Reinforcement Quantum Annealing: A Quantum-Assisted Learning Automata Approach

Ramin Ayanzadeh[1,*]**, Milton Halem**[1]**, and Tim Finin**[1]

[1]Department of Computer Science and Electrical Engineering, University of Maryland, Baltimore County, Baltimore, MD 21250, United States
[*]ayanzadeh@umbc.edu

ABSTRACT

We introduce the reinforcement quantum annealing (RQA) scheme in which an intelligent agent interacts with a quantum annealer that plays the stochastic environment role of learning automata and tries to iteratively find better Ising Hamiltonians for the given problem of interest. As a proof-of-concept, we propose a novel approach for reducing the NP-complete problem of Boolean satisfiability (SAT) to minimizing Ising Hamiltonians and show how to apply the RQA for increasing the probability of finding the global optimum. Our experimental results on two different benchmark SAT problems (namely factoring pseudo-prime numbers and random SAT with phase transitions), using a D-Wave 2000Q quantum processor, demonstrated that RQA finds notably better solutions with fewer samples, compared to state-of-the-art techniques in the realm of quantum annealing.

Introduction

Quantum artificial intelligence and quantum machine learning are emerging fields that leverage quantum information processing to address a certain types of problems, which are intractable in the realm of classical computing[1–3]. There are several models for the physical realization of quantum computers[4]. Among the quantum computing models, adiabatic quantum computers are currently more readily available at user sites due to recent advancements in commercializing programmable quantum annealers by D-Wave Systems[5].

Quantum annealing is a meta-heuristic that uses Quantum-mechanical fluctuations to address discrete optimization problems (i.e., combinatorial optimization problems), which are intractable in the realm of classical computing[6–9]. Quantum annealers are a type of adiabatic quantum computer that provides a hardware implementation for minimizing Hamiltonians whose ground states represent optimum solutions of the original problems of interest. The D-Wave quantum annealer is a programmable (and commercialized) Ising processing unit (IPU) that can minimize the transverse Ising Hamiltonian or its equivalent quadratic unconstrained binary optimization (QUBO) form[5]. More precisely, the D-Wave quantum annealer receives coefficients of an Ising Hamiltonian (here $\mathbf{h}$ and J) as an executable quantum machine instruction (QMI) and returns the vector $\mathbf{z}$ that minimizes the following energy function:

$$E_{\text{Ising}}(\mathbf{z}) = \sum_{i=1}^{N} \mathbf{h}_i \mathbf{z}_i + \sum_{i=1}^{N} \sum_{j=i+1}^{N} J_{ij} \mathbf{z}_i \mathbf{z}_j, \tag{1}$$

where N denotes the number of quantum bits (qubits) and $\mathbf{z}_i \in \{-1, +1\}$. To solve a problem on a D-Wave quantum annealer, one needs to define an Ising Hamiltonian, shown in Eq. (1), whose ground state represents a solution for the original problem of interest[10].

The current generation of the D-Wave quantum annealer (the Chimera architecture) includes more than 2,000 qubits and about 6,000 couplers, while the next generation (the Pegasus topology of the Advantage) will include more than 5,000 qubits and about 40,000 couplers[11]. Recent studies have revealed the potential of quantum annealers (namely the D-Wave quantum processors) to address certain classes of real-world problems that are intractable in the realm of classical computing[12]—including, but not limited to, planning[13], scheduling[14, 15], discrete optimization problems[16], constraint satisfaction problems[17], Boolean satisfiability[18, 19], matrix factorization[20], cryptography[21], fault detection and system diagnosis[22], compressive sensing[23], control of automated vehicles[24] and protein folding[25]. In addition, by sampling from high-dimensional probability distributions, one can use the D-Wave quantum annealers for many applications in artificial intelligence, machine learning and signal processing[12, 26, 27].

Beside all aforementioned applications, the D-Wave quantum annealer architecture has limitations that not only restrict the process of mapping problems into an executable QMI (namely the sparse connectivity) but also lower the quality of results. Reducing a problem to an Ising Hamiltonian that represents the solution of the problem in its ground state does not

guarantee that executing the corresponding QMI on a D-Wave quantum processing unit (QPU) will attain the optimum solution. For a given QMI, the D-Wave QPU draws samples from a problem-dependent pseudo-Boltzmann distribution at cryogenic temperatures[12]. The energy value of samples from the D-Wave QPU follow a Gaussian distribution. Thus, when we increase the number of reads/samples, we expect that the average parameter in the corresponding Gaussian distribution to approach the ground state energy of the corresponding Ising Hamiltonian—i.e., the probability of finding the global minimum approaches one. There are several drawbacks, nevertheless, that prevent quantum annealers from attaining a global minimum—including, but not limited to, confined anneal time, coefficients' range and precision limitations, noise, and decoherence.

From a quantum computing perspective, an adiabatic quantum computer needs to minimize a non-stoquastic Hamiltonian in order to be universal (which would make them equivalent to gate models); nevertheless, the D-Wave QPU minimizes an Ising Hamiltonian which is stoquastic[9, 28]. Since coupling every qubit to every other qubit in a quantum annealer is impractical, the D-Wave QPU has a sparse structure/architecture. Hence, we entangle multiple qubits to represent virtual qubits with higher connectivity. Chaining physical qubits substantially reduces the capacity of QPUs—e.g., 2,048 qubits in the Chimera architecture is equivalent to a clique of size 64. It is possible to implicitly leverage the capacity of the current D-Wave QPUs[29], albeit executing multiple QMIs. In addition, virtual qubits are vulnerable to breaking—the longer the chains, the higher the probability they break during the annealing process. Although we can remediate broken chains by applying postprocessing methods on classical computers (e.g., voting among the physical qubits on a chain), some chains break because they represent a state with lower energy.

The required anneal time in a quantum annealer to keep the process adiabatic has a reverse exponential relation to the energy gap between the ground state (global minimum) and the first excited state (a state right above the global minimum)[9]. In the current generation of the D-Wave QPU, $\mathbf{h}_i \in [-2, +2]$ and $J_{ij} \in [-1, +1]$. Thus, one needs to scale the resulting Ising model, Eq. (1), by dividing all coefficients with a large-enough positive number to satisfy the QPU hardware constraints. However, scaling the Ising model reduces the energy gap between the ground and the first excited states. As a result, the required annealing time can quickly exceed the maximum possible anneal time on a physical quantum annealer (for example 2,000 micro-seconds on the D-Wave QPUs) and makes the process diabatic, which exponentially reduces the probability of getting to the ground state[9].

The current generation of the D-Wave QPUs use 8–9 bits for representing coefficients in Eq. (1). Hence, the D-Wave QPU truncates coefficients of a QMI prior to putting qubits in their superposition, which can result in the Ising model having a different ground state—compared to the original QMI. Consequently, the D-Wave QPU may solve a different problem whose result is either infeasible or less accurate than the original problem of interest[30, 31]. Applying preprocessing techniques[32] and classical postprocessing heuristics[33] can remarkably enhance the performance of the D-Wave QPU; however, from a problem-solving viewpoint, the D-Wave quantum annealer cannot guarantee achieving a global optimum.

In this study, we view quantum annealers from two different perspectives simultaneously: (i) a meta-heuristic for solving discrete optimization problems that can find very high-quality solutions in near-constant time; and (ii) a physical process that naturally draws samples from a problem-dependent Boltzmann distribution in cryogenic temperatures. Unlike most current research in Quantum artificial intelligence that applies quantum computing models to hard AI problems, in this paper, we explore how we might apply AI techniques to improve quantum information processing. This paper presents the Reinforcement Quantum Annealing (RQA) scheme that leverages the idea of learning automata[34] to iteratively improve the quality of results, attained by the quantum annealers, and implicitly address the aforementioned issues. In this scheme, the agent maps the problem to an executable QMI and interacts with a quantum annealer that plays the role of a stochastic environment in learning automata. As a proof-of-concept, we first introduce a novel approach for reducing Boolean satisfiability (SAT) instances[35] to minimizing Ising Hamiltonians and then demonstrate that adopting the proposed RQA scheme results in notably better solutions.

Method

Assume that the given problem Π that we aim to solve on a quantum annealer contains a finite set of constraints (components), denoted by π_i for $i \in \{1, 2, \ldots, M\}$, over the same variables as follows:

$$\Pi := \{\pi_i, \pi_2, \ldots, \pi_M\}, \tag{2}$$

where M indicates the number of constraints and our ultimate objective is to find a solution that addresses all constraints. Let H_i be an Ising Hamiltonian whoes ground state represents a solution for π_i and H_Π be the corresponding Ising Hamiltonian of Π that all are acting on same spins (variables). In addition, let $E_0^{H_\Pi}$ be the ground state energy of H_Π and $E_0^{H_i}$ be ground state energy of corresponding H_i. If there exists $\mathbf{z}$ that puts H_i ($\forall i, i \in \{1, 2, \ldots, M\}$) in their ground states (i.e., satisfies all constraints in Π), then $\mathbf{z}$ also puts H_Π in its ground state[36]—in other words,

$$E_0^{H_\Pi} = E_0^{H_1} + E_0^{H_2} + \cdots + E_0^{H_M}. \tag{3}$$

Hence, we can represent H_Π as follows:

$$H_\Pi = H_1 + H_2 + \cdots + H_M. \tag{4}$$

This setting appears in a vast range of problem formulations—including, but not limited to SAT[19], constraint satisfaction problems[37,38], planning and scheduling[13–15], fault detection and diagnosis[17,22], and compressive sensing[23,39]—specifically when we adopt the idea of penalty methods to reduce problems of interest to minimizing Ising Hamiltonians.

Theorem 1. *For any problem in class NP, there are infinite different Ising models whose ground states are all identical to the solution of the original problem.*

Proof. According to Cook—Levin theorem, we can reduce any NP problem to minimizing an Ising Hamiltonian (which is also in the class NP) in polynomial-time[40]. Multiplying all coefficients of the Ising model by a positive non-zero real number results in a new Ising model whose ground state will be identical to the original Ising model. Since the number of positive real numbers are infinite, we can generate infinite different Ising models whose ground states represent the solution for the original problem of interest. □

According to Theorem 1, there are infinite different Ising Hamiltonians whose ground states all represent the solution of the original problem of interest—nevertheless owing to the range and precision limitations on the D-Wave QPUs, we have finite different Ising models for a problem. In theory, these different Ising models are equivalent to each other—i.e., an adiabatic annealing process always attains the ground state which is identical for all corresponding Ising models of a problem. In practice, however, each of these (theoretically) equivalent Ising models are analogous to a pseudo-Boltzman distribution whose parameters are different. Consequently, when we minimize the corresponding QMIs with a physical quantum annealer (like the D-Wave QPU), the probability of finding the global minimum for different Ising Hamiltonians of a given problem varies from zero to one. As an example, an annealing process on a D-Wave QPU may become diabatic because the required anneal time exceeds the maximum possible anneal time (2,000 micro-seconds), which can substantially reduce the probability of finding the global minimum. Note that for a given Ising Hamiltonian, we cannot estimate the probability of finding the ground state prior to executing the corresponding QMI.

We introduce a novel scheme, called reinforcement quantum annealing (RQA), in which an intelligent agent interacts with a quantum annealer as a stochastic environment and tries to iteratively find better Ising Hamiltonians that sampling from their ground state(s), by quantum annealers, results in a better distribution—i.e., the probability of finding the global optimum is increased over the time. To this end, we extend Eq. (4) as follows:

$$\widetilde{H}_\Pi = \widetilde{H}_1 + \widetilde{H}_2 + \cdots + \widetilde{H}_M, \tag{5}$$

such that:

$$\widetilde{H}_i = \chi\left(H_i, \rho_i\right), \tag{6}$$

where $\rho_i \in \mathbb{R}$ denotes the impact (or influence) factor of π_i (or H_i) and χ is a function that maps the input Hamiltonian to a different Hamiltonian which satisfies:

- any $\mathbf{z}$ that puts H_i in its ground state also puts $\widetilde{H}_i$ in its ground state, and vice versa;

- if $\rho_i^1 < \rho_i^2$ then $\chi\left(H_i, \rho_i^1\right) \geq \chi\left(H_i, \rho_i^2\right)$.

Learning automata (LA)[34] are adaptive decision-making models (i.e., type of reinforcement learning[41]) that try to maximize the accumulative reward when they are interacting with stochastic environments. In a similar manner to reinforcement learning, LA use Markov-decision-processes for representing the automaton-environment structure[34,41,42]. In a learning automaton, the agent has a set of r actions (denoted by $\alpha = \{\alpha_1, \alpha_2, \ldots, \alpha_r\}$) and each action has a corresponding probability (denoted by p_i and $\sum p_i = 1$). In each episode, the agent takes (applies) the action α_i (according to p), and (correspondingly) the stochastic environment returns its feedback β that specifies the performance evaluation of the action α_i. The agent uses this feedback to learn from the environment and aims to take optimal actions over the time. For $\beta \in [0,1]$ —so-called S-Type learning automata—in episode t, if the agent takes the action α_i and receives the feedback β^t, we can update p as follows:

$$p_j^{t+1} = \begin{cases} p_j^t - \theta_2\left(1 - \beta^t\right)p_j^t + \theta_1\beta^t\left(1 - p_j\right), & i = j; \\ p_j^t + \theta_2\left(1 - \beta^t\right)\left(\frac{1}{r-1} - p_i^t\right) - \theta_1\beta^t p_j, & i \neq j, \end{cases} \tag{7}$$

where $\beta = 0$ represents the lowest action performance and $\beta = 1$ represents the highest action performance, $\theta_1, \theta_2 \in [0,1]$ are learning factors, and $i, j \in \{1, 2, \ldots, r\}$[34].

We extend learning automata to allow the agent to take multiple actions in each episode. Let $\hat{\alpha}^t \subset \alpha$ denotes the set (list) of actions that the agents takes in episode t. We can extend the Eq. (7) as follows:

$$p_j^{t+1} = \begin{cases} p_j^t - \theta_2 \left(1 - \beta^t\right) p_j^t + \theta_1 \beta^t \left(1 - p_j\right), & \alpha_j \in \hat{\alpha}^t; \\ p_j^t - \theta_2 \left(1 - \beta^t\right) \left(\frac{1}{r - \hat{r}} - \hat{p}_{\hat{\alpha}}^t\right) - \frac{\theta_1 \beta^t}{r - \hat{r}} \left(\hat{r} - \hat{p}_{\hat{\alpha}}^t\right), & \alpha_j \notin \hat{\alpha}^t, \end{cases} \tag{8}$$

where $\hat{r} = |\hat{\alpha}^t|$ and,

$$\hat{p}_{\hat{\alpha}}^t = \sum p_i, \qquad \text{for } \alpha_i \in \hat{\alpha}^t.$$

Finally, we leverage multi-task learning automata—let $\rho_i = p_i$ and $M = r$—to propose the RQA scheme. RQA is an iterative process that we can start it with a uniform distribution of influence factors—i.e., $\rho = \{\frac{1}{M}\}^M$. In each iteration, the agent applies Eq. (5) and submits the corresponding QMI to a quantum annealer. After performing the necessary post-processing methods (like remediating broken-chains and applying post-quantum error correction heuristics), the agent estimates β according to the number of satisfied constraints (π_i) and employs Eq. (8) to update the influence factor ρ.

Proof of Concept: Solving SAT Instances

For a given Boolean formula $f(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$, the problem of Boolean satisfiability (SAT) determines whether a constant replacement of values ("True" or "False") for all Boolean variables can interpret f as "True"[35]. From a complexity perspective, SAT is NP-complete, and we can reduce all problems of class NP to SAT in polynomial-time[40]. The Boolean formula f is in conjunctive normal form (CNF) if it is a conjunction ("AND") of clauses (i.e., $f(\mathbf{x}) = C_1 \wedge C_2 \wedge \cdots \wedge C_M$), where each clause is a disjunction ("OR") of literals (a Boolean variable or its negation)—$C_i = \mathbf{l}_i \vee \mathbf{l}_j \cdots \vee \mathbf{l}_k$. The maximum satisfiability problem (MAX-SAT) is an NP-hard extension of the SAT problem that aims to maximize the number of satisfying clauses[35,43].

First, we adopt the idea of penalty methods to propose a novel heuristic that maps the problem of SAT to a QMI that is executable by the D-Wave quantum processors. In other words, we aim to find coefficients of Eq. (1) such that the ground state of the resulting Ising Hamiltonian represents the satisfying solution of the original SAT instance. In this formulation, $\mathbf{z}_i$ represents the Boolean variable $\mathbf{x}_i$, and we interpret -1 and +1 as "False" and "True", respectively. For a clause with k literals, there are 2^k different possibilities among which, we can distinguish the only state that makes the clause to be false—called infeasible state. Hence, we represent each clause of the given SAT instance with two inequalities as:

$$D_i \leq E_{\text{infeasible}}, \tag{9}$$

and,

$$D_i \geq \sum E_{\text{feasible}}, \tag{10}$$

where $D_i \in \mathbb{R}$ is the boundary variable corresponding to the clause C_i, AND E_{feasible} and $E_{\text{infeasible}}$ represent the contribution of the feasible and infeasible states in the ultimate energy function, respectively. For $C_i = \mathbf{x}_1 \vee \neg \mathbf{x}_4 \vee \mathbf{x}_9$, as an example, Eq. (9) reduces to:

$$D_i \leq -\mathbf{h}_1 + \mathbf{h}_4 - \mathbf{h}_9 - J_{1,4} + J_{1,9} - J_{4,9},$$

and we can represent Eq. (10) as follows:

$$\begin{aligned} D_i \geq & -\mathbf{h}_1 - \mathbf{h}_4 - \mathbf{h}_9 + J_{1,4} + J_{1,9} + J_{4,9} \\ & -\mathbf{h}_1 - \mathbf{h}_4 + \mathbf{h}_9 + J_{1,4} - J_{1,9} - J_{4,9} \\ & -\mathbf{h}_1 + \mathbf{h}_4 + \mathbf{h}_9 - J_{1,4} - J_{1,9} + J_{4,9} \\ & +\mathbf{h}_1 - \mathbf{h}_4 - \mathbf{h}_9 - J_{1,4} - J_{1,9} + J_{4,9} \\ & +\mathbf{h}_1 - \mathbf{h}_4 + \mathbf{h}_9 - J_{1,4} + J_{1,9} - J_{4,9} \\ & +\mathbf{h}_1 + \mathbf{h}_4 - \mathbf{h}_9 + J_{1,4} - J_{1,9} - J_{4,9} \\ & +\mathbf{h}_1 + \mathbf{h}_4 + \mathbf{h}_9 + J_{1,4} + J_{1,9} + J_{4,9}. \end{aligned}$$

A clause with k literals includes $2^k - 1$ different feasible states so the size of Eq. (10) grows exponentially with k.

Theorem 2. *Sum of the energy values for all possible states in every Ising model is zero.*

Proof. Let Z denotes the set of all possible states in Eq. (1). Because spins in the Ising model (here z_I) take their values from $\{-1, +1\}$, Z is a closed set under the complement operation (i.e., $\mathbf{z}, -\mathbf{z} \in Z$), and $|Z| = 2^N$. Accordingly, sum of the energy values for all possible states in the Ising model is:

$$\sum_{\mathbf{z} \in Z} E_{\text{Ising}}(\mathbf{z}) = \sum_{\mathbf{z} \in Z} \left(\sum_{i=1}^{N} \mathbf{h}_i \mathbf{z}_i + \sum_{i=1}^{N} \sum_{j=i+1}^{N} J_{ij} \mathbf{z}_i \mathbf{z}_j \right) = \sum_{i=1}^{N} \left(\frac{2^N}{2} \mathbf{h}_i + \frac{2^N}{2}(-\mathbf{h}_i) \right) + \sum_{i=1}^{N} \sum_{j=i+1}^{N} \left(\frac{2^N}{2} J_{ij} + \frac{2^N}{2}(-J_{ij}) \right) = 0.$$

$\square$

According to Theorem 2, we can rewrite Eq. (10) as follows:

$$D_i \geq -E_{\text{infeasible}},$$

that obeys:

$$0 \leq D_i. \tag{11}$$

Note that clauses in CNF representation are connected with "AND" operator. Hence, after representing each clause of the SAT with two inequalities, Eq. (9) and (11), we aggregate the resulting sub-systems of inequalities to form a larger system of inequalities. After representing the given SAT instance with M clauses with a system of $2M$ inequalities, we represent the D-Wave hardware restrictions through embedding:

$$-2 \leq \mathbf{h}_i \leq +2, \tag{12}$$

and,

$$-1 \leq J_{ij} \leq +1, \tag{13}$$

where $i, j \in \{1, 2, \ldots, N\}$ and $i < j$. Finally, we solve the following objective function:

$$\max_{\mathbf{h}, J, D} \sum_{i=1}^{M} D_i, \tag{14}$$

to obtain coefficients of the Ising Hamiltonian, shown in Eq. (1), that is executable by a D-Wave QPU. Note that inequalities (9), (11), (12) and (13) are linear. Thus, the objective function in Eq. (14) is tractable by linear programming and convex optimization techniques. Considering that biases and couplers on a D-Wave QPU are bounded, Eq. (12) and (13), problem (14) will always converge.

To adopt the proposed RQA scheme for addressing the NP-complete problem of SAT, we rewrite the inequality (11) as follows:

$$\rho_i \leq D_i, \tag{15}$$

where ρ_i denotes the influence factor of the clause C_i. Here, we define ρ as:

$$\rho_i = \frac{1}{M} - p_i, \tag{16}$$

where p_i is the corresponding probability of constraint π_i (here the clause C_i) in Eq. (8). Note that when $p_i = \frac{1}{M}$, the inequalities (11) and (15) are identical.

The architecture of the proposed agent contains the following components:

- Φ^t—set of unsatisfied clauses in episode t;

- ρ^t—tuple of M influence factors in episode t;

- QMI^t—action of the agent in episode t, the Ising Hamiltonian for solving the given SAT instance (according to Φ^t and ρ^t);

- $\mathbf{z}$—perception of the agent from the stochastic environment, resulting sample(s) from executing the QMI^t on a quantum annealer.

For a given Boolean formula in CNF, the agent initializes its internal state as:

$$\Phi^0 = \emptyset,$$

$$p^0 = \{\frac{1}{M}\}^M.$$

In each episode, the agent forms a system of inequalities with Eq. (9) and (15), and embeds Eq. (12) and (13). Afterward, the agent solves problem (14), and submits the resulting Ising Hamiltonian (QMI^t) to a D-Wave QPU. The environment (here the D-Wave QPU) draws sample(s) from the corresponding pseudo-Boltzmann distribution, and returns the resulting sample(s). The episode ends with updating the internal state of the agent as follows:

$$\Phi^t = \text{set of unsatisfied clauses with } \mathbf{z}^t,$$

$$\beta^t = 1 - \frac{|\Phi^t|}{M},$$

and (finally) updating probabilities with (8). In RQA, the action of the agent in episode t depends on ρ_{t-1}; therefore, Markov property holds here[44].

Results

In this section, we aim to evaluate the performance of RQA scheme on solving benchmark SAT instances, and compare it with recent software and hardware enhancements to the quantum annealers. In this study, we used Z3 (from Microsoft Research) as a framework for symbolic computing implementations in Python[45] and we executed each QMI on the D-Wave 2000Q quantum annealer (with lower noise), located at Burnaby, British Columbia.

For every SAT instance, we used inequalities (9), (15), (12) and (13) to represent the given SAT instance as a system of inequalities. Afterward, we solved problem (14) for reducing the SAT to an executable QMI on a D-Wave quantum processor. Note that solving problem (14) will result in an Ising Hamiltonian which is not necessarily compatible with the D-Wave hardware graph (Chimera topology for the current generation). Therefore, we applied the minor-embedding heuristic[46] for embedding the problem to the physical lattice of qubits on a D-Wave QPU. To avoid the impact of chaining physical qubits in our evaluations, we employed fixed embeddings of cliques in all instances.

Recent studies have revealed that using spin-reversal transforms (a.k.a. gauge transforms)—i.e., flipping the qubits randomly without altering the ground state of the original Ising Hamiltonian—can reduce analog errors of the quantum annealers[32]. Thus, as a preprocessing technique, we applied spin-reversal transforms prior to submitting the QMIs to the physical QPU. We also put a delay between measurements to reduce the sample-to-sample correlation, albeit longer run-time.

To remediate possible broken chains in the resulting raw samples from the D-Wave QPU, we performed voting among the physical qubits of chains. After disembedding samples (i.e., representing variables in the original problem domain), we applied the multi-qubit correction (MQC) heuristic[33] which has demonstrated a significant ability to improve the probability of finding the global minimum, attained by the D-Wave QPU. Finally, we performed a local search heuristic, so-called single-qubit correction (SQC), to construct the final solution of the given SAT[33].

Experiment A: Factoring Pseudo-Prime Numbers

In number theory, the problem of integer factoring refers to decomposing a composite integer number into the product of smaller integers, and prime factorization restricts these factors to prime numbers. Although there are debates on the class (or complexity) of this problem, there is no known efficient (non-quantum) algorithm for factoring numbers in polynomial-time[47].

In this study, we use the problem of prime factorization as a benchmark to evaluate the performance of RQA. It is worth noting that our research objective in this paper is not to set a new record for quantum factorized integers, which for the current generation of the D-Wave quantum annealers is 1,005,973[21]. Indeed, since the security of the modern public-key cryptography systems (like RSA) mainly relies on the difficulty of factoring very large pseudo-prime numbers[21,48], we relied on the difficulty of prime factorization problem for generating benchmark SAT instances. Let $f(\mathbf{x}_1, \mathbf{x}_2)$ be a Boolean function as follows:

$$\mathbf{q} = f(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1 \times \mathbf{x}_2, \tag{17}$$

where $\mathbf{x}_1 \in \{0,1\}^{n_1}$ and $\mathbf{x}_2 \in \{0,1\}^{n_2}$ are integer-valued numbers in binary representation (here, $\mathbf{x}_1, \mathbf{x}_2 \geq 2$), and the multiply operator is in binary base—each element of the vector $\mathbf{q}$ is a Boolean function of $\mathbf{x}_1$ and $\mathbf{x}_2$. Assume that $\hat{\mathbf{q}}$ is a pseudo-prime integer number in binary base (i.e., $\hat{\mathbf{q}}$ has two prime factors. We can map the problem of factoring $\hat{\mathbf{q}}$ to SAT as follows

$$g = f_{\text{SAT}}(\mathbf{q}, \hat{\mathbf{q}}) = \bigwedge_{i=1}^{n} \neg(\mathbf{q}_i \oplus \hat{\mathbf{q}}_i), \tag{18}$$
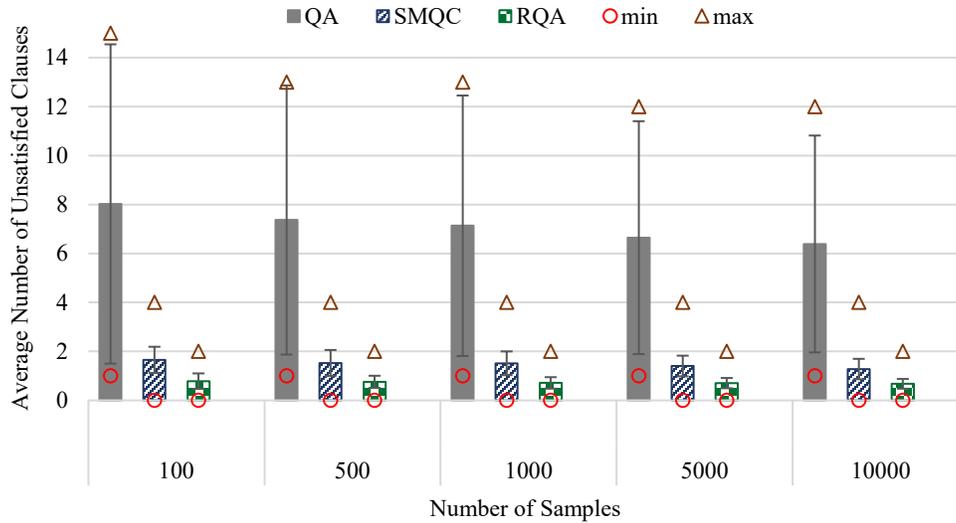
Figure 1. Experiment results for solving 136 satisfiable SAT instances for factoring pseudo-prime numbers with quantum annealing (QA), quantum annealing with classical post-processing (SMQC) and reinforcement quantum annealing (RQA).

where $n = n_1 + n_2$ denotes the length of $\mathbf{q}$. We can look at the process of generating SAT instances from a reverse-engineering viewpoint. To this end, we generated pseudo-prime numbers via multiplying two prime numbers, and represented them in binary base (denoted by $\hat{\mathbf{q}}$). For each instance, we then used the Eq. (18) to map the factorization of $\hat{\mathbf{q}}$ to a satisfiable Boolean formula. Since g is a Boolean expression of x, we applied the Tseitin transformation[49] to represent g in CNF. We also performed pre-processing techniques, namely "ctx-solver-simplify", "recover-01", "propagate-values" and "reduce-args" tactics from Z3[50]. Note that applying the Tseitin transformation can increase the size of g linearly, due to defining auxiliary variables. Since the capacity of the current D-Wave 2000Q quantum processors is limited to a complete graph of size 63, we eliminated SAT instances (in CNF) with more than 63 Boolean variables which resulted in 136 satisfiable SAT instances.

Figure 1 illustrates results—minimum (circles), maximum (triangles), average and variance of number of unsatisfied clauses—for solving these 136 satisfiable SAT instances, and compares the performance of the proposed RQA scheme with quantum annealing (QA) and quantum annealing with multiple post-quantum processes (SMQC). To enhance the standard quantum annealing technique, we used two spin-reversal-transforms[32], as well as the delay between measurements to reduce the inter-sample correlation. In the second method (SMQC), we first used the multi-qubit correction (MQC) method[33], in problem variable level—which is the state-of-the-art technique in the realm of post-quantum correction for quantum annealers—and then applied a local search to maximize the quality of results, attained by the SMQC arrangement.

To update influence factors of clauses in RQA, Eq. (8), we used $\theta_1 = 0.1$ and $\theta_2 = 0$. Learning automata generally require a notable number of episodes to converge to an optimal (or sub-optimal) policies. In this experiment, nevertheless, the agent terminates the process after at most $T = 10$ episodes (due to QPU time limitations) or finding a solution that satisfies all clauses. Hence, we formed a hall-of-fame—a set of final solutions from all episodes—and applied MQC (followed by SQC) on them to obtain the ultimate solution of RQA. Our empirical observations showed that this technique can implicitly address the limited number of allowed episodes in RQA. Note that RQA utilizes at most same number of samples as QA and SMQC.

Experiment B: Uniform Random 3-SAT with Phase Transitions

Sampling from the phase transition region of uniform Random 3-SAT is a common practice for generating benchmark SAT (and MAX-SAT) problems[51–54]. In this experiment, as our second study case, we used the satisfiable benchmark test-set of uniform random 3-SAT with phase transitions[55]. Considering the capacity of the current generation of the D-Wave quantum annealers—we can embed a clique of size at most 63 on chimera architecture—we employed the test-set with 50 Boolean variables.

Figure 2 demonstrates results—minimum (circles), maximum (triangles), average and variance of number of unsatisfied clauses—for solving the first 100 instances from the benchmark test-set, and (similar to the previous experiment) compares the performance of the proposed RQA scheme with quantum annealing (QA) and quantum annealing with multiple post-quantum processes (SMQC). The setting for this experiment was identical to the previous experiment, except the number of instances (136 vs. 100) and the number of variables (variant vs. 50).
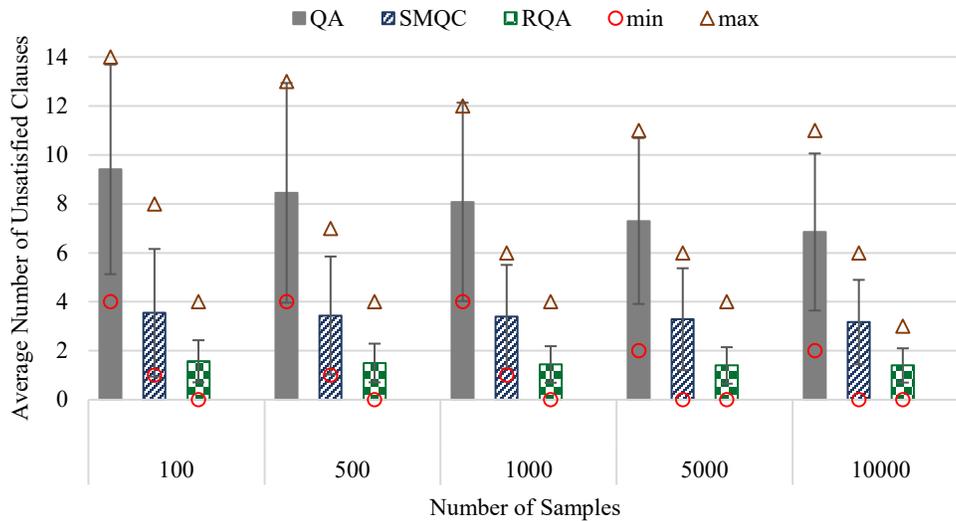
Figure 2. Experiment results for solving 100 satisfiable uniform random 3-SAT instances with phase transitions using quantum annealing (QA), quantum annealing with classical post-processing (SMQC) and reinforcement quantum annealing (RQA).

Experiment C: Run-Time Evaluation

In this experiment, we aim to evaluate the average run-time of RQA scheme and compare it with QA and SMQC approaches. We implemented all experiments in Python 3.7.4, and executed them on a 64-bit Windows 10 based system with 32 GB RAM and Intel Xeon processor at 3.00 GHz. Figure 3 shows the average run-time of solving 100 SAT instances for QA, SMQC and RQA methods on a D-Wave 2000Q quantum processor. Note that, in this experiment, we did not include the computation time for finding the embedding of the QMI on a working graph—we used the pre-defined embeddings of cliques for the chimera architecture. For all instances, we used two spin-reversal transforms and we also enabled the inter-sample delay between samples' reads.

Discussion

In this study, we introduced a novel scheme—called reinforcement quantum annealing (RQA)—that leverages reinforcement learning (more specifically learning automata) to ENHANCE the quality of results, attained by the quantum annealers. Ramezanpour (2018) has proposed to improve the simulated quantum annealing algorithm by adding reinforcement to the standard quantum annealing algorithm[56]. Simulated quantum annealing is an iterative algorithm (similar to simulated annealing) that is implemented and run on classical computers. On the other hand, quantum annealers are physical, single-instruction, quantum processing units where the entire annealing process is atomic. Thus, we cannot modify or adjust the annealing process after starting the annealing (i.e., putting quantum bits on their superposition). in RQA, similar to the standard reinforcement learning scheme, iterations emulate the interactions between an agent and its environment. Ramezanpour's method, however, has an adaptive optimization scheme in which iterations simulate one quantum annealing process on a classical computer and is not applicable to physical quantum annealers.

As a proof-of-concept, we proposed a novel method for reducing SAT to minimizing Ising Hamiltonians and then demonstrated that applying the proposed RQA scheme (on a D-Wave quantum annealer) results in notably better solutions. It is worth highlighting that, however, the proposed approach (i.e., hybridization of reinforcement learning and quantum annealing) is applicable to a vast range of classic AI problems like constraint satisfaction, planning, and scheduling.

We applied the proposed RQA scheme on two different SAT problem sets, and compared its performance with quantum annealing (QA) and quantum annealing with post-quantum error corrections (so-called SMQC) which is the state-of-the-art in the realm of quantum annealing[33,57,58]. The first problem set includes 136 satisfiable SAT instances which represent factoring pseudo-prime numbers that have at most 63 Boolean variables, in CNF representation. Besides the length of the given composite numbers, the difficulty of integer factoring also depends on the properties of integer numbers. The hardest instances of this problem are factoring pseudo-prime numbers (product of two prime numbers) whose factors have the same size (in binary base). SAT instances in experiment A are not the hardest cases of prime-factoring— restricting the SAT instances in experiment A to a composite of the same size prime factors resulted in only 8 problems. It is worth noting that our main objective in this experiment was not to address the prime factorization nor to use quantum annealers for solving SAT or MAX-SAT problems.
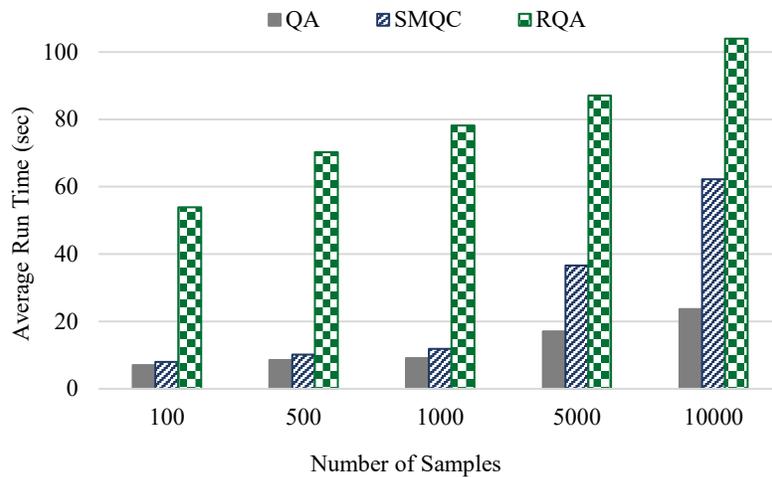
Figure 3. Average run-time for solving 100 SAT instances with quantum annealing (QA), quantum annealing with classical post-processing (SMQC) and reinforcement quantum annealing (RQA) approaches on a D-Wave 2000Q quantum processing unit.

Figure 1 illustrates results—minimum, maximum, average and variance of a number of unsatisfied clauses—of solving these 136 satisfiable SAT instances for 100, 500, 1,000, 5,000 and 10,000 samples. In RQA, increasing the number of samples from 100 to 10,000 reduces the average number of unsatisfied clauses from 1.57 to 1.40. Similarly, in QA and SMQC, the average number of unsatisfied clauses is reduced from 9.41 and 3.55 to 6.85 and 3.17, respectively. Although increasing the number of samples in all three methods reduces the average number of unsatisfied clauses, RQA with 100 samples outperforms both QA and SMQC approaches in all arrangements (even with 10,000 samples). It is worth highlighting that QA was not able to satisfy all clauses of any of these 136 SAT instances, even when we requested for 10,000 samples, while both SMQC and RQA methods were able to find a satisfying solution for at least one of the instances in all cases. From a robustness viewpoint, increasing the number of samples from 100 to 10,000 lowers the variance and range (the difference between maximum and minimum) of QA, SMQC and RQA from 4.28, 2.60 and 0.86 to 3.21, 1.73 and 0.70, respectively. Therefore, RQA demonstrated significantly better robustness (i.e., higher reproducibility rate), compared to both QA and SMQC approaches.

In the second study case, shown in Fig. 2, we used the first 100 SAT instances of the satisfiable benchmark test-set of uniform random 3-SAT with phase transitions[55]. Similar to the Fig. 1, Fig. 2 demonstrates that RQA with 100 samples outperforms both QA and SMQC approaches in all settings. More specifically, increasing the number of samples from 100 to 10,000 in QA, SMQC and RQA decreases the average number of unsatisfied clauses from 8.02, 1.65 and 0.79 to 6.39, 1.28 and 0.68, respectively. Also, the variances are reduced from 6.52, 0.53 and 0.31 to 4.43, 0.42 and 0.20, respectively. Note that the minimum number of unsatisfied clauses in RQA for all settings is zero while SMQC needed at least 5,000 samples to satisfy all clauses of at least one SAT instance.

Problem-solving with a D-Wave quantum processor, in practice, requires some preprocesses (e.g., embedding and gauge transforms) and post-processes (like error correction and broken-chain remediation) that extends the total run-time from 20–2000 microseconds (pure annealing time) to some seconds and even minutes. Run-time in many pre/post-processing techniques mainly depends on the number of samples. It is a common practice in applying quantum annealers to request a few thousand (at most 10,000 per QMI) samples to increase the probability of finding the global minimum. Figure 3 demonstrates that increasing the number of samples increases the total run-time of RQA at a significantly lower rate (compared to QA and more specifically SMQC)—since a request of 10 times fewer samples in each iteration of RQA yet has less pre/post-processing overhead. As an illustration, 100 samples (10 samples in each iteration of RQA), increases the total run-time 7.6x and 6.8x more than QA and SMQC, respectively. However, increasing the number of samples to 10,000, (1,000 samples in each iteration of RQA) only increases the total run time by 4.4x and 1.7x, compared to QA and SMQC, respectively.

Acknowledgements

References

1. Lamata, L. Basic protocols in quantum reinforcement learning with superconducting circuits. *Sci. reports* **7**, 1609 (2017).

2. Biamonte, J. *et al.* Quantum machine learning. *Nature* **549**, 195 (2017).

3. Dunjko, V. & Briegel, H. J. Machine learning & artificial intelligence in the quantum domain: a review of recent progress. *Reports on Prog. Phys.* **81**, 074001 (2018).

4. Ladd, T. D. *et al.* Quantum computers. *nature* **464**, 45 (2010).

5. Johnson, M. W. *et al.* Quantum annealing with manufactured spins. *Nature* **473**, 194 (2011).

6. Finnila, A., Gomez, M., Sebenik, C., Stenson, C. & Doll, J. Quantum annealing: a new method for minimizing multidimensional functions. *Chem. physics letters* **219**, 343–348 (1994).

7. Kadowaki, T. & Nishimori, H. Quantum annealing in the transverse ising model. *Phys. Rev. E* **58**, 5355 (1998).

8. Ohzeki, M. & Nishimori, H. Quantum annealing: An introduction and new developments. *J. Comput. Theor. Nanosci.* **8**, 963–971 (2011).

9. Nishimori, H. & Takada, K. Exponential enhancement of the efficiency of quantum annealing by non-stoquastic hamiltonians. *Front. ICT* **4**, 2 (2017).

10. Ayanzadeh, R., Halem, M. & Finin, T. An SAT-based quantum programming paradigm. *Univ. Maryland, Baltim. Cty.* (2019).

11. Boothby, K., Bunyk, P., Raymond, J. & Roy, A. Next-generation topology of d-wave quantum processors. Tech. Rep., Technical report (2019).

12. Biswas, R. *et al.* A nasa perspective on quantum computing: Opportunities and challenges. *Parallel Comput.* **64**, 81–98 (2017).

13. Rieffel, E. G. *et al.* A case study in programming a quantum annealer for hard operational planning problems. *Quantum Inf. Process.* **14**, 1–36 (2015).

14. Venturelli, D., Marchand, D. J. & Rojo, G. Quantum annealing implementation of job-shop scheduling. *arXiv preprint arXiv:1506.08479* (2015).

15. Tran, T. T. *et al.* A hybrid quantum-classical approach to solving scheduling problems. In *Ninth annual symposium on combinatorial search* (2016).

16. Bian, Z. *et al.* Discrete optimization using quantum annealing on sparse ising models. *Front. Phys.* **2**, 56 (2014).

17. Bian, Z. *et al.* Mapping constrained optimization problems to quantum annealing with application to fault diagnosis. *Front. ICT* **3**, 14 (2016).

18. Su, J., Tu, T. & He, L. A quantum annealing approach for boolean satisfiability problem. In *2016 53nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, 1–6 (IEEE, 2016).

19. Bian, Z. *et al.* Solving sat and maxsat with a quantum annealer: Foundations and a preliminary report. In *International Symposium on Frontiers of Combining Systems*, 153–171 (Springer, 2017).

20. O'Malley, D., Vesselinov, V. V., Alexandrov, B. S. & Alexandrov, L. B. Nonnegative/binary matrix factorization with a d-wave quantum annealer. *PloS one* **13**, e0206653 (2018).

21. Peng, W. *et al.* Factoring larger integers with fewer qubits via quantum annealing with optimized parameters. *SCIENCE CHINA Physics, Mech. & Astron.* **62**, 60311 (2019).

22. Perdomo-Ortiz, A., Fluegemann, J., Narasimhan, S., Biswas, R. & Smelyanskiy, V. N. A quantum annealing approach for fault detection and diagnosis of graph-based systems. *The Eur. Phys. J. Special Top.* **224**, 131–148 (2015).

23. Ayanzadeh, R., Mousavi, S., Halem, M. & Finin, T. Quantum annealing based binary compressive sensing with matrix uncertainty. *arXiv preprint arXiv:1901.00088* (2019).

24. Ohzeki, M., Miki, A., Miyama, M. J. & Terabe, M. Control of automated guided vehicles without collision by quantum annealer and digital devices. *arXiv preprint arXiv:1812.01532* (2018).

25. Perdomo-Ortiz, A., Dickson, N., Drew-Brook, M., Rose, G. & Aspuru-Guzik, A. Finding low-energy conformations of lattice protein models by quantum annealing. *Sci. reports* **2**, 571 (2012).

26. Adachi, S. H. & Henderson, M. P. Application of quantum annealing to training of deep neural networks. *arXiv preprint arXiv:1510.06356* (2015).

27. Vinci, W. *et al.* A path towards quantum advantage in training deep generative models with quantum annealers. *arXiv preprint arXiv:1912.02119* (2019).

28. Vinci, W. & Lidar, D. A. Non-stoquastic hamiltonians in quantum annealing via geometric phases. *npj Quantum Inf.* **3**, 38 (2017).

29. Okada, S., Ohzeki, M., Terabe, M. & Taguchi, S. Improving solutions by embedding larger subproblems in a d-wave quantum annealer. *Sci. reports* **9**, 2098 (2019).

30. Pudenz, K. L., Albash, T. & Lidar, D. A. Quantum annealing correction for random ising problems. *Phys. Rev. A* **91**, 042302 (2015).

31. Dorband, J. E. Extending the d-wave with support for higher precision coefficients. *arXiv preprint arXiv:1807.05244* (2018).

32. Pelofske, E., Hahn, G. & Djidjev, H. Optimizing the spin reversal transform on the d-wave 2000q. *arXiv preprint arXiv:1906.10955* (2019).

33. Dorband, J. E. A method of finding a lower energy solution to a qubo/ising objective function. *arXiv preprint arXiv:1801.04849* (2018).

34. Narendra, K. S. & Thathachar, M. A. *Learning automata: an introduction* (Courier Corporation, 2012).

35. Biere, A., Heule, M. & van Maaren, H. *Handbook of satisfiability*, vol. 185 (IOS press, 2009).

36. Mooney, G. J., Tonetto, S. U., Hill, C. D. & Hollenberg, L. C. Mapping np-hard problems to restricted adiabatic quantum architectures. *arXiv preprint arXiv:1911.00249* (2019).

37. Vyskocil, T. & Djidjev, H. Simple constraint embedding for quantum annealers. In *2018 IEEE International Conference on Rebooting Computing (ICRC)*, 1–11 (IEEE, 2018).

38. Vyskočil, T., Pakin, S. & Djidjev, H. N. Embedding inequality constraints for quantum annealing optimization. In *International Workshop on Quantum Technology and Optimization Problems*, 11–22 (Springer, 2019).

39. Mousavi, S., Taghiabadi, M. M. R. & Ayanzadeh, R. A survey on compressive sensing: Classical results and recent advancements. *arXiv preprint arXiv:1908.01014* (2019).

40. Garey, M. R. & Johnson, D. S. *Computers and intractability*, vol. 29 (wh freeman New York, 2002).

41. Kaelbling, L. P., Littman, M. L. & Moore, A. W. Reinforcement learning: A survey. *J. artificial intelligence research* **4**, 237–285 (1996).

42. Sutton, R. S. & Barto, A. G. *Reinforcement learning: An introduction* (MIT press, 2018).

43. Ayanzadeh, R., Halem, M. & Finin, T. SAT-based compressive sensing. *arXiv preprint arXiv:1903.03650* (2019).

44. Cox, D. R. *The theory of stochastic processes* (Routledge, 2017).

45. De Moura, L. & Bjørner, N. Z3: An efficient smt solver. In *International conference on Tools and Algorithms for the Construction and Analysis of Systems*, 337–340 (Springer, 2008).

46. Cai, J., Macready, W. G. & Roy, A. A practical heuristic for finding graph minors. *arXiv preprint arXiv:1406.2741* (2014).

47. Balasubramanian, K. & Abbas, A. M. Integer factoring algorithms. In *Algorithmic Strategies for Solving Complex Problems in Cryptography*, 228–240 (IGI Global, 2018).

48. Dridi, R. & Alghassi, H. Prime factorization using quantum annealing and computational algebraic geometry. *Sci. reports* **7**, 43048 (2017).

49. Li, C. M., Manyà, F. & Soler, J. R. Clausal form transformation in maxsat. In *2019 IEEE 49th International Symposium on Multiple-Valued Logic (ISMVL)*, 132–137 (IEEE, 2019).

50. De Moura, L. & Passmore, G. O. The strategy challenge in smt solving. In *Automated Reasoning and Mathematics*, 15–44 (Springer, 2013).

51. Cheeseman, P. C., Kanefsky, B. & Taylor, W. M. Where the really hard problems are. In *IJCAI*, vol. 91, 331–337 (1991).

52. Selman, B., Mitchell, D. G. & Levesque, H. J. Generating hard satisfiability problems. *Artif. intelligence* **81**, 17–29 (1996).

53. Achlioptas, D., Gomes, C., Kautz, H. & Selman, B. Generating satisfiable problem instances. *AAAI/IAAI* **2000**, 256–261 (2000).

54. Nudelman, E., Leyton-Brown, K., Hoos, H. H., Devkar, A. & Shoham, Y. Understanding random sat: Beyond the clauses-to-variables ratio. In *International Conference on Principles and Practice of Constraint Programming*, 438–452 (Springer, 2004).

55. Hoos, H. H. & Stützle, T. Satlib: An online resource for research on sat. *Sat* **2000**, 283–292 (2000).

56. Ramezanpour, A. Enhancing the efficiency of quantum annealing via reinforcement: A path-integral monte carlo simulation of the quantum reinforcement algorithm. *Phys. Rev. A* **98**, 062309 (2018).

57. Golden, J. K. & O'Malley, D. Pre-and post-processing in quantum-computational hydrologic inverse analysis. *arXiv preprint arXiv:1910.00626* (2019).

58. Ayanzadeh, R., Halem, M., Dorband, J. & Finin, T. Quantum-assisted greedy algorithms. *arXiv preprint arXiv:1912.02362* (2019).