IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 10 June 2024, accepted 3 July 2024, date of publication 9 July 2024, date of current version 23 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3425711

RESEARCH ARTICLE

Toward Practical Benchmarks of Ising Machines: A Case Study on the Quadratic Knapsack Problem

KENTARO OHNO[1,2], TATSUHIKO SHIRAI[3], (Member, IEEE), AND NOZOMU TOGAWA[2], (Member, IEEE)

[1]NTT, Tokyo 180-8585, Japan
[2]Department of Computer Science and Communications Engineering, Waseda University, Tokyo 169-8555, Japan
[3]Waseda Institute for Advanced Study, Waseda University, Tokyo 169-0051, Japan

Corresponding author: Kentaro Ohno (kentaro.ohno@togawa.cs.waseda.ac.jp)

ABSTRACT Combinatorial optimization has wide applications from industry to natural science. Ising machines bring an emerging computing paradigm for efficiently solving a combinatorial optimization problem by searching a ground state of a given Ising model. Current cutting-edge Ising machines achieve fast sampling of near-optimal solutions of the max-cut problem. However, for problems with additional constraint conditions, their advantages have been hardly shown due to difficulties in handling the constraints. In this work, we focus on benchmarks of Ising machines on the quadratic knapsack problem (QKP). To bring out their practical performance, we propose fast two-stage post-processing for Ising machines, which makes handling the constraint easier. Simulation based on simulated annealing shows that the proposed method substantially improves the solving performance of Ising machines and the improvement is robust to a choice of encoding of the constraint condition. Through evaluation using an Ising machine called Amplify Annealing Engine, the proposed method is shown to dramatically improve its solving performance on the QKP. These results are a crucial step toward showing advantages of Ising machines on practical problems involving various constraint conditions.

INDEX TERMS Combinatorial optimization, Ising machine, quadratic knapsack problem.

I. INTRODUCTION

Combinatorial optimization is an important research area with applications in various fields such as artificial intelligence and operations research. For example, the knapsack problem and its variants are famous and well-studied combinatorial optimization problems with numerous applications including production planning, resource allocation, and portfolio selection [1]. Theoretically, combinatorial optimization problems are often hard to solve exactly within a reasonable amount of time due to their NP-hardness. Therefore, various heuristics and meta-heuristics have been developed for dealing with large-scale combinatorial optimization problems.

Ising machines offer a new computing paradigm for tackling hard combinatorial optimization problems [2]. Ising machines search a ground state of a given Ising model, a model in statistical mechanics involving binary variables (called spins) and their interactions, and thus can be used for optimization over binary variables. For problems with additional constraint conditions on binary variables, the *penalty method* is typically used [3]. A constraint on binary variables $x = (x_1, \cdots, x_n)$ is translated into a penalty term $H_{\mathrm{con}}(x)$ added to the objective function $H_{\mathrm{obj}}(x)$ with a positive coefficient $\lambda > 0$ to construct an unconstrained binary optimization problem

$$\begin{aligned} &\text{minimize } H_{\mathrm{obj}}(x) + \lambda H_{\mathrm{con}}(x) \\ &\text{subject to } x \in \{0, 1\}^n, \end{aligned} \tag{1}$$

to which an Ising machine is applied. There exist several types of Ising machines depending on the way of physical implementation: examples are quantum annealers [4], [5],

The associate editor coordinating the review of this manuscript and approving it for publication was Tomás F. Pena.

coherent Ising machines [6], [7], and specialized-circuit-based digital machines [8], [9], [10], [11], [12]. These machines enable fast sampling of near-optimal solutions on the max cut problem, which is naturally formulated with an Ising model.

However, for problems with additional constraint conditions on binary variables, the superiority of Ising machines to other methods has not been observed. For example, previous benchmark results [13], [14], [15], [16] on the quadratic knapsack problem (QKP) and quadratic assignment problem (QAP) show that Ising machines are not competitive with existing (meta-)heuristic solvers. A critical performance issue is that Ising machines do not necessarily output feasible solutions, i.e., solutions satisfying constraints. The penalty coefficient λ in (1) is required to be large for outputs to be feasible, but large λ tends to degrade the objective value. This trade-off also makes it difficult to fairly compare the performance of Ising machines with that of other heuristic solvers. Therefore, it is crucial to resolve the trade-off to establish practical utility of Ising machines.

In this study, we focus on the benchmark of Ising machines on the QKP [17]. The QKP is a well-studied practical problem involving one inequality constraint over binary variables. Although it is presumably suitable for solving with Ising machines, existing Ising machine benchmarks [15] on the QKP only deal with relatively easy instances that can be solved by exact methods due to the difficulty in handling the constraint for Ising machines. Therefore, we explore a way to overcome this problem and enable effective performance comparison with existing heuristic solvers. The application to the QKP is taken as the first attempt in this direction and would be extended to other problems in the future.

Several methods to encode an inequality constraint into penalties have been proposed for applying Ising machines to the QKP [18], [19], [20], [21], since the choice of encoding methods has impacts on controlling the trade-off between the feasibility and objective value. Nevertheless, none of them have achieved better results than other heuristic solvers or even a simple greedy method. Moreover, each encoding method has different advantages and disadvantages, making it difficult to select the appropriate method for a given instance.

We take another approach to enhance the performance of Ising machines by exploiting the problem structure. We propose to incorporate efficient two-stage post-processing into the solving process using an Ising machine. The post-processing consists of *repair* and *improvement* procedures (Fig. 1). First, the repair procedure converts the output of the Ising machine, if it is infeasible, into a feasible solution. The obtained feasible solution is improved by a local improvement procedure. Since Ising machines are suited for global search, the improvement procedure takes a complementary role to achieve further improvement via local search. Although the post-processing consists of well-known greedy algorithms [17], [22], [23], the combination with Ising machines has not been fully explored so far. We believe it is valuable to thoroughly examine the effectiveness of the post-processing approach as it seems to be a promising straightforward way to resolve the technical issue.

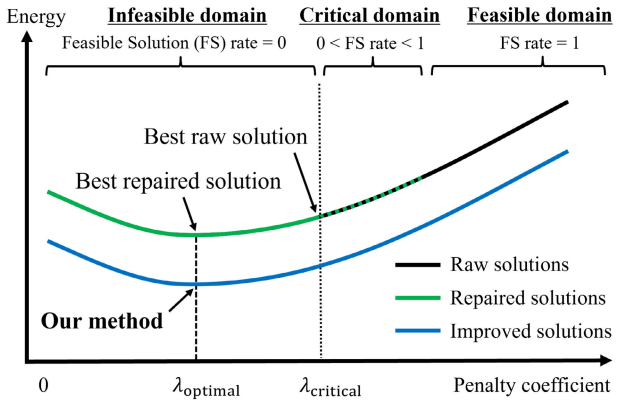


FIGURE 1. Conceptual figure of effect of two-stage post-processing. "Raw solutions" denote outputs of Ising machines, which are often infeasible when penalty coefficient λ is small (dashed line on "Critical domain"). Tuning of λ typically involves finding $\lambda_{critical}$ which achieves best trade-off between feasibility and objective. Repair procedure for infeasible solutions enables us to obtain feasible solutions even for smaller λ. Improvement procedure further enhances feasible solutions with local operations. Optimal penalty coefficient $\lambda_{optimal}$ is found to be much robust to choice of encoding methods for inequality constraint, in contrast to $\lambda_{critical}$ which heavily depends on encoding methods (see Section IV).

We conduct simulation experiments on medium-sized QKP instances using simulated annealing. The results show that the combined use of the repair and improvement procedures provides the synergistic effect on gaining the solving performance, achieving optimal solutions on more than 80% of the test instances within a reasonable time. Besides, we find that the post-processing greatly reduces the dependency of the solving performance on the choice of encoding methods of the inequality constraint into penalties, which might make practical use of Ising machines much easier.

We evaluate the performance of Amplify Annealing Engine (AE) [24], one of the state-of-the-art Ising machines, with our method on a data set of large QKP instances of size ranging from 1000 to 2000. AE combined with the post-processing achieves best known solutions on 77.5% of test instances and a small optimality gap on the rest instances. This result significantly exceeds the previous benchmark of Ising machines on the QKP [15], [16]. This is also the first result that an Ising-machine-based solver achieves a performance comparable to previous heuristics on the QKP [25], [26], [27], [28].

Our contribution is summarized as follows:

- We propose a method to solve the QKP with Ising machines combined with the post-processing consisting of the repair and improvement procedures to overcome the difficulty in handling the constraint condition.
- Through simulation experiments on medium-sized instances, we show that the post-processing is effective

in obtaining optimal solutions and making the performance robust to a choice of encoding methods.
- We show that the proposed method dramatically improves the solving performance of a state-of-the-art Ising machine on the QKP despite its simplicity, exceeding the previous benchmark results of Ising machines.

Since the proposed method is implemented on the basis of well-known naive algorithms, we expect that it can be extended and enhanced to show the advantage of Ising machines over previous heuristic methods on the QKP and other problems in the future. Therefore, our findings are a crucial step toward showing the practical utility of Ising machines.

The rest of the paper is organized as follows. Backgrounds are explained in Section II. We introduce the proposed method in Section III. The simulation experiment is conducted in Section IV. We evaluate the proposed method using the Ising machine in Section V. Related work and future work is discussed in Section VI. Section VII concludes this paper.

II. PRELIMINARIES

A. ISING MACHINES

We briefly review backgrounds on Ising machines. An *Ising model* is a model in statistical mechanics consisting of a number of binary variables $s_i \in \{\pm 1\}, i = 1, \cdots, n$ called spins and their interactions. The *energy* of a state $s = (s_1, \cdots, s_n) \in \{\pm 1\}^n$ is defined as

$$H = \sum_{i,j} J_{ij} s_i s_j + \sum_i h_i s_i, \tag{2}$$

where $J_{ij} \in \mathbb{R}$ represents the pairwise interaction between spin s_i and s_j and $h_i \in \mathbb{R}, i = 1, \cdots, n$ is called external field. A *ground state* of the Ising model is a state s that minimizes the energy H. *Ising machines* implement fast heuristics to search a ground state of the Ising model by analog computation using quantum annealing [29] or degenerate optical parametric oscillators [6], or by digital algorithms such as simulated annealing and simulated bifurcation [12] with massive parallelization.

The problem of finding a ground state of an Ising model can be also formulated as a *quadratic unconstrained binary optimization (QUBO)* problem [3], which is a class of optimization problems over binary variables $x_i \in \{0, 1\}, i = 1, \cdots, n$ defined by a square matrix $Q \in \mathbb{R}^{n \times n}$ as follows:

$$\text{minimize } x^\top Q x \tag{3}$$
$$\text{subject to } x \in \{0, 1\}^n. \tag{4}$$

The objective value $x^\top Q x$ is also called the energy of x.

B. QUADRATIC KNAPSACK PROBLEM

The quadratic knapsack problem (QKP) [17] is a generalization of the well-known knapsack problem and defined by data of n items and the knapsack capacity C. Each item i is associated with a weight $w_i > 0$ and a profit $p_i \geq 0$. In addition, for each pair i, j $(i < j)$ of items, a pairwise profit $p_{ij} \geq 0$ is defined and it is added to the total profit when both items are put into the knapsack. The QKP asks to maximize the total profit maintaining the total weight within the knapsack capacity. Namely, it is formulated as

$$\begin{aligned} &\text{maximize } H(x) := \sum_{i=1}^{n} p_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} p_{ij} x_i x_j \\ &\text{subject to } \sum_{i=1}^{n} w_i x_i \leq C, \\ &\qquad x_i \in \{0, 1\}, i = 1, \cdots, n. \end{aligned} \tag{5}$$

We define $p_{ij} := p_{ji}$ for $i > j$ to ease notation. We assume w_i and C are integers and satisfy $\min_i w_i \leq C < \sum_{i=1}^{n} w_i$ to avoid triviality. The QKP is an NP-hard optimization problem and the state-of-the-art exact solver can only solve some QKP instances of size up to 1500 in a reasonable time [30]. To solve large QKP instances efficiently, various heuristic approaches including the tabu search [26], [31], swarm optimization [32], dynamic programming [25], greedy randomized adaptive search procedure (GRASP) [26], and evolutionary algorithm [27], [33] have been proposed. The current best heuristic solver is based on the iterated hyperplane exploration approach [28].

As a particular problem structure, it is well-known that an optimum of the QKP is attained on the edge of the space of feasible solutions. Precisely, the following holds. For a proof, we refer to Appendix A.

Proposition 1 (cf. [23]): For a QKP instance defined as (5), an optimum is attained by a solution $x \in \{0, 1\}^n$ satisfying $C - \max_i w_i < \sum_{i=1}^{n} w_i x_i \leq C$.

The QKP can be reformulated as QUBO in the following way [31]. First, an integer slack variable $z \geq 0$ is introduced to represent the inequality constraint $\sum_{i=1}^{n} w_i x_i \leq C$ as an equality constraint $\sum_{i=1}^{n} w_i x_i + z = C$. By transforming the equality constraint into a penalty term in the standard way, we get a quadratic optimization problem:

$$\text{minimize } -H(x) + \lambda H_{\text{ineq}}(x, z), \tag{6}$$
$$H_{\text{ineq}}(x, z) = \left(\sum_{i=1}^{n} w_i x_i + z - C \right)^2, \tag{7}$$

where $\lambda > 0$ is a sufficiently large positive number. To further translate it into a QUBO problem, the integer variable z is represented by binary variables typically with binary expansion [31]. That is, taking sufficiently large integer $D > 0$ which is an upper bound of z, z is represented by

$$\begin{aligned} &k := \lfloor \log D \rfloor + 1, \ R := D + 1 - 2^{k-1}, \\ &z = \sum_{i=1}^{k-1} 2^{i-1} y_i + R y_k \end{aligned} \tag{8}$$

using additional binary variables $y_1, \cdots, y_k \in \{0, 1\}$. Other encoding methods of the integer variable are proposed

and evaluated for the use of Ising machine (without post-processing) [18], [19], [20]. Their performance will be compared in Section IV under the existence of post-processing.

We remark that a local optimum of the QUBO problem (6) does *not* necessarily correspond to that of the QKP (5). Recall that a local optimum of an optimization problem over binary variables is defined as the objective value of a feasible solution for which any flip (i.e. changing value from 0 to 1 or 1 to 0) of a variable cannot improve the objective value maintaining feasibility. For example, we consider a trivial feasible solution $x = (0, \cdots, 0)$ which clearly does not attain a local optimum of the QKP. In the QUBO setting, a solution with $x = (0, \cdots, 0)$ and y which gives $z = C$ corresponds to the solution. In fact, it attains a local minimum of the QUBO problem (5) for large λ since flipping x_i for any $i \in \{1, \cdots, n\}$ leads to a change of the objective value by $-p_i + \lambda w_i^2 > 0$ and similarly flipping y_i for any $i \in \{1, \cdots, k\}$ increases the objective value. In other words, a flip of x_i in the QKP is realized by multiple flips involving auxiliary variables y_i in the QUBO form. Hereafter, unless otherwise noted, we use the word "local" in the sense of the QKP and not of QUBO.

C. CHALLENGES IN ISING MACHINES SOLVING QKP

Since the QKP can be naturally formulated with a quadratic objective function of binary variables as above, it is presumably suited for benchmarks of Ising machines. However, in contrast to the max-cut problem on which Ising machines have achieved successful results [7], [12], even medium-sized QKP instances that can be handled by exact methods are not adequately optimally solved by Ising machines or simulation in the previous studies [15], [19], [20]. The biggest challenge is that Ising machines might output solutions violating the inequality constraint since the constraint is imposed only implicitly with the penalty term.

There is a trade-off that a large penalty is required to obtain feasible solutions with high probability whereas it also degrades the objective value. As shown in Section IV below, the recently proposed encoding methods of the inequality constraint [18], [19], [20] have a role to control this trade-off. Nevertheless, their improvement in Ising machine performance is not satisfactory, since they are still outperformed by a simple greedy method (see simulation results in Section IV). Our approach is to directly resolve the trade-off by incorporating local post-processing into Ising machines, instead of exploring the optimal encoding method.

III. PROPOSED METHOD

We propose to incorporate post-processing utilizing the problem structure into the solving process with Ising machines. The post-processing consists of two steps: repair and improvement. The repair procedure converts an infeasible solution into a feasible solution. It is commonly used for other meta-heuristics such as evolutionary algorithms [27], [34]. The improvement procedure takes a feasible solution as an input and improves the objective value by locally modifying the solution. Both procedures are building blocks of most heuristic combinatorial optimization algorithms, often combined with randomized operations to enable global search [28], [35]. In our case, they are used deterministically (i.e., without randomness) following a greedy strategy, since Ising machines have a role in the global search. We expect that Ising machines and the local post-processing work complementarily to efficiently enhance the solving performance. One important advantage of the proposed method is that the repair procedure enables us to set the penalty coefficient λ in (6) to small values and to tune λ according to the objective value, not to the rate of feasible solutions, since obtained solutions are always feasible. This effect, coupled with the local improvement, helps us to obtain the optimal solution more easily with Ising machines, as we will see in Sections IV and V. We explain the details of the method below.

Algorithm 1 Post-processing on QKP

Input: Solution $x = (x_1, \cdots, x_n) \in \{0, 1\}^n$ (possibly infeasible), Profits $(p_i)_i$, $(p_{ij})_{ij}$, Weights $(w_i)_i$, Capacity C
Output: Feasible solution x

for $i = 1, \cdots, n$ **do**
  $e_i \leftarrow (p_i + \sum_{j=1}^{i-1} p_{ji}x_j + \sum_{j=i+1}^{n} p_{ij}x_j)/w_i$
while $\sum_k w_k x_k > C$ **do**
  Take $j \in \mathrm{argmin}\{e_i \mid x_i = 1\}$
  $x_j \leftarrow 0$ ▷ Remove an item
  Update $(e_i)_i$
for j s.t. $x_j = 0$ in decreasing order of e_j **do**
  if $\sum_k w_k x_k + w_j \le C$ **then**
    $x_j \leftarrow 1$ ▷ Add an item
    Update $(e_i)_i$
for i s.t. $x_i = 1$ in increasing order of e_i **do**
  for j s.t. $x_j = 0$ in decreasing order of e_j **do**
    if $\sum_k w_k x_k - w_i + w_j \le C$ and $e_i w_i < e_j w_j - p_{ij}$ **then**
      $x_i \leftarrow 0, x_j \leftarrow 1$ ▷ Swap items
      Update $(e_i)_i$
return x

A. POST-PROCESSING ALGORITHM ON QKP

Both the repair and improvement procedures are built upon well-known greedy heuristics used in the previous studies [17], [22], [23]. We review the ideas of both procedures briefly to make the argument self-contained.

For the repair procedure, we note that an infeasible solution can be made into a feasible solution by removing several items from the knapsack since the weights are positive and there is a trivial feasible solution $x = (0, \cdots, 0)$. To reduce the loss of the objective value, items to be removed are selected one by one greedily. On the simple knapsack problem with the linear objective, a greedy strategy is typically based on a metric called *efficiency* defined by a ratio of the profit and weight of the item. In the QKP, the efficiency $e_i(x)$ of item i with respect to an incumbent solution x is defined as

$$e_i(x) := \frac{p_i + \sum_{j=1}^{i-1} p_{ji}x_j + \sum_{j=i+1}^{n} p_{ij}x_j}{w_i}. \tag{9}$$

Consequently, item i with $x_i = 1$ achieving minimum $e_i(x)$ is removed iteratively until the constraint is satisfied. Note that this greedy removal operation is previously used for a constructive heuristic with an input $x = (1, \cdots, 1)$ [22], [23].

The improvement procedure consists of so-called *fill-up and exchange* (FE) operation [17], which is widely used in heuristic methods on the QKP [25], [27]. The fill-up operation puts items into the knapsack unless it violates the capacity constraint. Then, the exchange operation replaces an item in the knapsack with another item that is not in the knapsack, so that it improves the objective value maintaining feasibility. In other words, the fill-up operation modifies a feasible solution to a local optimum, and the exchange operation searches neighborhood local optima. In our method, the order of item selection for FE operation is again based on the greedy strategy with the efficiency $e_i(x)$. An item to be included in the knapsack is chosen following the descending order of $e_i(x)$ and an item to be removed from the knapsack is chosen following the ascending order of $e_i(x)$.

The overall process is summarized in Algorithm 1. Every time the solution x is changed, the efficiency e_i is updated with computational cost of order $O(n)$. The total complexity of the algorithm is $O(n^3)$ in the worst case, but the number of the exchange operation (which is the bottleneck) is typically much less than n^2, and so the algorithm runs practically fast. Indeed, a quadratic scaling of the processing time is observed in our experiments in Section V.

The post-processing above is closely related to a greedy heuristic proposed by Billionnet and Calmels [23]. Their method is to first obtain a feasible solution with the greedy removal operation for $x = (1, \cdots, 1)$ and then apply the FE operation. In particular, when the penalty coefficient λ in (6) is set to 0, then the optimal solution is obviously $x = (1, \cdots, 1)$. Thus, for sufficiently small λ, an Ising machine with the post-processing outputs the same solution as the one obtained by the greedy method.

The ideas of the repair and improvement procedures are not new as mentioned above. Besides, more elaboration on the post-processing can be made to improve the solving performance further with additional computational costs. In this study, however, the specific implementation is not of much interest. Rather, we aim to show that combining the simple post-processing based on the well-known ideas effectively overcomes the critical performance issue of Ising machines, which does not seem to be understood well in the existing studies [15], [18]. The simplicity of the proposed method is preferable in terms of extensibility: establishing the effectiveness with the naive implementation leads to expectation that the approach also works on other problems.

IV. SIMULATION EXPERIMENTS

We validate the proposed method via simulation of Ising machines on the basis of simulated annealing (SA) that takes a QUBO problem as an input. Note that most digital Ising machines are based on SA [8], [9], and also SA is treated as a classical counterpart of quantum annealing [36], [37]. Therefore, controlled experiments with SA provide informative insights on the use of Ising machines. For a test bed, we use a data set of 100 medium-sized QKP instances generated in the previous study [38]. There are 10 generated instances for each combination of the problem size $n \in \{100, 200, 300\}$ and density $d\%$ of the objective function for $d \in \{25, 50, 75, 100\}$ except for $(n, d) = (300, 75)$ and $(300, 100)$. Specifically, the pairwise profit p_{ij} $(i < j)$ is non-zero with probability $d/100$ in the generation procedure. The exact optimal solutions of these instances are known and the data set has been used in the existing benchmark of Ising machines [15], [19], [20]. Things to be verified are as follows: (i) better solutions (in particular, the optimal solutions) are obtained by utilizing the post-processing and (ii) the computational cost for the post-processing is sufficiently small compared to the rest of the whole process. Furthermore, we re-evaluate various encoding methods of the inequality constraints [18], [19], [20] under the existence of the post-processing to verify the robustness of the proposed method.

A. COMPUTATIONAL SET-UP

Each QKP instance is translated into a QUBO problem (6) with binary encoding (8) of the integer variable z where the upper bound D of z is set to the capacity C. The penalty coefficient λ is varied for $\lambda = 2^i$, $i = -6, -5, \cdots, 6, 7$. For each λ, SA is executed 10 times to obtain 10 solutions. The setting of SA is as follows. We use the public implementation of SA on D-Wave Ocean SDK[1] of version 6.4.1. In the algorithm, the temperature is successively decreased from the initial value to the end value, iterating an inner loop consisting of Monte-Carlo (MC) steps for all variables. Following the previous studies [18], [19], the number of inner loops is set to 10^6 and the initial and end temperatures are set to $n \max_{i,j} |Q_{i,j}|$ and 0.1, respectively. Here, $Q_{i,j}$ is the QUBO matrix for (6), i.e.,

$$\sum_{i,j:i\le j} Q_{i,j}\hat{x}_i\hat{x}_j = -H(x) + \lambda H_{\text{ineq}}(x, z), \tag{10}$$

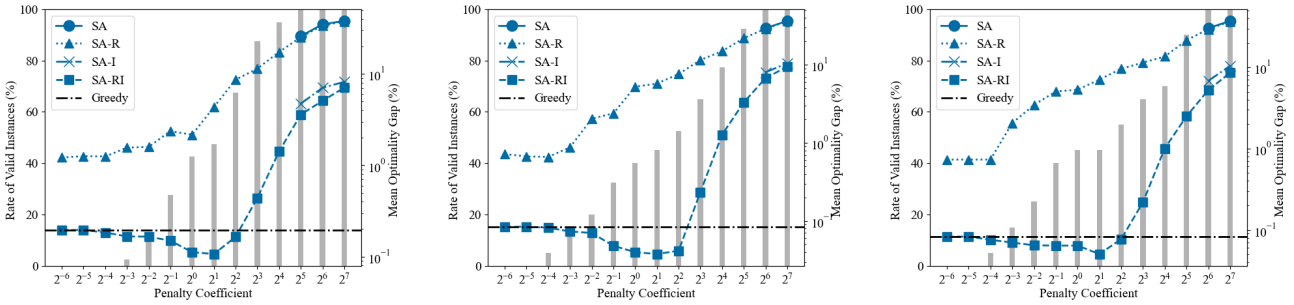
where $\hat{x} = (x_1, \cdots, x_n, y_1, \cdots, y_k)$ is a vector of the whole variables including $y_1, \cdots, y_k$ in (8). The experiment program is coded with python 3.11.4 and run on a CentOS (version 7.6.1810) server with Intel Xeon Gold 6130 chip.

We set SA without post-processing (which we simply call SA) and the greedy algorithm described in Section III as baselines, and compare them to SA with the repair and/or improvement procedure (which we call SA-R, SA-I, and SA-RI, respectively). We summarize the compared methods in Table 1. The quality of a solution is evaluated via the optimality gap

$$\text{Optimality Gap} = \frac{S_{\text{best}} - S}{S_{\text{best}}} \times 100\ (\%), \tag{11}$$

where S_{best} is the optimal value for the QKP instance and S is the objective value of the solution. For methods other than

[1] https://github.com/dwavesystems/dwave-ocean-sdk



(a) $n = 100$ (b) $n = 200$ (c) $n = 300$

FIGURE 2. Optimality gap (line graph) and number of instances on which feasible solutions are obtained with SA (bar chart) for each problem size n of QKP instances. Optimality gap for SA and SA-I is plotted only for λ producing feasible solutions on all instances. By combining repair and improvement procedures, SA-RI achieves smaller optimality gap than greedy method.

TABLE 1. Description of compared methods.

Name	Description
Greedy	Equivalent to post-processing on $x = (1, \cdots, 1)$
SA	SA without post-processing (may output infeasible solutions)
SA-R	SA with repair procedure
SA-I	SA with improvement procedure (only for feasible solutions)
SA-RI	SA with both repair and improvement procedures

the (deterministic) greedy method, the optimality gap is taken as the minimum over all feasible solutions obtained for each λ. For SA, we also count the number of instances on which a feasible solution is obtained, for each λ. The optimality gap for SA and SA-I is reported only for instances on which they obtain at least one feasible solution.

B. RESULTS

1) OBSERVATIONS FROM TUNING OF PENALTY COEFFICIENTS

The optimality gap of each method aligned with the penalty coefficient λ averaged over instances of the same size are shown in Fig. 2. We also show the rate of the number of instances where SA outputs at least one feasible solution (which we call valid instances) as bar charts. For SA and SA-I, the optimality gap is plotted only when feasible solutions are obtained on all instances for each λ and not shown otherwise. The first thing to observe from the results of SA is that the rate of valid instances increases for large λ, whereas large λ degrades the optimality gap. Therefore, SA achieves its smallest optimality gap on the minimum λ_{SA} among those giving feasible solutions on all instances, i.e., $\lambda_{\mathrm{SA}} = 32$ for $n = 100$ and $\lambda_{\mathrm{SA}} = 64$ otherwise. Note that the best optimality gap of SA is much worse than that of the greedy method. Since the greedy method runs several orders of magnitude faster than SA, we conclude that SA without post-processing is completely inferior to the greedy method on the QKP. When the repair method is applied, the optimality gap of SA-R roughly extrapolates that of SA, as expected. Accordingly, the optimality gap of SA-R achieves smaller values than that of SA for $\lambda < \lambda_{\mathrm{SA}}$. A similar phenomenon was observed by Fukada et al. [39] on a variant of the QAP.

TABLE 2. Number of medium-sized instances optimally solved.

n_d	Greedy	SA	SA-R	SA-I	SA-RI
100_25	3	0	3	6	**9**
100_50	4	1	1	8	**10**
100_75	4	1	4	6	**9**
100_100	4	0	2	8	**10**
200_25	2	0	0	5	**9**
200_50	4	0	2	3	**6**
200_75	4	0	1	3	**8**
200_100	2	0	1	**5**	**5**
300_25	4	0	2	3	**8**
300_50	4	0	1	5	**8**
Total	35	2	17	52	**82**

This result indicates the effectiveness of tuning λ based on the objective value instead of the rate of feasible solutions, which is realized thanks to the repair procedure. The optimality gap is further reduced after combining with the improvement procedure. Although using only either of the two procedures is not sufficient to outperform the greedy method, SA-RI using both procedures achieves a smaller optimality gap than that of the greedy method. This suggests that the two procedures improve the solving performance of Ising machines synergistically. Note that as λ gets closer to 0, the optimality gap of SA-RI converges to that of the greedy method. This is expected as we argued in Section III, that is, SA outputs the trivial solution $x = (1, \cdots, 1)$ for extremely small λ. The same argument applies to SA-R; as $\lambda \to 0$, the optimality gap converges to that of a weak version of the greedy algorithm that only repairs $x = (1, \cdots, 1)$.

There are two other interesting observations from Fig. 2 regarding the optimal penalty coefficient. One is that penalty coefficient λ minimizing the averaged optimality gap of SA-RI seems independent of the problem size n. We discuss this phenomenon in Section IV-D, where the dependence of the optimal λ on instance data including n and other factors is analyzed quantitatively. The other observation is that λ minimizing the optimality gap of SA-R and that of SA-RI completely differ: $\lambda_{\text{SA-R}}$ for SA-R is near 0 and $\lambda_{\text{SA-RI}}$ for SA-RI is around 2 for all problem size n. We analyze this in detail in Appendix B-A.

TABLE 3. Average optimality gap (%).

n_d	Greedy	SA	SA-R	SA-I	SA-RI
100_25	0.370	6.651	0.797	0.139	**0.047**
100_50	0.101	6.358	0.272	0.103	**0.000**
100_75	0.115	5.821	0.208	0.426	**8.4E-3**
100_100	0.196	10.202	0.395	7.0E-3	**0.000**
200_25	0.173	9.404	0.325	0.318	**5.1E-3**
200_50	0.049	8.624	0.122	0.421	**0.011**
200_75	0.049	8.624	0.200	2.259	**2.9E-3**
200_100	0.062	10.357	0.206	0.995	**0.034**
300_25	0.127	10.098	0.230	0.484	**1.4E-3**
300_50	0.038	11.329	0.245	1.415	**4.4E-4**
Mean	0.128	8.747	0.300	0.657	**0.011**

TABLE 4. Average processing time.

	Before Post-process (s)		Post-process (ms)	
n_d	Formulation	SA	Repair	Improve
100_25	0.07	4.4	0.9	1.0
100_50	0.07	4.3	1.0	1.0
100_75	0.07	4.0	1.0	0.9
100_100	0.07	3.7	1.0	0.8
200_25	0.23	17.3	3.5	3.7
200_50	0.24	17.0	3.8	3.7
200_75	0.24	14.0	4.1	2.9
200_100	0.25	12.8	3.9	2.8
300_25	0.51	34.4	8.1	7.0
300_50	0.52	36.5	8.8	7.0

2) RESULTS ON BEST OPTIMALITY GAP

The number of instances on which each method achieved the optimal solution is reported in Table 2. We also summarize the optimality gap averaged over 10 instances for each pair (n, d) in Table 3. SA achieves the optimal solutions on only two instances among 100 instances in total. Although SA-R achieves the optimum on several instances, the total number of such instances is less than that of the greedy method. SA-I obtains the optimal solutions more frequently than SA-R and the greedy method, but its averaged optimality gap is worse than the others. This means that the quality of solutions of SA-I has much variance over instances, which is often undesirable. SA-RI, the proposed method, successfully attains the optimum on 82 instances in total and achieves the smallest optimality gap for all pairs of (n, d). These results clearly demonstrate the effectiveness of combining the repair and improvement procedures as the post-processing for SA. For full results on each instance, see Appendix B-B.

3) RESULTS ON PROCESSING TIME

We evaluate the computational overhead of the post-processing. The average processing time for each process is reported in Table 4. In addition to the execution time of SA and the repair and improvement procedures, we include the processing time to create the input QUBO object after reading data of the corresponding QKP instance as the "Formulation" column. We see that time for each process increases roughly with an order of n^2. Note that we report processing time before the post-processing in seconds and time for the post-processing in milliseconds. The time required for the post-processing is more than 1000 times less than that of the annealing, and also much less than the formulation. Therefore, the proposed method improves the accuracy with a negligibly small amount of additional computational cost.

C. DEPENDENCY ON ENCODING METHODS

As described in Section II, the previous studies [18], [19], [20] suggest that other encoding methods of the slack variable z in (6) than the standard binary encoding (8) might enhance the quality of solutions obtained by Ising machines. Since their evaluation has been conducted without any post-processing, we re-evaluate various encoding methods with the proposed post-processing in this section. We report simulation results based on SA here since it reproduces well the results of the previous studies [18], [19], [20] as shown below. We also conducted the same experiment with a real Ising machine and obtained mostly similar results, see Appendix B-A for details.

The setting is as follows. We consider the following five variations of encoding methods of z in the QUBO problem (6) of the QKP. The first is the binary encoding shown in (8). Recall that it involves k auxiliary variables $y_1, \cdots, y_k$ with $k = \lfloor \log D \rfloor + 1$, where D denotes the upper bound of z. The second is the unary encoding defined as

$$z = \sum_{i=1}^{D} y_i, \tag{12}$$

which involves D auxiliary variables $y_1, \cdots, y_D$. The third is the hybrid encoding [19], which hybridizes the unary and binary encoding. As it has several degrees of freedom, we adopt the following form close to a method called $HE(1)$ in the previous experiment [19]:

$$z = \sum_{i=1}^{k} y_i + \sum_{i=k+1}^{2k} 2\, y_i, \ k := \lceil D/3 \rceil. \tag{13}$$

The hybrid encoding involves $2\lceil D/3 \rceil$ auxiliary variables. The fourth is the one-hot encoding, which uses an additional penalty term H_{onehot} and modify the objective function of the QUBO problem as

$$-H(x) + \lambda \left(H_{\text{ineq}}(x, z) + H_{\text{onehot}} \right), \tag{14}$$

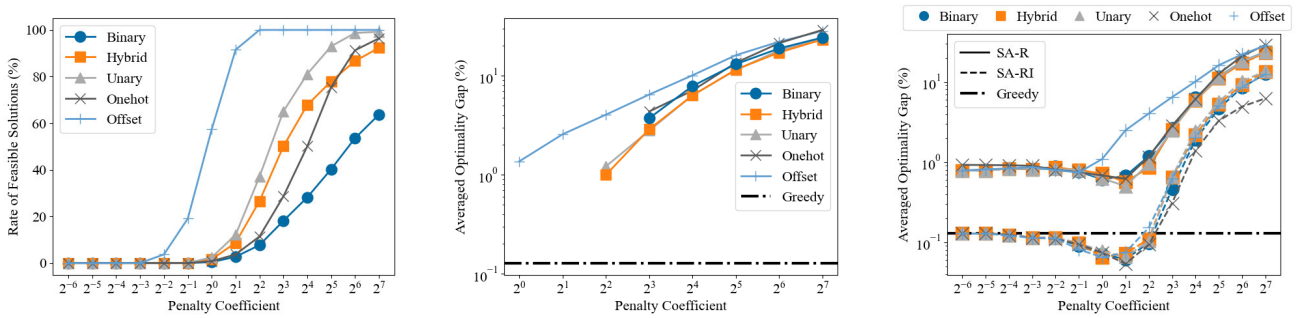
defining

$$z = \sum_{i=0}^{D} i y_i, \ H_{\text{onehot}} = \left(\sum_{i=0}^{D} y_i - 1 \right)^2. \tag{15}$$

The one-hot encoding involves $D+1$ auxiliary variables. The last is the offset encoding [20], which set z to a constant

$$z = W_{\text{offset}} \tag{16}$$

with some small number $W_{\text{offset}} \geq 0$. Since z does not work as a slack variable any more, the offset encoding does not preserve the equivalence of the optimization problems. Nevertheless, Bontekoe et al. [20] reported that it



(a) Rate of Feasible Solutions (b) Optimality Gap of SA (c) Optimality Gap with Post-processing

FIGURE 3. Performance comparison among various encoding methods of inequality constraint on 100 medium-sized QKP instances. (a)(b) Choice of encoding methods controls trade-off between rates of feasible solutions and objective values. (c) Solving performance of proposed method is much less dependent on choice of encoding methods.

outperformed other encoding methods. We set $W_{\text{offset}} = 3$ following the previous result. All methods other than the offset encoding involve the upper bound D of z. Note that it suffices to set D to a value greater than or equal to $\max_i w_i$ to translate the QKP to the QUBO problem preserving the optimum according to Proposition 1. On the other hand, since methods other than the binary encoding uses $O(D)$ auxiliary variables, D should be sufficiently small to effectively apply Ising machines. Therefore, we set D to $\max_i w_i$ in this experiment. Other settings are identical to those in the earlier experiment.

We remark that an output of Ising machines or SA can have a positive penalty $H_{\text{ineq}}(x, z) > 0$ (or $H_{\text{onehot}} > 0$ for the one-hot encoding) even if the solution x is feasible. Such situations include a case where $z \neq \sum_i w_i x_i$, as well as a case where $\sum_{i=0}^{D} y_i \neq 1$ for the one-hot encoding. It is in contrast to the previous evaluation [18] treating the solution as feasible only when it has zero penalty, and this difference in definition could lead to different results. In particular, for the one-hot encoding above, it is actually not necessary to impose the one-hot constraint $\sum_{i=0}^{D} y_i = 1$, since the inequality constraint can be satisfied even when $\sum_{i=0}^{D} y_i = 0$ or $\sum_{i=0}^{D} y_i \geq 2$. Note that this fact is also used in the previous study [20]. Therefore, the aforementioned case where x is feasible and $H_{\text{onehot}} > 0$ can particularly often occur, and we indeed observed this phenomenon in our experiment.

Fig. 3 shows the results over various λ. Fig. 3a shows the rate of feasible solutions (we call FS rate) over all instances for each encoding. On all methods, a larger penalty coefficient results in a high FS rate. Among the tested encoding methods, the binary encoding leads to the lowest, while the offset encoding achieves the highest. The difference might be explained by the number of flips of auxiliary variables y_i required for a flip of a variable x_i, which is mentioned in Section II. More precisely, multiple MC steps in SA are required to realize a single flip on the QKP. The offset encoding uses no auxiliary variables, and thus the penalty H_{ineq} might be easily decreased by local operations in SA, leading to the high FS rate. In contrast, a lot of MC steps are required for changing the value of z for the binary encoding, resulting in a low FS rate. The redundancy of the representation (i.e. representing a value of z by multiple combinations of values of $y_1, y_2, \cdots$) in the unary and hybrid encoding might help to make the number of required MC steps small [18], and thus they give the intermediate results. For the one-hot encoding, most solutions violate the one-hot constraint and as a result obtain a similar redundancy, which again explains the intermediate result. The optimality gap of the feasible solutions obtained by SA is shown in Fig. 3b. Here, we plot the optimality gap for λ that obtains a feasible solution on more than half of all instances to exclude outlier values. Again, for all methods, a smaller penalty coefficient leads to better objective values. The hybrid, unary, and offset encodings achieve a lower optimality gap than the others, due to the high FS rate at small λ. These results on the FS rate and optimality gap agree well with the previous studies [18], [19], [20].

Fig. 3c shows the optimality gap for each method combined with the post-processing. Interestingly, after the post-processing, the difference among the encoding methods gets almost negligible and all methods reach a similar minimum optimality gap at the similar value of λ. A subtle exception is the offset encoding; SA-R with the offset encoding attains the minimum optimality gap at $\lambda_{\text{SA-R}} = 0.5$, unlike the others. This is presumably because fixing the slack variable z to a constant changes the effect of penalty H_{ineq} on the behavior of SA. The overall result indicates that the proposed method is much robust to the choice of encoding methods, compared to SA without post-processing. A fundamental reason for the somewhat surprising similarity of the post-processed outputs over the various encoding methods is unclear and might be related to the behavior of the SA algorithm. Since a precise algorithmic analysis is beyond the scope of this paper, further investigation is left as future work.

For a quantitative performance comparison, we summarize the number of instances optimally solved and the optimality gap for each encoding with the proposed method in Table 5 and 6. We see that the binary and one-hot encodings slightly outperform the other methods on average in terms of both metrics. In particular, among the binary, unary, and

TABLE 5. Number of medium-sized instances optimally solved.

n_d	Binary	Hybrid	Unary	One-hot	Offset
100_25	10	9	8	10	10
100_50	9	9	9	9	9
100_75	9	8	8	8	8
100_100	8	8	8	9	7
200_25	9	8	10	8	8
200_50	7	7	6	7	7
200_75	8	9	9	8	8
200_100	6	6	6	6	6
300_25	7	7	7	8	8
300_50	10	9	9	8	9
Total	**83**	80	80	81	80

TABLE 6. Averaged optimality gap (×0.01 %).

n_d	Binary	Hybrid	Unary	One-hot	Offset
100_25	0.000	9.328	4.355	0.000	0.000
100_50	0.384	0.610	0.610	0.666	0.610
100_75	0.537	1.590	1.590	1.140	1.140
100_100	0.412	0.412	14.338	0.205	14.546
200_25	0.510	0.659	0.000	0.253	3.585
200_50	0.343	0.888	0.761	0.260	0.888
200_75	0.917	0.213	0.213	0.297	0.884
200_100	0.995	1.079	1.129	0.624	0.803
300_25	0.418	3.710	0.180	0.135	0.246
300_50	0.000	0.035	0.241	0.055	0.184
Mean	0.452	1.853	2.342	**0.363**	2.288

one-hot encodings, the unary encoding performs the worst (by a possibly negligible margin), in contrast to the previous evaluation without the post-processing [18]. In other words, whether or not a specific encoding method performs well can be easily changed by additional operations. This leads to an insight important to practitioners that performance evaluation of Ising machines should be carefully done in a practical situation when it involves pre- or post-processing of the problem or solutions.

D. ANALYSIS OF OPTIMAL PENALTY COEFFICIENTS

In the earlier experiments, we observed that the optimal penalty coefficient $\lambda_{\text{SA-RI}}$ for the proposed method varies depending on the problem instances (see Appendix B-B for full results including $\lambda_{\text{SA-RI}}$ for each instance). The optimal penalty coefficient could be estimated by some representative features of the instance data [39]. In this section, we analyze $\lambda_{\text{SA-RI}}$ over the tested instances to utilize the result for solving larger instances in the later section.

As representative features of the QKP, we consider the problem size n, density d of the objective function, and tightness ratio $\alpha = C/\sum_i w_i$ of the inequality constraint. Note that the tightness ratio α has not been mentioned in the QKP literature, whereas it is recognized as an important factor in the context of the multi-dimensional knapsack problem [34], [40]. We expect that n has weak correlation with $\lambda_{\text{SA-RI}}$, as we see from Fig. 2 for each n. On the other hand, the density d involves the scale of the increase in the objective value for putting an item into the knapsack. Since it is typically considered that the scales of the objective function and penalty should be balanced when applying the penalty method, we expect that $\lambda_{\text{SA-RI}}$ tends to be large for large d.

TABLE 7. Regression coefficients for optimal penalty coefficients.

A	c_n	c_d	c_α
1.14	0.09	0.84	−0.21

TABLE 8. Number of medium-sized instances optimally solved.

n_d	Gurobi	AE	AE-R	AE-I	AE-RI
100_25	10	10	10	10	10
100_50	10	10	10	10	10
100_75	10	10	10	10	10
100_100	10	10	10	10	10
200_25	10	7	8	10	10
200_50	9	8	8	10	10
200_75	10	7	8	10	10
200_100	10	7	7	10	10
300_25	10	5	7	10	10
300_50	10	6	8	10	10
Total	99	80	86	**100**	**100**

We model $\lambda_{\text{SA-RI}}$ as the product of the features by

$$\lambda_{\text{Estimate}} = A n^{c_n} d^{c_d} \alpha^{c_\alpha}, \tag{17}$$

where A, c_n, c_d, and c_α are parameters to be fit. We show the results of log-linear regression on $\lambda_{\text{SA-RI}}$ for the tested 100 instances in Table 7. As expected, the resulting coefficient for n is close to 0 and that for d is a large positive value. The coefficient c_α for α is negative, which means that λ should be lowered for large capacity C. This might be because large α implies that feasible solutions occupy a large fraction of the total space $\{0,1\}^n$, and thus the penalty is not required to be much emphasized for solving the QKP. Note that the overall analysis is on a data set created following a specific procedure of instance generation, and the result might depend on the distribution of problem instances. Since larger instances used in the later section are based on the same generating protocol as that of the instances used above, we make use of the analysis result to solve the larger instances.

V. EVALUATION USING ISING MACHINE

In this section, we evaluate the proposed method using one of the state-of-the-art Ising machines, Amplify Annealing Engine (AE) [24], on a broader set of QKP instances. Our aim in this experiment is to verify that the proposed method works also for a high-performance Ising machine as well as for naive SA. We use large QKP instances for which exact methods require high computational time to solve. We compare the performance of the Ising machine with that of existing heuristic solvers.

A. SETTING

In addition to the medium-sized instances in Section IV, we use another group of QKP instances generated in the previous study [26]. There are 10 instances for each combination of the problem size $n \in \{1000, 2000\}$ and density $d \in$

TABLE 9. Number of best known solutions obtained and average optimality gap (×0.01 %).

	Greedy		DP+FE [25]		GRASP+ Tabu [26]		IQIEA [27]		Gurobi		AE		AE-R		AE-I		AE-RI	
n_d	#BKS	Gap	#BKS	Gap	#BKS	Gap	#BKS	Gap	#BKS	Gap	#BKS	Gap	#BKS	Gap	#BKS	Gap	#BKS	Gap
1000_25	4	2.452	3	11.153	**10**	**0.000**	**10**	**0.000**	8	0.041	1	-	4	2.830	3	-	**10**	**0.000**
1000_50	1	1.928	1	0.434	**10**	**0.000**	**10**	**0.000**	8	0.010	0	-	4	0.428	2	-	8	0.184
1000_75	0	7.760	1	0.675	9	0.043	9	0.043	8	0.011	0	-	2	4.002	0	-	8	0.062
1000_100	0	3.782	2	0.495	9	0.121	**10**	**0.000**	7	0.259	0	-	1	1.833	0	-	7	0.032
2000_25	1	0.763	0	0.330	**10**	**0.000**	**10**	**0.000**	5	0.032	0	-	4	0.141	0	-	8	0.010
2000_50	3	1.297	2	0.337	9	0.034	9	0.037	7	0.042	0	-	4	0.360	0	-	8	0.101
2000_75	1	1.097	1	0.173	8	0.375	8	0.375	9	0.054	0	-	2	1.229	0	-	7	0.393
2000_100	0	2.577	1	0.257	9	0.533	9	0.512	5	0.152	0	-	2	2.873	0	-	6	0.597
All	10	2.707	11	1.732	74	0.138	75	0.121	57	0.075	1	-	20	1.712	5	-	62	0.172

Results are shown in boldface when the algorithm achieves the zero optimality gap on all instances, i.e., reaches the best known scores of IHEA [28].

TABLE 10. Performance comparison of AE-RI and each baseline.

Method	# better	# worse	Wilcoxon test statistic	Wilcoxon test p-value
Greedy	70	0	2485.0	2E-13
DP+FE [25]	66	5	2293.5	3E-9
GRASP+Tabu [26]	1	15	16.0	0.996
IQIEA [27]	1	15	6.0	0.999
IHEA [28]	0	18	0.0	>0.999
Gurobi	16	13	217.0	0.504

better/worse represents the number of instances on which AE-RI performs better/worse.

$\{25, 50, 75, 100\}$ of the objective function in the data set. Their exact optimal solutions have not been known and the current best known objective values are reported by Chen and Hao [28]. Therefore, we use their result to compute the optimality gap (11) in which S_{best} denotes the best known objective value.

The computational environment is the same as in Section IV. We provide additional details on the use of the Ising machine. We use AE of version 0.7.4 with A100 GPU. The timeout for the execution of AE is set to $0.01n$ seconds for problem size n, which is comparable with that of the existing solver [28]. We use Amplify SDK [24] of version 0.11.2 to translate the QKP into a QUBO problem and input it to AE. The slack variable z is encoded into binary variables by binary expansion (8) with $D = C$.

The penalty coefficient λ is heuristically varied as

$$\lambda = a\frac{d}{100}\sqrt{1/\alpha},\ a = 1, 2, \cdots, \tag{18}$$

using the density d and tightness ratio $\alpha = C/\sum_i w_i$, based on the result in Section IV-D. The upper bound of a is set to 10 for medium-sized instances and 20 for large instances, which is found to be sufficient to obtain good solutions with the proposed method. The Ising machine is executed 10 times to sample 10 solutions for each λ. The solutions are evaluated by optimality gap with and without the post-processing.

We compare the performance of AE with and without the post-processing. We call them AE, AE-R, AE-I, and AE-RI, respectively, following the same notation in Table 1. We use the greedy method described in Section III as a baseline. Besides, the results of the following heuristic solvers tailored for the QKP are taken from the existing papers [27], [28] and included as baselines: dynamic programming with fill-up and exchange (DP+FE) [25], GRASP combined with tabu search (GRASP+Tabu) [26], improved quantum-inspired evolutionary algorithm (IQIEA) [27], and iterated hyper-plane exploration approach (IHEA) [28]. We also use Gurobi [41], one of the state-of-the-art commercial solvers, to provide an insight into performance comparison with a general-purpose method. For each instance, we run Gurobi (version 9.1.2) with a time limit of 1 minute and report the best solution found.

B. RESULTS

We discuss the benchmark results on medium-sized and large QKP instances. For the full results on each instance, we refer to Appendix B-C. Since the best known solutions (BKS) produced by the IHEA algorithm are used for evaluation, IHEA trivially achieves zero optimality gap for all instances and thus is omitted from the results.

The results on the medium-sized instances are summarized in Table 8. For the previous methods, we omit the results since these instances are rather easy to reach optimality, and refer to the original results [28]. For the instances of size $n = 100$, AE successfully obtains the optimal solutions without the post-processing. As n increases, however, the number of instances solved optimally by AE decreases. Meanwhile, the post-processing enables us to obtain the optimal solutions on all instances of sizes up to 300. The result ensures that the proposed method can further enhance the solving performance of the state-of-the-art Ising machine.

The results on the large instances are shown in Table 9. The averaged optimality gap for AE and AE-I are omitted in Table 9 since they could not obtain a feasible solution on some instances for every (n, d). AE-RI achieves the best known solutions on 62 instances out of 80 instances, whereas AE obtains the best known solution on only one instance. Also, the greedy method performs poorly compared to other methods. Note that the greedy method applies the same local operations as the post-processing. Therefore, the result indicates that global search by the Ising machine and local search by the post-processing work

TABLE 11. Average running time (second) to sample a solution.

n	Greedy	DP+FE [25]	GRASP+ Tabu [26]	IQIEA [27]	IHEA [28]	Gurobi	AE-RI			
							Formulation	AE	Repair	Improve
1000	0.27	2917.7	28.0	307.4	6.0	60.0	2.17	9.93	0.08	0.06
2000	1.08	51695.8	329.7	3034.0	22.7	60.4	10.74	20.50	0.33	0.32

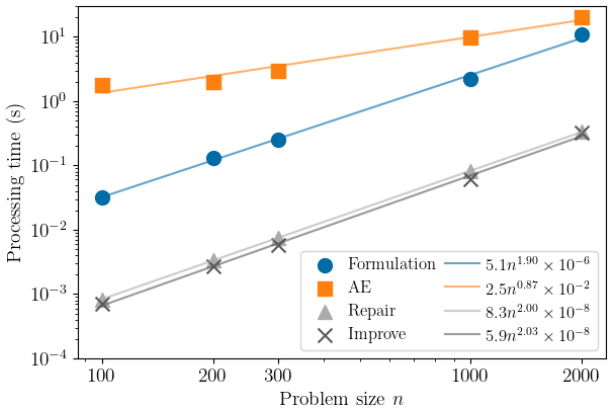


FIGURE 4. Processing time of processes in AE-RI. Lines are fitted with log-log regression. The execution time of AE is set to $O(n)$ and the other processes empirically take $O(n^2)$ runtime.

well in a complementary manner in the proposed method. Furthermore, AE-RI achieves comparable results with other heuristic solvers. This is the first result to achieve such high accuracy on the large-scale QKP using Ising machines, which might shed the light on the practical utility of Ising machines. To compare the performance of AE-RI and each baseline more directly, we also provide a result of statistical testing of the AE-RI performance against each baseline in Table 10. We conducted the one-sided Wilcoxon rank sum test with a null hypothesis that the performance of AE-RI is not better than a baseline. More precisely, we count the number of QKP instances on which AE-RI achieved a larger/smaller objective value than each baseline method and calculate the test statistic and p-value. The result shows that AE-RI indeed performs significantly better than the greedy and DP+FE methods, while it does not hold for other baselines. In summary, although our result significantly outperforms the previous benchmarks of Ising machines, there is still a performance gap between Ising machines and the state-of-the-art heuristic solvers such as IHEA. Filling the gap could be an important milestone for further software and hardware development of Ising machines.

We report the computational time of the proposed method in Table 11. The processing time for formulation, repair, and improvement procedures shows an expected scaling behavior extending Table 4 to larger n. The execution time of AE is around the timeout we set. We plot the processing times against the number of variables n in Fig. 4 with log-log regression curves. As in the case of SA-RI, we observe the quadratic scaling of processing time for formulation, repair, and improvement procedures. Overall, the results imply that the post-processing causes negligible computational overhead also for the Ising machine.

Averaged running time to obtain one solution for each baseline is also listed in Table 11. For methods other than the greedy method and Gurobi, the results are taken from the previous studies [27], [28]. We do not intend a fair comparison of running time across the baselines, due to differences in the computational environments. Moreover, since AE is a cloud service involving queue and communication time, defining a reasonable metric on computational time is itself a hard task. Here, we just aim to get insights into the scaling behavior of running time. The IHEA algorithm scales quite well for large n, and thus the comparable amount of time has been adopted for the timeout of AE in our experiments. Further precise benchmarks including evaluation of practical solving time should be conducted in the future after establishing a method for Ising machines to achieve sufficiently high accuracy.

VI. RELATED WORK AND DISCUSSION

There are several previous studies aiming to solve the QKP using Ising machines [15], [18], [19], [20]. All of them use relatively easy QKP instances which can be handled by exact methods. Our work is the first to solve large QKP instances ranging from 1000 to 2000 variables with Ising machines. Parizy and Togawa [15] propose an improvement algorithm for feasible solutions of the QKP, but their rate of instances optimally solved is only 77% with a high-performance Ising machine while ours achieves higher rates using naive SA. The difference might be caused by the use of the repair method. The other studies explore a good way of encoding inequality constraints [18], [19], [20]. Our work takes a completely different approach and shows that an encoding method is not an important factor for accuracy on the QKP under the existence of the post-processing as in Section IV.

The comprehensive experiments in the previous sections show that the naive post-processing leads to drastic improvement of solving performance of SA and the Ising machine. This finding is somewhat surprising given the simplicity of the method, and seems to have been overlooked by the existing studies. Considering that the Ising machine hardware is rapidly evolving to obtain better results and there could be room for enhancing and extending the post-processing, it also indicates the possibilities that Ising machines will be competent with other heuristic approaches in the future.

The proposed method could be extended to other problems on which a greedy heuristic is known. Such problems may

involve other types of constraints such as one-hot constraints. For example, the max k-cut [42] problem, a generalization of the max cut problem, admits an efficient implementation of a greedy local search algorithm [43]. On the other hand, Ising machines do not perform well on the max k-cut problem as it involves a lot of one-hot constraints. We expect that the post-processing approach using the greedy local search can be utilized to develop a high-performance Ising machine-based solver. We will investigate such extensions in our future work.

VII. CONCLUSION

Toward more practical benchmarks of Ising machines, we proposed a method to solve the QKP with Ising machines using the two-stage post-processing. The repair and improvement procedures improve the solving performance of Ising machines synergistically. From an empirical study using both simulation and an Ising machine, we demonstrated the effectiveness of the proposed method. We found that the performance of the proposed method was much less dependent on a choice of the encoding methods of the inequality constraint. Evaluation on large QKP instances showed that the Amplify Annealing Engine with the proposed post-processing achieved comparable performance with Gurobi and other heuristic methods tailored for the QKP, which is an important step toward practical utility of Ising machines. Future work includes the extension of the proposed method to other optimization problems and establishing a reasonable benchmarking framework considering computational time required for Ising machines.

REFERENCES

[1] V. Cacchiani, M. Iori, A. Locatelli, and S. Martello, "Knapsack problems—An overview of recent advances. Part II: Multiple, multidimensional, and quadratic knapsack problems," *Comput. Oper. Res.*, vol. 143, Jul. 2022, Art. no. 105693.

[2] N. Mohseni, P. L. Mcmahon, and T. Byrnes, "Ising machines as hardware solvers of combinatorial optimization problems," *Nature Rev. Phys.*, vol. 4, no. 6, pp. 363–379, May 2022.

[3] A. Lucas, "Ising formulations of many NP problems," *Frontiers Phys.*, vol. 2, p. 5, Feb. 2014.

[4] M. W. Johnson, M. H. S. Amin, S. Gildert, T. Lanting, F. Hamze, N. Dickson, R. Harris, A. J. Berkley, J. Johansson, P. Bunyk, and E. M. Chapple, "Quantum annealing with manufactured spins," *Nature*, vol. 473, no. 7346, pp. 194–198, 2011.

[5] A. A. Houck, H. E. Türeci, and J. Koch, "On-chip quantum simulation with superconducting circuits," *Nature Phys.*, vol. 8, no. 4, pp. 292–299, Apr. 2012.

[6] Z. Wang, A. Marandi, K. Wen, R. L. Byer, and Y. Yamamoto, "Coherent Ising machine based on degenerate optical parametric oscillators," *Phys. Rev. A, Gen. Phys.*, vol. 88, no. 6, Dec. 2013, Art. no. 063853.

[7] T. Honjo, T. Sonobe, K. Inaba, T. Inagaki, T. Ikuta, Y. Yamada, T. Kazama, K. Enbutsu, T. Umeki, R. Kasahara, K.-I. Kawarabayashi, and H. Takesue, "100,000-spin coherent Ising machine," *Sci. Adv.*, vol. 7, no. 40, Oct. 2021, Art. no. eabh0952.

[8] M. Yamaoka, C. Yoshimura, M. Hayashi, T. Okuyama, H. Aoki, and H. Mizuno, "A 20k-spin Ising chip to solve combinatorial optimization problems with CMOS annealing," *IEEE J. Solid-State Circuits*, vol. 51, no. 1, pp. 303–309, Jan. 2016.

[9] M. Aramon, G. Rosenberg, E. Valiante, T. Miyazawa, H. Tamura, and H. G. Katzgraber, "Physics-inspired optimization for quadratic unconstrained problems using a digital annealer," *Frontiers Phys.*, vol. 7, p. 48, Apr. 2019.

[10] K. Yamamoto, K. Kawamura, K. Ando, N. Mertig, T. Takemoto, M. Yamaoka, H. Teramoto, A. Sakai, S. Takamaeda-Yamazaki, and M. Motomura, "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*, vol. 56, no. 1, pp. 165–178, Jan. 2021.

[11] T. Wang, L. Wu, P. Nobel, and J. Roychowdhury, "Solving combinatorial optimisation problems using oscillator based Ising machines," *Natural Comput.*, vol. 20, no. 2, pp. 287–306, Jun. 2021.

[12] H. Goto, K. Tatsumura, and A. R. Dixon, "Combinatorial optimization by simulating adiabatic bifurcations in nonlinear Hamiltonian systems," *Sci. Adv.*, vol. 5, no. 4, Apr. 2019, Art. no. eaav2372.

[13] P. Codognet, D. Diaz, and S. Abreu, "Quantum and digital annealing for the quadratic assignment problem," in *Proc. IEEE Int. Conf. Quantum Softw. (QSW)*, Jul. 2022, pp. 1–8.

[14] T. Huang, J. Xu, T. Luo, X. Gu, R. Goh, and W.-F. Wong, "Benchmarking quantum(-inspired) annealing hardware on practical use cases," *IEEE Trans. Comput.*, vol. 72, no. 6, pp. 1692–1705, Jun. 2023.

[15] M. Parizy and N. Togawa, "Analysis and acceleration of the quadratic knapsack problem on an Ising machine," *IEICE Trans. Fundamentals Electron., Commun. Comput. Sci.*, vol. E104.A, no. 11, pp. 1526–1535, 2021.

[16] A. Ceselli and M. Premoli, "On good encodings for quantum annealer and digital optimization solvers," *Sci. Rep.*, vol. 13, no. 1, p. 5628, Apr. 2023.

[17] G. Gallo, P. L. Hammer, and B. Simeone, "Quadratic knapsack problems," in *Combinatorial Optimization*. Berlin, Germany: Springer, 1980, pp. 132–149.

[18] K. Tamura, T. Shirai, H. Katsura, S. Tanaka, and N. Togawa, "Performance comparison of typical binary-integer encodings in an Ising machine," *IEEE Access*, vol. 9, pp. 81032–81039, 2021.

[19] S. Jimbo, D. Okonogi, K. Ando, T. V. Chu, J. Yu, M. Motomura, and K. Kawamura, "A hybrid integer encoding method for obtaining high-quality solutions of quadratic knapsack problems on solid-state annealers," *IEICE Trans. Inf. Syst.*, vol. E105.D, no. 12, pp. 2019–2031, 2022.

[20] T. Bontekoe, F. Phillipson, and W. V. D. Schoot, "Translating constraints into QUBOs for the quadratic knapsack problem," in *Proc. Int. Conf. Comput. Sci.* Cham, Switzerland: Springer, 2023, pp. 90–107.

[21] K. Yonaga, M. J. Miyama, and M. Ohzeki, "Solving inequality-constrained binary optimization problems on quantum annealer," 2020, *arXiv:2012.06119*.

[22] P. Chaillou, P. Hansen, and Y. Mahieu, "Best network flow bounds for the quadratic knapsack problem," in *Combinatorial Optimization*, Rome, Italy; Springer, 1989, pp. 225–235.

[23] A. Billionnet and F. Calmels, "Linear programming for the 0–1 quadratic knapsack problem," *Eur. J. Oper. Res.*, vol. 92, no. 2, pp. 310–325, Jul. 1996.

[24] Fixstars Corporation. (2020). *Fixstars Amplify AE*. [Online]. Available: https://amplify.fixstars.com/en/engine

[25] F. D. Fomeni and A. N. Letchford, "A dynamic programming heuristic for the quadratic knapsack problem," *INFORMS J. Comput.*, vol. 26, no. 1, pp. 173–182, Feb. 2014.

[26] Z. Yang, G. Wang, and F. Chu, "An effective GRASP and Tabu search for the 0–1 quadratic knapsack problem," *Comput. Oper. Res.*, vol. 40, no. 5, pp. 1176–1185, May 2013.

[27] C. Patvardhan, S. Bansal, and A. Srivastav, "Parallel improved quantum inspired evolutionary algorithm to solve large size quadratic knapsack problems," *Swarm Evol. Comput.*, vol. 26, pp. 175–190, Feb. 2016.

[28] Y. Chen and J.-K. Hao, "An iterated 'hyperplane exploration,' approach for the quadratic knapsack problem," *Comput. Oper. Res.*, vol. 77, pp. 226–239, 2017.

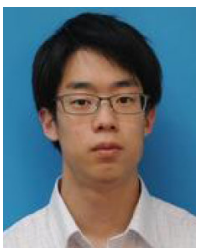
[29] T. Kadowaki and H. Nishimori, "Quantum annealing in the transverse Ising model," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 58, no. 5, pp. 5355–5363, Nov. 1998.

[30] W. D. Pisinger, A. B. Rasmussen, and R. Sandvik, "Solution of large quadratic knapsack problems through aggressive reduction," *INFORMS J. Comput.*, vol. 19, no. 2, pp. 280–290, May 2007.

[31] F. Glover, G. Kochenberger, B. Alidaee, and M. Amini, "Solving quadratic knapsack problems by reformulation and Tabu search: Single constraint case," in *Combinatorial and Global Optimization*. Singapore: World Scientific, 2002, pp. 111–121.

[32] X.-F. Xie and J. Liu, "A mini-swarm for the quadratic knapsack problem," in *Proc. IEEE Swarm Intell. Symp.*, Apr. 2007, pp. 190–197.

[33] C. Patvardhan, V. P. Prakash, and A. Srivastav, "Novel quantum-inspired evolutionary algorithms for the quadratic knapsack problem," *Int. J. Math. Oper. Res.*, vol. 4, no. 2, p. 114, 2012.
[34] P. C. Chu and J. E. Beasley, "A genetic algorithm for the multidimensional knapsack problem," *J. Heuristics*, vol. 4, no. 1, pp. 63–86, Jun. 1998.
[35] R. Jovanovic and S. Voß, "Solving the quadratic knapsack problem using GRASP," in *Metaheuristics for Machine Learning*. Cham, Switzerland: Springer, 2023, pp. 157–178.
[36] D. A. Battaglia, G. E. Santoro, and E. Tosatti, "Optimization by quantum annealing: Lessons from hard satisfiability problems," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 71, no. 6, Jun. 2005, Art. no. 066707.
[37] B. Heim, T. F. Rønnow, S. V. Isakov, and M. Troyer, "Quantum versus classical annealing of Ising spin glasses," *Science*, vol. 348, no. 6231, pp. 215–217, Apr. 2015.
[38] A. Billionnet and É. Soutif, "An exact method based on Lagrangian decomposition for the 0–1 quadratic knapsack problem," *Eur. J. Oper. Res.*, vol. 157, no. 3, pp. 565–575, Sep. 2004.
[39] K. Fukada, M. Parizy, Y. Tomita, and N. Togawa, "A three-stage annealing method solving slot-placement problems using an Ising machine," *IEEE Access*, vol. 9, pp. 134413–134426, 2021.
[40] R. R. Hill and C. H. Reilly, "The effects of coefficient correlation structure in two-dimensional knapsack problems on solution procedure performance," *Manage. Sci.*, vol. 46, no. 2, pp. 302–317, Feb. 2000.
[41] Gurobi Optimization, LLC. (2023). *Gurobi Optimizer Reference Manual*. [Online]. Available: https://www.gurobi.com
[42] A. Frieze and M. Jerrum, "Improved approximation algorithms for MAXk-CUT and MAX BISECTION," *Algorithmica*, vol. 18, no. 1, pp. 67–81, May 1997.
[43] F. Ma and J.-K. Hao, "A multiple search operator heuristic for the max-k-cut problem," *Ann. Oper. Res.*, vol. 248, nos. 1–2, pp. 365–403, Jan. 2017.



KENTARO OHNO received the B.Sci. and M.Sci. degrees in mathematics from The University of Tokyo, in 2017 and 2019, respectively. He is currently pursuing the Ph.D. degree with Waseda University. He is also with NTT as a Researcher. He is studying combinatorial optimization using Ising machines.

TATSUHIKO SHIRAI (Member, IEEE) received the B.Sci., M.Sci., and Dr.Sci. degrees from The University of Tokyo, in 2011, 2013, and 2016, respectively. He is currently an Assistant Professor with the Waseda Institute for Advanced Study, Waseda University. His research interests include quantum dynamics, statistical mechanics, and computational science. He is a member of JPS.

NOZOMU TOGAWA (Member, IEEE) received the B.Eng., M.Eng., and Dr.Eng. degrees in electrical engineering from Waseda University, in 1992, 1994, and 1997, respectively. He is currently a Professor with the Department of Computer Science and Communications Engineering, Waseda University. His research interests include quantum computation and integrated system design. He is a member of ACM, IEICE, and IPSJ.