

RESEARCH ARTICLE

Rejection-Free Monte Carlo Simulation of QUBO and Lechner–Hauke–Zoller Optimization Problems

YOSHIHIRO NAMBU 

NEC-AIST Quantum Technology Cooperative Research Laboratory, National Institute of Advanced Industrial Science and Technology, Tsukuba, Ibaraki 305-8560, Japan

e-mail: y-nambu@aist.go.jp

This work was supported by the New Energy and Industrial Technology Development Organization (NEDO), Japan, under Project JPNP16007.

ABSTRACT Many studies have attempted to develop simple and efficient methods for solving global optimization problems. Simulated annealing (SA) has been recognized as a powerful tool for performing this task. In this study, we implemented a rejection-free Monte Carlo (RFMC) algorithm, which is a kernel of rejection-free SA (RFSA). We showed its validity and advantage in obtaining globally optimal solution for quadratic unconstrained binary optimization (QUBO) and spin-glass problem embedded in the Lechner–Hauke–Zoller (LHZ) architecture. Landscapes of success probabilities of finding the globally optimal solution as well as feasible solutions were evaluated as a function of a set of hyper-parameters used to characterize the weights of cost and penalty functions. We demonstrated that the efficiency of RFMC was greatly enhanced compared to that of standard MC. Furthermore, we found that the landscapes for QUBO and those for LHZ problem show considerably different views. When we focus on a specific class of problems, the smaller the problem size, the more scattered is the success probability. We also show that the success probabilities can be further improved by avoiding inefficiencies due to cycling of state transitions using short-term memory mechanisms. We suggest a reasonable strategy to tune hyper-parameters and find the correct global solution using RFSA.


INDEX TERMS Monte Carlo, simulation, annealing, combinatorial optimization, quadratic unconstrained binary optimization, Ising model, Lechner–Hauke–Zoller (LHZ) architecture.

I. INTRODUCTION

Combinatorial optimization is a classical, but important, problem in the field of mathematical optimization. It focuses on finding the optimal object from a finite set of objects, where the set of feasible solutions is discrete or can be reduced to a discrete set. It has a wide range of practical applications such as logistics [1], supply chain optimization [2], and earth science problems [3], [4], [5]. Numerous algorithms have been developed to solve the combinatorial optimization problem; these can be classified into two classes: approximation (heuristics [6] and meta-heuristics [7]) and exact algorithms, widely known as NP-hard problems. Among

the approximation algorithms, simulated annealing (SA) [8] has become an increasingly powerful tool because of the development of faster computers. It is a nature-inspired approximate computational model and is based on probabilistic simulation of the dynamic process of a physical system. Recently, research interest in natural computing has increased, wherein nature-inspired novel computing hardware are analyzed based on interesting natural systems, such as neural networks, molecules, DNAs, and quantum computers.

SA, together with the Ising model [8], [9], is a classical and convenient tool for solving various classes of combinatorial optimization problems. The standard SA is usually implemented based on Markov chain Monte Carlo (MCMC) simulation with the Metropolis (or Gibbs) rule. Although

The associate editor coordinating the review of this manuscript and approving it for publication was Mostafa M. Fouda .

standard SA can be easily implemented on digital computers, finding the globally optimal solution is challenging because it requires tuning the hyper-parameter schedule for each problem. This is a common problem faced when using SA, heuristics, and meta-heuristics. In addition, the standard SA may become very inefficient under some situations; for example, at low temperatures where the spin state is trapped in local energy minima, i.e., the local solution. This is because the spin-selection algorithm of the standard SA, in which a spin to be inverted is chosen based on the acceptance–rejection (AR) method [10], is a bottleneck. In the AR method with the Metropolis rule, a trial spin is randomly proposed, and it is accepted to be inverted with a probability $a(\Delta E_i/T) = \min[1, \exp(-\Delta E_i/T)]$ where ΔE_i and T denote the energy shift due to inverting spin i and temperature of the spins, respectively. If the state is trapped in the local solution, then $\Delta E_i \geq \Delta E > 0$, where ΔE is an energy barrier separating the minima. The time required for spins to escape from a local minimum (escaping time [11] or waiting time [12], [13]) via thermal fluctuations should be $t_{SA} \sim \langle t_{AR} \rangle = b_{AR} \langle a(\Delta E_i/T)^{-1} \rangle \geq b_{AR} \exp(\Delta E/T)$, where b_{AR} is the computational time associated with a single acceptance–rejection decision and $\langle a(\Delta E_i/T)^{-1} \rangle$ is the average number of repetitions required until a proposal is accepted. The escaping time t_{AR} depends on the problem size N because ΔE in an exponential factor $\exp(\Delta E/T)$ depends on N and may diverge when $N \rightarrow \infty$ [14], [15]. For a small T and large ΔE (large N), t_{SA} becomes very large. Then, it is likely that the spin-selection algorithm will become a bottleneck and will dominate over the computational time in the standard SA.

The problem of this inefficiency in the standard SA has been frequently highlighted. It could be overcome by quantum annealing (QA). In [11], Denchev *et al.* suggested the advantages of QA over SA. They argued that the escaping time t_{QA} in QA is governed by quantum tunneling and could be smaller than t_{SA} for possible situations. In contrast, this inefficiency is reduced by modifying the spin-selection algorithm. This is possible because the AR method is not the only method for spin selection required by the MCMC simulation. More efficient algorithms can be chosen for the same task. For example, algorithms for roulette wheel selection in a genetic algorithm [16], which is equivalent to non-uniform random number generation, can also be used. The task is to return an index of a single item drawn from a set of weights $\{w_1, \dots, w_N\}$ according to a probability distribution $\mathbb{P}(i) = w_i/W$, where $w_i = a(\Delta E_i/T)$ and $W = \sum_{1 \leq i \leq N} w_i$. Numerous algorithms other than the AR method can be used for this task, including cumulative sum, Walker’s alias method [17], [18], and partial sum tree [19]. Numerous studies have focused on accelerating this task by employing parallel computing [18], [19], [20]. Notably, it is the dimensionality-reduction operation that converts a set of real values $\{w_1, \dots, w_N\}$ into a single integer $i \in [1, N] \in \mathbb{N}$. It is impossible to reduce the computational time of the reduction operation to $1/N$ by computing N subdivided, independent, and similar processes in parallel using a GPU or vector

processor. This is because a reduction operation requires communication among subdivided processes to obtain the correct result, the overhead of which limits the effectiveness of parallelization.

If we can implement efficient parallel reduction in the spin-selection algorithm so that its computational time t_{PR} is $t_{PR} \ll \langle t_{AR} \rangle \sim b_{AR} \exp(\Delta E/T)$, we can improve the efficiency of SA. This is known as rejection-free SA (RFSA), where time-wasting rejection events in the algorithm are eliminated [12], [13], [21], [22], [23], [24], [25], [26]. RFSA has attracted considerable interest from researchers since its invention [21]. Nevertheless, little is known regarding the results when it is applied to combinatorial optimization. The purpose of this study is to demonstrate validity and advantage of RFSA in finding correct solutions of combinatorial optimization problems. In this study, we investigated the potential of rejection-free Monte Carlo (RFMC), which is a kernel of RFSA. We considered two problems: quadratic unconstrained binary optimization (QUBO) [27], [28], [29], [30] and optimization of the maximum cut problem [30], [31] embedded in the Lechner–Hauke–Zoller (LHZ) architecture [32]. A common feature of these problems is that they have two types of objective functions to be optimized, i.e., cost and penalty functions, along with the associated hyper-parameters. The success probabilities of finding the globally optimal and feasible solutions are evaluated as a function of a set of these hyper-parameters. We present validity and advantages of RFSA as well as new findings, suggesting a reasonable strategy to tune the hyper-parameter settings using RFSA.

The rest of this paper is organized as follows. In Sec. II, we explain three essential elements to implement RFSA and principle of operation. In Sec. III, we define two hyper-parameters to reduce a constrained optimization problem to an unconstrained one. We also explain freedom of choosing these parameters and their mutual relationship. In Sec. IV, we describe our experimental approach. In Sec. V, we show various experimental results for QUBO and LHZ architecture. In Sec. VI, we suggest a strategy to efficiently tune the hyper-parameters from experimental observations. Section VII concludes the new knowledge obtained in this study. The Appendix provides supplemental information for simulating the LHZ architecture.

II. IMPLEMENTATION AND PRINCIPLE OF OPERATION

In this study, we formulated all the problems using Ising spins [8], [9], [28]. The maximum cut problem is equivalent to finding the ground state of spin-glass from this perspective. This section explains how to implement RFSA for the Ising model with all-to-all connectivity and for the LHZ architecture. There are three elements for RFSA that are not necessary but sometimes implemented in the standard SA. The first is forward computation of the energy ΔE_i ($i = 1, \dots, N$) before spin selection. We need knowledge regarding a set of values $\{\Delta E_1, \dots, \Delta E_N\}$ to obtain a set of weights $\{w_1, \dots, w_N\}$ in the spin selection. This technique has already been proposed

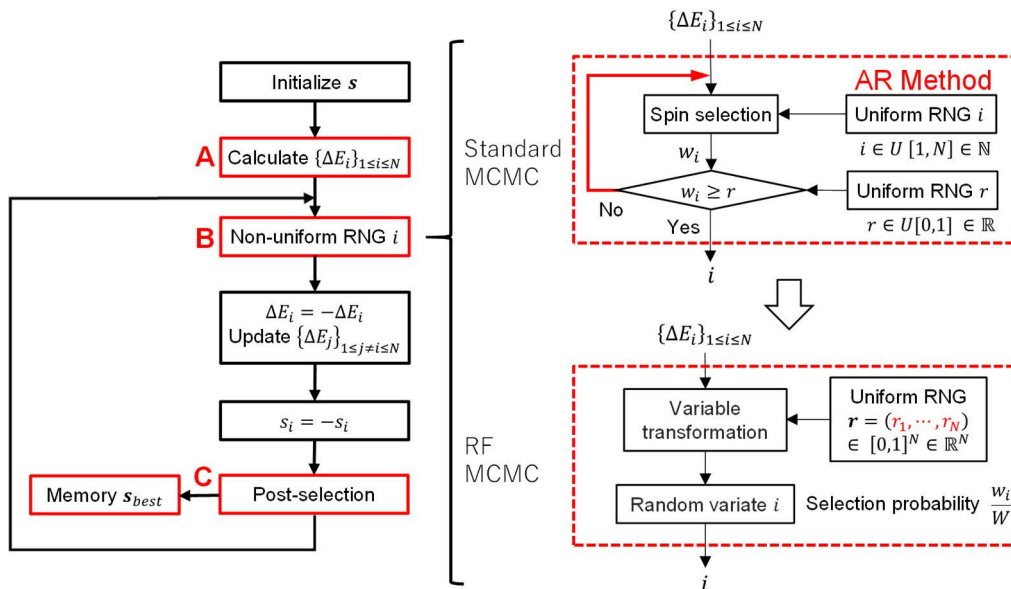


FIGURE 1. MCMC loop in the RFSA algorithm. The three elements (A-C) are indicated in this diagram. Non-uniform random number generation (RNG) is implemented by the AR method in the standard SA (upper right). In contrast, it is implemented by the other efficient dimensionality-reduction algorithm (lower right), which is considered as a transformation of uniform random variables into non-uniform ones.

to accelerate the standard SA using parallel computing [33]. The second is non-uniform random number generation to choose a spin to be inverted. It should generate a random number i with the probability distribution weighted by w_i . The third is the post-selection of the best solution so far and keeping track of it in a memory during the SA. In the standard SA, spins are gradually cooled close to absolute zero temperature, and the quasi-steady state at the end of the calculation is accepted as the solution. In contrast, Ising spins are kept at some finite temperature in our implementation. The resultant thermal fluctuations enable us to perform the solution search incessantly over time [34]. Figure 1 illustrates the MCMC loop in the RFSA algorithm (cooling loop is omitted here). In the following sections, we will discuss the three elements in the RFSA algorithm in some detail.

A. FORWARD COMPUTATION OF THE ENERGY SHIFT

At the initialization stage of the calculation, we generate an initial random spin configuration vector $s = (s_1, \dots, s_N)$, where s_i denotes the state of the i -th spin taking a random value of 1 or -1 , and compute ΔE_i for all i . Then, if spin j is chosen to be inverted, we invert the signs of s_j and ΔE_j . We also update ΔE_i associated with spin $i \neq j$ which is connected to spin j .

In the Ising model, spin i may be connected to all the spin $j \neq i$. The coupling between spin i and j is given by the matrix elements $\{J_{ij}\}_{1 \leq i < j \leq N}$, and ΔE_i should be updated by spin inversion as follows:

$$\Delta E_i \leftarrow -\delta_{ij} \Delta E_j + (1 - \delta_{ij}) (\Delta E_j - 4s_i J_{ij} s_j), \quad (1)$$

where s_i designates the state before its sign inversion. The calculation for $i \neq j$ includes the Hadamard product of the J -matrix and the tensor product $s \otimes s$ of the spin configuration

vector s , which can be performed in parallel with an efficient continuous data structure. Thus, it is possible to accelerate it considerably using a GPU and a vector processor.

In contrast, we could not simply represent the LHZ architecture within an Ising model because it involves four-way couplings. Although we can perform similar forward computations in the LHZ architecture, we only need to update the value of ΔE_i associated with a set of neighboring eight spins that are coupled to an inverted spin. Its calculation requires a small calculation power but involves a noncontiguous data structure because the memories storing eight spin configurations are not necessarily allocated sequentially. See the Appendix for more details on the LHZ architecture.

B. NON-UNIFORM RANDOM NUMBER GENERATION

Numerous parallel reduction algorithms can efficiently generate non-uniform random numbers, but the best choice is currently unproven. It should select integer $i \in [1, N] \in \mathbb{N}$ randomly with a selection probability w_i/W . Non-uniform random number generator (RNG) can be implemented as a transformation of N i.i.d. uniform random variables $r_i \in [0, 1] \in \mathbb{R}$ [26], [35] which can be generated using the standard RNG. In our calculations, we adopted a method based on the following properties of exponentially distributed random variables [35]. Let x_1, \dots, x_N be N independent exponentially distributed random variables with rate parameters $\lambda_1, \dots, \lambda_N$. Then,

$$\min \{x_1, \dots, x_N\} \quad (2)$$

is also exponentially distributed, with a rate parameter

$$\lambda = \lambda_1 + \dots + \lambda_N. \quad (3)$$

Then, the index of the random variable that achieves the minimum is distributed according to the categorical distribution

$$Pr(i = \arg \min \{x_1, \dots, x_N\}) = \frac{\lambda_i}{\lambda}. \quad (4)$$

Therefore, if we generate independent exponential random variables y_1, \dots, y_N , whose rate parameters are given by $\lambda_i = w_i$, and calculate

$$i = \arg \min \{y_1, \dots, y_N\}, \quad (5)$$

we obtain a random variable with a selection probability, w_i/W . The exponential random variables y_1, \dots, y_N with rate parameters w_1, \dots, w_N can be generated from the i.i.d. uniform random variables r_1, \dots, r_N with $r_i \in [0, 1] \in \mathbb{R}$ by the inversion method as follows:

$$y_i = -\frac{1}{w_i} \log r_i. \quad (6)$$

Note that random variable y_i is closely connected to the waiting time [12]. Furthermore, if we note that $\log x$ is a monotonically increasing function of x , it follows that

$$i = \arg \min_i \left\{ \log(-\log r_i) + \max\left(\frac{\Delta E_i}{T}, 0\right) \right\}, \quad (7)$$

where we have considered the Metropolis rule $w_i = a(\Delta E_i/T)$. In this algorithm, $\arg \min\{\}$ in eq.(5) is a dimensionality-reduction operation, since it involves finding the index of the minimum element of a set of N values. Because such a reduction operation is routinely used in machine learning, it is promising that efficient parallel reduction codes, libraries and hardware engines are available. We think this would be an advantage of the present algorithm.

C. POST-SELECTION OF THE INCUMBENT SOLUTION (MEMORY ALGORITHM)

To find the best solution in the trajectory of the spin states generated according to sequential application of the elements A and B, we post-select the spin configuration that is best thus far, record it, and keep track of it in the memory as an incumbent solution. This can be performed for an arbitrary timing, for example, every MC step or every annealing step. Frequent post-selection results in an exhaustive search of the fine solutions at the cost of calculation time because it avoids missing a solution. Thus, there is a trade-off between benefit and cost. Probably, the simplest strategy is as follows. We keep track of the total energy of spins E in addition to ΔE_i . We compare E and its incumbent value E^{best} at some timing, for example, every annealing step. If E is smaller than E^{best} , we update E^{best} as $E^{\text{best}} \leftarrow E$ and $s_{\text{best}} \leftarrow s$, where s_{best} is the incumbent solution. Finally, we output s_{best} as a solution. Because $E = -\frac{1}{4} \sum_i \Delta E_i$ and E should be updated such that $E \leftarrow E + \Delta E_i$ after chosen spin inverts for the Ising model, it is sufficient to keep track of ΔE_i in order to keep track of E . Similarly, it is also easy to keep track of ΔE_i and E for the LHZ architecture. Alternatively, we can consider a more sophisticated post-selection method. For example, consider the total energy of spins comprises two

parts: $E = E_1 + E_2$. We can keep track of energy E_1 and E_2 as well as their shifts $(\Delta E_1)_i$ and $(\Delta E_2)_i$ due to inversion of chosen spin separately. Then, for example, we can post-select the incumbent solution as follows. We compare E_2 and its incumbent value E_2^{best} at some timing. If E_2 is smaller than E_2^{best} , we update E_2^{best} as $E_2^{\text{best}} \leftarrow E_2$ and follow the next step: if E_1 is smaller than E_1^{best} , we update E_1^{best} as $E_1^{\text{best}} \leftarrow E_1$ and $s_{\text{best}} \leftarrow s$. Finally, we output s_{best} as the solution. For the QUBO problem and LHZ architecture, cost and penalty functions can be reasonably assigned to E_1 and E_2 , respectively. This sophisticated post-selection can search for the optimum solutions more exhaustively but incurs a higher calculation cost. The elements B and C enable us to perform an incessant solution search during calculation for our RFSA. Note that implementing these components on a digital computer is easy, but realizing them in QA is not necessarily easy. This is because we require some non-natural function or artificial intelligent agent to realize them in QA.

The performance of RFSA depends on machine time to be spent. The larger the machine time, the higher is the performance, which will be shown later. In contrast, it is usual to accept, for the solution, the quasi-steady state at the end of the calculation in QA. Then, an ensemble of solutions can be obtained by repeating a large number of independent calculations with a short duration to improve the performance of QA. The final solution is obtained by finding the best solution in the ensemble of solutions obtained by repeated calculations. Its performance depends on the number of repeated experiments, which specifies the machine time to be spent. Therefore, RFSA and QA obtain the solution in a different manner, that is, RFSA by a single long calculation and QA by a series of short calculations. In other words, whereas RFSA probabilistically approaches the solution through time series analysis, QA approaches it by means of ensemble analysis.

D. PRINCIPLE OF OPERATION

Before we proceed, let us discuss whether RFSA explores the globally optimum solution efficiently. To discuss this, let us consider three models of SA shown in Fig. 2. In this figure, model (a) indicates the most classical type of SA. In this model, the states when the proposed spin inversion is rejected (hereafter, rejected states) are retained together with the states when the proposed spin inversion is accepted and inverted (hereafter, accepted states) in the time series of a run, forming the original chain $\{S_n\}$. The chain $\{S_n\}$ is a Markov chain and converges to the thermal equilibrium state; that is, the statistical distribution of the states in $\{S_n\}$ approaches the Boltzmann distribution asymptotically in the limit of its length $\rightarrow \infty$. In contrast, model (b) is a modified version of model (a). In this model, the rejected states are simply disposed, and only the accepted states are retained in the time series of a run, which makes up the smaller chain $\{s_n\}$, called the jump chain [23]. The jump chain is also a Markov chain; however, it converges to the stationary distribution different from the Boltzmann distribution because the rejected states are absent [22], [23], [24].

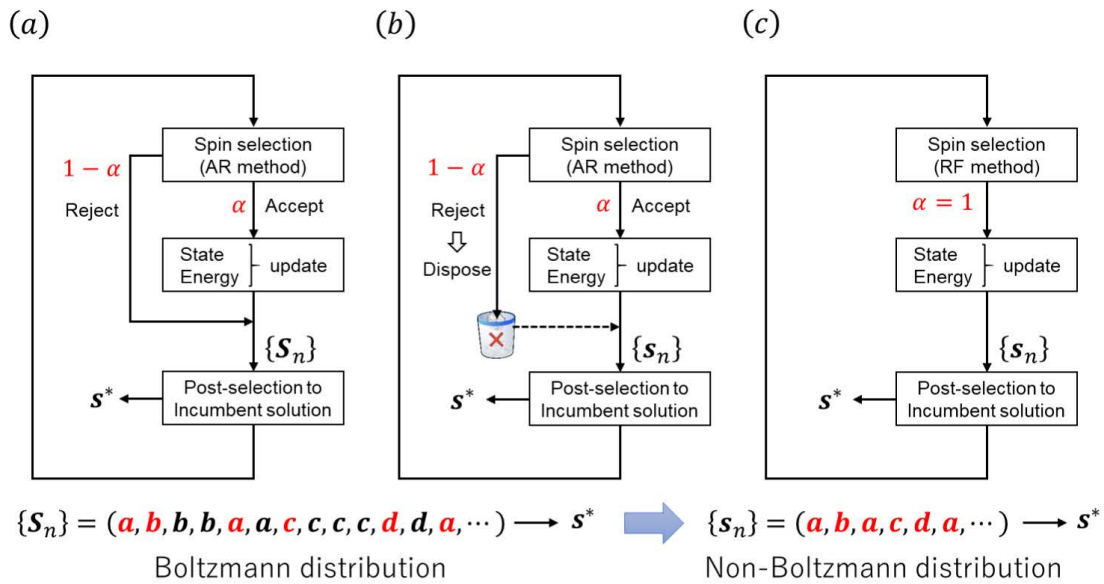
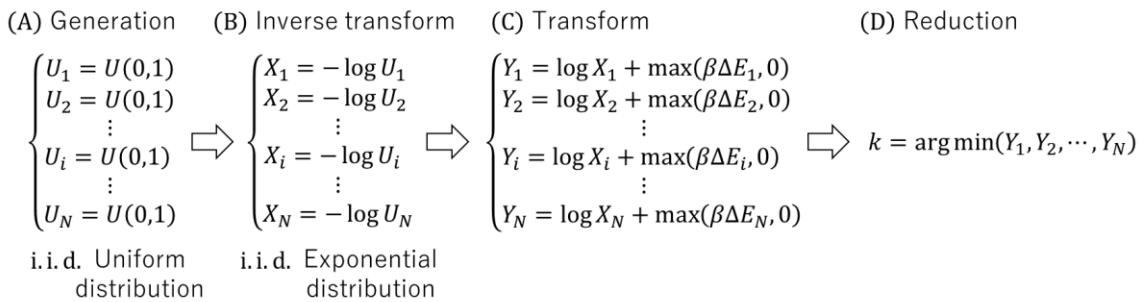


FIGURE 2. Schematic diagrams of three SA models. Only the MCMC loop is shown here. (a): Classical model where the original chain $\{S_n\}$ is observed. In this model, the rejected states as well as the accepted states are retained in $\{S_n\}$. In model (b), the rejected states are disposed and only the accepted states are retained in the jump chain $\{s_n\}$. In model (c), the spin-selection algorithm is different from the AR method but has the same function as the AR method. α : acceptance rate, s^* : optimal solution, a, b, c, d : some spin configurations.

RF SA



Standard SA

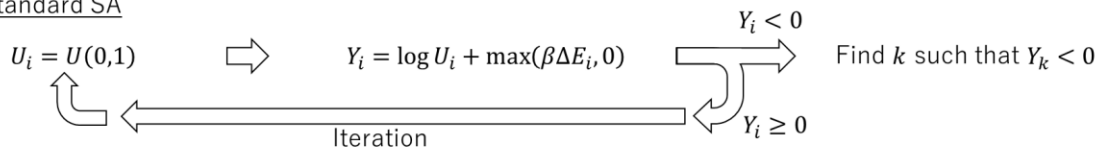


FIGURE 3. Schematic diagrams illustrating non-uniform random number generation in the spin-selection algorithm of the RFSA (upper) and the standard SA (lower).

Notably, the success probability of finding the globally optimal solution should solely depend on the number of accepted states, that is, the rejected states have no contribution because we keep track of the incumbent solution. Therefore, as long as the number of accepted states in $\{S_n\}$ and length of $\{s_n\}$ are the same, models (a) and (b) should have the same. Therefore, the performance should be independent of whether the rejected states are retained or disposed.

Next, we focus on model (c), which corresponds to the RFSA. In this model, no intrinsically rejected events exist, and the solution search is driven solely by the accepted states. A spin-selection algorithm presented in element B allows us to realize this model. It is evident that, as long as the length of $\{s_n\}$ remains the same, models (b) and (c) should show the same performance for finding the globally optimal solution. Nevertheless, an implicit difference exists between

models (b) and (c). Every spin state s_n in the chain $\{s_n\}$ is synchronously generated with spin selection in the model (c), whereas it is asynchronously generated in model (b) because a geometrically distributed random number of rejected events must be necessarily iterated after every accepted event in the original chain $\{S_n\}$ [23]. Then, the following question arises: is model (c) more efficient than models (a) and (b)?

To answer this question, recall that the most serious problem in the standard SA is inefficiency under the low temperature situation, where spins are frequently trapped in the local minima. In this case, spin selection by the AR method is inefficient and becomes a bottleneck because a large number of iterations of rejected events are required for spins to be successfully escaping from the local minima. The acceptance rate α in the AR method becomes much smaller than unity, which slows down the solution search. In contrast, we may avoid time-consuming rejected events in the RFSA at the cost of an increased calculation cost in the renewed spin-selection algorithm. Therefore, the question should be discussed in the light of the cost–benefit perspective.

To discuss the calculation cost of the spin-selection algorithm of the RFSA, we illustrate its four building-blocks (A)–(D) in Fig. 3.

- (A) Generation of N i.i.d. pseudo-random real variables (U_1, U_2, \dots, U_N) that are uniformly distributed in the $(0,1)$ interval.
- (B) Transform (U_1, U_2, \dots, U_N) to N i.i.d. exponentially distributed random variables (X_1, X_2, \dots, X_N) .
- (C) Transform (X_1, X_2, \dots, X_N) to N random variables (Y_1, Y_2, \dots, Y_N) .
- (D) A reduction operation that finds the minimum element in (Y_1, Y_2, \dots, Y_N) , and return its index i (eq.(5)).

At first glance, it looks much more complicated than that of the AR method. However, this is not necessarily serious because the calculation in blocks (A)–(C) comprises independent N calculations; thus, they can be accelerated by parallel computation. Therefore, the answer for the question depends on various factors. They include the problem size N , degree of parallelization in the program, how efficient the reduction operation (D) can be, and so on. Later, we will experimentally demonstrate two instances showing the advantage of the RFSA.

III. HYPER-PARAMETERS AND ANNEALING SCHEDULE

Although hyper-parameters such as temperature are dynamically controlled in the RFSA, we investigate its performance under static hyper-parameter setting in this study. In other words, we focus attention on the performance of the kernel of the RFSA, i.e., the rejection-free Monte Carlo (RFMC) loop. We consider problems that have two kinds of objective functions—cost and penalty functions—and associated hyper-parameters. The penalty function is introduced to replace the constrained optimization to an unconstrained one. We must tune two hyper-parameters that specify the weight of two functions to succeed in optimization. Numerous techniques have been developed to tune those

parameters in the standard SA. They involve automated training using machine learning, for which, alternative hyper-parameters should be tuned. The hyper-parameters depend not only on the underlying dataset but also on the solution search algorithm. Nonetheless, few studies have reported hyper-parameter tuning for combinatorial optimization based on RFSA [26]. Furthermore, we must pay attention to the fact that the hyper-parameters can be defined only up to scaling factors. Let us start with considering several choices of hyper-parameters and their mutual relations. We will further discuss the scaling properties of the probabilities on the hyper-parameters.

The constrained optimization problem can be generally formulated as finding an optimal solution s^* that minimizes the cost function:

$$s^* = \arg \min_{s \in S'} \mathcal{H}_c(s), \tag{8}$$

where S' denotes a finite set of spin configurations allowed by a given constraint, which is called the feasible region. It can be restated that

$$s^* = \arg \min_{s^\# \in S} \mathcal{H}_c(s^\#). \tag{9}$$

under the constraint that $s^\#$ is subject to

$$s^\# \in S' \text{ where } \mathcal{H}_p(s^\#) = 0, \tag{10}$$

where $S \ni S'$ denotes a finite set of extended spin configurations, which we call the search space, and $S - S'$ is called the infeasible region. \mathcal{H}_p is the penalty function [36], which is quadratic in s_i for QUBO and quartic in s_i for the LHZ architecture. Eq. (10) constrains s within a feasible region S' . In eqs. (8) and (10), we must minimize \mathcal{H}_p prior to \mathcal{H}_c to find the globally optimal solution. Note that we can use this requirement for exhaustive post-selection of the best solution as shown before. The task can be relaxed into a formally simpler unconstrained global optimization that minimizes a linear combination of \mathcal{H}_c and \mathcal{H}_p :

$$\mathcal{H} = \beta_c \mathcal{H}_c + \beta_p \mathcal{H}_p, \tag{11}$$

where β_c and β_p are positive parameters and are called cost and penalty hyper-parameters, respectively, or collectively referred as regularization (relaxation) hyper-parameters. They must be tuned so that the globally optimal solution becomes the ground state of \mathcal{H} as well as that of \mathcal{H}_p but not that of \mathcal{H}_c . To improve the performance, we may tune their value as well as their temporal schedule. Note that choice of hyper-parameters is not unique. For example, we can rewrite eq. (11) and define the new sets of hyper-parameters (T_c, k_p) and (k_c, T_p) as follows:

$$\mathcal{H} = \beta_c \left(\mathcal{H}_c + \frac{\beta_p}{\beta_c} \mathcal{H}_p \right) = \frac{1}{T_c} (\mathcal{H}_c + k_p \mathcal{H}_p), \tag{12}$$

$$\mathcal{H} = \beta_p \left(\frac{\beta_c}{\beta_p} \mathcal{H}_c + \mathcal{H}_p \right) = \frac{1}{T_p} (k_c \mathcal{H}_c + \mathcal{H}_p), \tag{13}$$

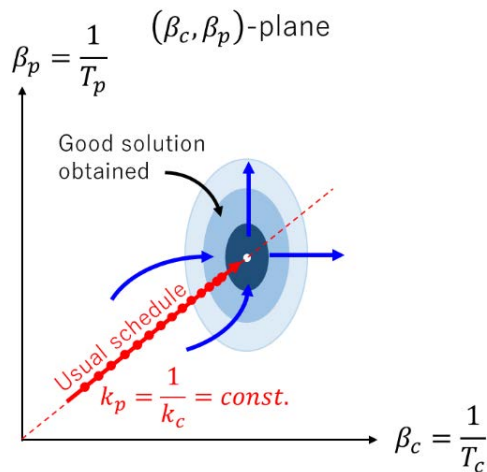


FIGURE 4. Hyper-parameter scheduling on their spaces. Hyper-parameter scheduling corresponds to the trajectory and density of a series of points $\{(\beta_c, \beta_p)\}$ on the (β_c, β_p) -plane.

where

$$T_c = \frac{1}{\beta_c}, \tag{14}$$

$$T_p = \frac{1}{\beta_p}, \tag{15}$$

$$k_p = k_c^{-1} = \frac{\beta_p}{\beta_c}. \tag{16}$$

In eqs. (12) and (13), T_c and T_p are parameters that are common for cost and penalty functions \mathcal{H}_c and \mathcal{H}_p , which can be regarded as temperature. It is considered that k_p should be large enough to satisfy eqs. (8) and (10).

Note that tuning the hyper-parameter schedule amounts to designing the sequences $\{(x, y)_k\}$, where $(x, y)_k$ denotes a value of the set of hyper-parameters (x, y) in the k -th iteration. For example, let us consider the sequence $\{(\beta_c, \beta_p)_k\}$. If we regard (β_c, β_p) as the coordinate of a point in a two-dimensional space, the trajectory and density of a set of the points associated with the sequence give the annealing schedule. In the standard SA, the schedule where $k_p = k_c^{-1} = const.$ is usually chosen, where its trajectory gives the straight line from the origin, as shown in Fig. 4. In this case, only one of (β_c, β_p) is a controllable hyper-parameter. In general, we can choose the arbitrary sequence $\{(\beta_c, \beta_p)_k\}$ where both of (β_c, β_p) can be freely controlled. As discussed above, the strategy based on varying hyper-parameters (β_c, β_p) during the simulation is called a dynamic penalty function strategy [36].

We must pay attention to the fact that \mathcal{H} is unique only up to scaling factors in the optimization problem. The globally optimal solution is invariant under the scaling transformation $\mathcal{H} \rightarrow \mu\mathcal{H}$, where μ is an arbitrary positive constant. This means that reported values of hyper-parameters may differ from researcher to researcher, although those parameters provide the objective function with the same globally optimal solution. For example, suppose that \mathcal{H} is written as eq. (11)

and that we transform \mathcal{H}_c and \mathcal{H}_p according to

$$\begin{cases} \mathcal{H}_c \rightarrow \kappa_c \mathcal{H}_c \\ \mathcal{H}_p \rightarrow \kappa_p \mathcal{H}_p, \end{cases} \tag{17}$$

where κ_c and κ_p are arbitrary positive constants, i.e., they are scaling factors. Then, if we also transform β_c and β_p according to

$$\begin{cases} \beta_c \rightarrow \mu\kappa_c^{-1}\beta_c \\ \beta_p \rightarrow \mu\kappa_p^{-1}\beta_p, \end{cases} \tag{18}$$

we get transformation $\mathcal{H} \rightarrow \mu\mathcal{H}$ from eqs. (17) and (18). Then, we see that the globally optimal solution is invariant under the scaling transformation:

$$\{\mathcal{H}_c, \mathcal{H}_p, \beta_c, \beta_p\} \rightarrow \{\kappa_c \mathcal{H}_c, \kappa_p \mathcal{H}_p, \mu\kappa_c^{-1}\beta_c, \mu\kappa_p^{-1}\beta_p\}.$$

Similarly, the globally optimal solution is invariant under the scaling transformations

$$\{\mathcal{H}_c, \mathcal{H}_p, k_p, T_c\} \rightarrow \{\kappa_c \mathcal{H}_c, \kappa_p \mathcal{H}_p, \mu(\kappa_c/\kappa_p)k_p, \mu\kappa_c T_c\}$$

and

$$\{\mathcal{H}_c, \mathcal{H}_p, k_c, T_p\} \rightarrow \{\kappa_c \mathcal{H}_c, \kappa_p \mathcal{H}_p, \mu(\kappa_p/\kappa_c)k_c, \mu\kappa_p T_p\}.$$

This indicates that there are an infinite number of objective functions and associated hyper-parameters leading to the same globally optimal solution. To the best of the author’s knowledge, no commonly accepted method for normalizing \mathcal{H}_c and \mathcal{H}_p is known. This might pose the problem that it is practically difficult for researchers to share or compare the numerical value of the optimal hyper-parameters of their own for various problem instances.

IV. EXPERIMENTS

A. SIMULATION PLATFORM

All the simulations were performed using the Mathematica® Ver. 13 platform on Windows 10/11 operating systems. We implemented the algorithm involving the common elements A and C for RFSA and standard SA simulations. For the standard SA experiment, we used the simple AR method for elements B. In advance, we confirmed that our method for the non-uniform random number generation for elements B and the AR method can generate random variates with the same probability distribution by independent simulation. In Mathematica®, certain calculations can be accelerated using listable built-in functions in a similar way as the Python interpreter. Ideally, if we want to perform N similar and independent calculations at the same time, they can be accelerated $\sim M$ times compared to sequential calculations if we can perform M calculations in parallel. This mechanism would be implemented in the listable built-in functions. An independent experiment implies that that listable built-in function in Mathematica® can perform $M \sim 200$ calculations in parallel on a single processor core. To achieve effective acceleration by parallel computing, we further designed a data structure suitable for sequential memory access.

The processes of the calculation of the energy shift ΔE_i after inverting spin i (see eq.(1)) in QUBO problems and calculation of the argument in eq. (7) may be accelerated as such. Furthermore, some functions can be largely accelerated by compiling with explicitly specified data types. Furthermore, the $\arg \min\{\}$ task in eq. (5) might be accelerated due to the parallel reduction technique in the built-in function of Mathematica[®]. Unfortunately, details of its mechanism are beyond the present investigation.

In addition to the above parallelization techniques introduced within the MCMC loop, which corresponds to fine-grain parallelism, we introduced coarse-grain parallelism into our simulation using the parallel computing mechanism built in Mathematica[®]. Our calculation was performed on nine distributed computers with their own multi-core processor/processors. The hardware specifications of the computers used for the experiment are listed in TABLE 1. In total, 83 processor cores were used, which constitute 82 slave nodes on a distributed memory MIMD architecture with a master node. This enables us to perform independent SA calculations in parallel across large numbers of independent, asynchronous nodes. Only input data and the outcome are shared across the slave nodes via the master node using Ethernet communication. After the initial states are generated, each node follows its own SA calculation independently, and no communication occurs until the end. Only the best results that were post-selected in the slave nodes are reported to the master node at the end of the calculation. Thus, no cooperation and interaction occur among the slave nodes. This is arguably the simplest parallelization scheme called parallel independent annealing (PIA) [37], [38], [39], [40], [41] or multiple independent runs (MIR) [42]. This coarse-grain parallelism was used for statistical analysis of the SA results. We made a set of 300 SA calculations to evaluate the success probabilities to find the exact or best-known solution for each fixed hyper-parameter setting. If the reported best result is feasible and its energy agrees with that of the exact or best-known solution, it is counted as a successful solution and success probabilities are evaluated.

In summary, we introduced both fine-grained parallelism for accelerating calculation in each SA and coarse-grain parallelism for accelerating statistical evaluation of the SA results.

B. OPTIMIZATION BASED ON THE STATIC HYPER-PARAMETER STRATEGY

A great deal of effort has been devoted to study how to tune the hyper-parameter schedule for combinatorial optimization using SA, but such research is still challenging. In the following section, we focus on studying optimization using SA under the simple static hyper-parameter strategy where hyper-parameters are assumed to be fixed during the simulation [36], because even if it is not optimal, we believe that it should be a starting point for tuning the annealing schedule further. We experimentally investigated and compared the performance of standard SA and RFSA. Because the

hyper-parameters are fixed, our experiment amounts to simple MCMC simulations, which is a kernel of SAs. We devote our efforts to evaluating the success probabilities of finding the exact or best-known solution as a function of three mutually connected sets of hyper-parameters (β_c, β_p) , (k_p, T_c) , and (k_c, T_p) . As a demonstrative example, we chose two types of problems: QUBO and optimization of the maximum cut problems embedded in the LHZ architecture. We visualized the evaluated success probabilities on a three-dimensional viewgraph.

TABLE 1. Computers used for the experiment.

System	CPU	Number of cores	Clock	Memory
#1 (master)	Dual Intel Xeon E5-2643v3	12	3.4 GHz	128GB
#2	Dual Intel Xeon Gold 5220	36	2.2 GHz	64GB
#3	Intel Core i7-9700T	8	2.0 GHz	16GB
#4	Intel Core i7-9700T	8	2.0 GHz	16GB
#5	Intel Core i7-9700	8	3.0 GHz	16GB
#6	Intel Core i7-9700	8	3.0 GHz	16GB
#7	Intel Core i7-10700	8	2.9 GHz	16GB
#8	Intel Core i7-10700	8	2.9 GHz	16GB
#9	Intel Core i7-8700T	6	2.4 GHz	16GB

V. RESULTS AND DISCUSSIONS

First, we experimentally compared the performances of three SA models (a)-(c) shown in Sec. II. The burma14 instance in TSPLIB [43] was used for a QUBO instance. We evaluated the success probability P_{suc}^o of finding the globally optimal solution as well as P_{suc}^f of finding the feasible solutions at the specific set of hyper-parameters (β_c, β_p) . To perform statistical analysis, we averaged the results over 2000 calculations for models (a) and (b) and those over 4000 calculations for model (c). P_{suc}^i ($i = o, f$) depend on the choice of (β_c, β_p) . We chose a special (β_c, β_p) which gives the optimal results for P_{suc}^o as shown later. In Fig. 5, the typical experimental results of P_{suc}^o (black circles) and P_{suc}^f (blue squares) associated with the three SA models are shown as a function of the number of performed spin inversions N_{SI} in the MCMC calculation. Let us denote length of chain $\{\mathbf{x}_n\}$ as $|\{\mathbf{x}_n\}|$. Then, in models (b) and (c), $N_{SI} = |\{s_n\}|$. In model (a), N_{SI} was counted using a counter variable that increments its value when the spin selection is accepted, which is implemented to evaluate the acceptance rate $\alpha = N_{SI} / |\{\mathbf{S}_n\}|$. The calculation cost per experiment was also plotted as the wall-clock time (red triangles). We can see that the wall-clock times are almost proportional to N_{SI} for all models. Note that the wall-clock times for models (a) and (b) are comparable, whereas that of model (c) is smaller by a factor of ~ 20 . This improved calculation efficiency clearly indicates the advantage of the RFSA over the standard SA.

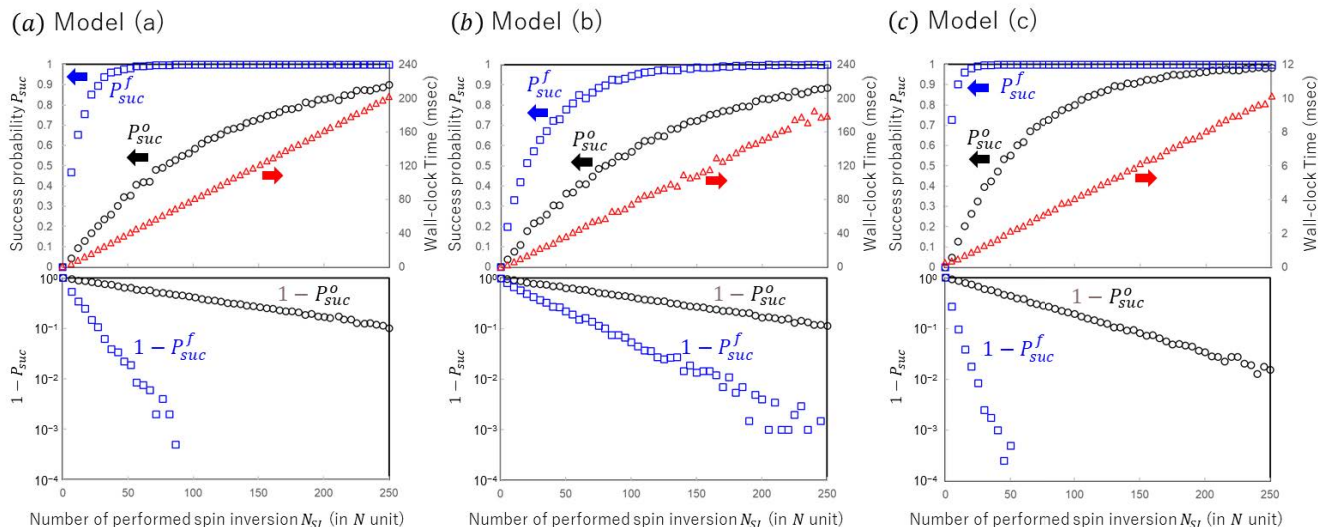


FIGURE 5. Results of simulation for the three SA models (a)–(c) with static hyper parameter settings. The upper diagrams show the success probabilities P_{suc}^o (black circles) and P_{suc}^f (blue squares) as well as wall-clock time (red open triangles) as a function of the number of performed spin inversion N_{SJ} . The lower diagrams show $\log(1 - P_{suc}^i)$ ($i = o, f$) as a function of N_{SJ} .

In contrast, we can see that N_{SJ} also characterizes the performance for the SA models. In the bottom diagrams in Fig. 5, the values of $\ln(1 - P_{suc}^i)$ are plotted against N_{SJ} . These diagrams show that $1 - P_{suc}^i$ exponentially decreases against N_{SJ} both for $i = o$ and f in all models. This implies that P_{suc}^i can be well described by a Bernoulli trial, which is a special case of the Markov chain. This is consistent if $\{s_n\}$ converges with its own stationary distribution very quickly. For example, consider the probability of finding the globally optimal solution s^* in the chain $\{s_n\}$ with length $N_{SJ} \rightarrow \infty$. This might be equal to the probability p^* of finding the ground state s_g in the stationary state of the chain $\{s_n\}$ if we appropriately choose (β_c, β_p) . Unfortunately, it is not clear whether $\{s_n\}$ approaches the stationary state if we consider the chain $\{s_n\}$ with finite length N_{SJ} . Nonetheless, we postulate that $\{s_n\}$ approaches its stationary state very quickly, so that the probability of finding s^* in the chain $\{s_n\}$ is p^* , even if N_{SJ} is very small. Then, if we identify the finding of s^* with a success event, we can map the chain $\{s_n\}$ to another chain $\{i_n\}$ consisting of success ($i_n = s$) and failure ($i_n = f$) events. Then, the chain $\{i_n\}$ can be identified with a Bernoulli trial series with success probability p^* . Because we can find s^* in the incumbent solution if at least one success event is involved in the chain $\{i_n\}$, the probability P_{suc}^o of successfully finding s^* in the incumbent solution should be given by

$$P_{suc}^o(N_{SJ}) = 1 - (1 - p^*)^{N_{SJ}} = 1 - e^{-\gamma N_{SJ}}, \quad (19)$$

where

$$\gamma = -\ln(1 - p^*). \quad (20)$$

This is exactly consistent with the experimental observations.

In Fig. 6, we depicted our intuitional picture for the exploration process in the RFSA. The jump chain $\{s_n\}$ generated

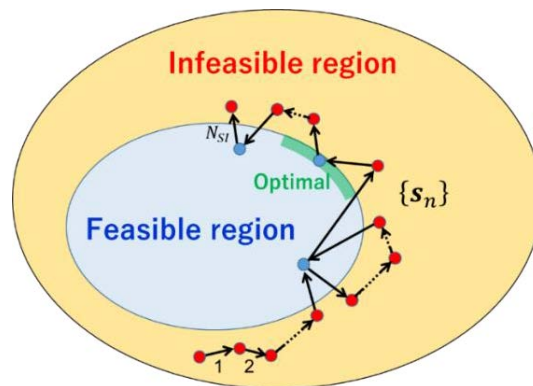


FIGURE 6. Intuitional picture of the exploring process in the RFSA. A run $\{s_n\}$ of the chain generated in the RFSA is schematically depicted. The blue and red circles indicate the feasible and infeasible solutions, respectively. The spin configurations are never trapped to the local solution and perform an incessant random walk. There are chances for a transient state to be in the feasible region as well as to arrive at the global optimal solution with a finite probability p^* . The trajectory of chain $\{s_n\}$ depends on the choice of the hyper-parameters. The best solution during the random walk is recorded as the incumbent solution.

in the MCMC iteration performs a random walk between feasible and infeasible regions and occasionally arrives at the globally optimal solution. In the RFSA, the trajectory incessantly fluctuates even though β_c or β_p is large enough. Therefore, the spins can always escape from the local minima and are never trapped in the local solution as in the standard SA. However, notably, there are subtle inconsistencies between our intuition and the experimental observations. For example,

1. As far as the feasible solution is concerned, P_{suc}^f in model (a) is apparently larger than P_{suc}^f in model (b), although it should be the same by intuition.
2. Both for $i = o$ and f , the P_{suc}^i for model (c) is apparently larger than the P_{suc}^i for models (a)

and (b), although it should be the same by intuition.

Concerning the second point, it is likely that P_{suc}^i is comparable for the three SA models because the solution search is only driven by spin inversion, and there is no difference among the three models. Unfortunately, the reason for this inconsistency is not clear. In these experiments, we carefully shared the codes for three models as far as possible to perform a fair comparison, so that differences lie only in the spin-selection algorithms shown in Fig. 4. Currently, we suspect that it may come from the quality of random numbers generated by the built-in function in Mathematica®, which uses a cellular-automata-based algorithm [44] called “ExtendedCA” [45]. This point needs further investigations.

Although there remains an unsettled problem, it is clear from Fig. 5 that if the computational bottleneck of the SA lies in the spin-selection algorithm, the RFSA has a potential advantage in avoiding the time-wasted rejected events and increasing the efficiency of the globally optimal solution search.

Now let us discuss another advantage of the RFSA. We evaluated P_{suc}^o as well as P_{suc}^f as a function of three sets of hyper-parameters (β_c, β_p) , (k_p, T_c) , and (k_c, T_p) . We evaluated P_{suc}^i on 50×50 hyper-parameter mesh points for every problem instance and visualized their landscapes on a three-dimensional viewgraph. Two typical problems were chosen for demonstrating the advantage of RFSA in comparison to the standard SA. The first problem is the QUBO problem in which both the cost and penalty functions are quadratic in spin variables. The second one is optimization of the max-cut problem (spin-glass problem in Ising spin model) embedded in the LHZ architecture where the cost function is linear but the penalty function is quartic in spin variables [32].

A. QUBO PROBLEM

We chose the traveling salesman problem (TSP) for demonstration. TSP is a typical QUBO problem; the problem is encoded using objective functions, which are a sum of the cost and the penalty functions that are the quadratic functions of N^2 binary variables. The sum of the two functions must be minimized under the condition that the penalty function is minimized. The penalty function is constructed by L2 regularization [36] which is an equally weighted sum of $2N$ errors measured by the L2-distance from the feasible region. The ground states of the penalty function are a set of states satisfying the $2N$ one-hot constraint [28], [46]. QUBO can be mapped to the isomorphic Ising problem and solved by finding the ground state of a total Hamiltonian \mathcal{H} of the form in eq.(11) [46]. \mathcal{H} describes the Hamiltonian of the Ising spins that have linear terms in spin variables associated with external fields acting on each spin and the quadratic terms associated with interactions between two spins. Because linear terms can be embedded into quadratic

terms, \mathcal{H} can be written as a sum of the quadratic terms in spin variables.

A set of the ground states of penalty function \mathcal{H}_p constitutes the feasible region. The solution is optimal if and only if it is in the feasible region and the ground state of \mathcal{H} . This condition imposes some restrictions on the area of (β_c, β_p) for successful optimization. Thus, the performance of SA depends on (β_c, β_p) . The efficiency of the solution search is sensitive to their choice. For sufficiently large values of β_p , SA will be pushed inside the feasible region very quickly and will produce feasible solutions with a greater probability. However, if β_p is too large relative to β_c , the search may be restricted to only feasible regions, and the boundary of the feasible and infeasible region on which the optimal solution might lie cannot be explored [36]; thus, a feasible solution that is far from the global optima is found. Conversely, if β_p is too small relative to β_c , then SA may search a very large region and will spend a lot of time in exploration of the infeasible regions. Then, SA cannot explore in the boundary of the feasible and infeasible regions. Eventually, the search will find only the local optima in the infeasible region. To efficiently find the globally optimal solution, we must find an efficient short cut through the infeasible region. Thus, it is reasonable that (β_c, β_p) should be tuned to assure convergence in the feasible region and still permit short-cuts leading to the optimum solution through the infeasible region [47], [48]. Thus, β_p/β_c should be kept as low as possible, just above the limit below which only infeasible solutions are obtained, which is known as the minimum penalty rule [16], [48], [49].

To confirm the minimum penalty rule, we investigated P_{suc}^i as a function of the static set of hyper-parameters (β_c, β_p) . We used the burma14 instance for illustrative demonstration. The distance matrix of the original problem is given by the following symmetric matrix (21), as shown at the bottom of the next page, where $d_{\alpha\beta} (= d_{\beta\alpha})$ shows the distance between cities α and β . To improve the calculation efficiency as much as possible, we performed a bias removal [50], [51] on d . We subtracted each row-minima from its corresponding row elements and repeated the same column-wise on the resultant matrix. Then, we replaced all the diagonal elements to zero because they are irrelevant to the calculation of spin dynamics using SA. Finally, we obtained matrix $d_{max}d^*$ that is a non-negative, asymmetric matrix with at least one zero in each row and in each column (22), as shown at the bottom of the next page, where $d_{max} = \max d_{max}d^*$, which is 753 in the present case. The asymmetric matrix d^* is called the reduced cost matrix normalized by d_{max} . The total of the row-minima and the subsequent column-minima being subtracted is called the “bias” of the matrix. We can calculate the total distance associated with the optimal traveling route from the associated value of the cost function using $d_{max}d^*$ and the bias value.

We prepared the local field strength and the coupling coefficient between spins using normalized matrix d^* according

to the prescription given in [28] and [46]. Explicit forms of $\mathcal{H}_c(s)$ and $\mathcal{H}_p(s)$ are given as follows:

$$\mathcal{H}_i(s) = -\frac{1}{2} \sum_{j=1}^N \sum_{k=1}^N \sum_{\alpha=1}^N \sum_{\beta=1}^N J_{j\alpha,k\beta}^{(i)} s_{j\alpha} s_{k\beta} - \sum_{j=1}^N \sum_{\alpha=1}^N h_{j\alpha}^{(i)} s_{j\alpha} + C^{(i)}, \quad (23)$$

where $i = c, p$ denotes the cost and penalty functions, respectively, and

$$J_{j\alpha,k\beta}^{(c)} = -\frac{1}{4} \left(d_{\alpha\beta}^* \delta_{k,j+1} + d_{\beta\alpha}^* \delta_{k,j-1} \right), \quad (24)$$

$$h_{j\alpha}^{(c)} = -\frac{1}{4} \sum_{k=1}^N \sum_{\beta=1}^N \left(d_{\alpha\beta}^* \delta_{k,j+1} + d_{\beta\alpha}^* \delta_{k,j-1} \right), \quad (25)$$

$$C^{(c)} = \frac{1}{4} N \sum_{\alpha=1}^N \sum_{\beta=1}^N d_{\alpha\beta}^* + \text{bias}, \quad (26)$$

$$J_{j\alpha,k\beta}^{(p)} = -\frac{1}{4} (1 - \delta_{j,k} \delta_{\alpha,\beta}) (\delta_{j,k} + \delta_{\alpha,\beta}), \quad (27)$$

$$h_{j\alpha}^{(p)} = -(N - 2), \quad (28)$$

$$C^{(p)} = \frac{1}{2} N (N^2 - 3N + 4), \quad (29)$$

In eqs.(23) and (24), $s_{j\alpha}$ ($j, \alpha = 1, \dots, N$) denotes the spin variable taking values $+1$ or -1 , which indicate whether city α is passed at time j . Notably, the present algorithm is, in some sense, considered a hybrid algorithm performing pre-processing to obtain d^* and SA in sequence. In the following sections, we will present and compare the experimental results obtained for standard SA and RFSA.

1) STANDARD SA

For a reference, we performed a simulation associated with the standard SA model (a). To make a fair comparison, we focused on the following points:

1. We shared as many codes between the standard and RFSA as possible. The difference lies within the spin-selection algorithms, i.e., AR method or non-uniform random generation.
2. The computational effort was chosen so that the measured wall-clock time of the standard SA and that of the RFSA were comparable.

The top diagrams in Fig. 7 show the results for the standard SA. In these figures, P_{suc}^i has been plotted as a function of

$$d = \begin{pmatrix} \infty & 153 & 510 & 706 & 966 & 581 & 455 & 70 & 160 & 372 & 157 & 567 & 342 & 398 \\ 153 & \infty & 422 & 664 & 997 & 598 & 507 & 197 & 311 & 479 & 310 & 581 & 417 & 376 \\ 510 & 422 & \infty & 289 & 744 & 390 & 437 & 491 & 645 & 880 & 618 & 374 & 455 & 211 \\ 706 & 664 & 289 & \infty & 491 & 265 & 410 & 664 & 804 & 1070 & 768 & 259 & 499 & 310 \\ 966 & 997 & 744 & 491 & \infty & 400 & 514 & 902 & 990 & 1261 & 947 & 418 & 635 & 636 \\ 581 & 598 & 390 & 265 & 400 & \infty & 168 & 522 & 634 & 910 & 593 & 19 & 284 & 239 \\ 455 & 507 & 437 & 410 & 514 & 168 & \infty & 389 & 482 & 757 & 439 & 163 & 124 & 232 \\ 70 & 197 & 491 & 664 & 902 & 522 & 389 & \infty & 154 & 406 & 133 & 508 & 273 & 355 \\ 160 & 311 & 645 & 804 & 990 & 634 & 482 & 154 & \infty & 276 & 43 & 623 & 358 & 498 \\ 372 & 479 & 880 & 1070 & 1261 & 910 & 757 & 406 & 276 & \infty & 318 & 898 & 633 & 761 \\ 157 & 310 & 618 & 768 & 947 & 593 & 439 & 133 & 43 & 318 & \infty & 582 & 315 & 464 \\ 567 & 581 & 374 & 259 & 418 & 19 & 163 & 508 & 623 & 898 & 582 & \infty & 275 & 221 \\ 342 & 417 & 455 & 499 & 635 & 284 & 124 & 273 & 358 & 633 & 315 & 275 & \infty & 247 \\ 398 & 376 & 211 & 310 & 636 & 239 & 232 & 355 & 498 & 761 & 464 & 221 & 247 & \infty \end{pmatrix}, \quad (21)$$

$$d_{max} d^* = \begin{pmatrix} 0 & 0 & 440 & 558 & 664 & 511 & 385 & 0 & 90 & 69 & 87 & 497 & 272 & 328 \\ 0 & 0 & 269 & 433 & 612 & 445 & 354 & 44 & 158 & 93 & 157 & 428 & 264 & 223 \\ 299 & 128 & 0 & 0 & 301 & 179 & 226 & 280 & 434 & 436 & 407 & 163 & 244 & 0 \\ 447 & 322 & 30 & 0 & 0 & 6 & 151 & 405 & 545 & 578 & 509 & 0 & 240 & 51 \\ 566 & 514 & 344 & 13 & 0 & 0 & 114 & 502 & 590 & 628 & 547 & 18 & 235 & 236 \\ 562 & 496 & 371 & 168 & 149 & 0 & 149 & 503 & 615 & 658 & 574 & 0 & 265 & 220 \\ 331 & 300 & 313 & 208 & 158 & 44 & 0 & 265 & 358 & 400 & 315 & 39 & 0 & 108 \\ 0 & 44 & 421 & 516 & 600 & 452 & 319 & 0 & 84 & 103 & 63 & 438 & 203 & 285 \\ 117 & 185 & 602 & 683 & 715 & 591 & 439 & 111 & 0 & 0 & 0 & 580 & 315 & 455 \\ 96 & 120 & 604 & 716 & 753 & 634 & 481 & 130 & 0 & 0 & 42 & 622 & 357 & 485 \\ 114 & 184 & 575 & 647 & 672 & 550 & 396 & 90 & 0 & 42 & 0 & 539 & 272 & 421 \\ 548 & 479 & 355 & 162 & 167 & 0 & 144 & 489 & 604 & 646 & 563 & 0 & 256 & 202 \\ 218 & 210 & 331 & 297 & 279 & 160 & 0 & 149 & 234 & 276 & 191 & 151 & 0 & 123 \\ 187 & 82 & 0 & 21 & 193 & 28 & 21 & 144 & 287 & 317 & 253 & 10 & 36 & 0 \end{pmatrix}, \quad (22)$$

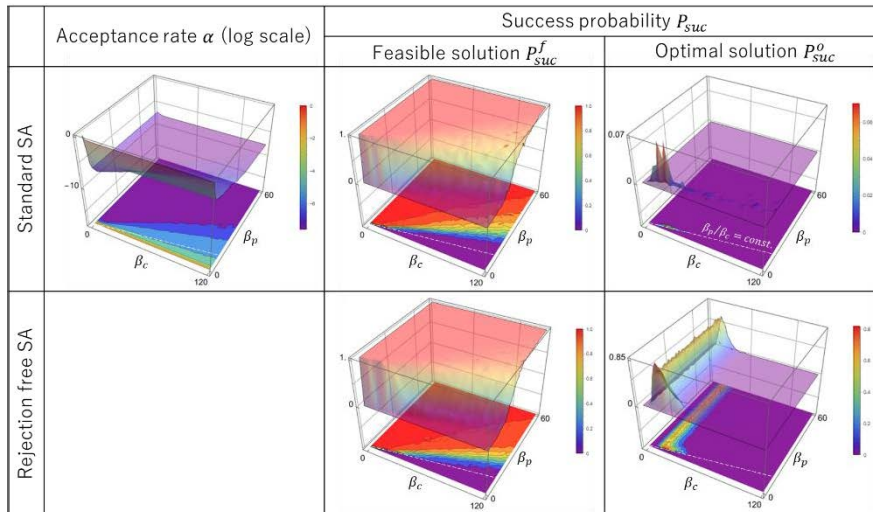


FIGURE 7. Results of simulation on QUBO problem. P^o_{suc} (right), P^f_{suc} (middle), and $\log\alpha$ (left) as a function of the set of hyper-parameters (β_c, β_p) for standard SA (top) and RFSA (bottom).

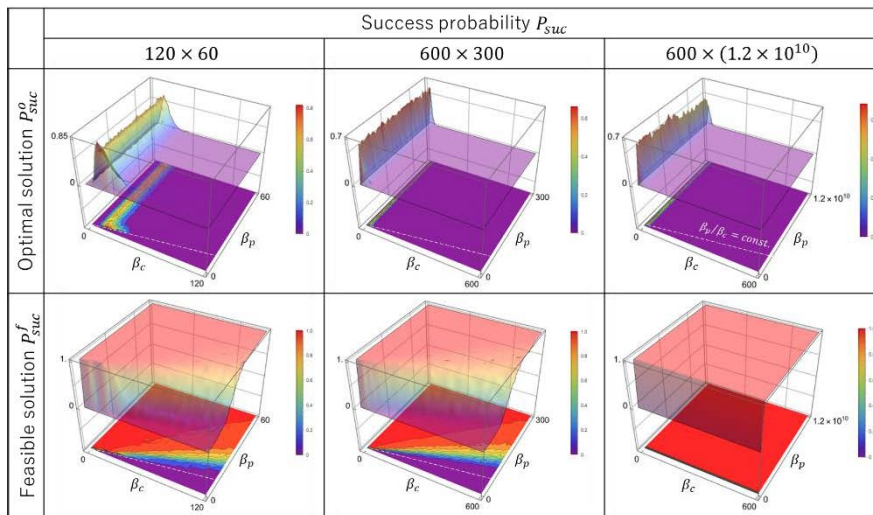


FIGURE 8. Results of simulation on QUBO problem. P^o_{suc} (top) and P^f_{suc} (bottom) as a function of hyper-parameters (β_c, β_p) for RFSA. The plot area is 120×60 , 600×300 , and $600 \times (1.2 \times 10^{10})$ from left to right.

a set of hyper-parameters (β_c, β_p) . The ground plane corresponds to a two-dimensional hyper-parameter space (β_c, β_p) , and P^i_{suc} was visualized on three-dimensional and contour plots. Landscapes of P^f_{suc} and P^o_{suc} are shown in the middle and right diagrams, respectively. In the left diagram, the evaluated acceptance rate $\alpha = N_{SI} / |\{S_n\}|$ is shown in a logarithmic scale. Hereafter, let us abbreviate the areas on the (β_c, β_p) -plane where $P^f_{suc} > 0$ and $P^f_{suc} \sim 0$ are ‘feasible’ and ‘infeasible’ areas, respectively, and the areas on the (β_c, β_p) -plane where $\alpha > 0$ and $\alpha \sim 0$ are ‘accepted’ and ‘rejected’ areas, respectively. Then, we see from Fig. 7 that $P^o_{suc} \sim 0$ if (β_c, β_p) is in the infeasible area or the rejected area. This is reasonable because the globally optimal solution must be feasible, and $\alpha = N_{SI} / |\{S_n\}| \rightarrow 0$ implies $P^o_{suc} \rightarrow 0$

as seen from Fig. 5 (a). Note that the feasible and accepted areas are mutually contradictory. Thus, we can obtain the globally optimal solution only if (β_c, β_p) is in the very small area that is feasible as well as an accepted area, which lies in the vicinity of the boundary of the feasible and infeasible areas.

2) RFSA

In turn, the bottom diagrams in Fig. 7 show the results associated with the RFSA model (c). Because it is trivial that $\alpha = 1$ in the RFSA, the bottom left diagram is omitted. We found that although the landscape of P^f_{suc} is like that of the standard SA, P^o_{suc} is considerably improved compared with that of the standard SA in wide ranges of (β_c, β_p) . This is

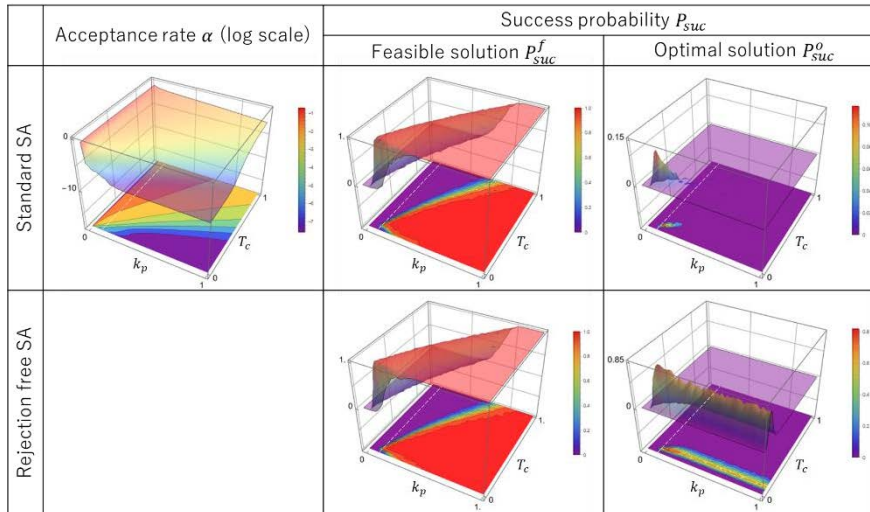


FIGURE 9. Results of simulation on QUBO problem. P_{suc}^o (right), P_{suc}^f (middle), and $\log \alpha$ (left) as a function of the set of hyper-parameters (k_p, T_c) for standard SA (top) and RFSA (bottom).

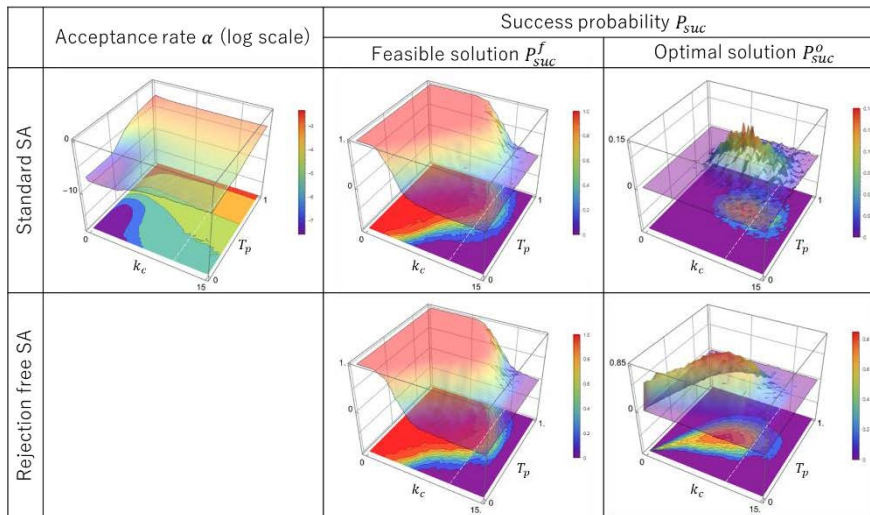


FIGURE 10. Results of simulation on QUBO problem. P_{suc}^o (right), P_{suc}^f (middle), and $\log \alpha$ (left) as a function of the set of hyper-parameters (k_c, T_p) for standard SA (top) and RFSA (bottom).

because $\alpha = 1$ in the RFSA. Therefore, the RFSA can release the potentiality for exploring the globally optimal solution hidden in the rejected area for the standard SA. We see that P_{suc}^o is largely improved in the range $b_c^{min} < \beta_c < b_c^{max}$ and $b_p^{min} < \beta_p < b_p^{max}$. Although it is limited within a small range in β_c , the upper limit b_p^{max} is remarkably enlarged. Fig. 8 indicates the extent to which it is enlarged. The top and bottom diagrams show P_{suc}^o and P_{suc}^f , respectively. The left diagrams are plotted in a 120×60 area on the (β_c, β_p) -plane, which are same as the ones in Fig. 7. The middle and right diagrams are plotted in 600×300 and $600 \times (1.2 \times 10^{10})$ areas, respectively. Remarkably, P_{suc}^o has an appreciable value even for $\beta_p \sim 10^{10}$. Note that P_{suc}^o has its maxima in the vicinity of the boundary of the feasible and infeasible areas. This is consistent with the minimum penalty rule [16], [48], [49],

although it is not necessarily clear why P_{suc}^o is improved in such a vicinity. We suppose the following scenario. When we set (β_c, β_p) in the feasible area and near the infeasible area, the probability that the state of spins stays in the infeasible region increases. Staying in the infeasible region for long may allow us to enlarge the solution search space effectively and may un-cover any hidden short-cuts leading to the globally optimum solution. This resembles the technique known as strategic oscillation in the Tabu search method [52]. Note that its purpose is consistent with the purpose of the relaxation approach including the penalty function method, where the search space is extended to include infeasible regions to change the difficult problem to an easier one.

Similar results have been obtained in the case of different choices of a set of hyper-parameters. Figures 9 and 10 show

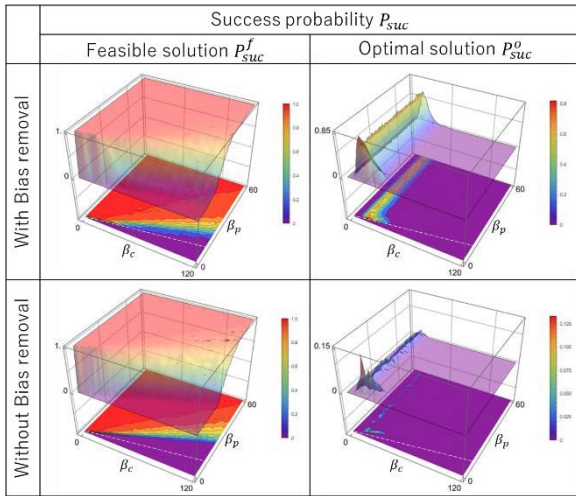


FIGURE 11. Results of simulation on QUBO problem. P_{suc}^o (right) and P_{suc}^f (left) with bias removal (top) and without bias removal (bottom) are shown.

P_{suc}^i ($i = o, f$) and $\log \alpha$ as a function of a set (k_p, T_c) as well as a set (k_c, T_p) . Their landscapes are clearly different from those obtained for the case of choice (β_c, β_p) . However, because (β_c, β_p) and (k_p, T_c) are connected by eqs. (14) and a (16), and (β_c, β_p) and (k_c, T_p) are connected by eqs. (15) and (16), these diagrams hold the same information as those in Fig. 7.

In the above simulations, we used the bias removed distance matrix d^* . Fig. 11 compares P_{suc}^i ($i = o, f$) with and without removed biases. We see that using the bias removed matrix d^* considerably increases the success probabilities. Thus, a better use of d^* would be for the problem in which d^* is known to be effective so far, such as the TSP and assignment problem [50], [51].

3) SCALING CHARACTERISTICS

We found that RFSA can reveal the landscape of P_{suc}^o that is hidden in the standard SA. Now, we focus on the scaling characteristics of P_{suc}^i and reveal their behaviors. We found novel characteristics that may be helpful for strategical tuning for hyper-parameters. To demonstrate the scaling characteristics, we used the burma14 instance as a basis. We randomly picked up N cities ($N = 8, 9, 10,$ and 14), and reconstructed new N -city TSP problems. Using the RFSA, we searched and obtained the globally optimal solution for these problems, which is the exact one for $N = 8, 9, 10$ and the best-known one for $N = 14$. Figures 12–14 show the evaluated landscape of P_{suc}^i as a function of (β_c, β_p) , (k_p, T_c) , and (k_c, T_p) , respectively. Scattering of P_{suc}^o on the hyper-parameter plane increases as N decreases. This would be reasonable because the cost function is a function of a set of $\{J_{ij}, h_j\}$, the values of which are scattered within some range. Further discussion lies outside the scope of this paper. We assume this observation to be general at least for TSPs and will discuss how this observation may be used for tuning hyper-parameters.

Notably, some other properties were scaled with N . For example, Fig. 15 indicates the peak location β_c^{peak} on the β_c -axis of marginal probability $\bar{P}_{suc}^o(\beta_c)$. We found that β_c^{peak} is approximately proportional to N . Because the number of data points is insufficient, it is unclear whether β_c^{peak} actually scales linearly in N , especially in the range $N \geq 14$. Although it is fairly certain that some experimentally observable quantities may depend on the size of the problem, further investigation is required.

4) AVOIDING CYCLING USING SHORT-TERM MEMORY MECHANISM

The RFSA avoids spins from getting stuck in local optima. We have confirmed that it significantly saves computational time and increases the efficiency for finding the globally optimal solution. Nevertheless, inefficiencies remain which may waste computational resources with no contribution to finding the solution. Our RFSA scheme keeps track of the best solution as an incumbent solution during the solution search. In this case, it is desirable for the trajectory of spin states to be as diverse as possible, that is, RFSA produces trajectories as extensively as possible. This is because we eventually find the globally optimal solution if we find at least one trajectory leading to it.

Cycling may be problematic for diversity of trajectory. Suppose that spins are in the local minima. RFSA probabilistically selects one of the spins and inverts it with certainty even if it increases the energy of spins. Then, it is most probable that the same spin is selected again for next inversion because this is the only choice that decreases the energy of the spins. Suppose that the current solution is s , and the next solution s' in the neighborhood of s is chosen. If s was the local optimum, it is likely that the next move from s' will be to go back to s . The spin state could cycle between the two states. This cycling will cost an extra computational time with no contribution to the solution search.

Cycling can be avoided by incorporating short-term memory, like Tabu lists in the Tabu search algorithm [53], [54], which records the most recent history of the spin inversion. Reverse inversion of the last spin inversion is then penalized by temporally introducing a penalty in the energy shift associated with the last spin inversion. The algorithm is schematically shown in Fig. 16 We confirmed that its calculation cost is negligible. Using this mechanism, the spin selection in the next transition depends on the spin selection in the current transition. Therefore, the next state depends not only on the current state but also on the previous state. Because this mechanism breaks the Markovian property of the chain $\{s_n\}$, it may not necessarily meet our expectations.

We experimentally confirmed the validity of this algorithm, as shown in Fig. 17. We compared the experimental results with and without avoiding cycling for the same lengths of $\{s_n\}$. The middle and left diagrams represent the landscapes of $P_{suc}^o(\beta_c, \beta_p)$ and $P_{suc}^f(\beta_c, \beta_p)$ with and without avoiding cycling, respectively. P_{suc}^o clearly increased in the wide

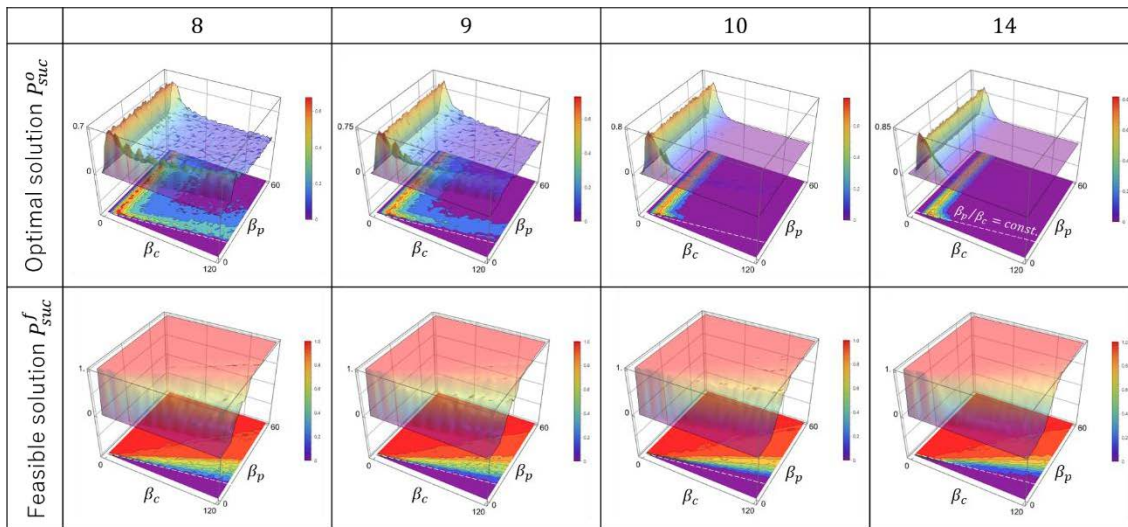


FIGURE 12. Results of simulation on QUBO problem. P^o_{suc} (top) and P^f_{suc} (bottom) as a function of the set of hyper-parameters (β_c, β_p) for RFSa. The problem size N is 8, 9, 10 and 14 from left to right.

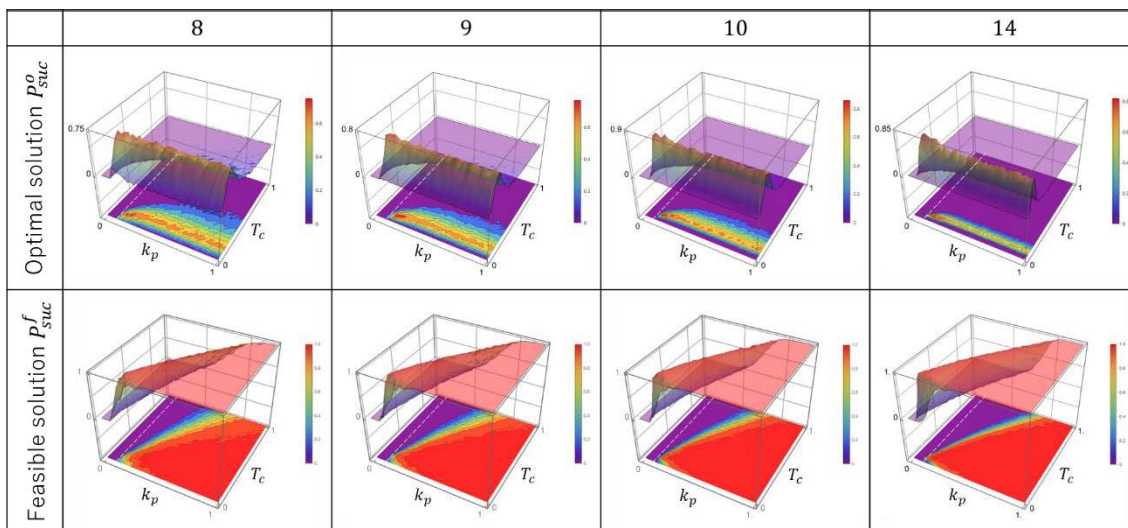


FIGURE 13. Results of simulation on QUBO problem. P^o_{suc} (top) and P^f_{suc} (bottom) as a function of the set of hyper-parameters (k_p, T_c) for RFSa. The problem size N is 8, 9, 10 and 14 from left to right.

area on the (β_c, β_p) -plane. The right diagrams indicate the improvement of P^o_{suc} and P^f_{suc} , which are the differences of the probabilities with and without avoiding cycling. We see that the present method was valid, at least, for a burma14 instance. Further investigation is required to prove the general validity of this method.

B. LHZ ARCHITECTURE

Next, let us consider the problems associated with the LHZ architecture. Since the LHZ architecture was proposed [32], significant research has been devoted to highlight its implementation and usability in the context of QA [55], [56]. In this study, we applied the RFSa for finding ground state

of the spin-glass embedded in the LHZ architecture, which is mathematically equivalent to, optimization of the maximum cut problem, and investigated its characteristics. Several peculiar characters in the landscape of P^o_{suc} as well as P^f_{suc} are presented.

The LHZ architecture is an embedding model designed for QA hardware. It comprises an extended set of physical spins arranged on a two-dimensional lattice with local field acting on each physical spin and four-way coupling among the nearest neighboring physical spins. This architecture is promising for realizing scalable QA hardware based on semiconductor manufacturing technology. The model can embed arbitrary N logical spins with connectivity between any two logical spins in an extended set of $\frac{N(N+1)}{2}$ physical spins. The dataset of

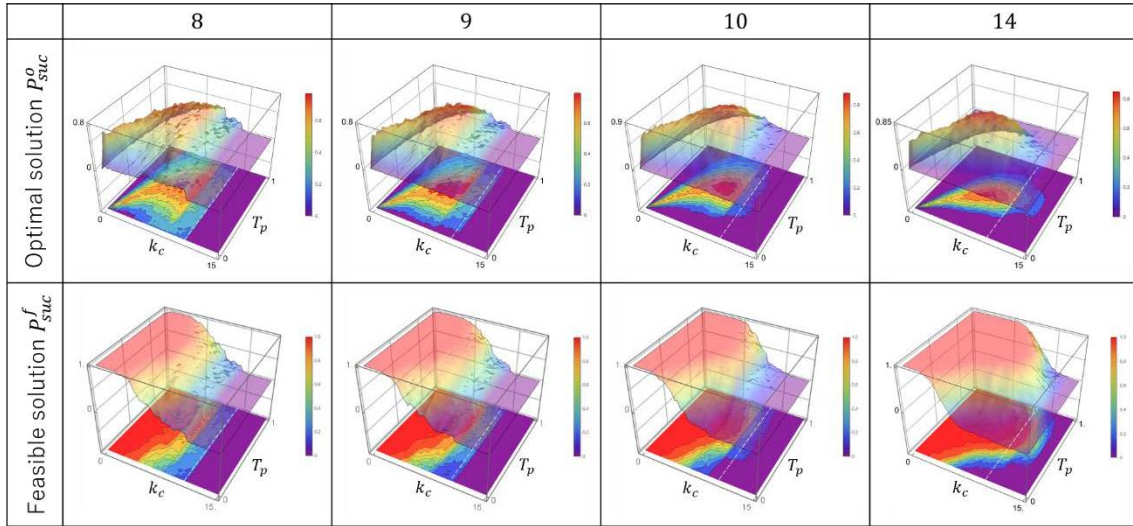


FIGURE 14. Results of simulation on QUBO problem. P^o_{suc} (top) and P^f_{suc} (bottom) as a function of the set of hyper-parameters (k_c, T_p) for RFSA. The problem size N is 8, 9, 10 and 14 from the left to right.

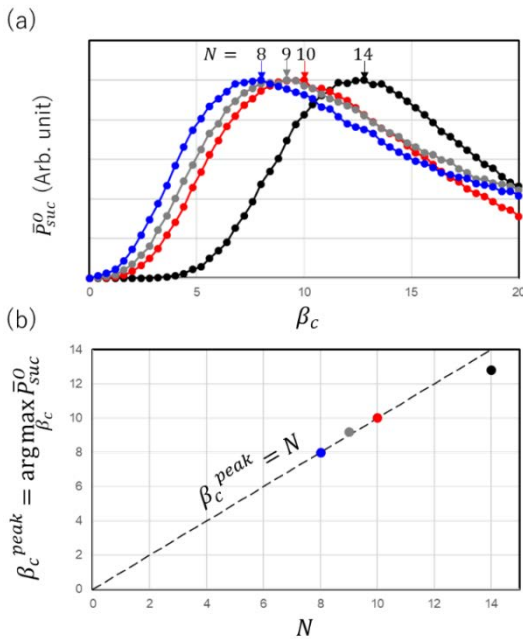


FIGURE 15. (a) Marginal probability distribution $\bar{P}^o_{suc}(\beta_c)$ obtained by $P^o_{suc}(\beta_c, \beta_p)$ averaged over $6 \leq \beta_p \leq 60$ and normalized by their peak values shown for $N = 8, 9, 10, 14$. (b) The peak location β_c^{peak} of $P^o_{suc}(\beta_c)$ is plotted as a function of N .

the optimization problems is encoded in the local fields acting on physical spins. Details regarding the LHZ architecture are shown in Appendix.

In this architecture, the Hamiltonian also consists of two components, a cost function describing the interaction of spins with local fields (eqs.(A1) and (A12)) and a penalty function describing four-way couplings among the nearest neighboring spins (eq.(A2)). The cost function is not quadratic in spin variables and the penalty function is not

based on L2 regularization as in the Ising model. Nonetheless, little attention has been given to this point and little is known about how these differences influence the P^i_{suc} . We investigated the landscape of P^i_{suc} using RFSA. As an illustrative instance, we chose a spin-glass problem in which coupling coefficients are chosen from random variables that are uniformly distributed in the range $[-1, 1]$. We prepared an $N \times N$ random matrix where $N = 8, 10, 12, 14$ and encoded it in a cost function. The ground state of the spin-glass corresponds to the globally optimal solution of the maximum cut problem. Fig. 18 shows the evaluated P^f_{suc} and P^o_{suc} as a function of a set of hyper-parameters (β_c, β_p) for the standard SA (top figure) and for the RFSA (bottom figure). We see that $P^o_{suc} \sim 0$ if (β_c, β_p) is in the infeasible area or the rejected area, which is similar to the QUBO case. Figures 19–21 show P^f_{suc} and P^o_{suc} as a function of sets (β_c, β_p) , (k_p, T_c) , and (k_c, T_p) for $N = 8, 10, 12, 14$. We see that the scattering of P^o_{suc} on the hyper-parameter plane increases as N decreases, which is also similar to the QUBO problem (see Figs. 12–14).

Despite these similarities in the experimental results of the QUBO and LHZ architecture, we found certain differences as well. For example, although we see that P^o_{suc} is almost independent of β_p in the wide range of $\beta_p < \beta_p^{max} \sim 10^{10}$ (Fig. 12) for the QUBO problem, Fig. 19 indicates that P^o_{suc} strongly depends on β_p for the LHZ architecture. In addition, if we compare the middle diagrams in Figs. 7 and 18, we can see that the improvement in P^o_{suc} using RFSA is rather limited compared to that in the QUBO problem. Most surprisingly, we see that even the landscapes of P^f_{suc} largely differ for the QUBO and the LHZ architecture. For example, Fig. 12 shows that P^f_{suc} is almost unity for any β_p satisfying $\beta_p/\beta_c > k_p^L$ in the QUBO problem where k_p^L denotes the lower limit, whereas P^f_{suc} largely depends on β_p in the LHZ architecture. Thus, the feasible area is clearly much more limited for

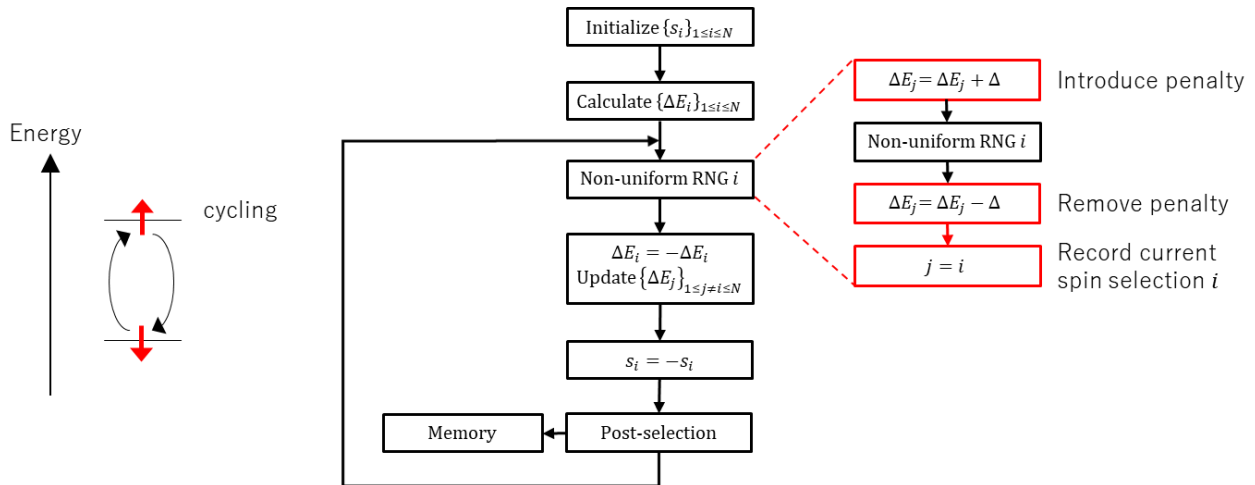


FIGURE 16. Block diagram of our spin-selection algorithm to avoid cycling using the short-term memory mechanism. Last spin selection is recorded and used to penalize it in the next spin selection.

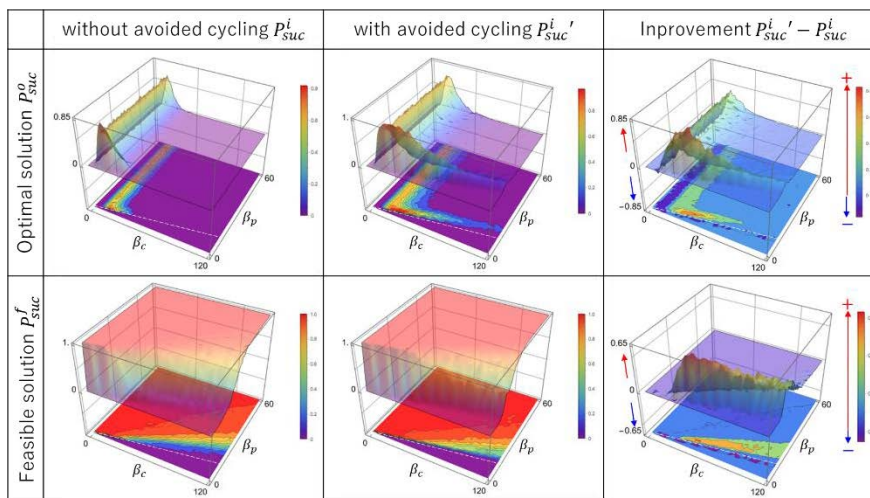


FIGURE 17. Demonstration of our proposed algorithm to avoid cycling. P_{suc}^o (top middle) and P_{suc}^f (bottom middle) on QUBO problem with avoiding cycling are shown. Left: probabilities without avoiding cycling are shown for reference. Right: improvements in success probabilities are shown.

the LHZ architecture than the QUBO problem. Because we can find globally optimal solution only if (β_c, β_p) is in the feasible area, the limited feasible area eventually limits the area where we can find the globally optimal solution.

This limited feasible area in LHZ architecture is interesting as well as critical for application. For example, let us consider the case $\beta_c = 0$. In this case, an objective function solely consists of a penalty function. We may conjecture from the experimental observation on the QUBO problem as well as the widespread consensus that we can successfully find feasible solutions as long as $\beta_p > 0$. However, this conjecture is not valid in the LHZ architecture. The feasible solution is found only around some specific β_p for the LHZ architecture as shown in Fig. 19. This surprising difference indicates that the conjecture based on experience in the QUBO problem is not necessarily valid for the LHZ architecture.

Experimental results indicate that $\beta_p = 1/T_p$ should not be very large to find the feasible solution in the LHZ architecture. In other words, we can find some optimal fixed temperature T_p [57], [58] for finding the ground state of the penalty function in the LHZ architecture, although we cannot find such an optimal temperature for finding the ground state of the penalty function associated with the one-hot constraint. This poses a question regarding the origin of this difference. Currently, the origin is not clear, but it is certainly caused by the difference in the structure of the penalty functions.

Finally, we applied the algorithm to avoid cycling and confirmed its validity even for the LHZ architecture. Fig. 22 shows the results, which is the counterpart of Fig. 17. The figure indicates that the proposed method is still valid for the LHZ architecture. Note, however, that an improvement

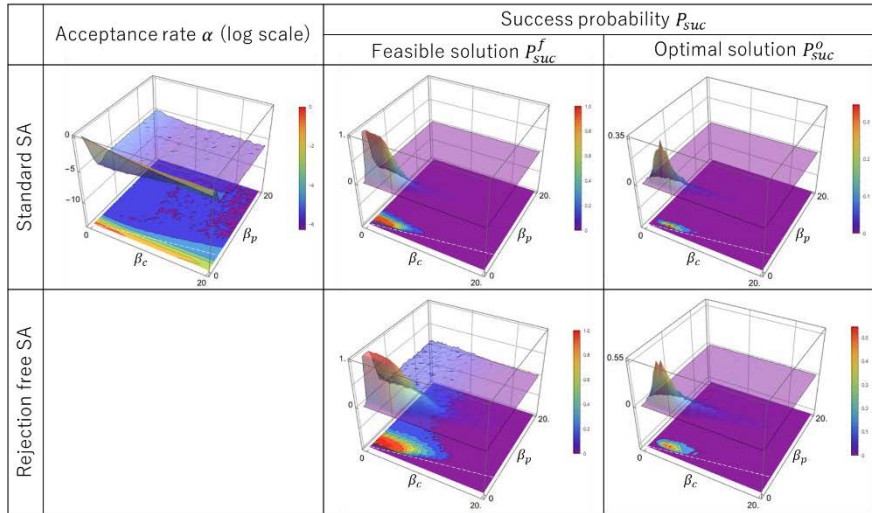


FIGURE 18. Results of simulation on LHZ architecture. P^o_{suc} (right), P^f_{suc} (middle), and $\log \alpha$ (left) as a function of the set of hyper-parameters (β_c, β_p) for standard SA (top) and RFSa (bottom).

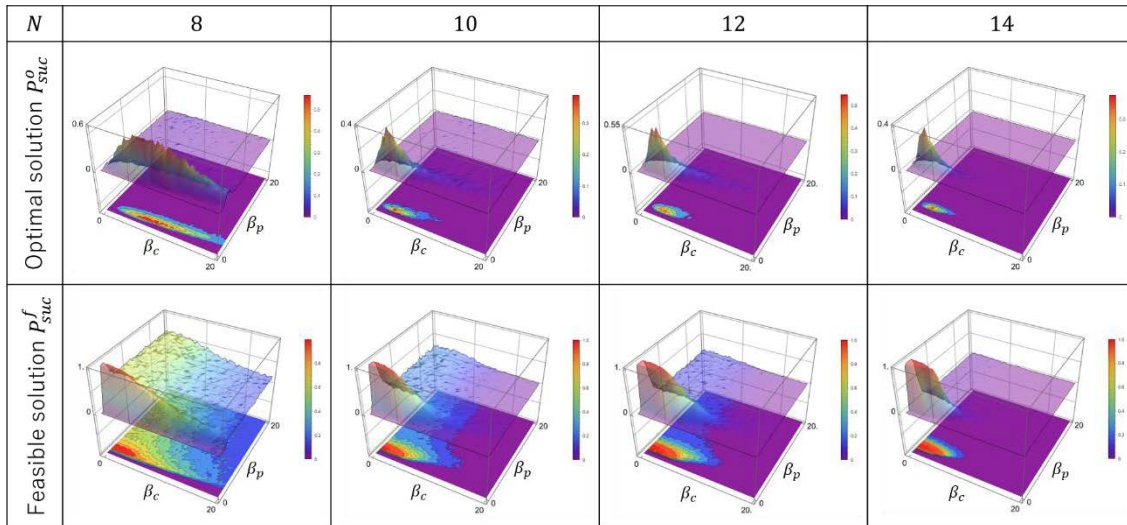


FIGURE 19. Results of simulation on LHZ architecture. P^o_{suc} (top) and P^f_{suc} (bottom) as a function of the set of hyper-parameters (β_c, β_p) for RFSa. The problem size N is 8, 10, 12, and 14 from left to right.

occurs largely on P^f_{suc} , which is contrasted with the results for the QUBO problem where improvement occurs largely on P^o_{suc} .

VI. STRATEGY TO EFFICIENTLY TUNE THE HYPER-PARAMETERS

Let us consider a reasonable strategy to efficiently tune hyper-parameters. We summarize the experimental observations that may contribute to solve this problem:

1. It is likely that we can find the globally optimal solution efficiently if we choose the hyper-parameters in the vicinity of the boundary of the feasible and infeasible areas.
2. Feasible solutions may be found more easily and efficiently than the globally optimal solution.

3. If we reconstruct a downscaled problem by randomly selecting nodes from the original problem, we can find its globally optimal solution more easily than the original problem. Furthermore, the landscape of P^i_{suc} for such a downscaled problem are scattered over a larger area with a slight shift.

We propose the following strategy for tuning hyper-parameters. Our strategy consists of the following three steps. The first step involves searching the boundary of the feasible and infeasible areas on the hyper-parameter plane. Because we have full knowledge regarding the feasible solutions a priori, we can always evaluate P^f_{suc} . Therefore, this task may be fulfilled easily by evaluating P^f_{suc} as a function of the set of the hyper-parameters considered. If we successfully find the boundary, we can proceed to the next step. In the second

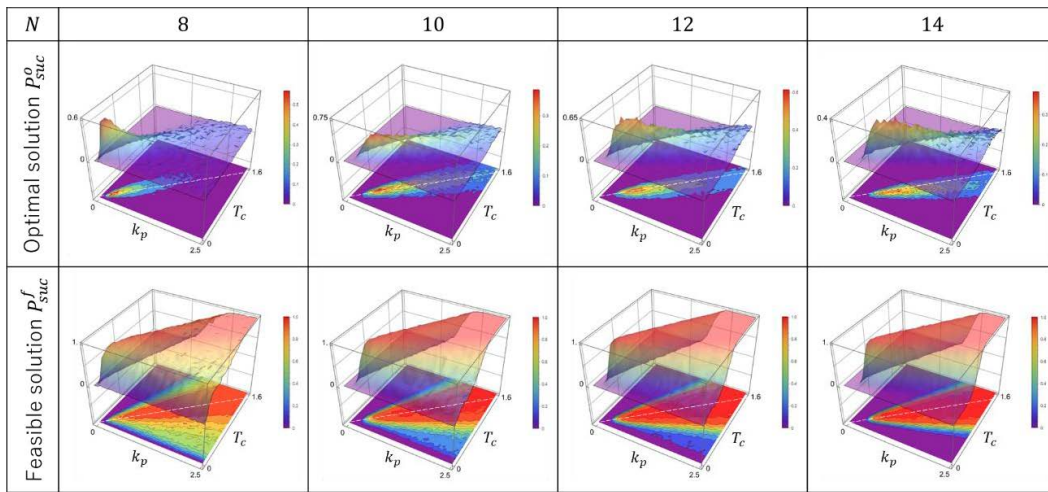


FIGURE 20. Results of simulation on LHZ architecture. P^o_{suc} (top) and P^f_{suc} (bottom) as a function of the set of hyper-parameters (k_p, T_c) for RFSA. The problem size N is 8, 10, 12, and 14 from left to right.

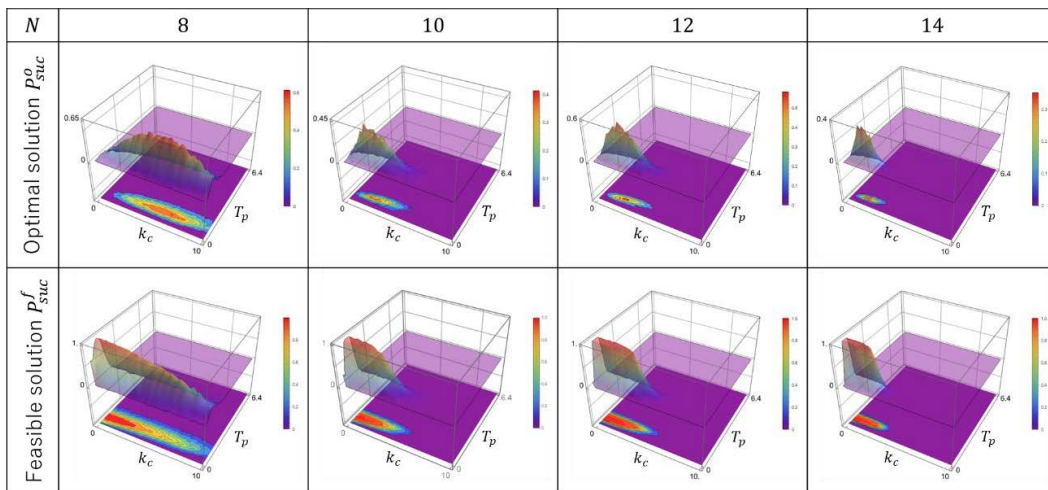


FIGURE 21. Results of simulation on LHZ architecture. P^o_{suc} (top) and P^f_{suc} (bottom) as a function of the set of hyper-parameters (k_c, T_p) for RFSA. The problem size N is 8, 10, 12, and 14 from left to right.

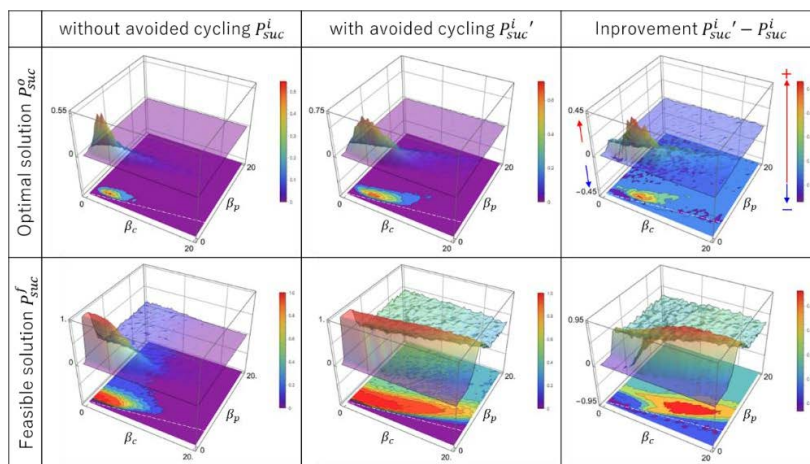


FIGURE 22. Validity of our proposal for avoiding cycling on the LHZ architecture. P^o_{suc} (top middle) and P^f_{suc} (bottom middle) on LHZ architecture with avoiding cycling are shown. Left: results without avoiding cycling are shown for reference. Right: improvements in success probabilities are shown.

step, we construct an artificially downscaled problem and evaluate the total energy of spins of the incumbent solution

in the vicinity of the observed boundary as well as within the feasible region. Then, we choose the hyper-parameter setting

that minimizes the evaluated total energy. In the final step, we apply the setting obtained in the second step to the original problem and fine-tune it to obtain better results. We expect that this procedure can be integrated with the RFSA code and can be automated.

VII. CONCLUSION

In this study, we implemented RFSA and demonstrated its validity and advantage over the standard SA for combinatorial optimization. We applied it to the QUBO problem and the spin-glass problem embedded in the LHZ architecture. The efficiency of finding the globally optimal solution was highly improved, especially for the QUBO problem. We investigated the success probabilities of finding the globally optimal solution and feasible solutions as a function of a set of hyper-parameters that specify the weights of the cost and the penalty functions. The RFSA can reveal structures in the landscapes of the success probabilities, which are invisible in the standard SA. We confirmed that the globally optimal solution can be most efficiently found in the vicinity of the boundary of the feasible and infeasible areas on the hyper-parameter plane. We compared the landscapes of success probabilities for the QUBO problem and the LHZ architecture, which highlighted the different characteristics of the penalty functions in these problems, i.e., L2 regularization and four-way couplings for spin variables. We also proposed and demonstrated a novel algorithm that incorporates short-term memory in the spin-selection algorithm to avoid inefficiency due to cycling. Our experimental results show that our proposed method successfully improved the efficiency of finding the globally optimal solution both for the QUBO problem and the LHZ architecture. Finally, we proposed a reasonable strategy to tune the hyper-parameter settings using RFSA.

Our experimental study shows that the landscapes of the success probabilities depend on the objective function, which leaves room for further investigation. The penalty function and the associated feasible solutions can be theoretically specified if we specify a class of the problem. For example, if we choose TSP, the penalty function is associated with one-hot constraint, which can be theoretically specified if the problem size N is given. In contrast, a dataset specifying the individual problem such as a distance matrix d is encoded only in the cost function. If we note that the globally optimal solution can be found only within a feasible area on the hyper-parameter plane, it might be interesting and important from an application point of view to investigate and compare the landscape of success probability of finding ground state for various classes of penalty functions, or in other words, for various classes of problems.

Finally, we note the following two remarks obtained from this study. Nature-inspired computational models offer us very instructive suggestions to use simulations of dynamic processes of physical systems for solving some mathematical problems. However, we must note that our objective is not physical simulation but obtaining some computational results. To obtain better computational results, we can

incorporate any artificial intelligence in the simulation. For example, we can freely incorporate intelligent agents like Maxwell’s daemon that can refer to the states of the spins and use this information to obtain better computational results in the simulation. If the benefit of incorporating such an agent dominates over the computational cost and overhead induced by it, we may use it to obtain better computational results. In our RFSA algorithm, post-selection and keeping track of the best solution as well as non-uniform random number generation depending on the spin states correspond to such intelligent agents.

APPENDIX

COST AND PENALTY FUNCTIONS FOR THE LHZ ARCHITECTURE

This appendix presents a mathematical formulation of the LHZ architecture that may be convenient for computer simulation. We explicitly show the updating rule for energy shift after a spin inversion. Fig. 23 schematically shows the configuration of the LHZ architecture. In this figure, blue and red circles indicate the spin and plaquette introduced in [32], respectively. If we introduce indices as shown in Fig. 23 and denote the associated spin and plaquette variables as s_{jk} and S_{jk} , respectively, where $0 \leq j < k \leq N$, the Hamiltonians for the cost and penalty functions are written as follows:

$$\mathcal{H}_c \left(\{s_{jk}\}_{0 \leq j < k \leq N} \right) = - \sum_{j=0}^{N-1} \sum_{k>j}^N J_{jk} s_{jk}, \tag{A1}$$

$$\mathcal{H}_p \left(\{S_{jk}\}_{0 \leq j < k \leq N} \right) = - \sum_{j=0}^{N-2} \sum_{k>j}^{N-1} S_{jk}, \tag{A2}$$

where, we have embedded the external magnetic field h_i into J_{0i} , and introduced $(N + 3) \times (N + 3)$ matrices \hat{J} , \hat{s} , and \hat{S} as

$$\hat{J}_{jk} = \begin{cases} J_{j-2k-2} & 2 \leq j < k \leq N + 2 \\ 0 & \text{otherwise,} \end{cases} \tag{A3}$$

$$\hat{s}_{jk} = \begin{cases} s_{j-2k-2} & 2 \leq j < k \leq N + 2 \\ 1 & 1 \leq j = k \leq N + 3 \\ 0 & \text{otherwise,} \end{cases} \tag{A4}$$

$$\hat{S}_{jk} = \begin{cases} S_{j-2k-2} & 2 \leq j < k \leq N + 1 \\ 0 & \text{otherwise,} \end{cases} \tag{A5}$$

By introducing dummy matrix elements with fixed values 0 or 1 for \hat{J}_{jk} , \hat{s}_{jk} and \hat{S}_{jk} , we can represent \hat{S}_{jk} simply as follows:

$$\hat{S}_{jk} = \hat{s}_{jk} \hat{s}_{jk+1} \hat{s}_{j+1k} \hat{s}_{j+1k+1} \quad (1 \leq j < k \leq N + 3). \tag{A6}$$

See Fig. 24 for reference. Using \hat{J} , \hat{s} and \hat{S} , we can represent \mathcal{H}_c and \mathcal{H}_p as

$$\mathcal{H}_c \left(\{\hat{s}_{jk}\}_{1 \leq j < k \leq N+3} \right) = - \langle \hat{J}, \hat{s} \rangle_F = - \sum_{j=1}^{N+3} \sum_{k=1}^{N+3} \hat{J}_{jk} \hat{s}_{jk} \tag{A7}$$

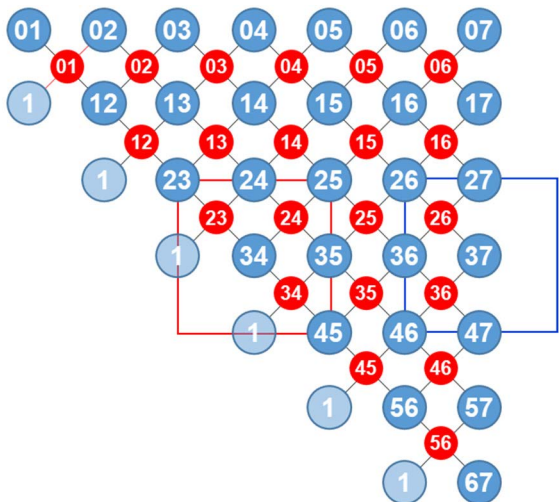


FIGURE 23. An illustration of the configuration of the LHZ architecture (for $N = 7$). Blue and red circles show variable s_{jk} associated with spin and variable, and S_{jk} associated with plaquette given in [32], respectively. Note that s_{jj} is fixed to unity for $1 \leq j \leq N - 1$.

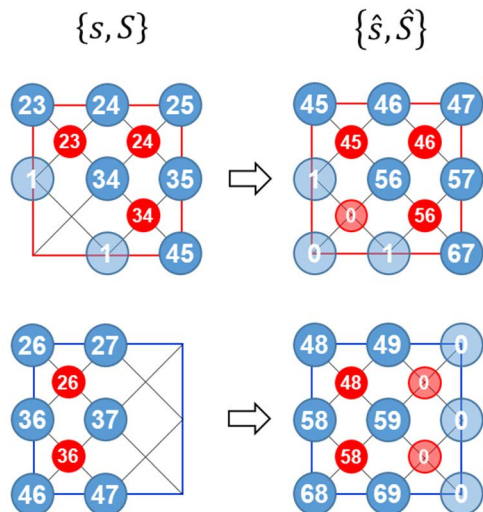


FIGURE 24. A function of dummy matrix elements given in eqs.(A3)-(A5). A set of $\frac{1}{2}N(N + 1)$ logical spins and $\frac{1}{2}N(N - 1)$ plaquette variables $\{s, S\}$ are embedded into a set of $(N + 3) \times (N + 3)$ matrices $\{\hat{s}, \hat{S}\}$ in which dummy elements are embedded. Dummy elements are assigned fixed values of 0 or 1, depending on the matrix and their indices. As a result, H_p given by eq.(A2) is consistent with that given by eqs. (A8) and (A6).

$$\mathcal{H}_p \left(\left\{ \hat{S}_{jk} \right\}_{1 \leq j < k \leq N+3} \right) = -\langle \hat{1}, \hat{S} \rangle_F = -\sum_{j=1}^{N+3} \sum_{k=1}^{N+3} \hat{S}_{jk}, \quad (\text{A8})$$

where $\langle A, B \rangle_F$ indicates the Frobenius inner product of two matrices A and B [59]. From the above formulation, the initial values of the energy shifts of cost and penalty functions associated with an inversion $\hat{s}_{jk} \rightarrow -\hat{s}_{jk}$ are

$$(\Delta E_c)_{jk} = 2\hat{J}_{jk}\hat{s}_{jk}, \quad (\text{A9})$$

$$(\Delta E_p)_{jk} = 2 \left(\hat{S}_{j-1k-1} + \hat{S}_{j-1k} + \hat{S}_{jk-1} + \hat{S}_{jk} \right), \quad (\text{A10})$$

respectively. If the total Hamiltonian is written as the form in eq. (11), the energy shift is

$$\Delta E_{jk} = \beta_c (\Delta E_c)_{jk} + \beta_p (\Delta E_p)_{jk}. \quad (\text{A11})$$

Next, let us consider updating rules for \hat{s} , \hat{S} , ΔE_c , and ΔE_p . Suppose that a spin $l'm'$ ($0 \leq l' < m' \leq N$) is chosen for next inversion, the sign of \hat{s}_{lm} is to be inverted, where $l = l' + 2$ and $m = m' + 2$. Note that only eight neighboring spins and only four neighboring plaquettes are directly connected to \hat{s}_{lm} and are affected by spin inversion. See Fig. 25. Then, we can update ΔE_c and ΔE_p as follows:

$$\begin{aligned} (\Delta E_c)_{lm} &\rightarrow -(\Delta E_c)_{lm}, \\ (\Delta E_p)_{lm} &\rightarrow -(\Delta E_p)_{lm}, \\ (\Delta E_p)_{l-1m-1} &\rightarrow (\Delta E_p)_{l-1m-1} - 4\hat{S}_{l-1m-1}, \\ (\Delta E_p)_{l+1m-1} &\rightarrow (\Delta E_p)_{l+1m-1} - 4\hat{S}_{l-1m-1} \\ (\Delta E_p)_{l-1m+1} &\rightarrow (\Delta E_p)_{l-1m+1} - 4\hat{S}_{l-1m}, \\ (\Delta E_p)_{l+1m+1} &\rightarrow (\Delta E_p)_{l+1m+1} - 4\hat{S}_{l-1m}, \\ (\Delta E_p)_{lm-1} &\rightarrow (\Delta E_p)_{lm-1} - 4 \left(\hat{S}_{l-1m-1} + \hat{S}_{lm-1} \right), \end{aligned}$$

$$\begin{aligned} (\Delta E_p)_{l-1m} &\rightarrow (\Delta E_p)_{l-1m} - 4 \left(\hat{S}_{l-1m-1} + \hat{S}_{l-1m} \right), \\ (\Delta E_p)_{lm+1} &\rightarrow (\Delta E_p)_{lm+1} - 4 \left(\hat{S}_{l-1m} + \hat{S}_{lm} \right), \\ (\Delta E_p)_{l+1m} &\rightarrow (\Delta E_p)_{l+1m} - 4 \left(\hat{S}_{lm} + \hat{S}_{lm-1} \right). \end{aligned} \quad (\text{A12})$$

Here, note that the variables \hat{s} , \hat{S} , ΔE_c , and ΔE_p involved in eq.(A12) are those before spin inversion. After calculation of eq.(A12), the signs of \hat{s} and \hat{S} must be inverted according to

$$\begin{aligned} \hat{s}_{lm} &\rightarrow -\hat{s}_{lm} \\ \hat{S}_{lm} &\rightarrow -\hat{S}_{lm} \\ \hat{S}_{l-1m} &\rightarrow -\hat{S}_{l-1m} \\ \hat{S}_{lm-1} &\rightarrow -\hat{S}_{lm-1} \\ \hat{S}_{l-1m-1} &\rightarrow -\hat{S}_{l-1m-1} \end{aligned} \quad (\text{A13})$$

Note that dummy elements should be used for calculation in eqs.(A6), (A10), and (A12) if they are involved. Concerning the energy shifts ΔE_c and ΔE_p , we must keep track of only $\frac{1}{2}N(N + 1)$ elements associated with the product of two logical spin variables. Evidently, the present formulation is redundant and introduces a noncontiguous data structure. This might influence the efficiency of calculation because it makes memory access slower. This is the inevitable price to pay in the present formulation. To solve this inefficiency due to memory access, we must omit dummy elements in ΔE_c and ΔE_p . This avoids the noncontiguous data structure and may speed up the calculation of eq.(7). The price to pay is that we must implement some algorithm to pick up the correct pair of elements appearing in ΔE_p and \hat{S} in eq.(A12) to calculate them. This may introduce some overhead in the

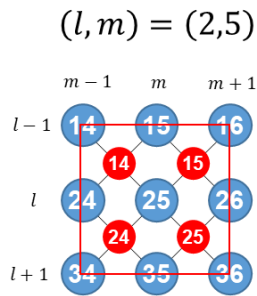


FIGURE 25. Only values of eight neighboring spins (blue circles) and four neighboring plaquettes (red circles) are influenced when the sign of \hat{s}_{lm} is inverted (the case $(l, m) = (2, 5)$ is shown).

calculation. The optimal strategy for efficiently calculating the energy shift ΔE for the LHZ architecture is currently an open question.

ACKNOWLEDGMENT

The author acknowledges Dr. Yuki Kobayashi and Dr. Masayuki Shirane at NEC Corporation for their helpful discussions and comments.

REFERENCES

- [1] A. Sbihi and R. W. Eglese, "Combinatorial optimization and green logistics," *Ann. Oper. Res.*, vol. 5, pp. 99–116, Jun. 2007, doi: [10.1007/s10288-007-0047-3](https://doi.org/10.1007/s10288-007-0047-3).
- [2] M. Eskandarpour, P. Dejax, J. Miemczyk, and O. Péton, "Sustainable supply chain network design: An optimization-oriented review," *Omega*, vol. 54, pp. 11–32, Jul. 2015, doi: [10.1016/j.omega.2015.01.006](https://doi.org/10.1016/j.omega.2015.01.006).
- [3] A. Hobé, D. Vogler, M. P. Seybold, A. Ebigbo, R. R. Settgaast, and M. O. Saar, "Estimating fluid flow rates through fracture networks using combinatorial optimization," *Adv. Water Resour.*, vol. 122, pp. 85–97, Dec. 2018, doi: [10.1016/j.advwatres.2018.10.002](https://doi.org/10.1016/j.advwatres.2018.10.002).
- [4] W. Wu and O. Daescu, Eds., *Combinatorial Optimization and Applications*. Kailua-Kona, HI, USA: Springer, 2010.
- [5] V. T. Paschos, Ed., *Applications of Combinatorial Optimization*, 2nd ed. Hoboken, NJ, USA: Wiley, 2014.
- [6] J. Pearl, *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Reading, MA, USA: Addison-Wesley, 1984.
- [7] V. Maniezzo, M. A. Boschetti, and T. Stützle, *Matheuristics, Algorithms and Implementations*. Cham, Switzerland: Springer, 2021.
- [8] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983, doi: [10.1126/science.220.4598.671](https://doi.org/10.1126/science.220.4598.671).
- [9] S. G. Brush, "History of the Lenz–Ising model," *Rev. Mod. Phys.*, vol. 39, no. 4, pp. 883–893, Oct. 1967, doi: [10.1103/RevModPhys.39.883](https://doi.org/10.1103/RevModPhys.39.883).
- [10] L. Leemis and B. Schmeiser, "Random variate generation for Monte Carlo experiments," *IEEE Trans. Rel.*, vol. R-34, no. 1, pp. 81–85, Apr. 1985, doi: [10.1109/TR.1985.5221941](https://doi.org/10.1109/TR.1985.5221941).
- [11] V. S. Denchev, S. Boixo, S. V. Isakov, N. Ding, R. Babbush, V. Smelyanskiy, J. Martinis, and H. Neven, "What is the computational value of finite-range tunneling?" *Phys. Rev. X*, vol. 6, no. 3, Aug. 2016, Art. no. 031015, doi: [10.1103/PhysRevX.6.031015](https://doi.org/10.1103/PhysRevX.6.031015).
- [12] J. Dall and P. Sibani, "Faster Monte Carlo simulations at low temperatures. The waiting time method," *Comput. Phys. Commun.*, vol. 141, no. 2, pp. 260–267, Nov. 2001.
- [13] H. Watanabe, S. Yukawa, M. A. Novotny, and N. Ito, "Efficiency of rejection-free dynamic Monte Carlo methods for homogeneous spin models, hard disk systems, and hard sphere systems," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 74, no. 2, Aug. 2006, Art. no. 026707, doi: [10.1103/PhysRevE.74.026707](https://doi.org/10.1103/PhysRevE.74.026707).
- [14] D. Sherrington and S. Kirkpatrick, "Solvable model of a spin-glass," *Phys. Rev. Lett.*, vol. 35, p. 1792, Dec. 1975, doi: [10.1103/PhysRevLett.35.1792](https://doi.org/10.1103/PhysRevLett.35.1792).
- [15] D. Sherrington and S. Kirkpatrick, "Infinite-ranged models of spin-glasses," *Phys. Rev. B, Condens. Matter*, vol. 17, p. 4384, Jun. 1978, doi: [10.1103/PhysRevB.17.4384](https://doi.org/10.1103/PhysRevB.17.4384).
- [16] L. Davis, *Genetic Algorithms and Simulated Annealing*. New York, NY, USA: Pitman, 1987.
- [17] A. J. Walker, "An efficient method for generating discrete random variables with general distributions," *ACM Trans. Math. Softw.*, vol. 3, no. 3, pp. 253–256, Sep. 1977, doi: [10.1145/355744.355749](https://doi.org/10.1145/355744.355749).
- [18] H.-P. Lehmann, L. Hübschle-Schneider, and P. Sanders, "Weighted random sampling on GPUs," 2021, *arXiv:2106.12270*.
- [19] C. K. Wong and M. C. Easton, "An efficient method for weighted sampling without replacement," *SIAM J. Comput.*, vol. 9, no. 1, pp. 111–113, Feb. 1980, doi: [10.1137/0209009](https://doi.org/10.1137/0209009).
- [20] L. Hübschle-Schneider and P. Sanders, "Parallel weighted random sampling," 2019, *arXiv:1903.00227*.
- [21] A. B. Bortz, M. H. Kalos, and J. L. Lebowitz, "A new algorithm for Monte Carlo simulation of Ising spin systems," *J. Comput. Phys.*, vol. 17, pp. 10–18, Jan. 1975, doi: [10.1016/0021-9991\(75\)90060-1](https://doi.org/10.1016/0021-9991(75)90060-1).
- [22] H. Zhu, N. Sun, M. Sasaki, K. Eguchi, T. Tabata, and F. Ren, "A Boltzmann machine with non-rejective move," *IEICE Trans. Fundam.*, vol. E85-A, pp. 1229–1235, Jun. 2002.
- [23] J. S. Rosenthal, A. Dote, K. Dabiri, H. Tamura, S. Chen, and A. Sheikholeslami, "Jump Markov chains and rejection-free metropolis algorithms," *Comput. Statist.*, vol. 36, pp. 2789–2811, Mar. 2021, doi: [10.1007/s00180-021-01095-2](https://doi.org/10.1007/s00180-021-01095-2).
- [24] J. W. Greene and K. J. Supowitz, "Simulated annealing without rejected moves," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. CAD-5, no. 1, pp. 221–228, Jan. 1986, doi: [10.1109/TCAD.1986.1270190](https://doi.org/10.1109/TCAD.1986.1270190).
- [25] H. Zhu, K. Eguchi, N. Sub, and T. Tabata, "A parallel algorithm of Boltzmann machine with rejectionless method," *WSEAS Trans. Power Syst.*, vol. 4, pp. 320–324, Apr. 2005.
- [26] A. Dote, J. S. Rosenthal, K. Dabiri, H. Tamura, S. Chen, and A. Sheikholeslami, "Rejection-free MCMC for QUBO optimization and Boltzmann sampling," Presented at the Adiabatic Quantum Comput. Conf. (AQC), Tokyo, Japan, Jun. 2021. [Online]. Available: https://aqc2021.org/poster2/poster_D5.html
- [27] A. Lucas, "Ising formulations of many NP problems," *Frontiers Phys.*, vol. 2, p. 5, Feb. 2014, doi: [10.3389/fphy.2014.00005](https://doi.org/10.3389/fphy.2014.00005).
- [28] K. Tanahashi, S. Takayanagi, T. Motohashi, and S. Tanaka, "Application of Ising machines and a software development for Ising machines," *J. Phys. Soc. Jpn.*, vol. 88, no. 6, Jun. 2019, Art. no. 061010, doi: [10.7566/JPSJ.88.061010](https://doi.org/10.7566/JPSJ.88.061010).
- [29] G. Kochenberger, J.-K. Hao, F. Glover, M. Lewis, Z. Lü, H. Wang, and Y. Wang, "The unconstrained binary quadratic programming problem: A survey," *J. Combinat. Optim.*, vol. 28, no. 1, pp. 58–81, Jul. 2014, doi: [10.1007/s10878-014-9734-0](https://doi.org/10.1007/s10878-014-9734-0).
- [30] F. Glover, G. Kochenberger, and Y. Du, "A tutorial on formulating and using QUBO models," 2018, *arXiv:1811.11538*.
- [31] E. Boros and P. L. Hammer, "The max-cut problem and quadratic 0–1 optimization; polyhedral aspects, relaxations and bounds," *Ann. Oper. Res.*, vol. 33, no. 3, pp. 151–180, Mar. 1991, doi: [10.1007/BF02115753](https://doi.org/10.1007/BF02115753).
- [32] W. Lechner, P. Hauke, and P. Zoller, "A quantum annealing architecture with all-to-all connectivity from local interactions," *Sci. Adv.*, vol. 1, no. 9, Oct. 2015, Art. no. e1500838, doi: [10.1126/sciadv.1500838](https://doi.org/10.1126/sciadv.1500838).
- [33] H. Goto, K. Tatumura, 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, doi: [10.1126/sciadv.aav2372](https://doi.org/10.1126/sciadv.aav2372).
- [34] M. H. Alrefaei and S. Andradóttir, "A simulated annealing algorithm with constant temperature for discrete stochastic optimization," *Manag. Sci.*, vol. 45, no. 5, pp. 748–764, 1999, doi: [10.1287/mnsc.45.5.748](https://doi.org/10.1287/mnsc.45.5.748).
- [35] P. S. Efraimidis and P. G. Spirakis, "Weighted random sampling with a reservoir," *Inf. Process. Lett.*, vol. 97, no. 5, pp. 181–185, Mar. 2006, doi: [10.1016/j.ipl.2005.11.003](https://doi.org/10.1016/j.ipl.2005.11.003).
- [36] A. E. Smith and D. W. Coit, "Constraint-handling techniques—Penalty functions," in *Handbook of Evolutionary Computation*, T. Baeck, D. Fogel and Z. Michalewicz, Eds. London, U.K.: Oxford Univ. Press, 1997.
- [37] H. Sanvicente-Sánchez and J. Frausto-Solís, "MPSA: A methodology to parallelize simulated annealing and its application to the traveling salesman problem," in *MICAI 2002: Advances in Artificial Intelligence* (Lecture Notes in Computer Science), C. A. C. Coello, A. de Albornoz, L. E. Sucar and O. C. Battistutti, Eds. Berlin, Germany: Springer, 2002, pp. 89–97.

- [38] E. Aarts and J. Korst, “A stochastic approach to combinatorial optimization and neural computing,” in *Simulated Annealing and Boltzmann Machines*. Hoboken, NJ, USA: Wiley, 1989, p. 272.
- [39] R. Azencott, “Parallelization techniques,” in *Simulated Annealing*. Hoboken, NJ, USA: Wiley, 1992, p. 200.
- [40] H. Sanvicente-Sánchez and J. Frausto-Solís, “A methodology to parallel the temperature cycle in simulated annealing,” in *MICAI 2000: Advances in Artificial Intelligence (Lecture Notes in Computer Science)*, vol. 1793, O. Cairó, L. E. Sucar, and F. J. Cantu, Eds. Berlin, Germany: Springer, 2000, pp. 63–74, doi: [10.1007/10720076_6](https://doi.org/10.1007/10720076_6).
- [41] K. Krishna, K. Ganeshan, and D. J. Ram, “Distributed simulated annealing algorithms for job shop scheduling,” *IEEE Trans. Syst., Man, Cybern.*, vol. 25, no. 7, pp. 1102–1109, Jul. 1995, doi: [10.1109/21.391290](https://doi.org/10.1109/21.391290).
- [42] Z. Lou. (2012). *The 100-Processor Barrier in Parallel Simulated Annealing Algorithms*. [Online]. Available: https://newtraell.cs.uchicago.edu/files/ms_paper/zhlou.pdf
- [43] G. Reinelt, “TSPLIB—A traveling salesman problem library,” *ORSA J. Comput.*, vol. 3, no. 4, pp. 376–384, Nov. 1991, doi: [10.1287/ijoc.3.4.376](https://doi.org/10.1287/ijoc.3.4.376).
- [44] S. Wolfram, “Random sequence generation by cellular automata,” *Adv. Appl. Math.*, vol. 7, pp. 123–169, Jun. 1986, doi: [10.1016/0196-8858\(86\)90028-X](https://doi.org/10.1016/0196-8858(86)90028-X).
- [45] H. Ruskeepää, “Probability,” in *Mathematica Navigator*, 3rd ed. New York, NY, USA: Academic, 2009, pp. 961–1002.
- [46] K. Takehara, D. Oku, Y. Matsuda, S. Tanaka, and N. Togawa, “A multiple coefficients trial method to solve combinatorial optimization problems for simulated-annealing-based Ising machines,” in *Proc. IEEE 9th Int. Conf. Consum. Electron. (ICCE-Berlin)*, Sep. 2019, pp. 64–69.
- [47] R. L. Riche, C. Knopf-Lenoir, and R. Haftka, “A segregated genetic algorithm for constrained structural optimization,” in *Proc. 6th Int. Conf. Genet. Algorithms*, 1995, pp. 558–565.
- [48] I. S. Fathi, R. M. Abo-Bakr, and R. M. Farouk, “Some penalty-based constraint handling techniques with ant lion optimizer for solving constrained optimization problems,” *Int. J. Comput. Appl.*, vol. 181, no. 30, pp. 24–36, Nov. 2018, doi: [10.5120/ijca2018917412](https://doi.org/10.5120/ijca2018917412).
- [49] X. Li and G. Zhang, “Minimum penalty for constrained evolutionary optimization,” *Comput. Optim. Appl.*, vol. 60, no. 2, pp. 513–544, Mar. 2015, doi: [10.1007/s10589-014-9676-6](https://doi.org/10.1007/s10589-014-9676-6).
- [50] Z. H. Ahmed, “A data-guided lexisearch algorithm for the asymmetric traveling salesman problem,” *Math. Problems Eng.*, vol. 2011, Jan. 2011, Art. no. 750968, doi: [10.1155/2011/750968](https://doi.org/10.1155/2011/750968).
- [51] J. D. C. Little, K. G. Murty, D. W. Sweeney, and C. Karel, “An algorithm for the traveling salesman problem,” *Oper. Res.*, vol. 11, no. 6, pp. 972–989, Dec. 1963, doi: [10.1287/opre.11.6.972](https://doi.org/10.1287/opre.11.6.972).
- [52] J. L. Calvet, J. Cardillo, J. C. Hennet, and F. Szigeti, “Method of relaxation applied to optimization of discrete systems,” in *Proc. Colloq. Differ. Equ. Appl.*, Maracaibo, Venezuela, 2005, pp. 13–19.
- [53] F. Glover, “Tabu search—Part I,” *ORSA J. Comput.*, vol. 1, no. 3, pp. 190–206, Aug. 1989, doi: [10.1287/ijoc.1.3.190](https://doi.org/10.1287/ijoc.1.3.190).
- [54] F. Glover, “Tabu search—Part II,” *ORSA J. Comput.*, vol. 2, no. 1, pp. 4–32, Feb. 1990, doi: [10.1287/ijoc.2.1.4](https://doi.org/10.1287/ijoc.2.1.4).
- [55] S. Puri, C. K. Andersen, A. L. Grimsmo, and A. Blais, “Quantum annealing with all-to-all connected nonlinear oscillators,” *Nature Commun.*, vol. 8, no. 1, p. 15785, Aug. 2017, doi: [10.1038/ncomms15785](https://doi.org/10.1038/ncomms15785).
- [56] A. Rocchetto, S. C. Benjamin, and Y. Li, “Stabilizers as a design tool for new forms of the Lechner-Hauke-Zoller annealer,” *Sci. Adv.*, vol. 2, no. 10, Oct. 2016, Art. no. e1601246, doi: [10.1126/sciadv.1601246](https://doi.org/10.1126/sciadv.1601246).
- [57] M. Fielding, “Simulated annealing with an optimal fixed temperature,” *SIAM J. Optim.*, vol. 11, no. 2, pp. 289–307, Jan. 2000, doi: [10.1137/S1052623499363955](https://doi.org/10.1137/S1052623499363955).
- [58] D. T. Connolly, “An improved annealing scheme for the QAP,” *Eur. J. Oper. Res.*, vol. 46, no. 1, pp. 93–100, May 1990, doi: [10.1016/0377-2217\(90\)90301-Q](https://doi.org/10.1016/0377-2217(90)90301-Q).
- [59] R. Horn and C. Johnson, *Topics in Matrix Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 1991.



YOSHIHIRO NAMBU received the B.Eng., M.Eng., and Dr.Eng. degrees in engineering science from Kyoto University, in 1985, 1987, and 2006, respectively. He was a Researcher at the NEC Corporation, from 1987 to 2021. He is currently an Invited Senior Researcher with the NEC-AIST Quantum Technology Cooperative Research Laboratory, National Institute of Advanced Industrial Science and Technology. His research interests include quantum information, quantum cryptography, quantum optics, quantum computing, and statistical mechanics. He is a member of JPS and JSAP.

• • •