Finding optimal pathways in chemical reaction networks using Ising machines

Yuta Mizuno [1,2,3,*] and Tamiki Komatsuzaki [1,2,3,4]

[1]*Research Institute for Electronic Science, Hokkaido University, Sapporo, Hokkaido 001-0020, Japan*
[2]*Institute for Chemical Reaction Design and Discovery (WPI-ICReDD), Hokkaido University, Sapporo, Hokkaido 001-0021, Japan*
[3]*Graduate School of Chemical Sciences and Engineering, Hokkaido University, Sapporo, Hokkaido 060-8628, Japan*
[4]*SANKEN, Osaka University, Ibaraki, Osaka 567-0047, Japan*

(Received 8 August 2023; accepted 21 December 2023; published 29 January 2024)

Finding optimal pathways in chemical reaction networks is essential for elucidating and designing chemical processes, with significant applications such as synthesis planning and metabolic pathway analysis. Such a chemical pathway-finding problem can be formulated as a constrained combinatorial optimization problem, aiming to find an optimal combination of chemical reactions connecting starting materials to target materials in a given network. Due to combinatorial explosion, the computation time required to find an optimal pathway increases exponentially with the network size. Ising machines, including quantum and simulated annealing devices, are promising novel computers dedicated to such hard combinatorial optimization. However, to the best of our knowledge, there has yet to be an attempt to apply Ising machines to chemical pathway-finding problems. In this article, we present the Ising/quantum computing application for chemical pathway-finding problems. The Ising model, translated from a chemical pathway-finding problem, involves several types of penalty terms for violating constraints. It is not obvious how to set appropriate penalty strengths of different types. To address this challenge, we employ Bayesian optimization for parameter tuning. Furthermore, we introduce a technique that enhances tuning performance by grouping penalty terms according to the underlying problem structure. The performance evaluation and analysis of the proposed algorithm were conducted using a D-Wave Advantage system and simulated annealing. The benchmark results reveal challenges in finding exact optimal pathways. Concurrently, the results indicate the feasibility of finding approximate optimal pathways, provided that a certain degree of relative error in cost value is acceptable.

DOI: 10.1103/PhysRevResearch.6.013115

I. INTRODUCTION

Since the inception of modern chemistry in the 18th century, scientists have designed and discovered numerous chemical reactions. These discoveries are recorded in chemical databases such as Reaxys [1] and SciFinder^n [2]. The vast collections of chemical reactions stored in these databases are big data in chemistry and have experienced rapid growth. For instance, Reaxys contains more than 60 million chemical reactions as of 2023. The number of chemical reactions added to Reaxys each year is more than hundreds of thousands, which has more than doubled between 2000 and 2015 [3]. Additionally, emerging technologies for automatic reaction exploration, such as chemical synthesis robots [4,5], artificial intelligence and machine learning [6,7], and quantum chemical calculation [8], will further accelerate the growth of chemical databases.

A chemical reaction database can be represented by a gigantic chemical reaction network. A chemical reaction network (CRN) is a graph-theoretical representation of a collection of chemical reactions, modeled as a directed bipartite graph with two types of nodes: chemical reactions and species [9,10]. This representation is used in chemistry and systems biology to visualize, elucidate, and design the mechanisms of complex chemical processes.

Finding optimal pathways in CRNs is a static analysis of CRNs, which has significant applications such as synthesis planning [10–13] and metabolic pathway analysis [14,15]. Such a chemical pathway-finding problem can be formulated as a combinatorial optimization problem to find an optimal combination of chemical reactions connecting starting materials to target materials in a given CRN. In chemical synthesis, chemists aim to find synthesis pathways from readily available materials to target compound(s). Moreover, they seek synthesis pathways with better properties, such as low monetary costs, short execution times, low risks, and ecofriendliness. In systems biology, identifying metabolic pathways is fundamental to characterizing the function and mechanism of a living system. Furthermore, metabolic engineers aim to design artificial metabolic pathways to maximize the target metabolite production of cells. There are several algorithms for these pathway-finding problems, including recursive network search algorithms [11,12], optimization/enumeration

*mizuno@es.hokudai.ac.jp

algorithms using a mixed integer programming solver [13,15], and a simulated annealing algorithm designed for synthesis planning [11].

However, finding optimal pathways in CRNs is NP-hard in general [14]; the computation time to find an optimal solution increases exponentially as the network size increases due to combinatorial explosion. Consequently, the continuous improvement of software and hardware for pathway finding in CRNs is crucial to address the rapidly increasing computational demands associated with growing data size in chemistry.

In recent years, Ising machines have gained considerable attention as next-generation hardware devices dedicated to combinatorial optimization [16]. Ising machines are specially designed to find the ground state(s) of an Ising model characterized by the Hamiltonian

$$H_{\mathrm{Ising}}(\boldsymbol{\sigma}) = -\sum_{i=1}^{N_\sigma - 1} \sum_{j=i+1}^{N_\sigma} J_{ij}\sigma_i\sigma_j - \sum_{i=1}^{N_\sigma} h_i\sigma_i. \tag{1}$$

Here, $\sigma_i \in \{-1, +1\}$, $\boldsymbol{\sigma}$, N_σ, J_{ij}, and h_i denote, respectively, a spin variable, the spin configuration $(\sigma_1, \sigma_2, \ldots, \sigma_{N_\sigma})$, the number of spins, the interaction coefficient between two spins σ_i and σ_j, and the local field interacting with σ_i. Ising machines include quantum annealing devices developed by D-Wave Systems [17,18] and classical simulated annealing devices [19–23]. Many combinatorial optimization problems are efficiently reducible to the ground state finding problems of Ising models [24,25]. If chemical pathway-finding problems are also reducible to Ising model problems, these physics-inspired, novel computers may potentially offer a solution to the challenges posed by the accelerating growth in chemical data and the concurrent slowdown in performance growth for conventional von Neumann computers due to the end of Moore's law [26]. However, to the best of our knowledge, there has yet to be an attempt to apply Ising machines to chemical pathway-finding problems.

Pathway-finding problems in CRNs, as potential application targets of Ising machines and quantum annealing, exhibit the following notable features: (1) The available data size is enormous and increasing rapidly. (2) Various problem settings exist, including multiobjective and mixed integer programming formulation of synthesis planning [13] and exhaustive enumeration of possible metabolic pathways [15]. (3) The bipartite graph representation of a CRN can be viewed as a Petri net [27], a graphical, mathematical modeling tool for discrete event systems. Hence, chemical pathway-finding problems can be model cases for developing various general-purpose algorithms using Ising machines, which are anticipated to be transferable to other systems.

In this article, we present an Ising computing framework for pathway finding in CRNs. In Sec. II, we formulate a synthesis-planning problem as a typical chemical pathway-finding problem for algorithm development. In Sec. III, we detail our proposed algorithm, which includes a translation procedure of a chemical pathway-finding problem into an Ising model problem and postprocessing methods for enhancing the feasibility and optimality of solutions. Additionally, to address the challenge of tuning penalty strengths in the translated Ising model, we introduce Bayesian optimization and a technique to enhance tuning performance by reducing the search complexity in Sec. III C. In Sec. IV, we evaluate the performance of the proposed algorithm using a D-Wave Advantage quantum annealing machine [18] and simulated annealing on a classical computer. Finally, we summarize the present study and remark on possible directions for the future development of chemical pathway-finding algorithms using Ising machines in Sec. V.

II. PROBLEM FORMULATION

A. Chemical reaction networks and pathways

Figure 1 shows an example of chemical reaction networks (CRNs) in the bipartite graph representation [9,10], also known as the Petri net representation [27]. The network illustrates the Solvay process, an industrial chemical production process of soda ash (Na_2CO_3). The overall process is represented as $2NaCl + CaCO_3 \rightarrow Na_2CO_3 + CaCl_2$, which consists of five chemical reactions:

(1) $NaCl + CO_2 + NH_3 + H_2O \rightarrow NaHCO_3 + NH_4Cl$
(2) $2NaHCO_3 \rightarrow Na_2CO_3 + CO_2 + H_2O$
(3) $CaCO_3 \rightarrow CaO + CO_2$
(4) $CaO + H_2O \rightarrow Ca(OH)_2$
(5) $Ca(OH)_2 + 2NH_4Cl \rightarrow CaCl_2 + 2NH_3 + 2H_2O$

In Fig. 1, these five chemical reactions involved in the Solvay process are depicted as squares, and 11 chemical species are represented as ellipses. Each directed edge (arrow) connects a reaction with one of its reactants or products. The edge direction is either from a reactant species to a reaction or from a reaction to a product species. The number of directed edges between a reaction and a species indicates the stoichiometric coefficient, that is, the coefficient of the species in the chemical equation of the reaction.

A CRN may include inflow and outflow *dummy* reactions representing mass exchange between a chemical system and its environment. The network depicted in Fig. 1 includes four dummy reactions represented as dashed squares: the inflow reactions of NaCl and $CaCO_3$ (the purchase of the starting materials); the outflow reaction of Na_2CO_3 (the shipping of the chemical product); and the outflow reaction of $CaCl_2$ (either the shipping or disposal of the byproduct). Dummy reactions are useful for modeling pathway-finding problems, as detailed below. Henceforth, we will use the term"reaction" to refer to both chemical and dummy reactions.

We define a *pathway* in a CRN as a combination of chemical and inflow/outflow reactions, taking into account reaction multiplicity. Reaction multiplicity, which indicates the number of occurrences of a reaction, is essential for a quantitative description of complex reaction processes [13,15,28]. For instance, in Fig. 1, the numbers in black circles indicate the multiplicity of each reaction of the Solvay process. During a single cycle of the overall reaction, chemical reaction 1 occurs twice, while the other chemical reactions 2–5 occur once each. This specific combination of the multiplicities of the five chemical reactions enables the reuse of ammonia and carbon dioxide, resulting in an economical chemical transformation. In addition, the multiplicity of each inflow or outflow reaction corresponds to the consumption of the starting material or production of the (by)product during a

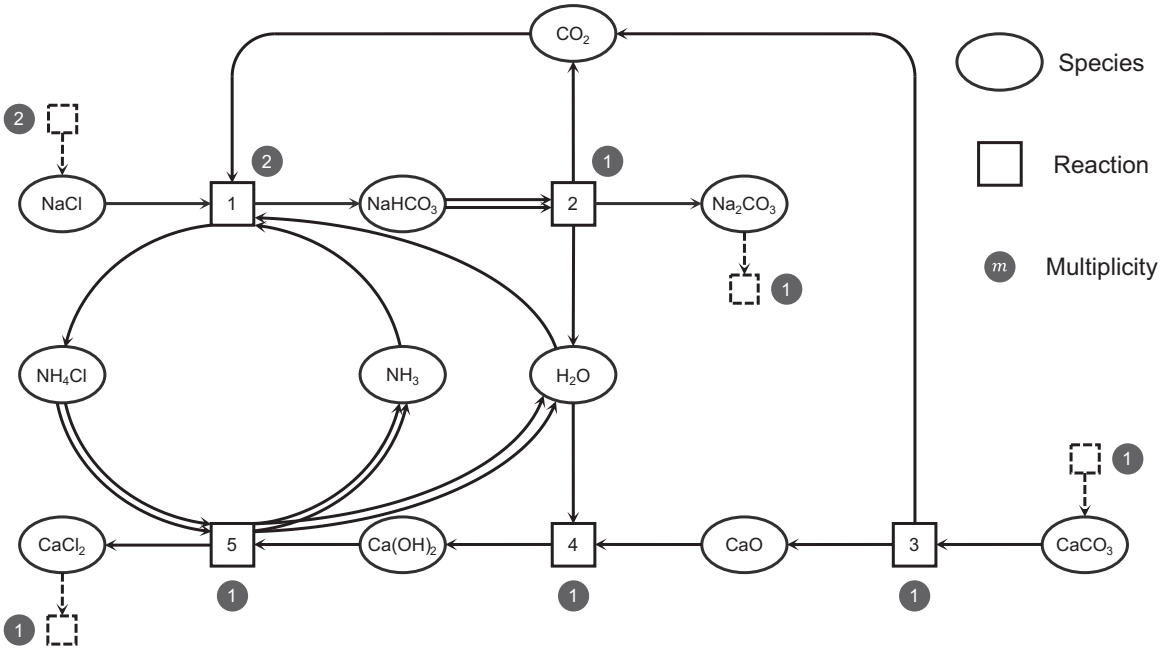


FIG. 1. An example of a chemical reaction network and a pathway: The Solvay process.

single overall process, respectively. It is important to mention, however, that the definition of pathways here does not consider the order of occurrence of chemical reactions, as in [13,15].

Physically feasible pathways must satisfy the mass balance equations for all chemical species in a network. The mass balance equation for species s is given by $\sum_{r\in R} \nu_{sr} m_r = 0$, where R, m_r, and ν_{sr} are the set of all reactions in the network, the multiplicity of reaction r, and the *signed* stoichiometric coefficient of species s in reaction r, respectively. The sign of ν_{sr} is defined as positive if r is a chemical reaction producing s or an inflow reaction supplying s into the system; conversely, it is defined as negative if r is a chemical reaction consuming s or an outflow reaction exporting s from the system; otherwise, it is zero. The mass balance equation states that the sum of the total production and input equals the sum of the total consumption and output. The mass balance constraints ensure that no species are created from nothing nor annihilated to nothing during the chemical process, in accordance with the law of conservation of mass.

B. Synthesis-planning problem

Let us formulate a synthesis-planning problem to find the minimum monetary cost synthesis pathway from commercially available substrates to target compound(s).

Let R be a set of chemical and inflow/outflow reactions potentially used for synthesizing the target species. Let S be the set of all chemical species participating in chemical reactions of R. Here, the candidate reactions in R and species in S can be listed in advance while the multiplicities of reactions of the desired synthesis pathway are unknown. For example, the chemical reactions in R can be either extracted from a chemical reaction database (see Appendix A) or generated by computer-aided retrosynthesis [13]. Only commercially available species have inflow reactions in R. All target species must have outflow reactions, while others may also have outflow reactions representing disposal.

The decision variables in the synthesis-planning problem are the multiplicities of each reaction in R. Let x_r denote the variable representing the multiplicity of $r \in R$. In this article, we assume each variable x_r is a nonnegative integer variable with lower bound l_r ($\geqslant 0$) and upper bound u_r. Since feasible synthesis pathways must ensure the positive outflow of each target species, the lower bound of the outflow reaction multiplicity for each target species must be positive; in this way, positive-outflow constraints specify which species are targets of the chemical synthesis.

The monetary costs can be categorized into two types: variable costs and fixed costs. Variable costs vary with changes in the multiplicities of reactions. For example, the costs of substrate purchase and byproduct disposal are proportional to the substrate inflows and the byproduct outflows, respectively. The simplest model of such variable costs can be represented as $c_r^{\text{unit}} x_r$, where c_r^{unit} is the unit cost of reaction r. On the other hand, fixed costs depend only on whether reactions are carried out. For example, the costs of equipment preparation for reactions are overhead costs for carrying out the reactions and are independent of the amounts of the reactions. These fixed costs can be modeled as $c_r^{\text{fixed}} \chi_+(x_r)$, where c_r^{fixed} is the fixed cost of reaction r and χ_+ denotes the positivity indicator function defined as $\chi_+(x) = 0$ for $x = 0$ and $\chi_+(x) = 1$ for $x > 0$.

In summary, the synthesis-planning problem is formulated as follows:

$$\begin{aligned} \underset{\boldsymbol{x}\in\mathbb{N}_0^{|R|}}{\text{minimize}} \quad & \sum_{r\in R} c_r^{\text{unit}} x_r + \sum_{r\in R} c_r^{\text{fixed}} \chi_+(x_r), \\ \text{subject to} \quad & \forall s \in S, \ \sum_{r\in R} \nu_{sr} x_r = 0, \\ & \forall r \in R, \ l_r \leqslant x_r \leqslant u_r. \end{aligned} \tag{2}$$

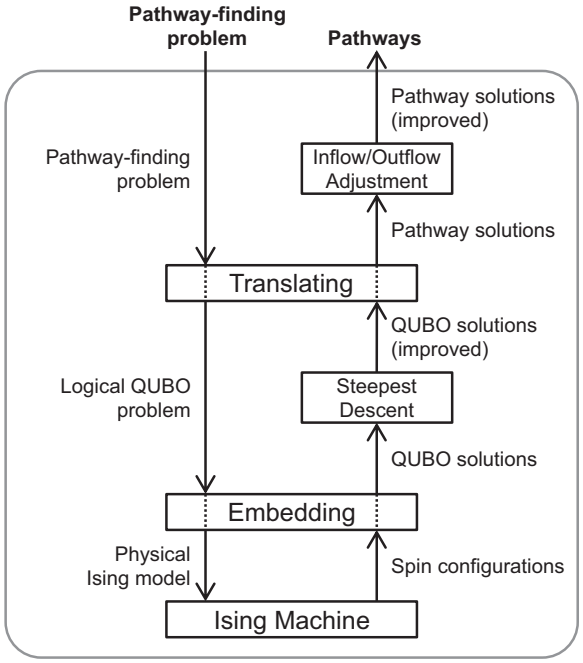


FIG. 2. The overview of the proposed algorithm for finding pathways in chemical reaction networks using Ising machines.

Here, $\boldsymbol{x}$ denotes the vector representation of the reaction multiplicity variables $\{x_r\}$.

Note that a gap still exists between the Ising model [Eq. (1)] and the pathway-finding problem [Eq. (2)]. In the next section, we will present a connection between them.

III. PROPOSED ALGORITHM

This section presents an algorithm for finding pathways in chemical reaction networks using Ising machines. Figure 2 illustrates the overview of the proposed algorithm. This algorithm first translates a chemical pathway-finding problem into a quadratic unconstrained binary optimization (QUBO) problem, mathematically equivalent to an Ising model problem. Then, the translated problem is solved by an Ising machine. Here, the algorithm embeds the *logical* QUBO problem into a *physical* Ising model of the device if the device hardware has limited spin connectivity. Finally, the algorithm decodes binary solutions returned from the Ising machine and applies postprocessing methods to enhance the feasibility and optimality of the solutions.

In the following subsections, we will elaborate on (1) the translating procedure, (2) the postprocessing methods, and (3) the automatic tuning of parameters in the QUBO/Ising form of a pathway-finding problem. In addition, we review Ising machines and the embedding process in Appendix B for readers unfamiliar with Ising computing.

A. Translating into QUBO

We translate our pathway-finding problem [Eq. (2)] into a quadratic unconstrained binary optimization (QUBO) problem. This QUBO problem is mathematically equivalent to the energy minimization problem of an Ising model. QUBO is

TABLE I. Integer-to-binary encoding methods. Here, an integer variable $x \in [l, u]$ is expressed by n binary variables $\boldsymbol{q} \in \{0, 1\}^n$. The number n of required binary variables is d $(:= u - l)$ for the unary and order encoding methods, $K + 1$ $(K := \lfloor \log_2 d \rfloor)$ for the log encoding method, and $d + 1$ for the one-hot encoding method.

Type	Integer variable expression $x(\boldsymbol{q})$	Additional penalty $P_x(\boldsymbol{q})$
Unary	$l + \sum_{k=1}^{d} q_k$	0
Order	$l + \sum_{k=1}^{d} q_k$	$\sum_{k=1}^{d-1} q_{k+1}(1 - q_k)$
Log	$l + \sum_{k=0}^{K-1} 2^k q_k$ $+ [d - (2^K - 1)]q_K$	0
One-hot	$l + \sum_{k=0}^{d} k q_k$	$(\sum_{k=0}^{d} q_k - 1)^2$

defined as

$$\underset{\boldsymbol{q}\in\{0,1\}^{N_q}}{\text{minimize}} \quad \sum_{i=1}^{N_q}\sum_{j=i}^{N_q} Q_{ij} q_i q_j, \tag{3}$$

where $q_i \in \{0, 1\}$, $\boldsymbol{q}$, N_q, and Q_{ij} denote, respectively, a binary variable, the binary variables vector $(q_1, q_2, \ldots, q_{N_q})$, the number of binary variables, and a constant associated with q_i and q_j. The binary variables are interconvertible with the Ising spin variables as $q_i = (1 - \sigma_i)/2$. Note that the diagonal term $Q_{ii}q_i^2$ is essentially linear due to the idempotent law $q_i^2 = q_i$ for $q_i \in \{0, 1\}$.

First, we translate the objective function into a quadratic function. To achieve this, we devised the following mapping from the positivity indicator function into a quadratic expression. The positivity indicator function of nonnegative integer variable x can be expressed as

$$\chi_+(x) = \min_{y\in\{0,1\}} [y + (1 - y)x]. \tag{4}$$

Therefore, we can replace $\chi_+(x_r)$ in the original cost function with $[y_r + (1 - y_r)x_r]$, adding an auxiliary binary variable $y_r \in \{0, 1\}$.

Next, instead of imposing the mass balance constraints directly, we add the penalty term

$$\sum_{s\in S} M_s \left(\sum_{r\in R} \nu_{sr} x_r \right)^2 \tag{5}$$

to the QUBO objective function. Here, M_s is a positive constant. This term equals zero if solution $\boldsymbol{x}$ satisfies all the mass balance constraints and a positive value otherwise. Thus, this term penalizes solutions violating the mass balance constraints. The penalty strength is determined by M_s. In theory, large-enough penalty strength ensures that the optimal solution of this unconstrained problem equals that of the original problem. However, the penalty strength requires careful tuning in practice, as extremely large penalty strength can impair the annealing performance. Section III C describes the penalty strength parameter tuning in detail.

Finally, we encode integer variables into binary variables. We employ four types of integer-to-binary encoding methods: unary, order, log, and one-hot [29]. Table I summarizes these encoding methods. In the table, an integer variable $x \in [l, u]$

is expressed by a linear expression $x(\boldsymbol{q})$ of n binary variables $\boldsymbol{q} \in \{0,1\}^n$. The number n of required binary variables is d $(:= u - l)$ for the unary and order encoding methods, $\lfloor \log_2 d \rfloor + 1$ for the log encoding method, and $d+1$ for the one-hot encoding method. In addition, the order and one-hot encoding methods require a penalty term in the QUBO objective function. The additional penalty in the order encoding method corresponds to constraints that $q_k = 0 \Rightarrow q_{k+1} = 0$ for $k = 1, \ldots, d-1$. Note that the order constraints are not necessarily hard constraints; even if the order constraints are not satisfied, the integer variable expression $x(\boldsymbol{q})$ takes an integer value in $[l, u]$. On the other hand, the additional penalty in the one-hot encoding method corresponds to a constraint that only one binary variable takes the value one and all the others zero. In contrast to the order encoding method, if the one-hot constraint is violated, $x(\boldsymbol{q})$ may take an infeasible value of x; thus, the one-hot constraint is a hard constraint. We will compare the performance of these four encoding methods in Sec. IV B.

In summary, the synthesis-planning problem [Eq. (2)] can be represented in QUBO form as follows:

$$\underset{\boldsymbol{q}\in\{0,1\}^{N_q}}{\text{minimize}} \sum_{r\in R} c_r^{\text{unit}} x_r(\boldsymbol{q}_r) + \sum_{r\in R} c_r^{\text{fixed}} [y_r + (1-y_r) x_r(\boldsymbol{q}_r)] + \sum_{s\in S} M_s \left(\sum_{r\in R} \nu_{sr} x_r(\boldsymbol{q}_r) \right)^2 + \sum_{r\in R} L_r P_{x_r}(\boldsymbol{q}_r). \tag{6}$$

Here, each x_r is expressed by binary variables $\boldsymbol{q}_r$ by one of the four encoding methods listed in Table I. $P_{x_r}(\boldsymbol{q}_r)$ and L_r (> 0) are the additional penalty term and its strength parameter for the encoding of x_r, respectively. Each y_r $(\in \{0,1\})$ is an auxiliary binary variable for translating the positivity indicator function $\chi_+(x_r)$. Finally, the symbol $\boldsymbol{q}$ collectively denotes all binary variables $\{\boldsymbol{q}_r \mid r \in R\} \cup \{y_r \mid r \in R\}$, and N_q is the total number of the binary variables.

B. Postprocessing

Ising machines often return nonoptimal or infeasible solutions. To improve the quality of solutions, we employ two postprocessing methods: steepest descent and inflow/outflow adjustment.

Steepest descent is a standard postprocessing method in Ising computing. This method iteratively descends the energy landscape of an Ising model by single-spin flips. At each step, it performs the spin flip that reduces the energy the most among all possible single-spin flips. This postprocessing method is implemented in the D-Wave Ocean Software [30]. We employ it to enhance the optimality and feasibility of Ising/QUBO solutions returned by an Ising machine.

Inflow/outflow adjustment, which we developed, is aimed at adjusting the inflow and outflow of each species so that the mass balance equals zero. Specifically, this method (1) sets the inflow and outflow of each species to their minimum values and (2) increases either the inflow or outflow as much as necessary to satisfy the mass balance constraint. The pseudocode is given in Fig. 3. We apply this postprocessing method at the end of the workflow to enhance the feasibility of solutions.

Algorithm 1 Adjust the inflow and outflow of each chemical species

Input: S, R, ν, l, u, m
Output: $\overline{m}$

```
for all r ∈ R do
    if r is an inflow or outflow reaction then
        m̄_r ← l_r
    else
        m̄_r ← m_r
    end if
end for
for all s ∈ S do
    Δ ← Σ_{r∈R} ν_sr m̄_r
    if Δ > 0 and s has the outflow reaction r_s^out then
        m̄_{r_s^out} ← min(m̄_{r_s^out} + |Δ|, u_{r_s^out})
    else if Δ < 0 and s has the inflow reaction r_s^in then
        m̄_{r_s^in} ← min(m̄_{r_s^in} + |Δ|, u_{r_s^in})
    end if
end for
return m̄
```

FIG. 3. Inflow/outflow adjustment algorithm. The input arguments are a set of chemical species S, a set of chemical and inflow/outflow reactions R, a signed stoichiometric coefficient function $\nu : S \times R \to \mathbb{Z}; (s,r) \mapsto \nu_{sr}$, a lower bound function $l : R \to \mathbb{N}_0; r \mapsto l_r$, an upper bound function $u : R \to \mathbb{N}_0; r \mapsto u_r$, and the multiplicity function representing a pathway $m : R \to \mathbb{N}_0; r \mapsto m_r$. This algorithm, first, resets the multiplicity of each inflow/outflow reaction to its lower bound. Then, it increases the outflow of overproduced species $(\Delta > 0)$ and the inflow of overconsumed species $(\Delta < 0)$. Finally, it returns the multiplicity function of the improved pathway $\overline{m} : R \to \mathbb{N}_0; r \mapsto \overline{m}_r$. Note that this postprocessing does not ensure that a postprocessed pathway satisfies all mass balance constraints. For example, if a species is overconsumed, but the associated inflow is unavailable, this method cannot adjust the inflow/outflow to make the mass balance zero.

C. Penalty strength tuning

The QUBO form of the pathway-finding problem [Eq. (6)] has penalty strength parameters. Moreover, when the QUBO problem is embedded into a physical Ising machine with limited connectivity of spins, the physical Ising model has chain strength parameters (see Appendix B 3). In the following discussion, we consider parameter tuning to improve algorithm performance.

Parameter tuning aims to find parameter values that maximize the quality of solutions returned by the Ising computing algorithm. The parameter tuning problem is formulated as

$$\underset{\boldsymbol{\lambda}\in D}{\text{minimize}}\, f(\boldsymbol{\lambda}), \tag{7}$$

where $\boldsymbol{\lambda}$ is the parameter vector to be tuned (in this study, penalty and chain strength parameters), D is the domain of $\boldsymbol{\lambda}$, and $f : D \to \mathbb{R}$ is a function measuring the quality of solutions (smaller is better).

Let us define the parameter search space D. In the present case, we can estimate an upper bound of penalty strength parameters, which is large enough to ensure the equivalence

between the original and QUBO problems. A simple upper bound is the maximum cost among all feasible and infeasible pathways,

$$\overline{C} = \sum_{r\in R} \left(c_r^{\text{unit}} u_r + c_r^{\text{fixed}}\right). \tag{8}$$

If all penalty strength parameters (i.e., $\{M_s\}$ and $\{L_r\}$) equal $\overline{C}$, the QUBO objective function value for any feasible solution is less than $\overline{C}$[1]; in contrast, that for any infeasible solution is greater than or equals to $\overline{C}$. Therefore, the penalty strength $\overline{C}$ ensures the equivalence between the original and QUBO problems. This upper bound estimation is also valid for chain strength parameters; if the chain strength equals $\overline{C}$, a spin configuration with chain breaks has an energy greater than any feasible solution without chain breaks. Therefore, we define the search space as

$$D = [0, \overline{C}]^{N_{\text{params}}} \subset \mathbb{R}^{N_{\text{params}}}, \tag{9}$$

where N_{params} is the total number of penalty and chain strength parameters to be tuned.

Next, we define the evaluation function f. Multiple runs of the algorithm produce a set of synthesis pathways. This set contains feasible and infeasible pathways with various synthesis costs. Our goal here is (1) to get many feasible solutions and (2) to get low costs solutions. Therefore, we use the following function to evaluate a single solution:

$$E(\boldsymbol{x}) = \sum_{r\in R} \left(c_r^{\text{unit}} x_r + c_r^{\text{fixed}} \chi_+(x_r)\right) + \overline{C} \sum_{s\in S} \left(\sum_{r\in R} \nu_{sr} x_r\right)^2. \tag{10}$$

Here, the first term evaluates the total costs of synthesis, and the second is the penalty for infeasible solutions. Since the penalty strength equals $\overline{C}$ in the formula, all infeasible solutions have a higher value of E than any feasible solution. Using this measure of single solution quality, we define $f(\boldsymbol{\lambda})$ as the expected value of $E(\boldsymbol{x})$,

$$\langle E\rangle_{\boldsymbol{\lambda}} = \sum_{\boldsymbol{x}\in\mathbb{N}_0^{|R|}} E(\boldsymbol{x}) p(\boldsymbol{x}|\boldsymbol{\lambda}), \tag{11}$$

where $p(\boldsymbol{x}|\boldsymbol{\lambda})$ is the probability that the algorithm returns solution $\boldsymbol{x}$ when using parameters $\boldsymbol{\lambda}$. In practice, $\langle E\rangle_{\boldsymbol{\lambda}}$ is evaluated by a sample mean of $E(\boldsymbol{x})$ from a finite set of samples of $\boldsymbol{x}$.

We adopt Bayesian optimization (BO) to minimize $\langle E\rangle_{\boldsymbol{\lambda}}$ over parameters $\boldsymbol{\lambda}$. Specifically, we employ a BO algorithm implemented in the Optuna library [31,32]. BO is a sequential black-box optimization technique that selects points to be evaluated in an iteration step based on a surrogate model of f constructed with previous evaluation results. The Optuna's algorithm uses the tree-structured Parzen estimator (TPE) [33] as a surrogate model. BO-TPE is known to be effective in a wide range of hyperparameter tuning in machine learning and has less computational time complexity than standard BO algorithms using the Gaussian process [34].

However, the high dimensionality of the search space D may impair the efficiency of BO. Although BO is applicable to parameter tuning in a complex and high-dimensional search space, its efficiency decreases rapidly as the dimension of the search space increases [34]. The dimension of D is N_{params}, which is on the same order of magnitude as the total numbers of species and reactions. Therefore, reducing the dimensionality of D is crucial for efficient parameter tuning.

To address the issue of high dimensionality in D, we devised parameter grouping methods. In these grouping methods, penalty and chain strength parameters are grouped according to the associated constraint type, and all parameters in each group are constrained to take the same value. We first classify parameters into three major categories: (1) penalty strength parameters associated with mass balance constraints, (2) penalty strength parameters related to integer-to-binary encoding, and (3) chain strength parameters for embedding. In this article, we further consider four types of subdivisions of the mass balance constraints group according to chemical species classifications listed in Table II and detailed below. For simplicity, we leave the other groups related to encoding and embedding intact, although these groups can be further subdivided according to reaction classifications.

TABLE II. Grouping methods for penalty strength parameters associated with mass balance constraints. The right column lists species groups corresponding to mass balance constraint groups. See the text for details on the chemical species classifications.

Type	Groups
Unified	all species
Degree	degree-1 species, degree-2 species, ...
Depth	targets, depth-1 precursors, depth-2 precursors, ..., byproducts
Category	targets, substrates, intermediates, substrates/intermediates, byproducts

The details of the four grouping methods for mass balance constraints are as follows. The first method, "unified," consolidates all mass balance constraints into one group without any further subdivision. The second method, "degree," classifies mass balance constraints by the associated species' node degree. Here, the node degree is defined as the number of reactions involving the species. The third and fourth methods have been designed based on the specific structure of synthesis-planning problems. The third method, named "depth," organizes mass balance constraints by the associated species' depth. The depth of a precursor species is defined as the minimum number of synthesis steps from the species to one of the target species, as detailed in Appendix A. The fourth method, termed "category," categorizes mass balance constraints by the associated species' category: "targets," "substrates" (species that are exclusively starting materials and not synthesized from others), "intermediates" (species that need to be synthesized from others), "substrates/intermediates" (species that can be either starting materials or synthesized from others), and "byproducts." We

[1]If solution $x_r = u_r$ $(\forall r \in R)$ is feasible, this statement is false; in that case, a value greater than $\overline{C}$ can be employed as the upper bound of the parameter values. However, in every benchmark problem used in the present article, the solution $x_r = u_r$ is confirmed to be infeasible.

will compare the performance of parameter tuning using these grouping methods in Sec. IV B.

We note that the Optuna library has already been used in parameter tuning for Ising computing in [35,36]. In [35], Optuna was applied to tuning parameters of the Digital Annealer, not penalty strength parameters. In [36], all penalty strength parameters are consolidated into one group, and the single, unified penalty strength parameter was tuned by Optuna. In contrast to these studies, the present study systematically elucidates the effects of several different parameter grouping methods on tuning performance. In the next section, we will demonstrate that appropriate choices of parameter grouping methods significantly contribute to performance improvement.

IV. PERFORMANCE EVALUATION

In this section, we present a performance evaluation of the proposed algorithm using a D-Wave Advantage quantum annealing machine and a simulated annealing software. The primary objective of these experiments is to assess the feasibility and efficacy of the proposed algorithm.

A. Experimental setting

We generated 100 benchmark problems of synthesis planning based on the USPTO dataset of chemical reactions [37]. In our benchmark problems, fixed reaction costs and substrate purchase costs are randomly assigned to all chemical reactions and commercially available species, respectively. The benchmark problem generation process is detailed in Appendix C. These benchmark problems have been designed to assess the feasibility of the proposed algorithm using a D-Wave Advantage quantum annealer with a limited number of qubits. Thus, they are relatively small-scale problems: the number of integer variables N_x, equivalent to the number of chemical and inflow/outflow reactions $|R|$, ranges from 3 to 116; the number of chemical species $|S|$, ranges from 2 to 80. All of these problems can be embedded into the D-Wave Advantage's Pegasus topology QPU [38], with the number of required physical qubits ranging from 18 to 3752 when utilizing the unary or order encoding method. Refer to Appendix D 1 for the details of the embedding results.

We solved the benchmark problems using our proposed algorithm. The Ising machines employed were (1) the D-Wave Advantage system 4.1 quantum annealer [18] and (2) the `SimulatedAnnealingSampler` of the D-Wave Ocean Software [30]. We used the `minorminer.find_embedding` function of the D-Wave Ocean Software to find embeddings of QUBO problems into the D-Wave Advantage system 4.1. Note that the simulated annealing software does not require the embedding process, as it can solve Ising models with any spin connectivity. We employed the `SteepestDescentSolver` of the D-Wave Ocean Software for the steepest descent postprocessing. We implemented the other components of the algorithm, that is, the translation process based on SymPy [39] and the inflow/outflow adjustment postprocessing, in Python. We integrated all these components into the workflow illustrated in Fig. 2. We utilized the multivariate version of the `TPESampler` of the Optuna library [32] to tune penalty and chain strength parameters. Unless otherwise noted, we used default settings for the functions from the D-Wave Ocean Software and the Optuna library in the benchmarking. The simulated annealing, embedding finding, translating, postprocessing, and parameter tuning were executed on a local Linux machine with Intel Xeon Gold 6248R processors (3.0 GHz, 24 cores × 2). Our local machine in Hokkaido, Japan, communicated with the D-Wave Advantage system 4.1 in British Columbia, Canada, via the internet, using the Leap quantum cloud service.

For performance comparison, we solved the benchmark problems also with the Gurobi Optimizer version 9.5.1 [40]. The Gurobi Optimizer is a commercial state-of-the-art solver for mathematical programming, including integer linear programming, and was employed in a previous study on chemical pathway-finding problems [13]. To solve the synthesis planning problems with the Gurobi Optimizer, we translated them into integer linear programming by replacing the positivity indicator function in a fixed cost term, $\chi_+(x_r)$, with an auxiliary binary variable $y_r \in \{0, 1\}$ satisfying $x_r \leqslant u_r y_r$. The chemical pathway-finding algorithm with the Gurobi Optimizer was executed on our local machine using up to 32 threads.

B. Results and discussion

1. Best choice of parameters

We first investigated the best combination of an integer-to-binary encoding method (unary, order, log, or one-hot) and a parameter grouping method (unified, degree, depth, or category). This investigation was conducted on five representative problems selected from the 100 benchmark problems to cover a wide range of problem sizes. The problems selected were (a) the 20th, (b) the 40th, (c) the 60th, (d) the 80th, and (e) the 100th in ascending order of the number of integer variables. For each combination of encoding and grouping methods, we tuned penalty and chain strength parameters for the five representative problems. During the tuning processes, the annealing time for quantum annealing (QA) was fixed to 20 μs, and the number of sweeps (corresponding to the annealing time) for simulated annealing (SA) was fixed to 1000. These are the default values for the D-Wave Advantage system and the SA program we used. We set the number of Bayesian optimization iterations to 300, which we consider sufficient to check for convergence trends. We sampled 200 solutions per tuning iteration to compute the tuning score function, Eq. (11), with balancing statistical uncertainty and computational cost. We measured the performance of each combination of encoding and grouping methods by the best score of the tuning objective function over the 300 tuning trials. The performance comparisons for the D-wave Advantage system and SA are shown in Figs. 4 and 5, respectively.

Concerning the encoding methods, the unary and order encoding methods demonstrate the highest performance. Subsequently, the log encoding method also yields satisfactory results, whereas the one-hot encoding method exhibits the poorest performance. These results on the performance of the encoding methods are consistent with a previous study [41].

Among the four grouping methods, the depth and category grouping methods outperform the others when employing the unary or order encoding methods for both QA and SA. The

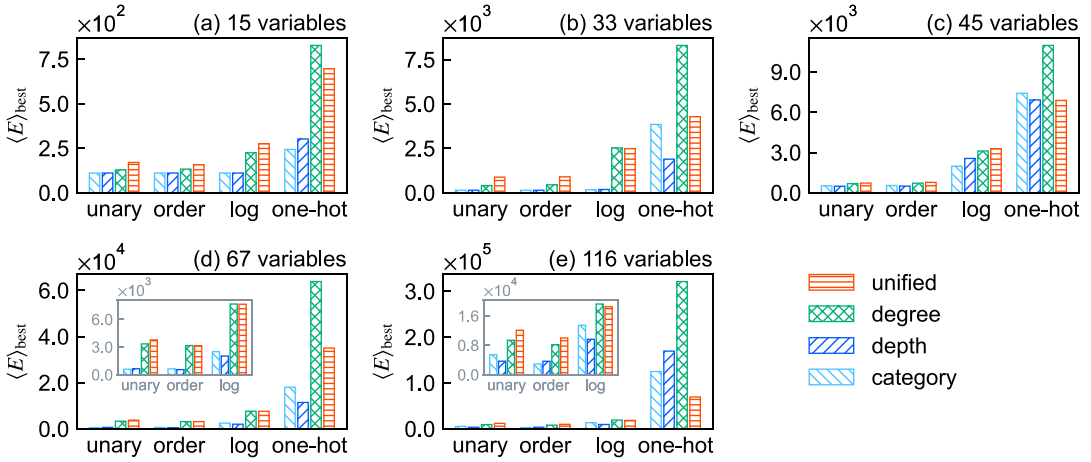


FIG. 4. Performance comparison of different combinations of an integer-to-binary encoding method (unary, order, log, or one-hot) and a parameter grouping method (unified, degree, depth, or category) in the proposed algorithm using the D-Wave Advantage system 4.1. Each panel displays the results for (a) the 20th, (b) the 40th, (c) the 60th, (d) the 80th, and (e) the 100th from the 100 benchmark problems sorted in ascending order by the number of integer variables, respectively. The performance measure $\langle E \rangle_{\text{best}}$ is the best score of the tuning objective function, Eq. (11), during 300 iterations of Bayesian optimization (smaller is better). The insets in panels (d) and (e) provide magnified views.

performance difference between these two methods and the others is prominent, especially for large problems (d) and (e). Furthermore, the degree grouping method exhibits the least performance in many cases, although the degree grouping method offers more degrees of freedom for parameter tuning due to the subdivision of mass balance constraints than the unified grouping method. These findings suggest that the appropriate design of parameter groups based on the specific structure of problems is crucial for performance improvement.

Next, we examined the optimized penalty and chain strength values for the depth and category grouping methods to clarify the reason for the advantage of these grouping methods as well as the trends in penalty and chain strength parameter values of each group with respect to the problem size. This examination was conducted on all 100 benchmark problems and employed the unary and order encoding methods. The settings for the tuning processes were the same as those described above.

Figure 6 depicts the penalty and chain strength parameter values optimized through Bayesian optimization for the combination of the category grouping and order encoding methods. We found that: (1) the penalty strength for the mass balance constraints of the target species is larger than that of the others and has a strong linear correlation with the number of integer variables N_x; (2) the chain strength parameter also exhibits a linear correlation with N_x; and (3) the other penalty strength parameters do not show any noticeable linear correlation with N_x. These penalty strength scaling patterns (1)–(3) are common across all four combinations of the unary/order encoding and depth/category grouping methods. Refer to Appendix D 4 for the results of the other combinations not shown in Fig. 6.

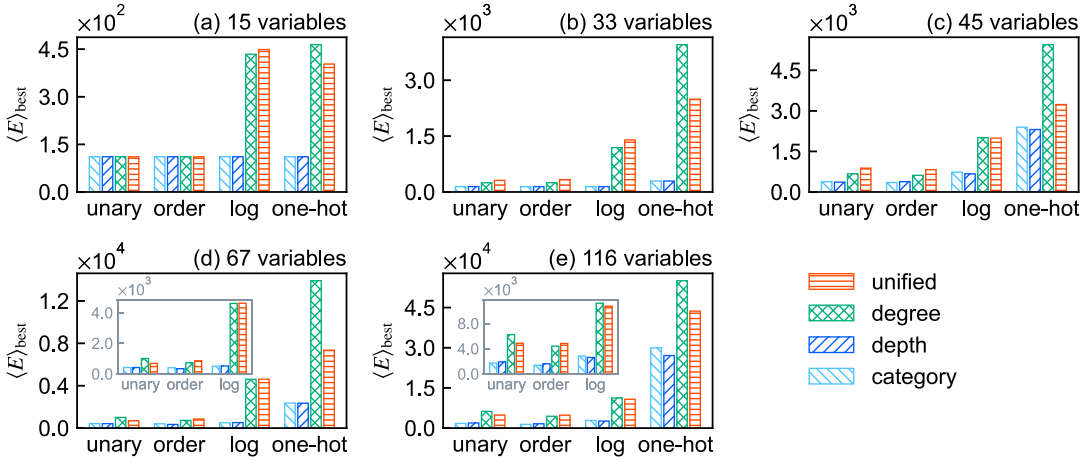


FIG. 5. Performance comparison of different combinations of an integer-to-binary encoding method (unary, order, log, or one-hot) and a parameter grouping method (unified, degree, depth, or category) in the proposed algorithm using simulated annealing. The notation is the same as that in Fig. 4.

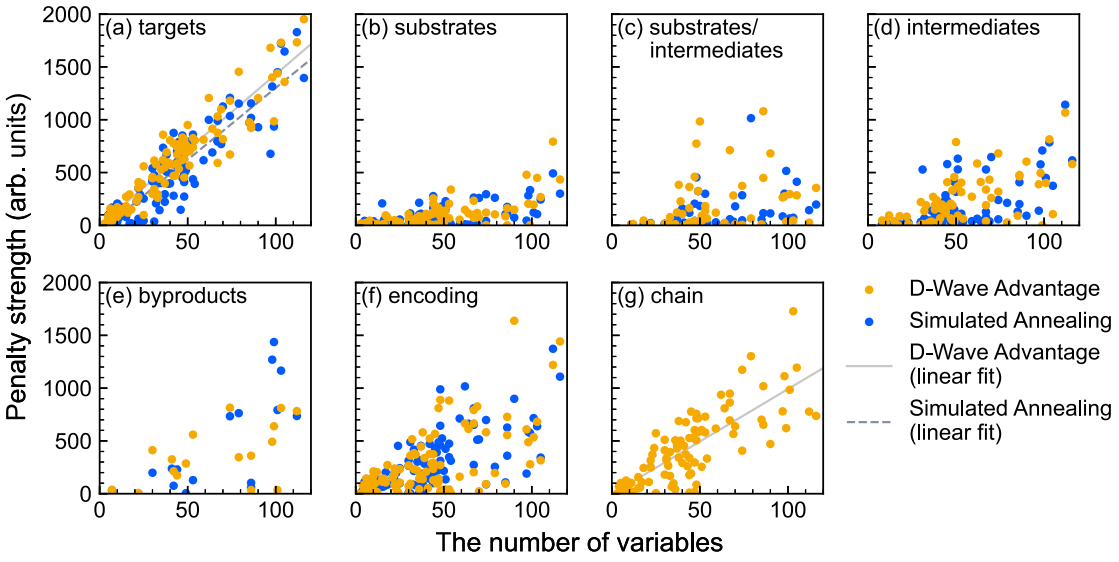


FIG. 6. Scaling of the tuned penalty and chain strength parameter values with respect to the number of integer variables in the case of using the order encoding and category grouping methods. Panels (a)–(e) depict the penalty strength parameters for mass balance constraints of species in each group, panel (f) shows the penalty strength parameter for order encoding, and panel (g) shows the chain strength for embedding. Linear fits are plotted for datasets with a coefficient of determination (R^2) greater than 0.5. The R^2 values are 0.88 [panel (a) for D-Wave Advantage], 0.83 [panel (a) for simulated annealing], and 0.65 [panel (g) for D-Wave Advantage], respectively.

The observed penalty strength scaling patterns and the advantage of the depth and category grouping methods can be explained as follows. The minimum-cost infeasible solution, $\boldsymbol{x} = \mathbf{0}$ (i.e., the plan of doing nothing), violates the mass balance constraints for the target species. An Ising machine is likely to sample this solution with a high probability unless the penalty for violating the mass balance constraints of the target species is sufficiently larger than feasible solutions' cost values. The steepest descent postprocessing may not improve this solution as it relies on the same QUBO formulation as the Ising machine. The inflow/outflow adjustment postprocessing also cannot reform the solution $\boldsymbol{x} = \mathbf{0}$ because the target species are overconsumed in the pathway, but the inflow reactions of the target species are unavailable by definition. Therefore, the penalty strength for the mass balance constraints of the target species must be large enough so that the Ising machine hardly samples the solution $\boldsymbol{x} = \mathbf{0}$. Furthermore, as the order of magnitude of feasible solutions' cost values increases almost linearly with respect to N_x, the penalty strength for the mass balance constraints of the target species must also increase linearly. In contrast, the violation of the mass balance constraints for nontarget species can often be reconciled by adjusting the associated inflow and outflow, although the effectiveness of the postprocessing may depend on the problem structure. Thus, the penalty strength associated with nontarget species can be moderate. Keeping penalty strengths moderate may enhance the exploration in the solution space during the annealing process, potentially leading to high performance in finding solutions. For the depth and category grouping methods, the penalty strength for the mass balance constraints of the target species and the others can be tuned separately; this is likely the reason for their advantage.

2. *Computation time*

We next examined the computation time scaling of the pathway-finding algorithms using the D-Wave Advantage system and SA. Since the algorithms output approximate solutions in a stochastic manner, the computation time should be evaluated by the time taken to find a solution within a certain cost tolerance ρ with a specified tolerant failure probability ϵ. This computation time metric is known as time-to-solution (TTS) [16,18], defined by

$$\mathrm{TTS}(\rho, \epsilon, \tau) = \tau_{\mathrm{algo}}(\tau)\left\lceil \frac{\log \epsilon}{\log\left(1 - p_{\mathrm{s}}(\rho, \tau)\right)} \right\rceil. \tag{12}$$

Here, τ is the annealing time; $\tau_{\mathrm{algo}}(\tau)$ is the average execution time of a single run of the algorithm under the annealing time τ; $p_{\mathrm{s}}(\rho, \tau)$ is the success probability for the algorithm under the annealing time τ to find a solution whose cost is at most ρC_{min}, where C_{min} denotes the minimum cost of feasible solutions.

The computation time analysis was conducted on all 100 benchmark problems and employed the order encoding and category grouping methods. We set the cost tolerance ρ to (a) 1, (b) 2, or (c) 3 and fixed the failure tolerance ϵ to 0.01. We measured the TTS for different annealing time τ: 1, 2, 5, 10, 20, 50, 100, 200, and 500 μs for QA; 10, 20, 50, 100, 200, 500, 1000, 2000, 5000, and 10 000 sweeps for SA. For each problem and τ, we ran the algorithm 1000 times with tuned penalty and chain strengths as illustrated in Fig. 6. In the measurement, we took into account only the runtime of annealing and postprocessing as τ_{algo} and excluded the execution time of embedding, translating, tuning, and network communication from τ_{algo}. Lastly, we recorded the minimum value of TTS with respect to τ as the computation time metric for each ρ.

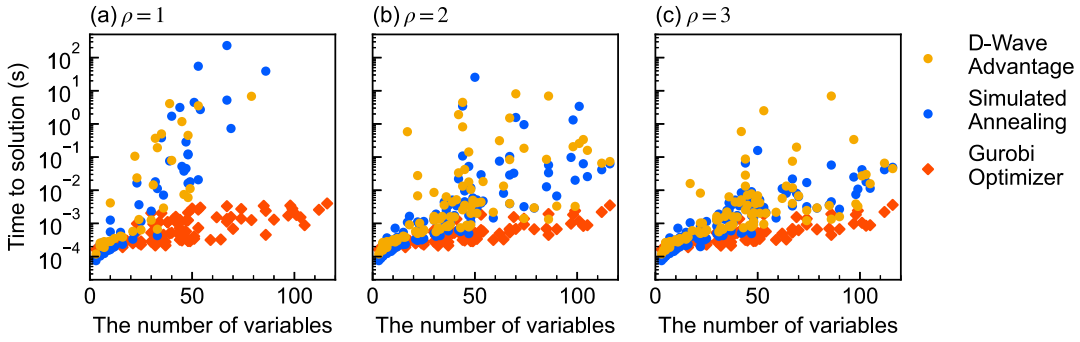


FIG. 7. Computation time scaling for the pathway-finding algorithms using the D-Wave Advantage system, simulated annealing, and the Gurobi Optimizer. The computation times of the algorithms using the D-Wave Advantage system and simulated annealing are the time-to-solution defined by Eq. (12), the time taken to find a solution whose cost is at most $\rho \times$ (the minimum cost) with a probability of at least 99%. Here, the cost tolerance ρ is (a) 1, (b) 2, and (c) 3, respectively. For the Gurobi Optimizer, the time-to-solution refers to the runtime it takes the solver to find a solution guaranteed to satisfy the cost tolerance requirement. Refer to the main text for additional details on the computing procedure.

We also computed the average runtime of the pathway-finding algorithm using the Gurobi Optimizer for comparison. For each cost tolerance ρ, the "MIPGap" parameter of the Gurobi Optimizer was set to $1 - 1/\rho$ so that the optimizer terminates when it is assured that the cost value of the current best solution is less than or equal to $\rho C_{\min}$. We recorded the average runtime of 1000 runs for each problem and ρ.

Figure 7 shows the scaling of computation times for the pathway-finding algorithms using the D-Wave Advantage system, SA, and the Gurobi Optimizer. When $\rho = 1$, the computation times for the D-Wave Advantage system and SA increase drastically as the number of variables N_x increases, compared to that for the Gurobi Optimizer. As ρ increases, the rates of increase in the computation times with respect to N_x for the D-Wave Advantage system and SA become slower. Consequently, these computation times approach that of the Gurobi Optimizer. In addition, the pathway-finding algorithm using the D-Wave Advantage system succeeded in finding optimal solutions ($\rho = 1$) for 34 out of the 100 benchmark problems, while the SA-based algorithm found optimal solutions for 43 problems. Approximate solutions with $\rho = 2$ were found for 96 problems when using the D-Wave Advantage system and all problems when using SA. Both methods found approximate solutions with $\rho = 3$ for all problems.

In the case of $\rho = 1$, the fast increasing rate of the TTS of SA with respect to N_x is likely attributed to the increase in the maximum penalty strength $M_{\max}$. The reasoning behind this statement is as follows. As discussed in Appendix B 1, the initial temperature in SA should be proportional to the largest absolute energy difference between any two states capable of direct transition, denoted by Δ. Such a high initial temperature enhances exploration in the solution space. The final temperature, in contrast, should be sufficiently low relative to the energy gap between the optimal and the second optimal solutions, denoted by δ. This ensures that the Boltzmann factor, and consequently the sampling probability, for the optimal solution(s) are sufficiently larger than those for the others. A larger difference between the initial and final temperatures, or equivalently between Δ and δ, leads to longer annealing time. In fact, according to an estimate for the computational complexity of SA [Eq. (B3) in Appendix B 1], the necessary annealing time exponentially increases with respect to Δ/δ. In the present Ising model, Δ is $O(M_{\max})$ as the penalty terms dominate energy barrier heights. Thus, the computational complexity is expected to depend on $M_{\max}/\delta$. This ratio increases as N_x increases, as shown in Fig. 8. This occurs because $M_{\max}$ is $O(N_x)$ as already shown in Fig. 6, whereas the cost difference δ may not necessarily depend on the problem size. Therefore, the increase of $M_{\max}$ is likely to cause the

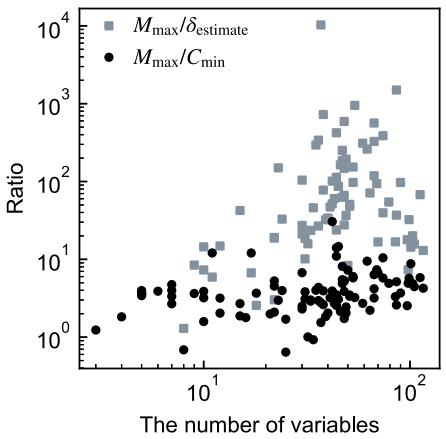


FIG. 8. The ratio of the maximum penalty strength $M_{\max}$ relative to (1) the cost difference δ between the optimal and the second optimal solutions and (2) the minimum cost $C_{\min}$ of feasible solutions. The maximum penalty strength $M_{\max}$ was computed for the optimized penalty and chain strengths for simulated annealing (SA) shown in Fig. 6. The minimum cost $C_{\min}$ was computed by the Gurobi Optimizer. The cost difference δ was estimated as follows: first, we gathered nonoptimal feasible solutions from the samples used to compute the time-to-solution for SA shown in Fig. 7; then, among these nonoptimal feasible solutions, we identified the minimum cost, denoted by C'; finally, we estimated δ as $\delta_{\text{estimate}} = C' - C_{\min}$. This estimate provides an upper bound of δ as C' is always greater than or equal to the true second minimum cost. Note that for some problems (especially with small sizes), all feasible solutions sampled by the SA-based algorithm are the optimal solution, thus $M_{\max}/\delta_{\text{estimate}}$ is not plotted.

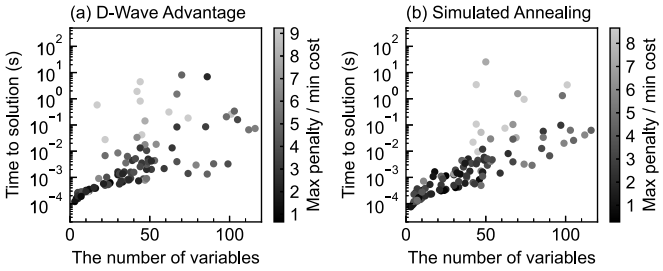


FIG. 9. The relationship between the time-to-solution and the ratio of the maximum penalty strength to the minimum cost. Panels depict the time-to-solution with cost tolerance $\rho = 2$ for the algorithms using (a) the D-Wave Advantage system and (b) simulated annealing, respectively. The time-to-solution data correspond to that in panel (b) in Fig. 7. Each data point is colored according to the value of the ratio of the maximum penalty strength to the minimum cost. For clarity, outlier values (colors) of the penalty-cost ratio were treated using the following standard method based on the inter-quartile range: values outside the interval $[Q_1 - 1.5IQR, Q_3 + 1.5IQR]$ were truncated, where Q_1 and Q_3 represent the first and third quartiles, respectively, and $IQR(:= Q_3 - Q_1)$ is the inter-quartile range; smaller outliers were set to the lower bound of the interval, while larger outliers were set to the upper bound.

increase of Δ/δ, leading to the rapid increase in the TTS of SA for $\rho = 1$.

In the cases of $\rho = 2$ and 3, the relatively slow increasing rate of the TTS of SA with respect to N_x can be explained by a theoretical expectation that the computational complexity of SA depends on $M_{\max}/[(\rho - 1)C_{\min}]$ instead of $M_{\max}/\delta$, contrasting the case of $\rho = 1$. This expectation is based on the following arguments. The sampling probability of the optimal solution(s) is desired to be sufficiently higher than those of nonallowable solutions whose costs are greater than $\rho C_{\min}$ but not necessarily than those of solutions within the cost tolerance range. This insight suggests that the final temperature should be sufficiently low compared to the cost difference between the optimal and the least-cost nonallowable solutions, $(\rho - 1)C_{\min}$, instead of δ in the case of $\rho = 1$. Therefore, based on a similar discussion presented in Appendix B 1, the necessary annealing time is expected to depend on $M_{\max}/[(\rho - 1)C_{\min}]$. In fact, as demonstrated in Fig. 9, problems with larger $M_{\max}/C_{\min}$ tend to have longer TTS for $\rho = 2$. Furthermore, $M_{\max}/C_{\min}$ does not exhibit a clear increasing trend with respect to N_x, as shown in Fig. 8. This observation is explained by the fact that the minimum cost $C_{\min}$ also tends to increase almost linearly as the problem size increases. Therefore, the reason for the moderate increase in the TTS of SA for $\rho = 2$ and 3 is likely that $M_{\max}/C_{\min}$ does not necessarily increase with respect to N_x.

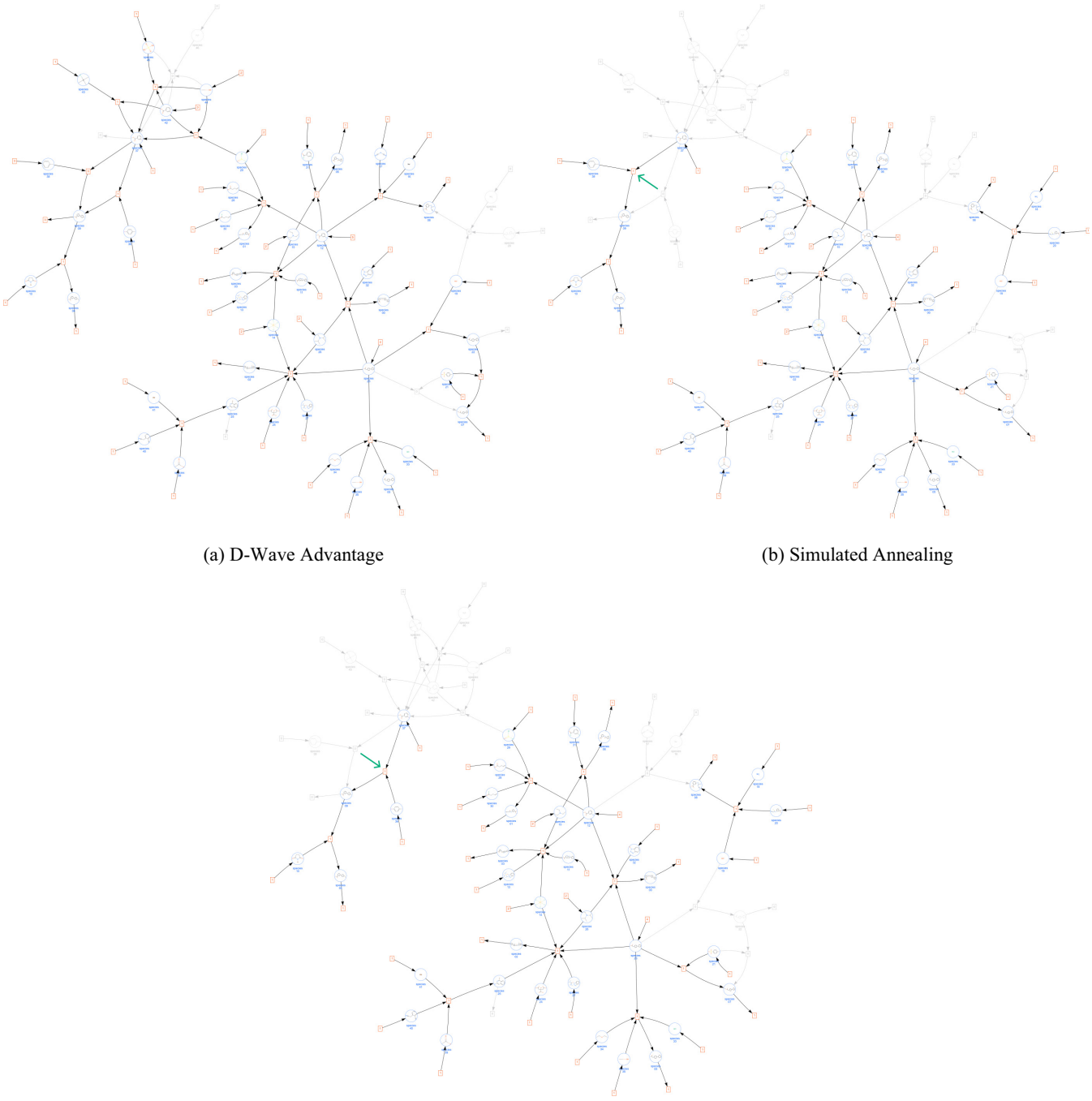
For the case of using the D-Wave Advantage system, the increasing trend of the TTS with respect to N_x can be partially explained in terms of $M_{\max}/\delta$ and $M_{\max}/C_{\min}$, in a way similar to the case of SA. According to an estimate for the computational complexity of QA [Eq. (B8) in Appendix B 2], the necessary annealing time required to find the optimal solution(s), i.e., for the case of $\rho = 1$, increases exponentially as $\log(1/\delta')$ increases, where δ' denotes the energy gap between the ground and the first excited states of the physical Ising model. When using the D-Wave Advantage system, the Ising Hamiltonian must be rescaled by the maximum of $|J_{ij}|$ and $|h_i|$ due to their finite ranges. This rescaling factor scales as $O(M_{\max})$. Therefore, the factor $1/\delta'$ in the necessary annealing time is $O(M_{\max}/\delta)$ for $\rho = 1$. Similar to the above arguments for SA with $\rho > 1$, the factor $1/\delta'$ can be replaced by $O(M_{\max}/C_{\min})$ for $\rho > 1$. Thus, $M_{\max}/\delta$ and $M_{\max}/C_{\min}$ are likely important factors determining the computational time scaling for QA as well as SA. In fact, Fig. 9 demonstrates that problems with longer TTS tend to have larger $M_{\max}/C_{\min}$ value, for problems with fewer than 50 variables.

However, despite the theoretical prediction that QA exhibits an advantage over SA in terms of the computational complexity scaling with respect to $M_{\max}/\delta$ or $M_{\max}/C_{\min}$ (see also Appendix B 2), the results presented in Fig. 7 do not demonstrate an apparent quantum advantage. This gap between theory and reality can be due to various factors, including device-specific characteristics of the D-Wave Advantage system, which are further discussed in the following paragraph.

Device-specific characteristics, such as the finite setting precision of J_{ij} and h_i and embedding overheads, must impact the TTS for the D-Wave Advantage system. First, the precision in setting J_{ij} and h_i is limited. Since the energy gap between the ground and the first excited states of the rescaled Ising Hamiltonian is $O(\delta/M_{\max})$, the energy gap can be smaller than the setting errors of J_{ij} and h_i for large-size problems, leading to poor performance of sampling the optimal solution(s). Second, minor embedding necessitates more physical qubits compared with the original logical Ising/QUBO formulation. As presented in Appendix D 1, the necessary number of physical qubits to represent a pathway-finding problem on the device increases almost quadratically as the number of logical binary variables increases. This embedding overhead contributes to the increased computation time of QA. Lastly, other hardware characteristics, such as thermal noises and readout errors, may also affect performance. These limitations specific to QA hardware devices may nullify the theoretical quantum advantage in the computational complexity scaling with respect to $M_{\max}/\delta$ or $M_{\max}/C_{\min}$.

3. *Computed synthesis pathways*

Finally, we compare synthesis pathways computed by the D-Wave Advantage system, SA, and the Gurobi Optimizer. Figs. 10 and 11 depict synthesis pathways computed for two representative problems shown in panels (d) and (e) of Figs. 4 and 5, that is, the 80th and 100th problems in ascending order of the number of integer variables, respectively. In this computation, we employed the order encoding method and the penalty and chain strength parameters optimized using the category grouping method. The annealing time for the D-Wave Advantage system and the number of sweeps for SA were set to their default values, i.e., 20 μs and 1000, respectively. We ran the proposed Ising-computing algorithm 1000 times for each problem and each Ising machine. The feasible synthesis pathways at the minimum cost among the 1000 solutions for each problem and each Ising machine are shown in panels (a) and (b) of Figs. 10 and 11. The synthesis



(a) D-Wave Advantage

(b) Simulated Annealing

(c) Gurobi Optimizer

FIG. 10. Synthesis pathways computed by (a) the D-Wave Advantage system, (b) simulated annealing, and (c) the Gurobi Optimizer. The problem solved is the 80th problem in ascending order of the number of integer variables [the same problem shown in panel (d) of Figs. 4 and 5]. Circles and squares represent chemical species and reactions, respectively. Each reaction's multiplicity is indicated inside the reaction node. Colored parts correspond to computed synthesis pathways, whereas light gray parts are not included in these pathways; in other words, the multiplicity of each light gray reaction is zero. The green arrows in the upper left area of panels (b) and (c) indicate different choices of alternative reactions in these two pathways. The costs of the three synthesis pathways are (a) 372.7, (b) 251.4, and (c) 250.0, respectively.

pathways computed by the Gurobi Optimizer shown in panel (c) of these figures are the exact optimal synthesis pathways for each problem.

The synthesis pathways computed by the D-Wave Advantage system and SA differ from the optimal synthesis pathways computed by the Gurobi Optimizer in the following two aspects. First, it has been observed that synthesis pathways computed by the D-Wave Advantage system and SA make suboptimal choices when alternative reactions are available, in certain cases. Pathways in panels (b) and (c) of

(a) D-Wave Advantage

(b) Simulated Annealing

(c) Gurobi Optimizer

FIG. 11. Synthesis pathways computed by (a) the D-Wave Advantage system, (b) simulated annealing, and (c) the Gurobi Optimizer. The problem solved is the 100th problem in ascending order of the number of integer variables [the same problem shown in panel (e) of Figs. 4 and 5]. The green frame in the upper left area of panel (a) encloses a subpathway that produces an unnecessary precursor. See the caption of Fig. 10 for the details of other symbols' meaning. The costs of the three synthesis pathways are (a) 552.1, (b) 471.4, and (c) 328.3, respectively.

Fig. 10 exemplify such a situation; these two pathways differ only in the choice of reactions indicated by the green arrows in the upper left area. Second, there are instances where synthesis pathways computed by the D-Wave Advantage system and SA include synthesis subpathways which produce precursors not used for synthesizing any target. Panel (a) of Fig. 11 clearly exemplifies such a situation; the subpathway enclosed by the green frame in the upper left area is disconnected with the other part and is not included in the optimal pathway; this subpathway produces a precursor not used for synthesizing any target. In general, such unnecessary subpathways are not always separated from main pathways and often overlap with them.

These differences between the optimal pathways and those computed by the D-Wave Advantage system and SA may be attributed to the large penalty strength issue and device-

specific noise and errors mentioned above. When the penalty strength is high, constraint satisfaction is more prioritized than cost minimization, often leading to feasible but suboptimal solutions. If the multiplicity of an unnecessary reaction in a chemical reaction network becomes positive due to device-specific noise or errors, the postprocessing methods may add additional reactions to make the returned solution feasible, which may form an unnecessary subpathway.

It may be possible to address the issue of unnecessary subpathways through the following postprocessing steps: (1) detecting disposed precursors; (2) finding subpathways that produces the excess precursors at maximum cost in the original pathway, using the proposed pathway-finding algorithm; (3) subtracting the subpathways—which are likely unnecessary—from the original pathway. Unnecessary subpathways may also be suppressed by imposing additional penalties for precursors disposal in the QUBO formulation. Introducing these treatments, however, necessitates careful consideration as precursor disposal does not necessarily indicate wasteful pathways and economical reactions may produce precursors as byproducts in some cases. Methods to remove or suppress unnecessary subpathways remain open for further investigation.

4. Other examinations

Additionally, we confirmed that the BO-based parameter tuning used in the present study contributes to performance improvement, outperforming random search (see Appendix D 2). An examination into the postprocessing methods showed that the postprocessing methods indeed enhance the algorithm's performance (see Appendix D 3).

V. CONCLUSIONS

We have developed an algorithm for finding optimal pathways in chemical reaction networks using Ising machines. A pathway-finding problem in a chemical reaction network is formulated as a combinatorial optimization problem that involves integer variables representing reaction multiplicity and mass balance constraints corresponding to the law of conservation of mass. The proposed algorithm translates a chemical pathway-finding problem into the ground-state finding problem of an Ising model and solves the translated problem using an Ising machine. In addition, the proposed algorithm applies two postprocessing methods to enhance the feasibility and optimality of solutions returned from the Ising machine.

We utilized Bayesian optimization to tune parameters determining penalty strengths for constraint violations. To enhance tuning performance, we have devised a dimensionality reduction technique for the parameter search space. In this technique, parameters are grouped together according to the associated constraint type, and all parameters in each group are constrained to take the same value. We systematically elucidated the effects of several different parameter grouping methods on tuning performance. We found that it is crucial to design parameter groups tailored according to the specific structure of problems for performance improvement. Most Ising/QUBO formulations do not consider such elaborate parameter grouping; instead, all penalty strengths of the same type are often consolidated into one group, like our "unified" method (for instance, see [24] and applications reviewed in [25]). Our findings may inform the future development of automatic parameter tuning methods for complex constrained problems in Ising computing.

Our performance evaluation and analysis of the proposed algorithm using the D-Wave Advantage system and simulated annealing indicate that: (1) the computation time required to find optimal pathways rapidly increases as the problem size increases, likely due to the growth of penalty strengths with respect to the problem size and hardware limitations of quantum annealing devices; (2) the scaling of the computation time can be moderated if a relative error in a cost value is allowable and if the maximum penalty strength to the minimum cost hardly increases with respect to the problem size. These findings provide insights into future directions of the algorithm development for chemical pathway-finding problems.

In the development of chemical pathway-finding algorithms utilizing the penalty-based Ising/QUBO formulation, it might be advisable for the algorithms to aim at finding approximate solutions within a relative cost error tolerance. In addition, appropriate applications may be problems with a moderate ratio of the maximum penalty strength to the minimum cost. For further performance improvement, the advancement of parameter tuning methods is needed. For instance, annealing schedule optimization [42,43] likely reduces the required annealing time, and transfer learning techniques [43] may reduce the overhead runtime for parameter tuning by leveraging tuning results for other problem instances. Hardware improvements, especially reducing embedding overheads, are also necessary. Furthermore, a method presented in [44] may mitigate the challenges in the penalty-based formulation by transforming quadratic penalty terms into linear terms using the Hubbard–Stratonovich transformation.

An alternative approach is to apply penalty-free quantum optimization algorithms, such as constrained quantum annealing [45] and quantum alternating operator ansatz (QAOA) [46], which may help bypass the penalty related issues. However, these algorithms introduce their own challenges, such as the need for a quantum processing unit capable of handling complex qubit interactions like XX and YY interactions and the difficulty of designing an effective driver Hamiltonian [47]. Despite these technical difficulties, advancements in algorithms along this direction hold promise for resolving the challenges identified in the present study when solving chemical pathway-finding problems using Ising machines.

Furthermore, extending the scope of the chemical pathway-finding problems targeted by the Ising computing framework is a future challenge. For example, a chemical pathway-finding problem may involve continuous variables representing real-valued reaction multiplicity in moles. Real-valued reaction multiplicity is necessary to be considered in synthesis planning when taking into account reaction yields because the optimal multiplicity may differ from a theoretical integer value based on stoichiometry. In such a case, the problem can be generally formulated as mixed integer programming. For another instance, a chemical pathway-finding

problem can be formulated as multiobjective optimization involving multiple objectives such as low monetary costs and low harmfulness [10]. Moreover, chemists are sometimes interested in enumerating near-optimal pathways or even all feasible pathways [12,15].

Several Ising/quantum-computing algorithms aim at solving mixed integer programming [48,49], multiobjective optimization [50], and exhaustive enumeration [51,52]. These algorithms may unlock further potential in chemical reaction network analysis using Ising machines. Note also that all chemical pathway-finding problems inherently involve mass balance constraints due to the law of conservation of mass; hence, our methods and findings related to penalty strength for mass balance constraints are likely broadly applicable across these problems.

We hope this study will serve as a starting point for future interdisciplinary technological advancements in chemical reaction network analysis using next-generation computers such as Ising machines and quantum computers.

ACKNOWLEDGMENT

This work was supported by JST, PRESTO Grant No. JPMJPR2018, Japan.

APPENDIX A: CHEMICAL REACTION NETWORK UNDERLYING A SYNTHESIS-PLANNING PROBLEM

A synthesis-planning problem to find the optimal synthesis pathway toward specified targets does not require consideration of all chemical reactions in chemical databases such as Reaxys. As illustrated in Fig. 12, chemical reactions can be classified into two types: (1) chemical reactions *relevant* to synthesizing the targets, which should be taken into account in the synthesis planning, and (2) other *irrelevant* chemical reactions. Relevant chemical reactions are recursively defined as follows:

(1) All chemical reactions that directly produce target species are relevant to the synthesis planning.

(2) If a chemical reaction produces any potential precursor for the target species, that is, a reactant of another relevant reaction, the chemical reaction is relevant to the synthesis planning.

We also define *relevant* chemical species as chemical species participating in the relevant chemical reactions. Such species can be either targets, precursors, or byproducts. In Fig. 12, species 0a and 0b are targets, species 1a–1e and 2a–2i are precursors, and species Πa is a byproduct. Here, the number in the label of a target or precursor node indicates the *depth* of the species node. The node depth is defined as the minimum number of synthesis steps from the species node to any target node. The depth of a byproduct node is undefined.

The above recursive definition establishes an algorithm for collecting chemical species and reactions relevant to the synthesis planning (Fig. 13). This algorithm significantly reduces the number of chemical reactions to be considered compared to that of the whole dataset.

In addition to the relevant chemical reactions enumerated by Algorithm 2 (Fig. 13), the inflow and outflow dummy reactions associated with the relevant species are also relevant to the synthesis planning. In synthesis planning, only commercially available species have their inflow reactions. All target species must have outflow reactions; others may also have outflow reactions representing disposal. Note that inflow reactions of purchasable relevant byproducts and outflow reactions of relevant precursors never produced by chemical reactions in the database (e.g., species 1c in Fig. 12) can be omitted because the purchase of byproducts and disposal of purchased substrates are not economical choices.

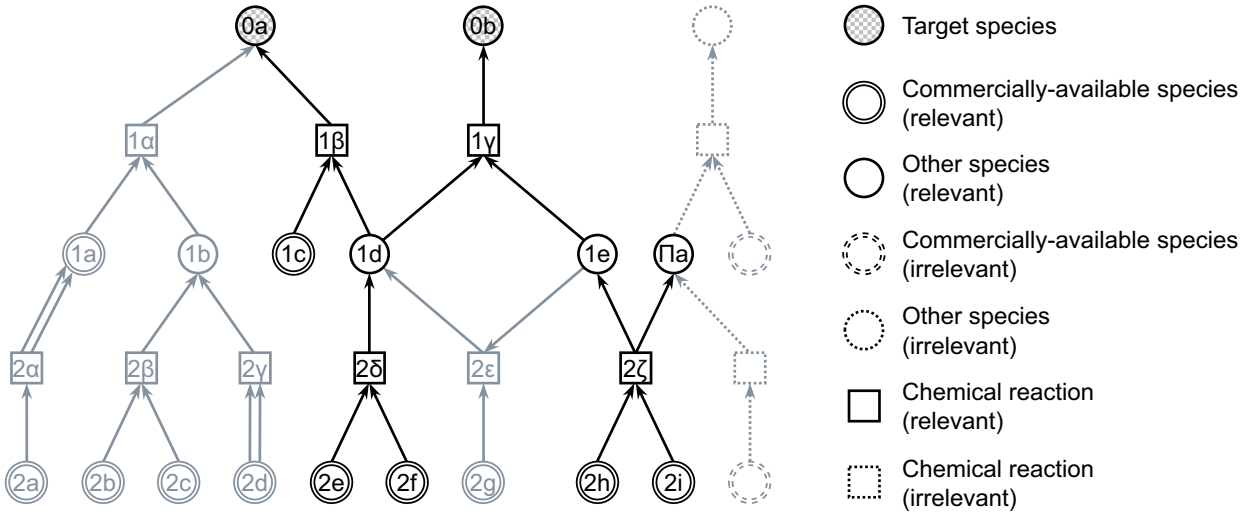


FIG. 12. Chemical reactions and species relevant and irrelevant to the synthesis planning of specified targets. Relevant chemical reactions (solid squares) are candidates for synthesis steps because they yield one of the targets or the potential precursors for the targets. On the other hand, irrelevant chemical reactions (dashed square) are never used for synthesizing the targets. Relevant chemical species (solid circles) are defined as those participating in relevant chemical reactions. Note that not all relevant chemical reactions are always used in chemical synthesis. For instance, four chemical reactions, namely 1β, 1γ, 2δ, and 2ζ, constitute a feasible synthesis pathway, indicated in dark black in the figure. Determining the optimal combination of the candidate reactions is central to the synthesis-planning problem.

Algorithm 2 Collect chemical species and reactions relevant to the synthesis planning of given targets

Input: $\mathscr{S}$, $\mathscr{R}$, $T \subset \mathscr{S}$, $d_{\max}$
Output: $S \subset \mathscr{S}$, $R \subset \mathscr{R}$
1: $R \leftarrow \emptyset$
2: $S \leftarrow T$
3: $S_0 \leftarrow T$
4: **for** $d = 1$ **to** $d_{\max}$ **do**
5: $\quad R_d \leftarrow \bigcup_{s \in S_{d-1}} \{r \in \mathscr{R} \setminus R \mid r \text{ produces } s\}$
6: $\quad R \leftarrow R \cup R_d$
7: $\quad S_d \leftarrow \bigcup_{r \in R_d} \{s \in \mathscr{S} \setminus S \mid s \text{ is a reactant of } r\}$
8: $\quad S \leftarrow S \cup S_d$
9: **end for**
10: $S \leftarrow S \cup \bigcup_{r \in R} \{s \in \mathscr{S} \setminus S \mid s \text{ is a product of } r\}$
11: **return** S, R

FIG. 13. An algorithm for collecting relevant chemical species and reactions. The input arguments are a set of chemical species $\mathscr{S}$, a set of chemical reactions $\mathscr{R}$, a set of targets T, and the maximum search depth $d_{\max}$. This algorithm iteratively enumerates relevant chemical species and reactions up to depth $d_{\max}$, then returns the sets S and R of the collected species and reactions. Note that the maximum recursive depth $d_{\max}$ does not exceed the number of all chemical reactions $|\mathscr{R}|$. For example, in Fig. 12, a square node with a label starting with integer d represents a relevant reaction in R_d at line 5 of the code, and a circle node with a label starting with integer d signifies a relevant species in S_d at lines 3 and 7. In addition, the byproduct species "Πa" is added to S at line 10.

APPENDIX B: ISING MACHINES AND MINOR EMBEDDING

Ising machines are dedicated computing devices for finding the ground state(s) of Ising models [16]. Major algorithms underlying Ising machines are simulated annealing [53] and quantum annealing [54]. This Appendix surveys the simulated and quantum annealing algorithms. Moreover, we also explain the minor embedding required for using Ising machines with limited spin connectivity.

1. Simulated annealing

Simulated annealing (SA) is a physics-inspired heuristic algorithm for optimization, proposed by Kirkpatrick *et al.* in 1983 [53]. SA simulates a thermal annealing process in a statistical mechanics system by a Markov chain Monte Carlo method. The system is designed such that its lowest energy state(s) correspond to the optimal solution(s) of an optimization problem to be solved. In the annealing process, the temperature of the system gradually decreases. If the cooling process is slow enough, the system is expected to remain in thermal equilibrium. Since the Gibbs distribution assigns a high probability to the lowest energy state(s) at sufficiently low temperatures, the system is likely to reach the lowest energy state(s) after annealing.

The annealing schedule, which dictates the rate of temperature decrease, determines the success probability of sampling the lowest energy state(s). Geman and Geman [55] proved that the sampling-probability distribution in SA converges to the uniform distribution on the lowest energy state(s) as $t \to \infty$ if the temperature decreases in time as

$$T(t) = \frac{N_\sigma \Delta}{\log t} \quad (t > t_0). \tag{B1}$$

Here, t denotes time (Monte Carlo step), $t_0 (> 1)$ is a constant, $T(t)$ is the temperature at time t, N_σ is the number of spin variables, and Δ is the largest absolute energy difference between any two states such that the system can directly transition between them in a single step of the Markov chain.

The computational complexity of SA with the inverse-logarithmic annealing schedule [Eq. (B1)] is estimated as follows. Assume that the probability distribution stays close to the Gibbs distribution during the annealing process. Let E_0 and E_1 denote the lowest and the second lowest energies, respectively. The temperature at final time τ needs to be low enough compared with the energy gap δ $(:= E_1 - E_0)$; this is because the lowest energy state(s) should have a sufficiently larger Boltzmann factor than the second lowest energy state(s),

$$\frac{\mathrm{e}^{-\frac{E_0}{T(\tau)}}}{\mathrm{e}^{-\frac{E_1}{T(\tau)}}} = \mathrm{e}^{\frac{\delta}{T(\tau)}} \gg 1$$

$$\Rightarrow \quad T(\tau) = \frac{N_\sigma \Delta}{\log \tau} \ll \delta. \tag{B2}$$

This suggests that the necessary annealing time in SA scales as

$$\tau \sim \exp\left(c \frac{N_\sigma \Delta}{\delta}\right), \tag{B3}$$

where c is a positive constant of $O(N_\sigma^0)$. This computational complexity is consistent with that implied by a theoretical result in [56]: the total-variation distance between the sampling probability distribution in SA with the annealing schedule given by Eq. (B1) and the Gibbs distribution at absolute zero temperature is upper bounded by $O((t/N_\sigma)^{-\delta/N_\sigma \Delta})$.

Note that most SA algorithms do not employ the inverse-logarithmic schedule due to impractical slowness in real-world applications. Practical implementations of SA utilize other fast cooling schedules, such as the geometric schedule $T(t) = T(0) r_T^t$ $(0 < r_T < 1)$, to find approximate optimal solutions. For instance, the SA algorithm implemented in the D-Wave Ocean Software [30] employs the geometric annealing schedule as default, which was also used in our benchmarking in Sec. IV. Thus, we reference the theoretical computational complexity given by Eq. (B3) merely as a coarse measure to help interpret the observed computation time scaling of SA with respect to the problem size in Sec. IV.

2. Quantum annealing

Quantum annealing (QA) is a heuristic quantum optimization algorithm inspired by SA. QA for Ising models was first proposed by Kadowaki and Nishimori in 1998 [54]. The QA algorithm utilizes the time evolution of a quantum system with the time-dependent Hamiltonian

$$\hat{H}_{\mathrm{QA}}(t) = -A(t) \sum_{i=1}^{N_\sigma} \hat{\sigma}_i^x - B(t) \left[\sum_{i=1}^{N_\sigma - 1} \sum_{j=i+1}^{N_\sigma} J_{ij} \hat{\sigma}_i^z \hat{\sigma}_j^z + \sum_{i=1}^{N_\sigma} h_i \hat{\sigma}_i^z \right], \tag{B4}$$

from time 0 to τ. Here, $A(t)$ and $B(t)$ are monotonic functions such that $A(0) = 1, B(0) = 0$ and $A(\tau) = 0, B(\tau) = 1$, and $\hat{\sigma}_i^x$ and $\hat{\sigma}_i^z$ are the Pauli x and z operators acting on the ith qubit, respectively. The system Hamiltonian varies with time from the first term (the transverse field Hamiltonian) to the second term (the quantum Ising Hamiltonian). In the computation, the quantum system is initialized to the ground state of the initial Hamiltonian $|+\rangle^{\otimes N_\sigma}$, where $|+\rangle = (|\uparrow\rangle + |\downarrow\rangle)/\sqrt{2}$, and $|\uparrow\rangle$ and $|\downarrow\rangle$ are the eigenstates of the Pauli z operator with eigenvalues 1 and -1, respectively. According to the quantum adiabatic theorem, if the system Hamiltonian varies slowly enough, the quantum state is expected to stay close to the instantaneous ground state. Therefore, after the adiabatic time evolution of the quantum system, it reaches a quantum state close to the ground state of the Ising Hamiltonian at time τ, allowing us to observe the lowest-energy spin configuration(s) with a high probability.

A convergence condition of QA is known for the following transverse-field Ising model,

$$\hat{H}'_{\mathrm{QA}}(t) = -\Gamma(t)\sum_{i=1}^{N_\sigma}\hat{\sigma}_i^x - \sum_{i=1}^{N_\sigma-1}\sum_{j=i+1}^{N_\sigma} J_{ij}\hat{\sigma}_i^z\hat{\sigma}_j^z - \sum_{i=1}^{N_\sigma} h_i\hat{\sigma}_i^z. \tag{B5}$$

This Hamiltonian is related to the Hamiltonian in Eq. (B4) as $\hat{H}'_{\mathrm{QA}}(t) = \hat{H}_{\mathrm{QA}}(t)/B(t)$. The function $\Gamma(t)$ $[= A(t)/B(t)]$ controls the magnitude of quantum fluctuation induced by the transverse field. Morita and Nishimori [57,58] proved that the excitation probability is bounded by an arbitrarily small constant ϵ at each time if $\Gamma(t)$ decreases in time as

$$\Gamma(t) = a(\sqrt{\epsilon}t + b)^{-\frac{1}{2N_\sigma - 1}} \quad (t > t_0), \tag{B6}$$

where a and b are constants of $O(N_\sigma^0)$ and t_0 is a positive constant.

The computational complexity of QA with the power-law annealing schedule [Eq. (B6)] is estimated as follows. At the final time τ, the perturbative transverse field needs to be small enough compared with the energy gap δ between the ground and the first excited states of the nonperturbed Ising model. This is because the contributions of excited states of the nonperturbed Ising model to the perturbed ground state should be sufficiently small,

$$\left|\frac{\langle k|\Gamma(\tau)\sum_i \hat{\sigma}_i^x|0\rangle}{E_k - E_0}\right| \propto \frac{(\sqrt{\epsilon}\tau + b)^{-\frac{1}{2N_\sigma-1}}}{E_k - E_0} \quad (k \neq 0)$$
$$\leqslant \frac{(\sqrt{\epsilon}\tau + b)^{-\frac{1}{2N_\sigma-1}}}{\delta} \ll 1, \tag{B7}$$

where $|k\rangle$ denotes the $(k+1)$th lowest energy eigenstate of the nonperturbed Ising model with energy E_k. This suggests that the necessary annealing time in QA scales as

$$\tau \sim \exp\left[(2N_\sigma - 1)\log\left(\frac{c'}{\delta}\right)\right], \tag{B8}$$

where c' is a positive constant. Therefore, QA exhibits an advantage over SA in terms of the computational complexity scaling with respect to δ.

Note that D-Wave quantum annealing devices are based on the Hamiltonian defined by Eq. (B4) and do not employ the power-law annealing schedule given by Eq. (B6). Thus, the theoretical computational complexity given by Eq. (B8) is merely a coarse measure to help interpret the observed computation time scaling of QA with respect to the problem size. In addition, due to minor embedding, the necessary number of qubits N_σ is often greater than the number of spins of an Ising model that one wants to solve. We explain the minor embedding in the next subsection.

3. Minor embedding

Due to physical hardware topology, some Ising machines, including D-Wave quantum annealing devices, can only solve Ising models with limited spin interactions. Therefore, one needs preprocessing to represent the *logical* Ising model one wants to solve by the *physical* Ising model implemented in hardware.

The mapping from the logical model to the physical model is called minor embedding [59]. In the embedding, a collection of multiple physical spins, termed a *chain*, represents a single logical spin. To ensure that the physical spins in a chain collectively behave as a single logical variable, a constraint that all spins in the chain take the same value is imposed. This constraint can be physically implemented by strong ferromagnetic interactions ($J_{ij} \gg 1$) between adjacent spins in the chain. The strength of the ferromagnetic interactions is called *chain strength*. In addition, the embedding ensures that if logical spin variables σ and σ' interact, at least one physical spin in the chain of σ and one spin in the chain of σ' interact.

Finding minor embedding is an NP-hard problem. However, several heuristic algorithms exist for finding embeddings [59,60]. The algorithm proposed in [59] is available in the `minorminer` package of the D-Wave Ocean Software [30].

Physical spins in a chain often take different values. In such a case, postprocessing is performed on a classical computer, assigning the most common physical spin value in the chain to the corresponding logical spin variable. The probability of such chain breaks mainly depends on the chain strength. Therefore, careful tuning of the chain strength is necessary to obtain feasible, low-cost solutions. Section III C describes chain strength tuning in detail.

APPENDIX C: BENCHMARK PROBLEM GENERATION

The benchmark problems used in Sec. IV were generated as follows.

First, we constructed a chemical reaction network from the USPTO dataset, a dataset of chemical reactions extracted from the United States patents published between 1976 and September 2016 [37]. The dataset contains invalid entries, such as species with invalid chemical structures and chemical reactions with no reactants/products; we removed these invalid reactions from the network. In addition, we ignored catalysis information. The numbers of chemical reactions and species in the whole network of the USPTO dataset are around 1.1 and 1.5 million, respectively.

Second, we defined a set of commercially available substrates and a set of target candidates. Instead of referring to actual chemical makers' catalogs, we generated a set of commercially available substrates and a set of target candidates based solely on the USPTO network structure as

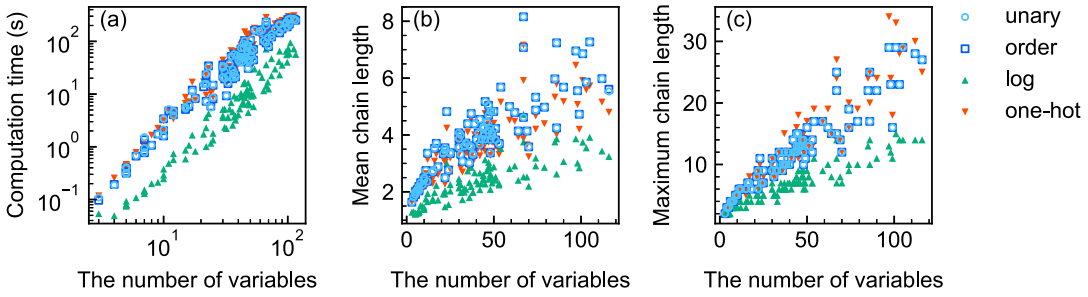


FIG. 14. Scaling of (a) the computation time required to find an embedding using the `minorminer.find_embedding` function, (b) the mean chain length, and (c) the maximum chain length, with respect to the number of integer variables.

follows: (1) We identified species with an indegree or outdegree of 20 or more as *hub*. We assumed that a hub species has an established standard way of purchasing or synthesizing it because it can be regarded as popular in chemistry. In other words, we assumed that the hub species are virtually commercially available and that finding synthesis pathways to them is unnecessary. Thus, we omitted deeper searches from hub species in Algorithm 2 (Fig. 13). This treatment prevents a synthesis planning problem from becoming too large and complex to solve by the D-Wave Advantage machine. (2) We classified nonhub species into *source* (with zero indegree), *sink* (with zero outdegree), and *nonterminal* (others). (3) We assumed that all source species are commercially available because source species are always used as starting materials in the dataset. (4) Nonterminal species may be either commercially available or not; we randomly assigned 25% of nonterminal species as commercially available substrates. (5) We designated all commercially unavailable species to target candidates. These target candidates comprise the sink species and the commercially unavailable nonterminal species.

Third, we randomly determined a unit purchase price of each commercially available species and a fixed execution cost for each chemical reaction. In our experiment, we used uniform distribution ranging from 1 to 10 for the random cost assignment.

Fourth, we randomly selected multiple targets for each synthesis planning: The number of targets was determined at uniformly random between 1 to 10; we then randomly selected a set of multiple species with high Tanimoto similarity [61] based on Morgan fingerprints [62,63] from the set of target candidates. Synthesis plans for multiple similar

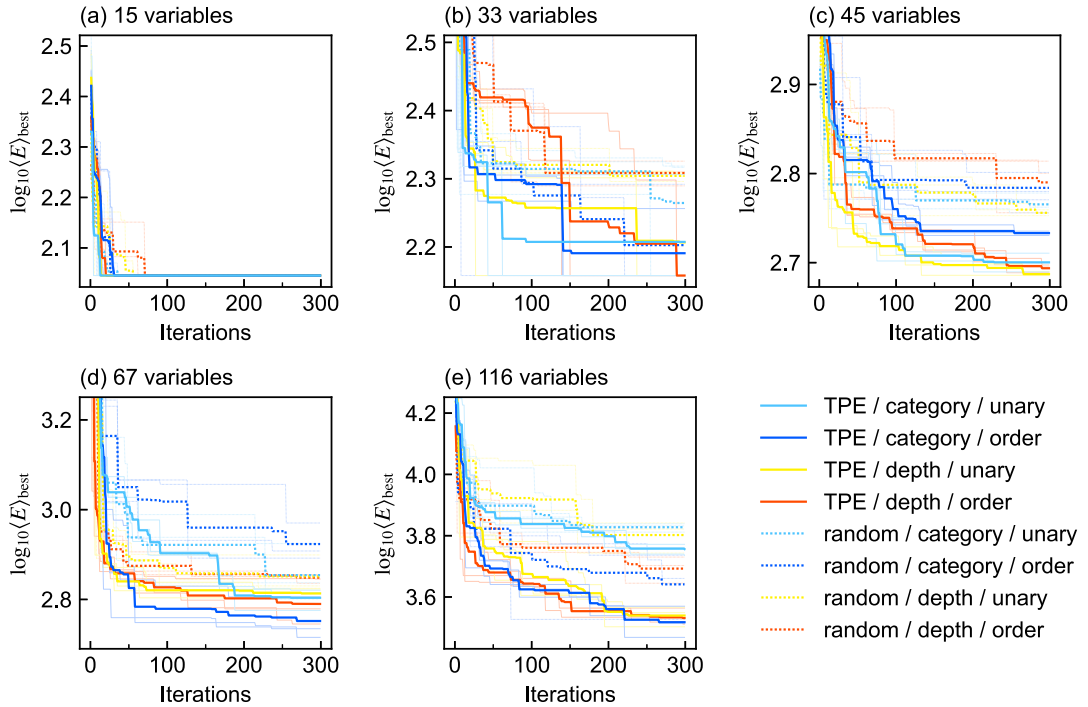


FIG. 15. Convergence plots of parameter tuning for the D-Wave Advantage case. The solid lines illustrate the convergence trends for Bayesian optimization with the multivariate tree-structured Parzen estimator (TPE), and the dotted lines illustrate those for random search. We performed parameter tuning three times for each problem, grouping, and encoding. The thick, dark lines represent the mean of the three tuning processes, while the thin, light lines depict each individual process to highlight variations in the tuning processes. The problems selected are the same as those in Fig. 4.

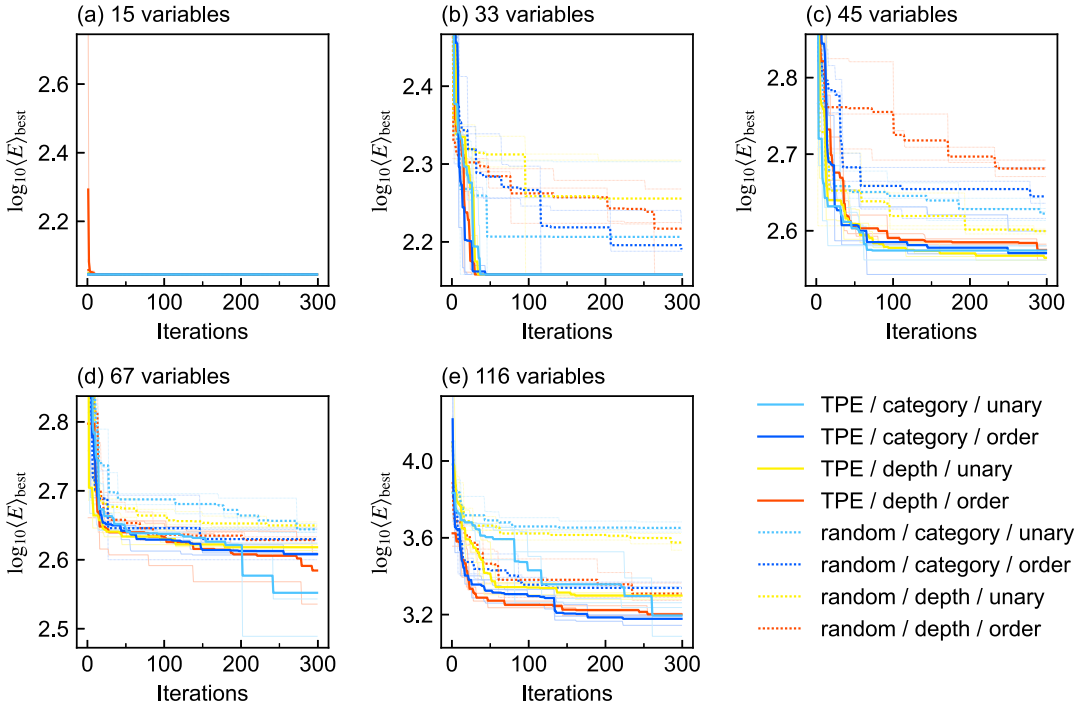


FIG. 16. Convergence plots of parameter tuning for the simulated annealing case. The notation is the same as that in Fig. 15.

species can save monetary costs by sharing precursors and reactions [11].

Fifth, we extracted a subnetwork underlying each synthesis planning from the USPTO chemical reaction network. Algorithm 2 (Fig. 13) identified the relevant chemical species and reactions for each synthesis. Here, the maximum search depth parameter $d_{\rm max}$ was set to $|\mathscr{R}|$, the upper bound of synthesis steps in a chemical reaction network with $|\mathscr{R}|$ reactions. Then, inflow dummy reactions of commercially available precursors were added to represent purchasing starting materials; outflow

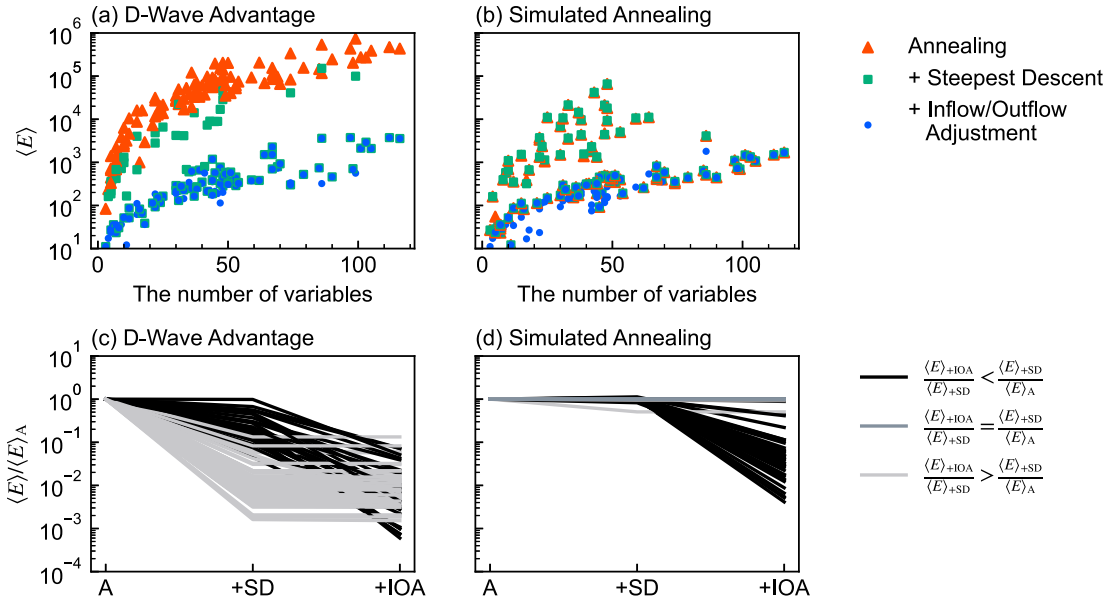


FIG. 17. Effectiveness of the postprocessing methods. Panels (a) and (b) depict the tuning score $\langle E\rangle$, defined by Eq. (11), for solutions at each step illustrated in Fig. 2, i.e., QUBO solutions, QUBO solutions (improved), and pathway solutions (improved). Red triangles represent the score of QUBO solutions sampled by an Ising machine $\langle E\rangle_{\rm A}$. Green squares denote the score of improved QUBO solutions processed by steepest descent $\langle E\rangle_{\rm +SD}$. Blue circles indicate the score of improved pathway solutions processed by both steepest descent and inflow/outflow adjustment $\langle E\rangle_{\rm +IOA}$. Panels (c) and (d) illustrate changes in the score relative to the score of raw solutions $\langle E\rangle_{\rm A}$. Points at position X (X = A, +SD, +IOA) represent the relative score $\langle E\rangle_X/\langle E\rangle_{\rm A}$. Each line corresponds to one problem and is colored according to the criteria shown in the lower legend to highlight the type of improvement process.

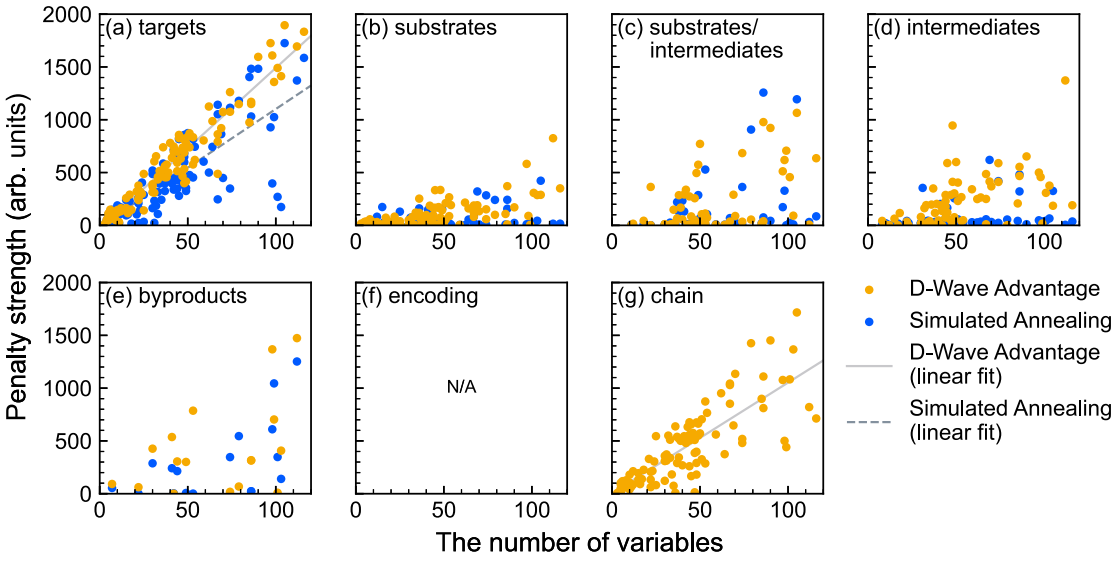


FIG. 18. Scaling of the tuned penalty and chain strength parameter values with respect to the number of integer variables in the case of using the unary encoding and category grouping methods. The notation is the same as that in Fig. 6 in Sec. IV B. Note that panel (f) is blank because the unary encoding method does not require any penalty for encoding. The R^2 values are 0.92 [panel (a) for D-Wave Advantage], 0.63 [panel (a) for simulated annealing], and 0.65 [panel (g) for D-Wave Advantage], respectively.

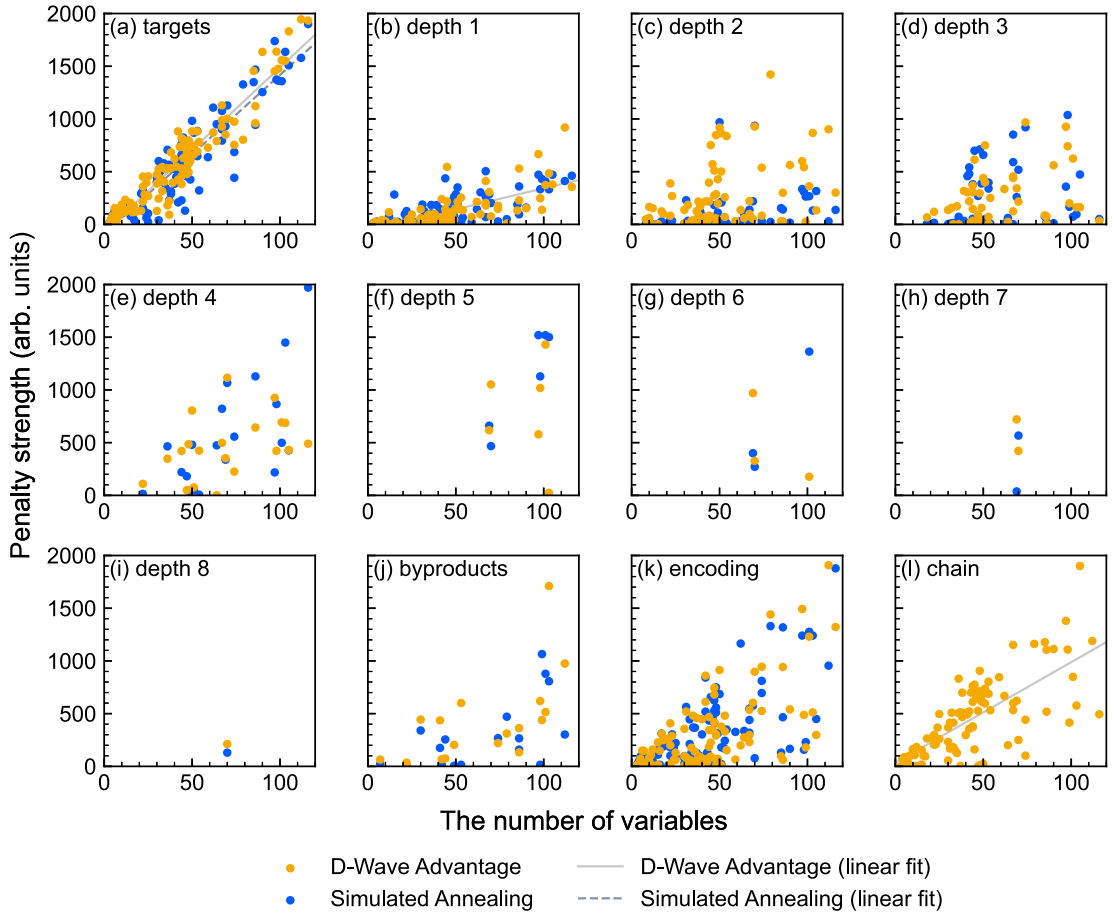


FIG. 19. Scaling of the tuned penalty and chain strength parameter values with respect to the number of integer variables in the case of using the order encoding and depth grouping methods. Panels (a)–(j) depict the penalty strength parameters for mass balance constraints of species in each group, panel (k) shows the penalty strength parameter for order encoding, and panel (l) shows the chain strength for embedding. Linear fits are plotted for datasets with a coefficient of determination (R^2) greater than 0.5, excluding those with fewer than ten data points. The R^2 values are 0.91 [panel (a) for D-Wave Advantage], 0.86 [panel (a) for simulated annealing], 0.52 [panel (b) for D-Wave Advantage], and 0.54 [panel (l) for D-Wave Advantage], respectively.

dummy reactions of species other than the source species (with zero indegree) were added to represent shipments of target chemicals or disposals of synthesized byproducts.

Finally, we created optimization problems of synthesis planning based on the random costs and the underlying chemical reaction networks. In all problems, the multiplicity of the outflow reaction of each target was fixed to 1 by setting the lower and upper bounds to 1, and the lower and upper bounds of the multiplicity of other reactions were set to 0 and 5, respectively.

Note that problems with disconnected relevant chemical reaction networks were not adopted as the benchmark problems because such problems can easily be decomposed into smaller problems. In addition, problems that are infeasible or unable to be embedded into the D-Wave Advantage machine were also not adopted as the benchmark problems.

APPENDIX D: SUPPLEMENTAL PERFORMANCE EVALUATION

1. Embedding performance

We calculated embeddings of the benchmark QUBO problems into the D-Wave Advantage Pegasus topology QPU using the `minorminer.find_embedding` function from the D-Wave Ocean Software [30]. For each problem, we performed the calculation ten times and selected the embedding with the shortest maximum and mean chain length from those obtained.

Figure 14 illustrates the scaling of (a) the computation time required to find an embedding, (b) the mean chain length, and (c) the maximum chain length, with respect to the number of integer variables. The computation time of the `minorminer.find_embedding` function tends to increase nearly quadratically as the problem size increases. The mean and maximum chain lengths exhibit roughly linear scaling, leading to quadratic scaling of the number of physical qubits with respect to the number of logical qubits. Since the log encoding method requires fewer binary variables than the other encoding methods, both the computation time and chain length for the log encoding method are less than those for the others.

2. Tuning performance

To evaluate the effectiveness of Bayesian optimization in penalty strength tuning, we compared its results with those of

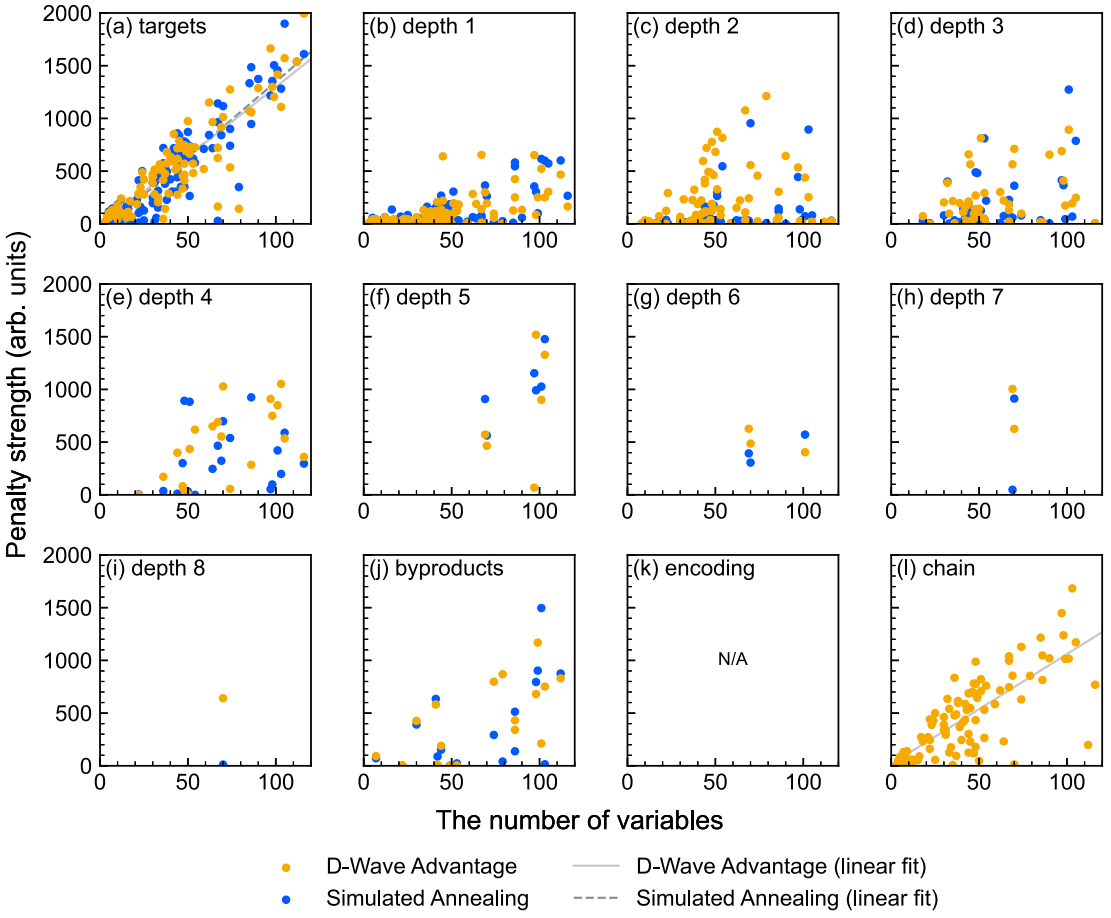


FIG. 20. Scaling of the tuned penalty and chain strength parameter values with respect to the number of integer variables in the case of using the unary encoding and depth grouping methods. The notation is the same as that in Fig. 19. Note that panel (k) is blank because the unary encoding method does not require any penalty for encoding. The R^2 values are 0.79 [panel (a) for D-Wave Advantage], 0.82 [panel (a) for simulated annealing], and 0.61 [panel (l) for D-Wave Advantage], respectively.

random search. The results for the D-Wave Advantage system and simulated annealing are depicted in Figs. 15 and 16, respectively. For the smallest size problem, both Bayesian optimization and random search converged rapidly to the same value. However, for larger problems, Bayesian optimization converged to a better value than random search when using the same encoding and grouping methods.

3. Postprocessing performance

To evaluate the effectiveness of the two postprocessing methods, we analyzed changes in the tuning score function $\langle E \rangle$, defined by Eq. (11), during postprocessing. The results are shown in Fig. 17. In the D-Wave Advantage case, the postprocessing enhances the $\langle E \rangle$ score by several orders of magnitude. The score improvement was primarily due to steepest descent (SD) or inflow/outflow adjustment (IOA). On the other hand, in the simulated annealing case, SD does not contribute to score improvement, whereas IOA does enhance solution quality for some problems.

4. Penalty strength scaling (supplemental)

Figure 6 in Sec. IV B and Figs. 18–20 in this Appendix demonstrate that the optimized penalty strength for the mass balance constraints of the target species tends to be larger than that of the others and exhibits a strong linear correlation with the number of integer variables when using the unary/order encoding and depth/category grouping methods.

[1] Elsevier, Reaxys (2023), https://www.reaxys.com.

[2] American Chemical Society, CAS SciFinder^n (2022), https://www.cas.org/solutions/cas-scifinder-discovery-platform/cas-scifinder.

[3] P.-M. Jacob, T. Lan, J. M. Goodman, and A. A. Lapkin, A possible extension to the RInChI as a means of providing machine readable process data, J. Cheminform. **9**, 23 (2017).

[4] J. M. Granda, L. Donina, V. Dragone, D.-L. Long, and L. Cronin, Controlling an organic synthesis robot with machine learning to search for new reactivity, Nature (London) **559**, 377 (2018).

[5] C. W. Coley, D. A. Thomas III, J. A. M. Lummiss, J. N. Jaworski, C. P. Breen, V. Schultz, T. Hart, J. S. Fishman, L. Rogers, H. Gao *et al.*, A robotic platform for flow synthesis of organic compounds informed by AI planning, Science **365**, eaax1566 (2019).

[6] M. H. S. Segler, M. Preuss, and M. P. Waller, Planning chemical syntheses with deep neural networks and symbolic AI, Nature (London) **555**, 604 (2018).

[7] N. Tsuji, P. Sidorov, C. Zhu, Y. Nagata, T. Gimadiev, A. Varnek, and B. List, Predicting highly enantioselective catalysts using tunable fragment descriptors, Angew. Chem. Int. Ed. **62**, e202218659 (2023).

[8] S. Maeda and Y. Harabuchi, Exploring paths of chemical transformations in molecular and periodic systems: An approach utilizing force, WIREs Comput. Mol. Sci. **11**, e1538 (2021).

[9] O. N. Temkin, A. V. Zeigarnik, and D. Bonchev, *Chemical Reaction Networks: A Graph-Theoretical Approach* (CRC Press, Boca Raton, FL, 1996).

[10] S. Szymkuć, E. P. Gajewska, T. Klucznik, K. Molga, P. Dittwald, M. Startek, M. Bajczyk, and B. A. Grzybowski, Computer-assisted synthetic planning: The end of the beginning, Angew. Chem. Int. Ed. **55**, 5904 (2016).

[11] M. Kowalik, C. M. Gothard, A. M. Drews, N. A. Gothard, A. Weckiewicz, P. E. Fuller, B. A. Grzybowski, and K. J. M. Bishop, Parallel optimization of synthetic pathways within the network of organic chemistry, Angew. Chem. Int. Ed. **51**, 7928 (2012).

[12] R. Shibukawa, S. Ishida, K. Yoshizoe, K. Wasa, K. Takasu, Y. Okuno, K. Terayama, and K. Tsuda, CompRet: A comprehensive recommendation framework for chemical synthesis planning with algorithmic enumeration, J. Cheminform. **12**, 52 (2020).

[13] H. Gao, J. Pauphilet, T. J. Struble, C. W. Coley, and K. F. Jensen, Direct optimization across computer-generated reaction networks balances materials use and feasibility of synthesis plans for molecule libraries, J. Chem. Inf. Model. **61**, 493 (2021).

[14] J. L. Andersen, C. Flamm, D. Merkle, and P. F. Stadler, Maximizing output and recognizing autocatalysis in chemical reaction networks is NP-complete, J. Sys. Chem. **3**, 1 (2012).

[15] J. L. Andersen, C. Flamm, D. Merkle, and P. F. Stadler, Chemical transformation motifs—modelling pathways as integer hyperflows, IEEE/ACM Trans. Comput. Biol. Bioinfor. **16**, 510 (2019).

[16] N. Mohseni, P. L. McMahon, and T. Byrnes, Ising machines as hardware solvers of combinatorial optimization problems, Nat. Rev. Phys. **4**, 363 (2022).

[17] M. W. Johnson, M. H. S. Amin, S. Gildert, T. Lanting, F. Hamze, N. G. Dickson, R. Harris, A. J. Berkley, J. Johansson, P. I. Bunyk *et al.*, Quantum annealing with manufactured spins, Nature (London) **473**, 194 (2011).

[18] C. McGeoch and P. Farré, Advantage Processor Overview, Tech. Rep. (D-Wave Systems Inc., 2022), https://www.dwavesys.com/media/3xvdipcn/14-1058a-a_advantage_processor_overview.pdf.

[19] M. Yamaoka, C. Yoshimura, M. Hayashi, T. Okuyama, H. Aoki, and H. Mizuno, 20k-spin Ising chip for combinational optimization problem with CMOS annealing, in *2015 IEEE International Solid-State Circuits Conference (ISSCC) Digest of Technical Papers* (IEEE, 2015), pp. 1–3.

[20] T. Okuyama, T. Sonobe, K. Kawarabayashi, and M. Yamaoka, Binary optimization by momentum annealing, Phys. Rev. E **100**, 012111 (2019).

[21] S. Matsubara, M. Takatsu, T. Miyazawa, T. Shibasaki, Y. Watanabe, K. Takemoto, and H. Tamura, Digital annealer for high-speed solving of combinatorial optimization problems and its applications, in *2020 25th Asia and South Pacific Design Automation Conference (ASP-DAC)* (IEEE, 2020), pp. 667–672.

[22] K. Yamamoto, K. Ando, N. Mertig, T. Takemoto, M. Yamaoka, H. Teramoto, A. Sakai, S. Takamaeda-Yamazaki, and M. Motomura, STATICA: A 512-spin 0.25 M-weight full-digital annealing processor with a near-memory

all-spin-updates-at-once architecture for combinatorial optimization with complete spin-spin interactions, in *2020 IEEE International Solid-State Circuits Conference (ISSCC)* (IEEE, 2020), pp. 138–140.

[23] K. Kawamura, J. Yu, D. Okonogi, S. Jimbo, G. Inoue, A. Hyodo, Á. L. García-Arias, K. Ando, B. H. Fukushima-Kimura, R. Yasudo, T. Van Chu, and M. Motomura, Amorphica: 4-replica 512 fully connected spin 336 MHz metamorphic annealer with programmable optimization strategy and compressed-spin-transfer multi-chip extension, in *2023 IEEE International Solid-State Circuits Conference (ISSCC)* (IEEE, 2023), pp. 42–44.

[24] A. Lucas, Ising formulations of many NP problems, Front. Phys. **2**, 5 (2014).

[25] S. Yarkoni, E. Raponi, T. Bäck, and S. Schmitt, Quantum annealing for industry applications: Introduction and review, Rep. Prog. Phys. **85**, 104001 (2022).

[26] D. A. Patterson and J. L. Hennessy, *Computer Organization and Design MIPS Edition: The Hardware/Software Interface*, 6th ed. (Morgan Kaufmann, Kidlington, UK; Waltham, MA, 2021).

[27] C. Chaouiya, Petri net modelling of biological networks, Brief. Bioinform. **8**, 210 (2007).

[28] J. Horiuti, Stoichiometric number and universal kinetic law in the neighbourhood of equilibrium. I, Proc. Jpn. Acad. **29**, 160 (1953).

[29] M. Zaman, K. Tanahashi, and S. Tanaka, PyQUBO: Python library for mapping combinatorial optimization problems to QUBO form, IEEE Trans. Comput. **71**, 838 (2022).

[30] D-Wave Systems Inc., D-Wave Ocean Software (2022), https://docs.ocean.dwavesys.com/en/stable/.

[31] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, Optuna: A next-generation hyperparameter optimization framework, in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Association for Computing Machinery, 2019), pp. 2623–2631.

[32] Preferred Networks, Inc., Optuna (2022), https://optuna.org.

[33] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, Algorithms for hyper-parameter optimization, in *Advances in Neural Information Processing Systems* (Curran Associates, Inc., 2011), Vol. 24.

[34] L. Yang and A. Shami, On hyperparameter optimization of machine learning algorithms: Theory and practice, Neurocomput. **415**, 295 (2020).

[35] M. Ayodele, Comparing the digital annealer with classical evolutionary algorithm arXiv:2205.13586.

[36] S. T. Goh, J. Bo, S. Gopalakrishnan, and H. C. Lau, Techniques to enhance a QUBO solver for permutation-based combinatorial optimization, in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO '22 (Association for Computing Machinery, 2022), pp. 2223–2231.

[37] D. Lowe, Chemical reactions from US patents (1976-Sep2016) (2017), https://figshare.com/articles/dataset/Chemical_reactions_from_US_patents_1976-Sep2016_/5104873.

[38] K. Boothby, P. Bunyk, J. Raymond, and A. Roy, Next-Generation Topology of D-Wave Quantum Processors, Tech. Rep. (D-Wave Systems Inc., 2019), https://www.dwavesys.com/media/jwwj5z3z/14-1026a-c_next-generation-topology-of-dw-quantum-processors.pdf.

[39] A. Meurer, C. P. Smith, M. Paprocki, O. Čertík, S. B. Kirpichev, M. Rocklin, A. Kumar, S. Ivanov, J. K. Moore, S. Singh *et al.*, SymPy: Symbolic computing in Python, PeerJ Comp. Sci. **3**, e103 (2017).

[40] Gurobi Optimization, LLC., Gurobi Optimizer (2022), https://www.gurobi.com.

[41] K. Tamura, T. Shirai, H. Katsura, S. Tanaka, and N. Togawa, Performance comparison of typical binary-integer encodings in an Ising machine, IEEE Access **9**, 81032 (2021).

[42] D. Herr, E. Brown, B. Heim, M. Könz, G. Mazzola, and M. Troyer, Optimizing schedules for quantum annealing, arXiv:1705.00420.

[43] Y.-Q. Chen, Y. Chen, C.-K. Lee, S. Zhang, and C.-Y. Hsieh, Optimizing quantum annealing schedules with Monte Carlo tree search enhanced with neural networks, Nat. Mach. Intell. **4**, 269 (2022).

[44] M. Ohzeki, Breaking limitation of quantum annealer in solving optimization problems under constraints, Sci. Rep. **10**, 3126 (2020).

[45] I. Hen and F. M. Spedalieri, Quantum annealing for constrained optimization, Phys. Rev. Appl. **5**, 034007 (2016).

[46] S. Hadfield, Z. Wang, B. O'Gorman, E. G. Rieffel, D. Venturelli, and R. Biswas, From the quantum approximate optimization algorithm to a quantum alternating operator ansatz, Algorithms **12**, 34 (2019).

[47] H. Leipold and F. M. Spedalieri, Constructing driver Hamiltonians for optimization problems with linear constraints, Quantum Sci. Technol. **7**, 015013 (2022).

[48] Z. Zhao, L. Fan, and Z. Han, Hybrid quantum Benders' decomposition for mixed-integer linear programming, in *2022 IEEE Wireless Communications and Networking Conference (WCNC)* (IEEE, 2022), pp. 2536–2540.

[49] A. Ajagekar, K. A. Hamoud, and F. You, Hybrid classical-quantum optimization techniques for solving mixed-integer programming problems in production scheduling, IEEE Trans. Quantum Engn. **3**, 1 (2022).

[50] B. Barán and M. Villagra, Multiobjective optimization in a quantum adiabatic computer, Electron. Notes Theor. Comput. Sci. **329**, 27 (2016).

[51] V. Kumar, C. Tomlin, C. Nehrkorn, D. O'Malley, and J. Dulny III, Achieving fair sampling in quantum annealing, arXiv:2007.08487.

[52] Y. Mizuno and T. Komatsuzaki, A note on enumeration by fair sampling, arXiv:2104.01941.

[53] S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, Optimization by simulated annealing, Science **220**, 671 (1983).

[54] T. Kadowaki and H. Nishimori, Quantum annealing in the transverse Ising model, Phys. Rev. E **58**, 5355 (1998).

[55] S. Geman and D. Geman, Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images, IEEE Trans. Pattern Anal. Mach. Intell. **PAMI-6**, 721 (1984).

[56] D. Mitra, F. Romeo, and A. Sangiovanni-Vincentelli, Convergence and finite-time behavior of simulated annealing, Adv. Appl. Probab. **18**, 747 (1986).

[57] S. Morita and H. Nishimori, Convergence of quantum annealing with real-time Schrödinger dynamics, J. Phys. Soc. Jpn. **76**, 064002 (2007).

[58] S. Morita and H. Nishimori, Mathematical foundation of quantum annealing, J. Math. Phys. **49**, 125210 (2008).

[59] J. Cai, W. G. Macready, and A. Roy, A practical heuristic for finding graph minors, arXiv:1406.2741.

[60] Y. Sugie, Y. Yoshida, N. Mertig, T. Takemoto, H. Teramoto, A. Nakamura, I. Takigawa, S. Minato, M. Yamaoka, and T. Komatsuzaki, Minor-embedding heuristics for large-scale annealing processors with sparse hardware graphs of up to 102,400 nodes, Soft Comput. **25**, 1731 (2021).

[61] J. Gasteiger and T. Engel, *Chemoinfomatics: A Textbook* (Wiley-VCH, Weinheim, Germany, 2003).

[62] D. Rogers and M. Hahn, Extended-connectivity fingerprints, J. Chem. Inf. Model. **50**, 742 (2010).

[63] G. Landrum, Morgan Fingerprints (Circular Fingerprints) (2023), https://www.rdkit.org/docs/GettingStartedInPython.html#morgan-fingerprints-circular-fingerprints.