} \text{ and } \Delta b_{i,j} = b_{i-1,j} - b_{i,j}. \quad (16)$$

Figure 4 illustrates examples of ΔA and ΔB with $n = 4$. The QUBO kernel for generating a permutation using the dual-matrix domain-wall technique is defined by the following expression, denoted as $E_d^{nn}(A, B)$:

$$E_d^{nn}(A,B) = \frac{1}{2}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}(\Delta a_{i,j})^2 + \frac{1}{2}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}(\Delta b_{i,j})^2 + \frac{1}{2}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}(\Delta a_{i,j} - \Delta b_{i,j})^2 \tag{17}$$

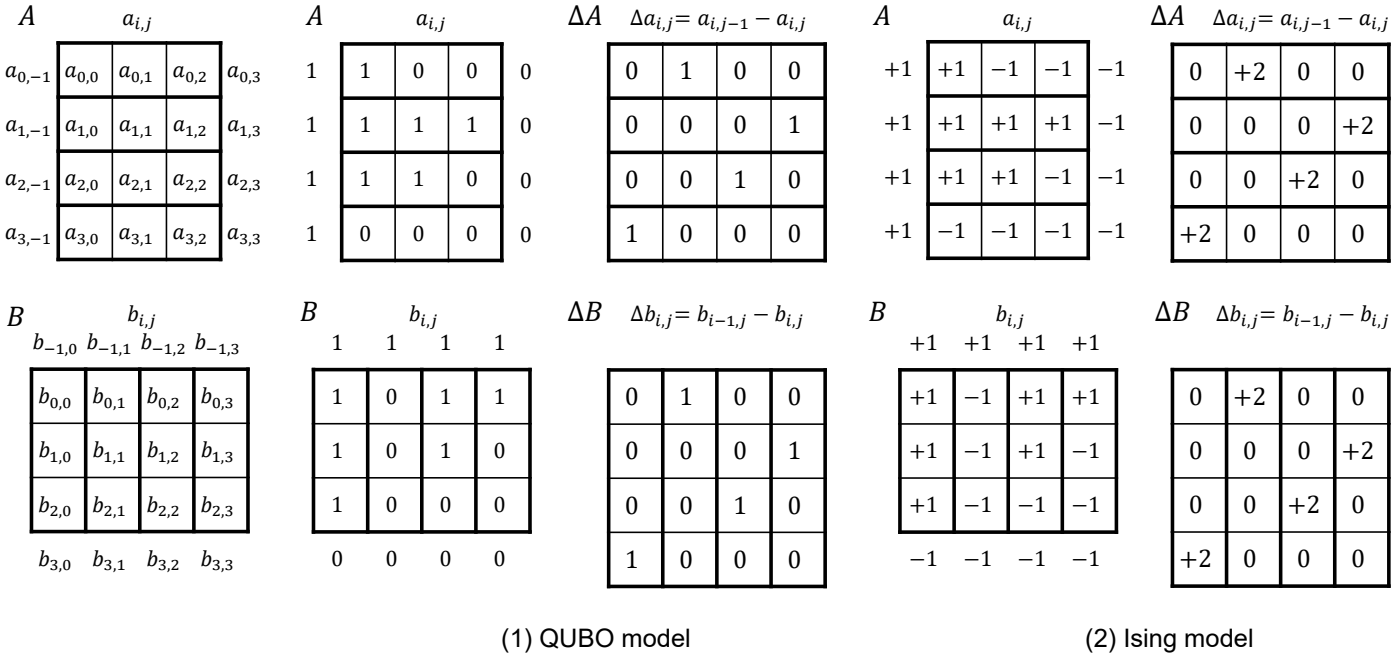

Figure 4. The dual-matrix domain-wall technique for representing permutations of $n = 4$ numbers on an Ising model.

The first summation term takes the minimum value of $\frac{1}{2}n$ if and only if the row domain-wall condition is satisfied. Similarly, the second summation term takes the minimum value of $\frac{1}{2}n$ if and only if the column domain-wall condition is satisfied. Lastly, the third summation term takes the minimum value of 0 if and only if $\Delta a_{i,j} = \Delta b_{i,j}$ for all i and j. In essence, this condition signifies the satisfaction of the dual-permutation condition. It is important to note that these three summation terms can all reach their minimum values simultaneously. Consequently, the energy function $E_d^{nn}(A, B)$ reaches its minimum value of n if and only if matrices A and B store dual permutations, satisfying all three conditions: the row domain-wall condition, the column domain-wall condition, and the dual-permutation condition.

For the Ising model, we can use the exact same mathematical expression as the QUBO model, resulting in the Hamiltonian function $H_d^{nn}(A, B)$:

$$H_d^{nn}(A,B) = \frac{1}{2}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}(\Delta a_{i,j})^2 + \frac{1}{2}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}(\Delta b_{i,j})^2 + \frac{1}{2}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}(\Delta a_{i,j} - \Delta b_{i,j})^2 \tag{18}$$

Both the first and second summation terms take the minimum value of $2n$ if and only if both the row and column domain-wall conditions are satisfied. The third summation term takes the minimum value of 0 if and only if the dual-permutation condition is satisfied. Since these three summation terms can take the minimum values at the same time, the Hamiltonian function $H_d^{nn}(A, B)$ takes the minimum value of $4n$ if and only if matrices A and B store dual permutations. It is worth noting that although $E_d^{nn}(A, B)$ and $H_d^{nn}(A, B)$ have the same mathematical expressions, their expansions differ. The energy function $E_d^{nn}(A, B)$ represents the QUBO model and takes the minimum value of n when all three

conditions (row domain-wall, column domain-wall, and dual-permutation) are met. On the other hand, the Hamiltonian function $H_d^{nn}(A, B)$ corresponds to the Ising model and achieves the minimum value of $4n$ when matrices A and B satisfy the dual-permutation condition along with the row and column domain-wall conditions. These QUBO/Ising models can be used as kernels for solving permutation-based combinational optimization problems as QUBO/Ising models.

The features of QUBO/Ising models obtained using our dual-matrix domain-wall encoding technique are presented in Table 3. It is evident from the table that these models consist of only $6n^2 - 12n + 4$ quadratic terms. In contrast, models designed using the conventional one-hot encoding approach contain $n^3 - n^2$ quadratic terms, while models created using the all-different domain-wall encoding method have $\frac{1}{2}n^3 - \frac{3}{2}n$ quadratic terms. Thus, our dual-matrix domain-wall technique significantly reduces the number of quadratic terms in the models.

Moreover, our encoding technique also leads to a reduction in the linear term coefficients in the Ising models. In Ising models obtained using conventional one-hot encoding and all-different domain-wall encoding, the linear terms have coefficients of $2n - 4$ and $n - 1$, respectively. However, Ising models obtained through dual-matrix domain-wall encoding feature linear terms with a coefficient of 2.

5. QUBO/Ising Kernels for Partial Permutations

For two positive integers satisfying $m \leq n$, a *partial permutation* refers to a sequence of m numbers selected from a set $0, 1, \ldots, n-1$ of n numbers without repetition. A permutation is a special case of a partial permutation where $m = n$. In this section, we primarily focus on the case where $m < n$. However, we can also consider a permutation of n numbers by substituting m with n. A partial permutation can be represented by an injection $\pi : \{0, 1, \ldots, m-1\} \to \{0, 1, \ldots, n-1\}$, where $\pi(i)$ denotes the i-th element in the partial permutation $[\pi(0), \pi(1), \ldots, \pi(m-1)]$. The total number of possible partial permutations is given by $n!/(n-m)!$. *An inverse* of a partial permutation π is a surjection $p : \{0, 1, \ldots, n-1\} \to \{0, 1, \ldots, m-1\}$ that satisfies $p(\pi(i)) = i$ for all i $(0 \leq i \leq m-1)$. It should be noted that $p(j)$ can take any value when $\pi^{-1}(j)$ is undefined, meaning that there is no i satisfying $\pi(i) = j$. We refer to such π and p as *dual permutations*.

This section first demonstrates partial-permutation encoding using one-hot vectors. Subsequently, we illustrate the application of the dual-matrix domain-wall technique to generate partial permutations. It is important to note that the all-different domain-wall technique cannot be applied in this case. Unlike permutation generation, the column-wise sums of the domain-wall encoding for a partial permutation are not fixed.

5.1. Partial-Permutation Encoding by One-Hot Vectors

We can construct QUBO/Ising kernels to generate partial permutations using zero-one-hot vectors. For this purpose, we utilize an $m \times n$ matrix $(m < n)$ of bits/qubits. Each row i $(0 \leq i \leq m-1)$ in the matrix represents $\pi(i)$ as a one-hot vector, while each column j $(0 \leq j \leq n-1)$ represents $\pi^{-1}(j)$ as a zero-one-hot vector. Here, we write $\pi^{-1}(j) = \varphi$ if there is no i satisfying $\pi(i) = j$. Figure 5 illustrates an example of a 3×5 matrix that represents a partial permutation. In this example, the rows correspond to the partial permutation $[\pi(0), \pi(1), \pi(2)] = [3, 0, 1]$ selected from the set $\{0, 1, 2, 3, 4\}$, and the columns represent the inverse $[\pi^{-1}(0), \pi^{-1}(1), \pi^{-1}(2), \pi^{-1}(3), \pi^{-1}(4)] = [1, 2, \varphi, 0, \varphi]$.

We can design the QUBO/Ising kernels $E_1^{mn}(X)/H_1^{mn}(S)$ generating a partial permutation as follows:

$$E_1^{mn}(X) = \frac{1}{2}\sum_{i=0}^{m-1}\left(1 - \sum_{j=0}^{n-1} x_{i,j}\right)^2 + \frac{1}{2}\sum_{j=0}^{n-1}\sum_{i=0}^{n-1} x_{i,j}\left(1 - \sum_{i=0}^{m-1} x_{i,j}\right) \tag{19}$$

$$H_1^{mn}(S) = \frac{1}{2}\sum_{i=0}^{m-1}\left((n-2) + \sum_{j=0}^{n-1} s_{i,j}\right)^2 + \frac{1}{2}\sum_{j=0}^{n-1}\left(m + \sum_{i=0}^{m-1} s_{i,j}\right)\left((m-2) + \sum_{i=0}^{m-1} s_{i,j}\right) \tag{20}$$

The first summation term in each expression takes the minimum value of 0 when every row is a one-hot vector. Similarly, the second summation term is evaluated as 0 when every column is a zero-one-hot vector. Consequently, the optimal solutions of $E_1^{mn}(X)/H_1^{mn}(S)$ represent dual permutations.

	1	2	φ	0	φ
3	0	0	0	1	0
0	1	0	0	0	0
1	0	1	0	0	0

(1) QUBO model

	1	2	φ	0	φ
3	−1	−1	−1	+1	−1
0	+1	−1	−1	−1	−1
1	−1	+1	−1	−1	−1

(2) Ising model

Figure 5. 3×5 matrix representing a partial permutation selecting 3 from 5 using one-hot and zero-one-hot vectors.

We should note that previous works such as [26,27] have already presented QUBO models for partial permutation using zero-one-hot vectors. However, the mathematical expressions used in these works differ. In [26], a slack variable y_i $(0 \leq i \leq n-1)$ was introduced for each column i, and the expression $\sum_{j=0}^{n-1}\left(1 - y_i - \sum_{i=0}^{m-1} x_{i,j}\right)^2$ was utilized to ensure that each column represented a zero-one-hot vector. Similarly, in [27], the expression $\sum_{j=0}^{n-1}\left(\frac{1}{2} - \sum_{i=0}^{m-1} x_{i,j}\right)^2$ was employed for the same purpose. In contrast, our proposed mathematical expression $\sum_{i=0}^{n-1} x_{i,j}\left(1 - \sum_{i=0}^{m-1} x_{i,j}\right)$ for zero-one-hot vectors is more intuitive and simpler as it does not yield any linear terms.

5.2. Partial-Permutation Encoding by Our Dual-Matrix Domain-Wall Encoding Technique

We utilize two matrices, namely $A = (a_{i,j})$ $(0 \leq i \leq m-1$ and $0 \leq j \leq n-2)$ of size $m \times (n-1)$ with bit/qubit variables and $B = (b_{i,j})$ $(0 \leq i \leq m-1$ and $0 \leq j \leq n-1)$ of size $(m-1) \times n$. These matrices are employed for our dual-matrix domain-wall technique and represent partial permutations. For an illustration, refer to Figure 6, which shows example matrices A and B with dimensions $m = 3$ and $n = 5$, respectively. Similarly to dual-matrix domain-wall encoding for permutations, we include fixed guard bits/qubits in the leftmost and rightmost columns of A, as well as the top and bottom rows of B, as depicted in Figure 6. By applying Equation (16), we can derive two matrices $\Delta A = (\Delta a_{i,j})$ and $\Delta B = (\Delta b_{i,j})$ $(0 \leq i \leq m-1, 0 \leq j \leq n-1)$.

To satisfy the three conditions, we make modifications to the mathematical expressions $E_d^{nn}(A,B)$ and $H_d^{nn}(A,B)$ for the QUBO and Ising kernels that generate permutations. These modifications ensure that the row domain-wall condition, column domain-wall condition, and dual-permutation condition are met by the two matrices A and B with sizes $m \times (n-1)$ and $(m-1) \times n$, respectively. The adjusted expressions are as follows:

$$E_d^{mn}(A,B) = \frac{1}{2}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(\Delta a_{i,j})^2 + \frac{1}{2}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(\Delta b_{i,j})^2 + \frac{1}{2}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(\Delta a_{i,j} - \Delta b_{i,j})^2 \quad (21)$$

$$H_d^{mn}(A,B) = \frac{1}{2}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(\Delta a_{i,j})^2 + \frac{1}{2}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(\Delta b_{i,j})^2 + \frac{1}{2}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(\Delta a_{i,j} - \Delta b_{i,j})^2 \quad (22)$$

We proceed to demonstrate that $E_d^{mn}(A,B)$ attains its minimum value of $n-m$ if and only if ΔA represents a partial permutation. The first summation term takes the minimum value of $\frac{1}{2}m$ when all rows of matrix A are domain-wall vectors, satisfying the row domain-wall condition. Similarly, the second summation terms will reach the minimum values of

$\frac{1}{2}n$ if the column domain-wall condition is satisfied. Suppose that the row domain-wall condition and the column domain-wall condition are satisfied. Since the numbers of 1s in ΔA and ΔB are m and n, respectively, the third summation term takes the minimum value of $\frac{1}{2}(n-m)$ if the dual-permutation condition is satisfied. Hence, $E_d^{mn}(A,B)$ takes the value $\frac{1}{2}m+\frac{1}{2}n+\frac{1}{2}(n-m)=n$ if the three conditions are satisfied. Unfortunately, this does not imply that $E_d^{mn}(A,B)$ takes the minimum value of n if and only if the three conditions are satisfied. The third summation term can be smaller than $\frac{1}{2}(n-m)$, say 0, if the row or column domain-wall condition is not satisfied. Hence, we need to prove that $E_d^{mn}(A,B)$ is larger than n if at least one of the three conditions is not satisfied. More specifically, we prove the following lemma:

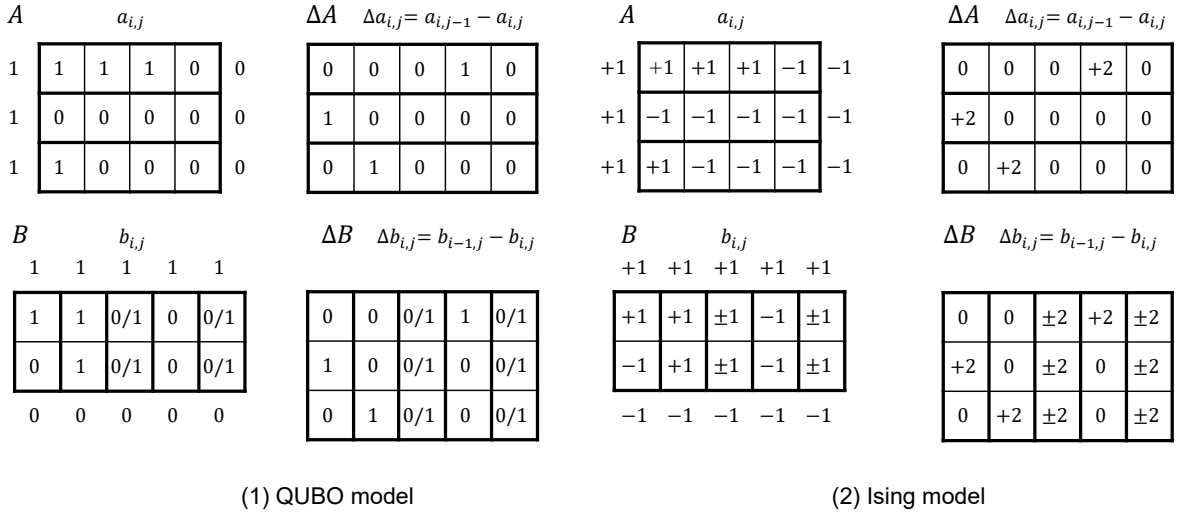

Figure 6. Dual-matrix domain-wall encoding presenting a partial permutation selecting 3 from 5 using domain-wall vectors.

Lemma 1. *$E_d^{mn}(A,B)$ is larger than n if A and B do not satisfy at least one of the row domain-wall, column domain-wall, and dual-permutation conditions.*

Proof. Suppose that the row and column domain-wall conditions are satisfied and the numbers of 1s in ΔA and ΔB are m and n, respectively. We can observe that the third summation term of $E_d^{mn}(A,B)$ exceeds $\frac{1}{2}(n-m)$ when ΔA and ΔB are not dual. Consequently, we can determine that $E_d^{mn}(A,B) > \frac{1}{2}m+\frac{1}{2}n+\frac{1}{2}(n-m)=n$.

Next, let us consider the case in which the row and/or column domain-wall conditions are not fulfilled. If a row of A is not a domain-wall vector, the numbers of $1/-1$s in ΔA are $(1+k)/k$ for some $k \geq 1$. For example, if a row of A is 11001011, the corresponding row of ΔA would be $0010(-1)1(-1)100$, containing three 1s and two -1s. Based on this observation, we can assign non-negative integers k_A and k_B such that the numbers of $1/-1$s in ΔA and ΔB are $(m+k_A)/k_A$ and $(n+k_B)/k_B$, respectively. It follows that ΔA and ΔB have $m+2k_A$ and $n+2k_B$ non-zero elements, respectively. Since we assume that the row and/or column domain-wall conditions are not satisfied, at least one of k_A and k_B must be greater than or equal to 1. Clearly, the first and second summation terms take values of $\frac{1}{2}m+k_A$ and $\frac{1}{2}n+k_B$, respectively. The third summation term must be at least

$$\left|(\frac{1}{2}m+\frac{1}{2}k_A)-(\frac{1}{2}n+\frac{1}{2}k_B)\right|+\left|\frac{1}{2}k_A-\frac{1}{2}k_B\right|. \tag{23}$$

We can now analyze the value of $E_d^{mn}(A,B)$ in the following cases:

Case 1: $(\frac{1}{2}m+\frac{1}{2}k_A) \leq (\frac{1}{2}n+\frac{1}{2}k_B)$

The lower bound of $E_d^{mn}(A,B)$ can be evaluated by the sum of the three summation terms as follows:

$$E_d^{mn}(A,B) \geq (\frac{1}{2}m + k_A) + (\frac{1}{2}n + k_B) + \left((\frac{1}{2}n + \frac{1}{2}k_B) - (\frac{1}{2}m + \frac{1}{2}k_A)\right)$$
$$= n + \frac{1}{2}k_A + \frac{3}{2}k_B > n. \quad (24)$$

Case 2: $(\frac{1}{2}m + \frac{1}{2}k_A) > (\frac{1}{2}n + \frac{1}{2}k_B)$

Similarly, $E_d^{mn}(A,B)$ can be evaluated by the sum of the first and second summation terms as follows:

$$E_d^{mn}(A,B) > (\frac{1}{2}m + k_A) + (\frac{1}{2}n + k_B) > (\frac{1}{2}n + \frac{1}{2}k_B) + (\frac{1}{2}n + k_B) > n. \quad (25)$$

Thus, this completes the proof of the lemma. □

Since we have shown that $E_d^{mn}(A,B) = n$ if the three conditions are satisfied, this lemma implies the following theorem:

Theorem 1. *$E_d^{mn}(A,B)$ takes the minimum value of n if and only if A and B satisfy the row domain-wall, column domain-wall, and dual-permutation conditions are satisfied.*

Next, we will evaluate the value of $H_d^{mn}(A,B)$ when the three conditions are satisfied. The first summation term and second summation term take minimum values of $2m$ and $2n$, respectively, if and only if the row and column domain-wall conditions are satisfied. Additionally, if the dual-permutation condition is satisfied, the third summation term takes a value of $2(n-m)$. Hence, $H_d^{mn}(A,B)$ is equal to $2m + 2n + 2(n-m) = 4n$ if the row domain-wall, column domain-wall, and dual-permutation conditions are all satisfied. Similarly, we can prove that $H_d^{mn}(A,B)$ is larger than $4n$ if at least one of the three conditions is not satisfied. Thus, we can state the following theorem:

Theorem 2. *$H_d^{mn}(A,B)$ takes the minimum value of $4n$ if and only if A and B satisfy the row domain-wall, column domain-wall, and dual-permutation conditions.*

This theorem establishes the condition for $H_d^{mn}(A,B)$ to attain its minimum value and confirms that it occurs when the row domain-wall, column domain-wall, and dual-permutation conditions are satisfied.

The readers should refer to Table 4, which provides the features of kernels $E_d^{mn}(A,B)$ and $H_d^{mn}(A,B)$. It can be observed that the number of quadratic terms in these models is given by $6mn - 4m - 4n$. They utilize only a quadratic number of quadratic terms, whereas kernels $E_1^{mn}(A,B)$ and $H_1^{mn}(A,B)$ require a cubic number of quadratic terms. Additionally, the maximum absolute value of the linear term coefficients in $H_d^{mn}(A,B)$ is only 2, while $H_1^{mn}(A,B)$ requires large coefficients of $m + n - 3$.

Table 4. QUBO/Ising kernels for generating a partial permutation selecting m numbers from n numbers.

		Domain-Wall	
Encoding Type	**One-Hot**	**Dual-Matrix**	**Extended**
Bit/qubit count	mn	$2mn - m - n$	$3mn - m - n$
QUBO models			
Quadratic formula	$E_1^{mn}(A,B)$	$E_d^{mn}(A,B)$	$E_e^{mn}(A,B,X)$
Linear term count	mn	$2mn - 2m - 2n$	$3mn - 3m - 2n + 1$
Linear term coefficients	-1	$+2$	$-1, +1, +2, +3$
Quadratic term count	$\frac{1}{2}m^2n + \frac{1}{2}mn^2 - mn$	$6mn - 6m - 6n + 4$	$6mn - 4m - 4n$
Quadratic term coefficients	$+1, +2$	$-2, -1, +1$	$-3, -2, -1, +1, +2$
Diameter	2	$m + n - 3$	$m + n$
Offset	m	$m + n - 1$	$\frac{3}{2}m + \frac{1}{2}n$
Optimal energy	0	n	$\frac{1}{2}m + \frac{1}{2}n$

Table 4. *Cont.*

Encoding Type	One-Hot	Domain-Wall	
		Dual-Matrix	Extended
Ising models			
Quadratic formula	$H_1^{mn}(A,B)$	$H_d^{mn}(A,B)$	$H_e^{mn}(A,B,S)$
Linear term count	mn	$2m+2n$	$mn+2m+2n$
Linear term coefficients	$m+n-3$	$-2,+2$	$-3,-1,+1,+2,+3,+4$
Quadratic term count	$\frac{1}{2}m^2n+\frac{1}{2}mn^2-mn$	$6mn-6m-6n+4$	$6mn-4m-4n$
Quadratic term coefficients	$+1$	$-2,-1,+1$	$-3,-2,-1,+1,+2$
Diameter	2	$m+n-3$	$m+n$
Offset	$\frac{1}{2}m^2n+\frac{1}{2}mn^2-2mn+2m$	$4mn-4$	$8mn-4m-2n$
Optimal Hamiltonian	0	$4n$	$2m+2n$

6. Dual-Matrix Domain-Wall Technique Extended with a One-Hot Matrix

When the QUBO kernels designed by the dual-matrix domain-wall technique are involved in QUBO models for solving permutation-based combinatorial optimization problems, $\Delta A = (\Delta a_{i,j})$ is used to compute the objective function of the optimization problems. For example, when checking if both $\pi(i) = j$ and $\pi(i') = j'$ are satisfied, a quadratic term $\Delta a_{i,j}\Delta a_{i',j'}$ is employed. However, since $\Delta a_{i,j}$s are not variables, they are expanded using the variable $a_{i,j}$s as shown below:

$$\begin{aligned}\Delta a_{i,j}\Delta a_{i',j'} &= (a_{i,j-1}-a_{i,j})(a_{i',j'-1}-a_{i',j'})\\ &= a_{i,j-1}a_{i',j'-1}-a_{i,j-1}a_{i',j'}-a_{i,j}a_{i',j'-1}+a_{i,j}a_{i',j'},\end{aligned} \tag{26}$$

Since a single quadratic term $\Delta a_{i,j}\Delta a_{i',j'}$ is expanded to the four quadratic terms involving the variable $a_{i,j}$s, the resulting QUBO model obtained through reduction contains numerous quadratic terms. To mitigate this issue, we introduce a matrix $X = (x_{i,j})$ with $n \times n$-bit variables, which satisfies $X = \Delta A$, into our QUBO kernel. Similarly, a one-hot matrix $S = (s_{i,j})$ with $n \times n$-qubit variables is added to the Ising kernel. The matrix ensures that $s_{i,j} = \Delta a_{i,j} - 1$ for all i and j, because each $\Delta a_{i,j}$ takes 0 or 2. Since these matrices X and S store a permutation as a one-hot encoding, we refer to them as *one-hot matrices*.

To extend our dual-matrix domain-wall technique for one-hot matrices, we introduce an additional condition along with the row domain-wall condition, column domain-wall condition, and dual-permutation condition. This new condition is defined as follows:

One-hot permutation condition. Each row of matrix X/S represents the permutation corresponding to matrix A in the form of a one-hot vector.

We begin by designing a QUBO model that satisfies these four conditions. The mathematical expression for this model is as follows:

$$\begin{aligned}E_e^{nn}(A,B,X) = &\frac{1}{2}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}(\Delta a_{i,j})^2+\frac{1}{2}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}(\Delta b_{i,j})^2\\ &+\frac{1}{2}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}(x_{i,j}-\Delta a_{i,j})^2+\frac{1}{2}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}(x_{i,j}-\Delta b_{i,j})^2\end{aligned} \tag{27}$$

Similarly to the previous model, the first and second summation terms achieve the minimum value of $\frac{n}{2}$ when the row and column domain-wall conditions are satisfied, respectively. The third and fourth summation terms reach the minimum value of 0 if both the dual-permutation condition and the one-hot permutation condition are met. Therefore, matrix X stores a permutation if and only if $E_e^{nn}(A,B,X)$ reaches its minimum value of n.

The same technique can also be applied to the Ising model. In this case, we introduce matrix $S = (s_{i,j})$ of size $n \times n$, where $s_{i,j} + 1 = \Delta a_{i,j}$ holds for all i and j. The Ising model $H_e^{nn}(A, B, S)$ can be designed as follows:

$$H_e^{nn}(A,B,S) = \frac{1}{2}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}(\Delta a_{i,j})^2 + \frac{1}{2}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}(\Delta b_{i,j})^2 + \frac{1}{2}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}((s_{i,j}+1) - \Delta a_{i,j})^2 + \frac{1}{2}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}((s_{i,j}+1) - \Delta b_{i,j})^2 \quad (28)$$

Similarly to the QUBO model, the four conditions are satisfied, and matrix S stores a one-hot encoding of a permutation if and only if $H_e^{nn}(A, B, S)$ reaches its minimum value of $4n$. Optimal solutions of $H_e^{nn}(A, B, S)$ correspond to a one-hot encoding stored in matrix S. The features of QUBO/Ising models obtained by expanding $E_e^{nn}(A, B, X)$ and $H_e^{nn}(A, B, S)$ are presented in Table 3.

We further apply the same technique for a partial permutation. However, a straightforward modification of $E_e^{nn}(A, B, X)/H_e^{nn}(A, B, S)$ to fit them to a partial permutation may not generate QUBO/Ising models correctly. The optimal solutions of such QUBO/Ising models may not satisfy the four conditions. We modify their fourth summation term and obtain the following QUBO model $E_e^{mn}(A, B, X)$:

$$E_e^{mn}(A,B,X) = \frac{1}{2}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(\Delta a_{i,j})^2 + \frac{1}{2}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(\Delta b_{i,j})^2 + \sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(x_{i,j} - \Delta a_{i,j})^2 + \sum_{i=0}^{m-1}\sum_{j=0}^{n-1}x_{i,j}(1 - \Delta b_{i,j}) \quad (29)$$

Note that the third and fourth summation terms do not have a coefficient of $\frac{1}{2}$, because the terms in the expanded formula will have a non-integer coefficient if they have a coefficient of $\frac{1}{2}$. The first and second summation terms take minimum values of $\frac{1}{2}m$ and $\frac{1}{2}n$, respectively, if and only if the row and column domain-wall conditions are satisfied. The third summation term takes the minimum value of 0 if X and ΔA are equal. The fourth summation term takes the minimum value of 0 if $x_{i,j} = 0$ or $\Delta b_{i,j} = 1$ holds for all i and j. Thus, $E_e^{mn}(A, B, X) = \frac{1}{2}m + \frac{1}{2}n$ if the four conditions are satisfied. We present the following lemma to prove that the "if" in the previous statement is "if and only if":

Lemma 2. *$E_e^{mn}(A, B, X)$ is larger than $\frac{1}{2}(m + n)$ if matrices A, B, and X do not satisfy at least one of the row domain-wall, column domain-wall, dual-permutation, and one-hot permutation conditions.*

Proof. For any A, B, and X, the first, second, third, and fourth summation terms in $E_e^{mn}(A, B, X)$ are at least $\frac{1}{2}m$, $\frac{1}{2}n$, 0, and 0, respectively. If the row/column domain-wall conditions are not satisfied, the first and second summation terms are larger than $\frac{1}{2}m$ and $\frac{1}{2}n$, respectively. Hence, if the row and/or the column domain-wall conditions are not satisfied, $E_e^{mn}(A, B, X) > \frac{1}{2}(m + n)$.

Suppose that the row and column domain-wall conditions are satisfied. Clearly, matrices A and B have m and n 1s, respectively, and all the other elements are 0. Thus, the first and second summation terms take minimum values of $\frac{1}{2}m$ and $\frac{1}{2}n$, respectively. If the dual-permutation condition is not satisfied, that is, ΔA and ΔB are not dual, then there exist i and j such that $\Delta a_{i,j} = 1$ and $\Delta b_{i,j} = 0$. For such an i and j, either $(x_{i,j} - \Delta a_{i,j})^2$ or $x_{i,j} \cdot (1 - \Delta b_{i,j})$ must be 1. Thus, at least one of the third or fourth summation terms is larger than 0. If the one-hot permutation condition is not satisfied, then X and ΔA are not equal, and the third summation term is larger than 0. Thus, $E_e^{mn}(A, B, X) > \frac{1}{2}m + \frac{1}{2}n$ holds. This completes the proof of Lemma 2. □

Since we have proved that $E_e^{mn}(A, B) = \frac{1}{2}m + \frac{1}{2}n$ if the four conditions are satisfied, we can state the following theorem:

Theorem 3. *$E_e^{mn}(A, B, X)$ takes the minimum value of $\frac{1}{2}m + \frac{1}{2}n$ if and only if A and B satisfy the row domain-wall, column domain-wall, dual-permutation, and one-hot permutation conditions.*

We can extend the Ising model for a partial permutation to have a one-hot matrix S in the same way as follows:

$$H_e^{mn}(A, B, S) = \frac{1}{2}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(\Delta a_{i,j})^2 + \frac{1}{2}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(\Delta b_{i,j})^2 + \sum_{i=0}^{m-1}\sum_{j=0}^{n-1}((s_{i,j}+1) - \Delta a_{i,j})^2 + \sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(s_{i,j}+1)\cdot(2 - \Delta b_{i,j}) \quad (30)$$

If the four conditions are satisfied, then the first, second, third, and fourth summation terms take the values $2m$, $2n$, 0, and 0, respectively. Thus, $H_e^{mn}(A, B, S) = 2m + 2n$ holds when the four conditions are satisfied. Similarly, we can prove that $H_e^{mn}(A, B)$ is larger than $2m + 2n$ if at least one of the four conditions is not satisfied. Thus, we can state the following theorem:

Theorem 4. *$H_e^{mn}(A, B, S)$ takes the minimum value of $2m + 2n$ if and only if A, B, and S satisfy the row domain-wall, column domain-wall, dual-permutation, and one-hot permutation conditions.*

The reader should refer to Table 4, which provides the features of $E_e^{mn}(A, B, X)$ and $H_e^{mn}(A, B, S)$. It can be observed that the number of quadratic terms in these kernels is given by $6mn - 4m - 4n$, while $E_d^{mn}(A, B)$ and $H_d^{mn}(A, B)$ have $6mn - 6m - 6n + 4$ quadratic terms. Therefore, the introduction of one-hot matrices results in a relatively small increment in the quadratic term counts. This reduction in the number of quadratic terms highlights the effectiveness of utilizing one-hot matrices X and S in the QUBO and Ising models for partial permutations. By employing these techniques, we can significantly enhance the efficiency of solving optimization problems associated with partial permutations.

7. Applications of QUBO/Ising Kernels for Generating Permutations in Combinatorial Optimization Problems

In this section, we begin by introducing *the Particle Placement Problem (PPP)*, which serves as a fundamental problem that many permutation-based combinatorial optimization problems can be reduced to with ease. We demonstrate that the PPP can be effectively reduced to QUBO/Ising models with kernels for generating permutations. Furthermore, we show that several permutation-based combinatorial optimization problems, such as the quadratic assignment problem (QAP), traveling salesman problem (TSP), and the subgraph isomorphism problem, can be equivalently reduced to the PPP. Consequently, these combinatorial optimization problems can be reduced to QUBO/Ising models through the PPP.

7.1. The Particle Placement Problem (PPP)

Suppose we have m particles with IDs $0, 1, \ldots, m-1$ and n positions with IDs $0, 1, \ldots, n-1$, where $m \leq n$ is satisfied. Let us consider the PPP, which involves placing m particles in n positions without conflicts. *The particle placement* is determined by injection $\pi : \{0, 1, \ldots, m-1\} \to \{0, 1, \ldots, n-1\}$, where each particle i ($0 \leq i \leq m-1$) is placed in position $\pi(i)$. The problem aims to find an optimal placement π that minimizes the total energy, which is defined by two types of energies as follows:

Potential $P_{i,j}$: the energy that occurs when particle i is placed in position j ($0 \leq i \leq m-1$ and $0 \leq j \leq n-1$).

Interaction $I_{i,j,i',j'}$: the energy that occurs between two particles i and i' when they are placed in positions j and j', respectively, where $(i, j, i', j') \in Q(m, n)$.

Here, $Q(m, n)$ is a set of all consistent quartets (i, j, i', j') satisfying $0 \leq i < i' \leq m-1$, $0 \leq j, j' \leq m-1$, and $j \neq j'$. Both potential $P_{i,j}$ and interaction $I_{i,j,i',j'}$ can be any integer,

including negative values. The total energy of π is the sum of all potentials and interactions, given by the equation

$$PPP(\pi) = \sum_{i=0}^{m-1} P_{i,\pi(i)} + \sum_{i=0}^{m-2} \sum_{i'=i+1}^{m-1} I_{i,\pi(i),i',\pi(i')} \tag{31}$$

The particle placement problem (PPP) aims to find an optimal placement π with the minimum $PPP(\pi)$ among all possible particle placements.

We will demonstrate that the total energy of π can be computed using the corresponding binary one-hot matrix, and we can reduce the PPP to a QUBO problem. In this sub-section, we focus on the PPP with m particles and n locations, where $m < n$ and π represents a partial permutation. However, it is worth noting that the same technique can be applied to the PPP with n particles and n locations, as it is relatively straightforward.

Suppose that an $m \times n$-bit matrix X stores a partial permutation π as one-hot vectors. In other words, $x_{i,j} = 1$ holds if and only if $\pi(i) = j$. Let $PPP(X)$ be a quadratic formula defined as follows:

$$PPP(X) = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} P_{i,j} x_{i,j} + \sum_{(i,j,i',j')}^{Q(m,n)} I_{i,j,i',j'} x_{i,j} x_{i',j'}. \tag{32}$$

Clearly, $x_{i,j} = 1$ if and only if $\pi(i) = j$, and $x_{i,j}x_{i',j'} = 1$ if and only if $\pi(i) = j$ and $\pi(i') = j'$. Thus, we have $PPP(X) = PPP(\pi)$. Since $PPP(X)$ is a quadratic formula of X, we can design a QUBO model using a QUBO kernel $E_1^{mn}(X)$ corresponding to the PPP as follows:

$$E_1^{PPP}(X) = \lambda E_1^{mn}(X) + PPP(X), \tag{33}$$

where λ is a parameter large enough to prioritize $E_1^{mn}(X)$ and guarantee that X is a one-hot vector. By finding an $n \times n$-bit matrix X that minimizes $E_1^{PPP}(X)$, we can obtain the optimal solution π of the PPP corresponding to such an X.

We can use the other QUBO kernels $E_d^{mn}(A, B)$ and $E_e^{mn}(A, B, X)$ instead of $E_1^{mn}(X)$:

$$E_d^{PPP}(A, B) = \lambda E_d^{mn}(A, B) + PPP(\Delta A) \tag{34}$$

$$E_e^{PPP}(A, B, X) = \lambda E_e^{mn}(A, B, X) + PPP(X) \tag{35}$$

Similarly, by finding their optimal solutions, we can obtain the optimal solution π of the PPP.

For an $m \times n$-qubit matrix $S = (s_{i,j})$, let $S' = (s'_{i,j})$ denote the matrix of the same size satisfying $s'_{i,j} = s_{i,j} + 1$ for all i and j. Clearly, $s'_{i,j} = 0/2$ if $s_{i,j} = -1/+1$, respectively. Thus, for $PPP(S')$ computed by the mathematical expression in Equation (32), $PPP(S') = 4PPP(\pi)$ is satisfied if S stores permutation π as one-hot vectors. Using this fact, we can design Ising models for solving the PPP as follows:

$$H_1^{PPP}(S) = \lambda H_1^{mn}(S) + PPP(S') \tag{36}$$

$$H_d^{PPP}(A, B) = \lambda H_d^{mn}(A, B) + PPP(\Delta A) \tag{37}$$

$$H_e^{PPP}(A, B, S) = \lambda H_e^{mn}(A, B, S) + PPP(S') \tag{38}$$

It should be clear that the optimal solutions of these Ising models give the optimal solutions of the PPP if λ is large enough to prioritize the Ising kernels for generating partial permutations.

The PPP can have a total of mn potentials and $\frac{1}{2}m^2n^2 - \frac{1}{2}m^2n - \frac{1}{2}mn^2 + \frac{1}{2}mn$ interactions. When the PPP is converted to the equivalent QUBO/Ising models, non-zero interactions may lead to quadratic terms. In essence, the total number of quadratic terms in the QUBO/Ising models for solving the PPP is the sum of the number of terms produced

by a kernel generating permutations and the number of quadratic terms for non-zero interactions in the PPP. As demonstrated in this paper, our dual-matrix domain-wall technique significantly reduces the count of quadratic terms. However, if the number of quadratic terms derived from non-zero interactions in the PPP is much larger than that for the kernel generating permutations, the advantage of our dual-matrix domain-wall technique becomes limited. It is worth noting that both the conventional one-hot technique and the all-different domain-wall technique require a cubic number of quadratic terms, whereas our dual-matrix domain-wall technique only uses a quadratic number of quadratic terms. If the PPP has full non-zero interactions, it would result in a number of quadratic terms that is proportional to the fourth power. In such a scenario, the resulting QUBO/Ising models would have a fourth power number of quadratic terms, and the reduction effect achieved by the dual-matrix domain-wall technique for the kernel terms would be relatively small. To evaluate the effect of the dual-matrix domain-wall technique on specific combinatorial optimization problems, we will present several examples and assess the number of non-zero interactions in the PPP.

7.2. Combinatorial Optimization Problems That Can Be Reduced to the PPP

This subsection demonstrates that several combinatorial optimization problems, namely the quadratic assignment problem (QAP), the traveling salesman problem (TSP), and the sub-graph isomorphism problem, which are known to be NP-hard [33], can be effectively reduced to the PPP. Consequently, the instances of these problems can be transformed into equivalent QUBO/Ising models. Additionally, we illustrate that the maximum weight matching problem can also be reduced to the PPP. While this problem is not NP-hard and can be solved in polynomial time [34], there is currently no known efficient parallel algorithm for general graphs [35]. Consequently, if a highly powerful QUBO/Ising solver becomes available in the future, it could be advantageous to solve this problem by utilizing the reduction to QUBO/Ising models.

7.2.1. The Quadratic Assignment Problem (QAP)

The quadratic assignment problem (QAP) [36] aims to find the optimal arrangement π of n facilities to n positions. This problem involves two types of instance parameters: flows and distances. Let $f_{i,i'}$ and $d_{j,j'}$ denote the flow from facility i to i' ($0 \leq i, i' \leq n-1$) and the distance from j to j' ($0 \leq j, j' \leq n-1$), respectively. The objective of QAP is to find a permutation π that minimizes the total logistics cost, defined by the following formula:

$$QAP(\pi) = \sum_{i=0}^{n-1} \sum_{i'=0}^{n-1} f_{i,i'} \cdot d_{\pi(i),\pi(i')}. \tag{39}$$

To reduce this problem to the PPP, we set $I_{i,j,i',j'} = f_{i,i'} \cdot d_{j,j'} + f_{i',i} \cdot d_{j',j}$ for all i, i', j, and j' ($(i,j,i',j') \in Q(n,n)$) and set $P_{i,j} = f_{i,i} \cdot d_{j,j}$ for all i and j ($0 \leq i, j \leq n-1$). Since $QAP(\pi) = PPP(\pi)$ holds for all permutation π, any QAP instance can be reduced to the PPP. The reduced PPP has up to $\frac{1}{2}n^4 - n^3 + \frac{1}{2}n^2$ non-zero interactions and n^2 non-zero potentials. If not all flows are non-zero, the number of non-zero interactions may be reduced. Let k be the number of pairs of facilities i and i' ($0 \leq i < i' \leq n-1$) such that at least one of $f_{i,i'}$ or $f_{i',i}$ is non-zero. Clearly, k is at most $\frac{1}{2}n^2 - \frac{1}{2}n$. In this case, the reduced PPP has only $\frac{1}{2}n^2k - \frac{1}{2}nk$ non-zero interactions.

7.2.2. The Traveling Salesman Problem (TSP)

The traveling salesman problem (TSP), which aims to find the shortest route to visit all n cities with IDs 0, 1, ..., $n-1$ can be reduced to the PPP as follows. Let $d_{j,j'}$ ($0 \leq j, j' \leq n-1$) denote the distance between two cities j and j'. We use a permutation π for representing a route, where the i-th visited city ($0 \leq i \leq n-1$) corresponds to the city

with ID $\pi(i)$. The objective of the TSP is to find a permutation π that minimizes the total route length computed using the following formula:

$$TSP(\pi) = \sum_{i=0}^{n-1} d_{\pi(i),\pi((i'+1) \bmod n)}. \tag{40}$$

To reduce the TSP to the PPP, we define $I_{i,j,i',j'} = d_{j,j'}$ for all i, i', j, and j' if $i' = (i+1) \bmod n$. For all other $I_{i,j,i',j'}$ and $P_{i,j}$, we set them to 0. This reduction allows us to observe that $TSP(\pi) = PPP(\pi)$ holds for all permutations π. Therefore, it is possible to reduce any TSP instance to the PPP. The reduced PPP has $n^3 - n^2$ non-zero interactions.

We can consider the traveling salesman problem (TSP) for a weighted undirected graph in which each edge is assigned a weight value. The goal of the TSP is to find a Hamiltonian cycle in the graph with the minimum total weight that visits all nodes. Figure 7 shows an example of a 40-node weighted graph. We assume that the weight of each edge is its length in the figure. The figure also depicts the optimal solution of the TSP. Note that if an input graph has no Hamiltonian cycle, the TSP has no feasible solution.

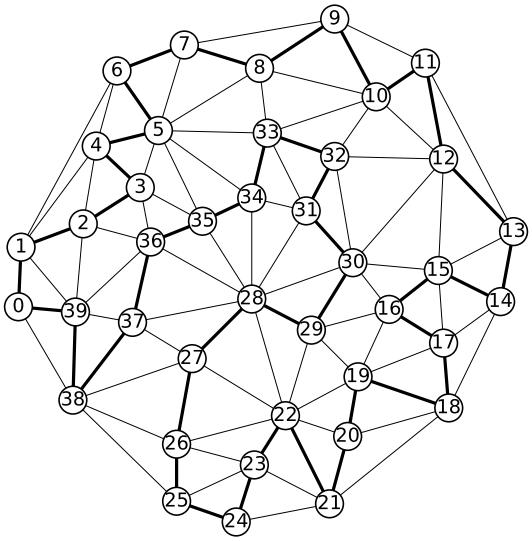

Figure 7. A 40-node weighted graph with the optimal TSP solution.

Suppose that we have a weighted graph with nodes 0, 1, ..., $n-1$, and let $d_{j,j'}$ ($0 \le j, j' \le n-1$) be the weight of an edge (j, j'). If the graph does not have an edge (j, j'), we assume $d_{j,j'} = +\infty$ to prevent selecting (j, j') as an edge of a cycle. The total route of a node permutation π is computed using Equation (40), based on the defined $d_{j,j'}$ values. For the real implementation, we set $d_{j,j'} = BIG$ instead of $+\infty$, where BIG is sufficiently larger than the maximum edge weight. An edge (j, j') with $d_{j,j'} = BIG$ is selected only if the TSP has no feasible solution.

We will introduce the technique for reducing the TSP into the PPP, so that the reduced PPP becomes sparse. For this purpose, we apply a fixed bias of $-BIG$, so that $I_{i,j,i',j'} = d_{j,j'} - BIG$ for all i, i', j, and j' if $i' = (i+1) \bmod n$. For all other $I_{i,j,i',j'}$ and $P_{i,j}$, we set them to 0. This reduction allows us to observe that $TSP(\pi) = PPP(\pi) + n \cdot BIG$ holds for all permutations π. Additionally, $I_{i,j,i',j'}$ is non-zero only if the graph has an edge (j, j') and $i' = (i+1) \bmod n$ holds. Thus, the PPP becomes so sparse that it has only $2en$ non-zero interactions, where e is the number of edges in the graph. For instance, if no two edges intersect each other, as illustrated in Figure 7, the graph has at most $3n - 6$ edges. Hence, the reduced PPP has at most $2(3n-6)n = 6n^2 - 12$ interactions, which is much smaller than the original TSP with $n^3 - n^2$ interactions.

7.2.3. The Sub-Graph Isomorphism Problem

Let $G = (V, E)$ and $G' = (V', E')$ be guest and host graphs such that $V = \{0, 1, \ldots, m-1\}$ and $V' = \{0, 1, \ldots, n-1\}$. The sub-graph isomorphism problem aims to find a permutation π such that $(\pi(i), \pi(i')) \in E'$ holds for all $(i, i') \in E$. If such a permutation π exists, it implies that G is a sub-graph of G'. To reduce the sub-graph isomorphism problem to the PPP, we assign -1 to $I_{i,j,i',j'}$ and $I_{i,j',i',j}$ if both $(i, i') \in E$ and $(j, j') \in E'$ hold, and assign 0 to all other $I_{i,j,i',j'}$ as well as all $P_{i,j}$. It is evident that $I_{i,\pi(i),i',\pi(i')} = -1$ holds for all edges (i, i') in G if π is a permutation that proves G is a sub-graph of G'. Therefore, $PPP(\pi) = -|E|$ if such a permutation exists. Consequently, any instance of the sub-graph isomorphism problem can be reduced to the PPP. The reduced PPP has $2 \cdot |E| \cdot |E'|$ non-zero interactions. If G and G' are connected graphs, then $m - 1 \leq |E| \leq \frac{1}{2}m^2 - \frac{1}{2}m$ and $n - 1 \leq |E'| \leq \frac{1}{2}n^2 - \frac{1}{2}n$ hold. Hence, the reduced PPP has from $2mn - 2m - 2n + 2$ to $\frac{1}{2}m^2n^2 - \frac{1}{2}m^2n - \frac{1}{2}mn^2 + \frac{1}{2}mn$ non-zero interactions.

7.2.4. The Maximum Weight Matching Problem

Let $G = (V, E, w)$ be a weighted graph, where $V = \{0, 1, \ldots, n-1\}$ and $w_{i,j}$ represents the weight of an edge (i, j) $(i < j)$ in E. We assume that $w_{i,j} = 0$ for all other pairs (i, j) such that $i \geq j$ or (i, j) is not in E. A subset of E is called a matching if none of its edges share a common node. The maximum weight matching problem aims to find a matching M $(\subseteq E)$ with the largest total weight. We can represent a matching M using a permutation π as follows. An edge (i, j) $(\in E)$ is in the matching M if both $\pi(i) = j$ and $\pi(j) = i$ hold, and node i is not connected to any edge in M if $\pi(i) = i$. To transform the maximum weight matching problem into the PPP, we assign $-w_{i,j}$ to $I_{i,j,j,i}$ for all $(i, j) \in E$, while assigning 0 to all other $I_{i,j,i',j'}$ as well as all $P_{i,j}$. Let π be a permutation that represents a matching M. Since $w_{i,\pi(i)} = 0$ whenever $i \geq \pi(i)$, we can compute the total weight of M using the obtained PPP as follows:

$$\sum_{(i,j)}^{M} w_{i,j} = \sum_{i=0}^{n-1} w_{i,\pi(i)} = -\sum_{i=0}^{n-1} I_{i,\pi(i),\pi(i),i} = -PPP(\pi). \tag{41}$$

Therefore, any instance of the maximum weight matching problem can be reduced to the PPP with $|E|$ interactions. If G is a connected graph, $n - 1 \leq |E| \leq \frac{1}{2}n^2 - \frac{1}{2}n$ holds, and so the number of interactions is from $n - 1$ to $\frac{1}{2}n^2 - \frac{1}{2}n$.

7.2.5. The Bipartite Maximum Weight Matching Problem

If an instance of the maximum weight matching problem is restricted to a bipartite graph, a more efficient reduction to the PPP can be used. Consider a weighted bipartite graph, denoted as $G = (V, V', E, w)$, where $V = \{0, 1, \ldots, m-1\}$, $V' = \{0, 1, \ldots, n-1\}$, $E \subseteq V \times V'$, and $w_{i,j}$ represents the weight of an edge $(i, j) \in E$. We can use a partial permutation $\pi : V \to V'$ to represent a matching M of G such that $(i, \pi(i)) \in M$. To transform the maximum weight matching problem for the weighted bipartite graph into the PPP, we assign $-w_{i,j}$ to $P_{i,j}$ for all $(i, j) \in E$, while assigning 0 to all other $P_{i,j}$ as well as all $I_{i,j,i',j'}$. Thus, we can compute the total weight of M using the obtained PPP as follows:

$$\sum_{(i,j)}^{M} w_{i,j} = \sum_{i=0}^{m-1} w_{i,\pi(i)} = -\sum_{i=0}^{m-1} P_{i,\pi(i)} = -PPP(\pi). \tag{42}$$

Therefore, any instance of the bipartite maximum weight matching problem can be reduced to the PPP. The resulting PPP has $|E|$ non-zero potentials but has no non-zero interactions.

7.2.6. The Quadratic Term Counts of the Ising Model

This section presents the quadratic term counts for various permutation-based combinatorial optimization problems. We examined the total number of quadratic terms in Ising models and permutation kernels to observe the impact of smaller permutation kernels on the size of Ising models. To facilitate this analysis, we introduced the concept of PPP density, which represents the ratio of non-zero interactions to the maximum number of interactions $|Q(m,n)| = \frac{1}{2}m^2n^2 - \frac{1}{2}m^2n - \frac{1}{2}mn^2 + \frac{1}{2}mn$.

We used PyQUBO [37], a Python tool designed to generate QUBO/Ising models from mathematical expressions, and SymPy [32], a Python library for symbolic mathematics, to validate the accuracy of the QUBO/Ising models. However, this approach, which operates on provided mathematical expressions, has a significant drawback in terms of its substantial time and space requirements. This limitation could make it impractical to solve particularly large problems. For optimization problems based on specific permutations, an alternative and more efficient approach is to create a custom computer program for generating QUBO/Ising models. This approach can leverage a low-level programming language like C++. Given the fixed and straightforward nature of the underlying mathematical expressions for such problems, it becomes feasible to design a program that efficiently generates QUBO/Ising models without the need to expand the mathematical expressions. This process can be achieved with linear time complexity and requires a memory proportional to the size of the models.

Figure 8 illustrates the quadratic term counts for the following problem instances:

1. A random instance of the QAP with $n = 50$: The reduced PPP consists of all 3,001,250 interactions, yielding a density of 1.00.
2. A random instance of the QAP with $n = 50$ and 10% non-zero flows: The reduced PPP contains 301,350 interactions out of 3,001,250 maximum interactions, yielding a density of 0.10.
3. A random instance of the TSP with $n = 100$: The reduced PPP includes 990,000 interactions out of 49,005,000 maximum interactions, yielding a density of 0.020.
4. The TSP for a randomly generated 40-node weighted graph as shown in Figure 7: The graph has 104 edges, and the reduced PPP includes 8320 interactions out of 1,216,800 maximum interactions, yielding a density of 0.0068.
5. The TSP for a randomly generated 300-node weighted graph: The graph is generated to avoid edge intersections, similar to the 40-node weighted graph. It has 277 edges, and the reduced PPP includes 524,400 interactions out of a maximum of 4,023,045,000 interactions, yielding a density of 0.00013.
6. A random instance of the sub-graph isomorphism problem for a three-regular guest graph with $m = 200$ nodes and a six-regular host graph with $n = 400$ nodes: The reduced PPP consists of 720,000 interactions out of 3,176,040,000 maximum interactions, yielding a density of 0.00023.
7. A random instance of the maximum weight matching problem for a complete graph with $n = 300$ nodes: The reduced PPP contains 44,850 interactions out of 4,023,045,000 maximum interactions, yielding a density of 0.000011.
8. A random instance of the bipartite maximum weight matching problem for a complete bipartite graph with $m = n = 300$ nodes: The reduced PPP has no interactions, yielding a density of zero.

The figure illustrates the total number of quadratic terms in the resulting Ising model, referred to as the "model total", for each problem. Additionally, it displays the number of quadratic terms contributed by the permutation kernels, referred to as the "permutation kernel", which is also included in the model total. The quadratic term counts are evaluated for four permutation kernels: the conventional one-hot technique (one-hot), the all-different domain-wall technique (all-different), the dual-matrix domain-wall technique (dual-matrix), and the extended technique with a one-hot matrix (extended).

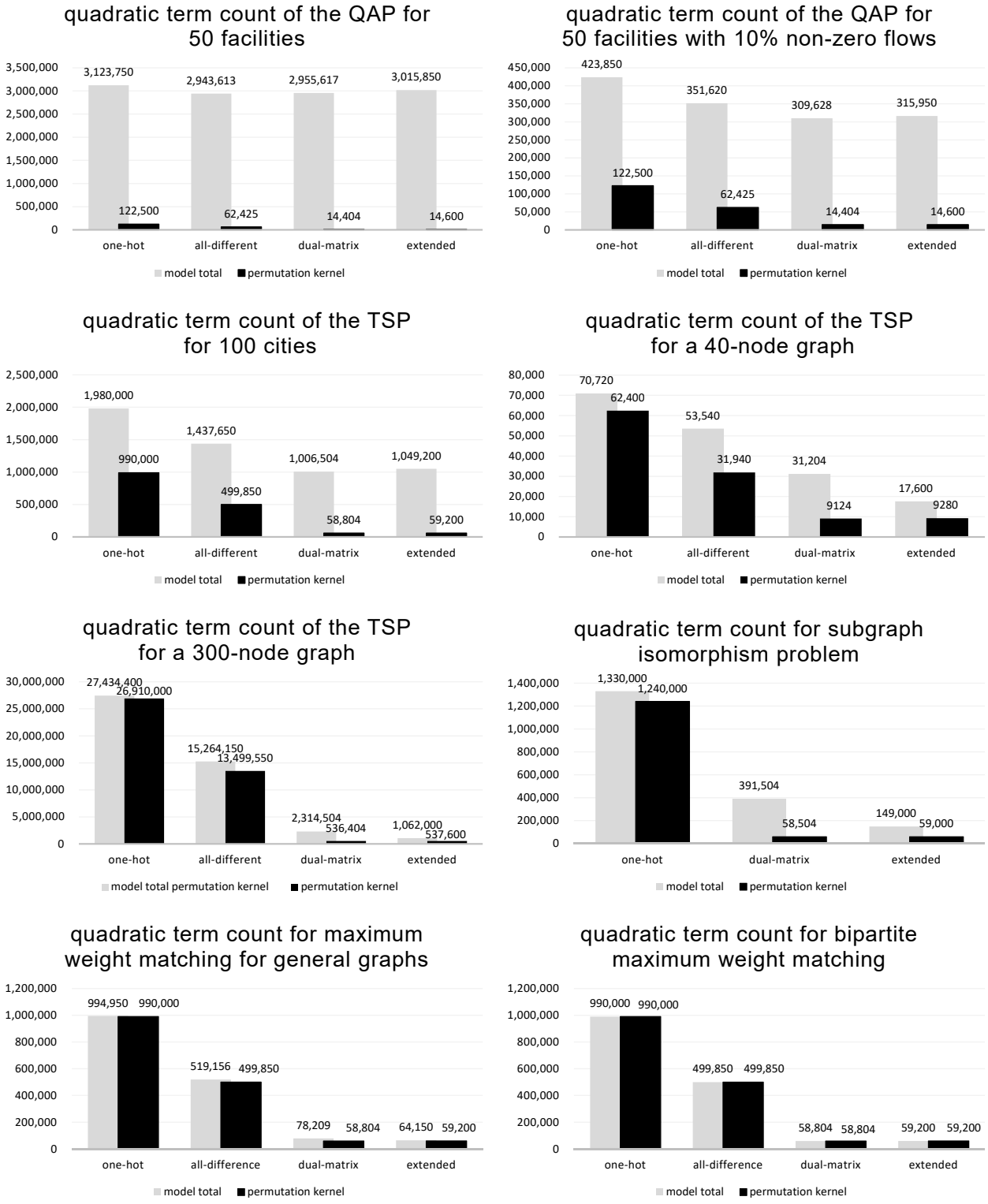

Figure 8. The quadratic term counts of Ising models reduced from combinatorial optimization problems by the conventional one-hot technique (one-hot), the all-different domain-wall technique (all-different), the dual-matrix domain-wall technique (dual-matrix), and that extended with a one-hot matrix (extended). We evaluated the total quadratic term counts of the resulting Ising models (model total), as well as those specifically for kernels generating permutations (permutation kernel).

From the figure, it is evident that our dual-matrix domain-wall technique is more effective in reducing the quadratic terms of resulting Ising models when the original PPPs have a lower density. When the density of the PPP is close to 1, the quadratic terms for

PPP interactions outweigh those for permutation kernels. On the other hand, when the PPP has a lower density, the total number of quadratic terms in the resulting Ising model can be significantly reduced. For instance, if the conventional one-hot encoding is used for the TSP with a randomly generated 300-node graph, the total number of quadratic terms in the resulting Ising model is 27,434,400. However, when our dual-matrix domain-wall technique extended by a one-hot matrix is applied, the total number of quadratic terms reduces to only 1,062,000. This represents a remarkable reduction factor of 25.8.

We should take note of the impact of extending by a one-hot matrix in our dual-matrix domain-wall technique. Recall that the original technique utilizes variables in matrix A for PPP interactions, while the extended technique introduces an one-hot matrix X/S. As shown in Equation (26), the dual-matrix domain-wall technique generates four quadratic terms of A for each PPP interaction. In contrast, the extended dual-matrix domain-wall technique produces only one quadratic term. When the PPP density is high and involves numerous interactions, the quadratic terms generated by A may overlap, resulting in the Ising model encompassing nearly all pairs of variables in A. Similarly, the Ising models produced by the extended dual-matrix domain-wall technique also include almost all pairs of variables in X/S. Consequently, the number of quadratic terms is nearly identical for both techniques, and the extended dual-matrix domain-wall technique does not effectively reduce them. Likewise, when the PPP density is too low and contains few interactions, the number of quadratic terms generated by variables $A/X/S$ is small. Since the extended dual-matrix domain-wall technique yields a slightly larger kernel size, it does not perform well in reducing the overall number of quadratic terms. Therefore, the extended dual-matrix domain-wall technique can effectively reduce the total number of quadratic terms only when the PPP density is moderate—neither too small nor too large. An example illustrating this is the sub-graph isomorphism problem depicted in Figure 8, where the extended dual-matrix domain-wall technique reduces the total number of quadratic terms to less than half. The quadratic term counts of the extended technique are slightly larger than those of the non-extended technique for the other problem. However, since the difference is quite small, we should select the extended dual-matrix domain-wall technique if we must use a fixed technique for some reason.

8. Embedding of Permutation Kernels in D-Wave Quantum Annealer

This section presents the implementation results of Ising kernels on a quantum annealer called the D-Wave Advantage 4.1. The purpose was to evaluate the suitability of the dual-matrix domain-wall technique for currently available quantum annealers and to verify the impact of quadratic term counts in Ising models on the cell counts. We employed Ising kernels designed using mathematical expressions such as $H_1^{nn}(S)$ (one-hot), $H_a^{nn}(S)$ (all-different), $H_d^{nn}(S)$ (domain-wall), and $H_e^{nn}(S)$ (extended) for generating permutations. Additionally, we used $H_1^{mn}(S)$ (one-hot), $H_d^{mn}(S)$ (domain-wall), and $H_e^{mn}(S)$ (extended) for partial permutations. These Ising kernels were embedded in the D-Wave Advantage 4.1, which was equipped with a 5760-node Pegasus graph designed for solving Ising models. However, due to faulty nodes and connections, only 5627 cells with 40,279 connections were available. Therefore, we retrieved the real graph topology and employed "minorminer", a heuristic tool for finding minor embeddings [38], to embed the Ising kernels.

Figure 9 illustrates the number of cells used to embed Ising kernels in the D-Wave Advantage 4.1. We embedded Ising kernels to generate permutations obtained by $H_1^{nn}(S)$ (one-hot), $H_a^{nn}(S)$ (all-different), $H_e^{nn}(S)$ (domain-wall), and $H_d^{nn}(S)$ (extended) for $n = 5$ until embedding failed. For partial permutations, we fixed the parameter $m = 12$ and evaluated embedding for $H_1^{mn}(S)$ (one-hot), $H_d^{mn}(S)$ (domain-wall), and $H_e^{mn}(S)$ (extended) for $n = 12$ until embedding failed. Ising kernels for generating permutations using the one-hot, all-different, domain-wall, and extended techniques were successfully embedded in the D-Wave Advantage 4.1 topology until $n = 16, 28, 35$, and 29, respectively. Regarding Ising kernels for partial-permutation generation by the one-hot, domain-wall, and extended techniques, successful embedding was achieved until $n = 22, 112$, and 75, respectively.

Our dual-matrix domain-wall technique significantly reduced the cell count required for permutation generation. However, the permutation kernels produced by the extended dual-matrix domain-wall technique utilize slightly more cells than those produced by the non-extended version. Additionally, the total number of cells used by the extended technique can be smaller, because it uses fewer quadratic terms for PPP interactions.

If the entire Ising models for solving combinatorial optimization problems are embedded, the upper bound for the value of n must be smaller. For such small values of n, conventional digital computers are capable of finding optimal solutions for the problems. Therefore, it does not make sense to utilize the D-Wave Advantage 4.1 for solving such problems. However, in the future, if a larger number of cells becomes available, the D-Wave Advantage 4.1 could potentially load Ising models reduced from larger combinatorial optimization problems. In this scenario, it could uncover optimal solutions that are beyond the reach of conventional digital computers. In such cases, kernels for generating permutations with larger values of m and n would be employed, and our dual-matrix domain-wall technique would be a powerful approach for designing them. For a quantum annealer with a fixed size, our dual-matrix domain-wall technique enables us to program larger-sized combinatorial optimization problems than the conventional one-hot encoding techniques, which require a cubic number of quadratic terms.

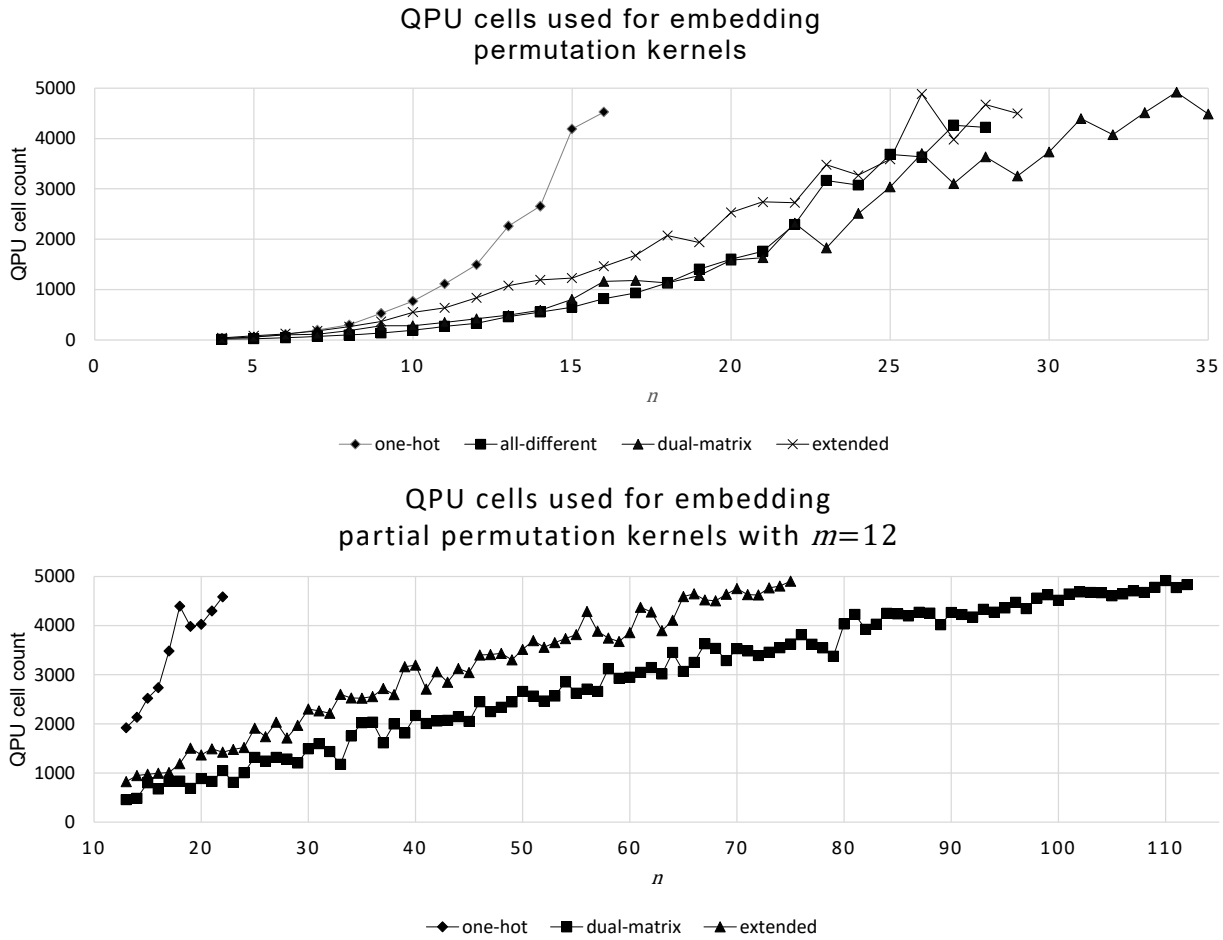

Figure 9. The numbers of cells of the D-Wave Advantage 4.1 used for embedding permutation kernels. We evaluated them for Ising kernels that generate permutations of n numbers and partial permutations of $m = 12$ numbers selected from n numbers.

9. Conclusions

We introduced a novel permutation encoding technique called the dual-matrix domain wall, which can be utilized to design QUBO/Ising kernels for generating permutations and

partial permutations. The conventional one-hot encoding technique for QUBO/Ising kernels requires a cubic number of quadratic terms, whereas our dual-matrix domain-wall technique enables a reduction to a quadratic number of quadratic terms. In this study, we provided a detailed mathematical analysis of QUBO/Ising kernels for permutation generation.

Additionally, we introduced a generic problem known as the particle placement problem (PPP), which can be easily converted to QUBO/Ising models by utilizing the QUBO/Ising kernels for generating permutations. Furthermore, we identified several permutation-based combinatorial optimization problems that can be reduced to the PPP. As a result, these problems can be efficiently converted to QUBO/Ising models using our dual-matrix domain-wall technique.

To evaluate the effectiveness of our approach, we analyzed the number of quadratic terms used in the QUBO/Ising models derived from these optimization problems. Finally, we provided implementation results on the D-Wave Advantage 4.1, a quantum annealer developed by D-Wave Systems.

Regrettably, the existing quantum annealers currently accessible are of insufficient size to showcase the practical efficacy of our novel technique. From a practical perspective, our methodology is specifically tailored for optimal performance exclusively on quantum annealers of a substantial scale. The dual-matrix domain-wall technique is not presently feasible with the D-Wave Advantage 4.1. Nonetheless, it possesses the potential to evolve into a valuable technique for forthcoming large-scale quantum annealers.

Author Contributions: Conceptualization, K.N.; methodology, K.N., S.T. and S.O.; validation, Y.I. and T.Y.; formal analysis, S.T., T.K., R.M. and R.K.; investigation, K.N., Y.I., J.Y., T.K. and S.O.; writing—original draft preparation, K.N.; writing—review and editing, K.N., J.Y., T.K., R.M. and R.K.; visualization, K.N.; supervision, K.N.; project administration, T.Y.; funding acquisition, T.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: A Python program for generating QUBO/Ising models, as utilized in this paper, is available on GitHub at Available online: https://github.com/nakanocs/dual-matrix (accessed on 16 October 2023).

Conflicts of Interest: Koji Nakano, Takashi Yazane, Junko Yano, Takumi Kato, Shiro Ozaki, Rie Mori, and Ryota Katsuki are the inventors listed on Japanese Patent Application No. 2023-125848. This manuscript discusses the inventions outlined in this patent application.

References

1. D-Wave Systems. Dimod—D-Wave Ocean Software Documentation. Available online: https://docs.ocean.dwavesys.com/en/stable/docs_dimod/reference/index.html (accessed on 16 October 2023).
2. Kochenberger, G.A.; Hao, J.; Glover, F.W.; Lewis, M.W.; Lü, Z.; Wang, H.; Wang, Y. The unconstrained binary quadratic programming problem: A survey. *J. Comb. Optim.* **2014**, *28*, 58–81. [CrossRef]
3. Tao, M.; Nakano, K.; Ito, Y.; Yasudo, R.; Tatekawa, M.; Katsuki, R.; Yazane, T.; Inaba, Y. A Work-Time Optimal Parallel Exhaustive Search Algorithm for the QUBO and the Ising model, with GPU implementation. In Proceedings of the International Parallel and Distributed Processing Symposium Workshops, New Orleans, LA, USA, 18–22 May 2020; pp. 557–566.
4. Lucas, A. Ising formulations of many NP problems. *Front. Phys.* **2014**, *2*, 5. [CrossRef]
5. Kadowaki, T.; Nishimori, H. Quantum annealing in the transverse Ising model. *Phys. Rev. E* **1998**, *E58*, 5355–5363. [CrossRef]
6. Zaborniak, T.; de Sousa, R. Benchmarking Hamiltonian Noise in the D-Wave Quantum Annealer. *IEEE Trans. Quantum Eng.* **2021**, *2*, 3100206. [CrossRef]
7. Oku, D.; Terada, K.; Hayashi, M.; Yamaoka, M.; Tanaka, S.; Togawa, N. A Fully-Connected Ising Model Embedding Method and Its Evaluation for CMOS Annealing Machines. *IEICE Trans. Inf. Syst.* **2019**, *102*, 1696–1706. [CrossRef]
8. Yamamoto, K.; Kawamura, K.; Ando, K.; Mertig, N.; Takemoto, T.; Yamaoka, M.; Teramoto, H.; Sakai, A.; Takamaeda-Yamazaki, S.; Motomura, M. STATICA: A 512-Spin 0.25M-Weight Annealing Processor with an All-Spin-Updates-at-Once Architecture for Combinatorial Optimization with Complete Spin–Spin Interactions. *IEEE J. Solid-State Circuits* **2021**, *56*, 165–178. [CrossRef]

9. Goto, H.; Endo, K.; Suzuki, M.; Sakai, Y.; Kanao, T.; Hamakawa, Y.; Hidaka, R.; Yamasaki, M.; Tatsumura, K. High-performance combinatorial optimization based on classical mechanics. *Sci. Adv.* **2021**, *7*, eabe7953. [CrossRef]
10. Goto, H.; Tatsumura, K.; Dixon, A.R. Combinatorial optimization by simulating adiabatic bifurcations in nonlinear Hamiltonian systems. *Sci. Adv.* **2019**, *5*, eaav2372. [CrossRef]
11. Kagawa, H.; Ito, Y.; Nakano, K.; Yasudo, R.; Kawamata, Y.; Katsuki, R.; Tabata, Y.; Yazane, T.; Hamano, K. High-throughput FPGA implementation for quadratic unconstrained binary optimization. *Concurr. Comput. Pract. Exp.* **2021**, *35*, e6565. [CrossRef]
12. Matsubara, S.; Tamura, H.; Takatsu, M.; Yoo, D.; Vatankhahghadim, B.; Yamasaki, H.; Miyazawa, T.; Tsukamoto, S.; Watanabe, Y.; Takemoto, K.; et al. Ising-Model Optimizer with Parallel-Trial Bit-Sieve Engine. In Proceedings of the International Conference on Complex, Intelligent, and Software Intensive Systems, Torino, Italy, 28–30 June 2017; pp. 432–438.
13. Okuyama, T.; Sonobe, T.; Kawarabayashi, K.; Yamaoka, M. Binary optimization by momentum annealing. *Phys. Rev. E* **2019**, *100*, 012111. [CrossRef]
14. Yasudo, R.; Nakano, K.; Ito, Y.; Katsuki, R.; Tabata, Y.; Yazane, T.; Hamano, K. GPU-accelerated Scalable Solver with Bit Permutated Cyclic-Min Algorithm for Quadratic Unconstrained Binary Optimization. *J. Parallel Distrib. Comput.* **2022**, *167*, 109–122. [CrossRef]
15. Yasudo, R.; Nakano, K.; Ito, Y.; Kawamata, Y.; Katsuki, R.; Ozaki, S.; Yazane, T.; Hamano, K. Graph-theoretic Formulation of QUBO for Scalable Local Search on GPUs. In Proceedings of the International Parallel and Distributed Processing Systems Workshops, Lyon, France, 30 May–3 June 2022; pp. 425–434.
16. Nakano, K.; Takafuji, D.; Ito, Y.; Yazane, T.; Yano, J.; Ozaki, S.; Katsuki, R.; Mori, R. Diverse Adaptive Bulk Search: A Framework for Solving QUBO Problems on Multiple GPUs. In Proceedings of the IEEE International Parallel and Distributed Processing Symposium Workshops, St. Petersburg, FL, USA, 15–19 May 2023; pp. 314–325.
17. Honjo, T.; Sonobe, T.; Inaba, K.; Inagaki, T.; Ikuta, T.; Yamada, Y.; Kazama, T.; Enbutsu, K.; Umeki, T.; Kasahara, R.; et al. 100,000-spin coherent Ising machine. *Sci. Adv.* **2021**, *7*, eabh0952. [CrossRef] [PubMed]
18. Inagaki, T.; Haribara, Y.; Igarashi, K.; Sonobe, T.; Tamate, S.; Honjo, T.; Marandi, A.; McMahon, P.L.; Umeki, T.; Enbutsu, K.; et al. A coherent Ising machine for 2000-node optimization problems. *Science* **2016**, *354*, 603–606. [CrossRef] [PubMed]
19. McGeoch, C.; Farré, P.; Bernoudy, W. *D-Wave Hybrid Solver Service + Advantage: Technology Update*; Technical Report; D-Wave Systems: Burnaby, BC, USA, 2020.
20. Ayodele, M. Penalty Weights in QUBO Formulations: Permutation Problems. In Proceedings of the Evolutionary Computation in Combinatorial Optimization. EvoCOP 2022. Lecture Notes in Computer Science, Madrid, Spain, 20–22 April 2022; Volume 13222, pp. 159–174.
21. Goh, S.T.; Bo, J.; Gopalakrishnan, S.; Lau, H.C. Techniques to enhance a QUBO solver for permutation-based combinatorial optimization. In Proceedings of the GECCO'22: Genetic and Evolutionary Computation Conference, Companion, Boston, MA, USA, 9–13 July 2022; pp. 2223–2231.
22. Codognet, P. Domain-Wall/Unary Encoding in QUBO for Permutation Problems. In Proceedings of the International Conference on Quantum Computing and Engineering, Broomfield, CO, USA, 18–23 September 2022; pp. 167–173.
23. Chancellor, N. Domain wall encoding of discrete variables for quantum annealing and QAOA. *Quantum Sci. Technol.* **2019**, *4*, 045004. [CrossRef]
24. Chen, J.; Stollenwerk, T.; Chancellor, N. Performance of Domain-Wall Encoding for Quantum Annealing. *IEEE Trans. Quantum Eng.* **2021**, *2*, 3102714. [CrossRef]
25. Berwald, J.; Chancellor, N.; Dridi, R. Understanding domain-wall encoding theoretically and experimentally. *Phil. Trans. R. Soc. A* **2022**, *381*, 20210410. [CrossRef]
26. Calude, C.S.; Dinneen, M.J.; Hua, R. QUBO formulations for the graph isomorphism problem and related problems. *Theor. Comput. Sci.* **2017**, *701*, 54–69. [CrossRef]
27. Yoshimura, N.; Tawada, M.; Tanaka, S.; Arai, J.; Yagi, S.; Uchiyama, H.; Togawa, N. Mapping Induced Subgraph Isomorphism Problems to Ising Models and Its Evaluations by an Ising Machine. *IEICE Trans. Inf. Syst.* **2021**, *104*, 481–489. [CrossRef]
28. McGeoch, C.C.; Harris, R.; Reinhardt, S.P.; Bunyk, P. Practical Annealing-Based Quantum Computing. *IEEE Comput.* **2019**, *52*, 38–46. [CrossRef]
29. McGeoch, C.; Farré, P. *The D-Wave Advantage System: An Overview*; Technical Report; D-Wave Systems: Burnaby, BC, USA, 2020.
30. Boothby, K.; Bunyk, P.; Raymond, J.; Roy, A. *Next-Generation Topology of D-Wave Quantum Processors*; Techreport 14-1026A-C; D-Wave Systems: Burnaby, BC, USA, 2019.
31. D-Wave Systems. Programming the D-Wave QPU: Setting the Chain Strength. 2020. Available online: https://www.dwavesys.com/media/vsufwv1d/14-1041a-a_setting_the_chain_strength.pdf (accessed on 16 October 2023).
32. SymPy Development Team. SymPy Documentation Release 1.12. 2023. Available online: https://www.sympy.org/en/index.html (accessed on 16 October 2023).
33. Garey, M.R.; Johnson, D.S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*; W H Freeman & Co.: New York, NY, USA, 1979.
34. Edmonds, J. Paths, Trees, and Flowers. *Can. J. Math.* **1965**, *17*, 449–467. [CrossRef]
35. Eppstein, D.; Vazirani, V.V. NC Algorithms for Computing a Perfect Matching and a Maximum Flow in One-Crossing-Minor-Free Graphs. *SIAM J. Comput.* **2021**, *50*, 1014–1033. [CrossRef]
36. Koopmans, T.C.; Beckmann, M. Assignment Problems and the Location of Economic Activities. *Econometrica* **1957**, *25*, 53–76. [CrossRef]

37. Recruit Communications Co., Ltd. Pyqubo Documentation. 2022. Available online: https://pyqubo.readthedocs.io/ (accessed on 16 October 2023).
38. Cai, J.; Macready, W.G.; Roy, A. A practical heuristic for finding graph minors. *arXiv* **2014**, arXiv:1406.2741.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.