Declaration of Academic Integrity

The participating team declares that the paper submitted is comprised of original research and results obtained under the guidance of the instructor. To the team's best knowledge, the paper does not contain research results, published or not, from a person who is not a team member, except for the content listed in the references and the acknowledgment. If there is any misinformation, we are willing to take all the related responsibilities.

Names of team members
Max B. Zhao
Signatures of team members

Name of the instructor
Fei Li
Signature of the instructor

Date
08/21/2025

S.T. Yau High School Science Award

Research Report

The Team

Name of team member: Max B. Zhao
School: Thomas Jefferson High School for Science and Technology
City, Country: Alexandria, VA 22312, USA

Name of supervising teacher: Fei Li
Job Title: Associate Professor
School/Institution: Department of Computer Science, George Mason University
City, Country: Fairfax, VA 22030, USA

Title of Research Report

How to solve Sudoku puzzles using quantum mechanics: reaching the ground state of Ising spin glass via Tensor Train Express

Date

August 23, 2025

How to solve Sudoku puzzles using quantum mechanics: reaching the ground state of Ising spin glass via Tensor Train Express

Max B. Zhao

Abstract

We present a novel quantum-inspired algorithm for solving Quadratic Unconstrained Binary Optimization (QUBO) problems, which are equivalent to finding the ground states of Ising spin glass Hamiltonians. The algorithm employs tensor trains to efficiently represent massive superpositions of spin configurations and utilizes a discrete driving schedule to guide the tensor train toward the ground state. At each step, a driver Hamiltonian incorporating a transverse magnetic field is mixed with the problem Hamiltonian to promote entanglement and facilitate spin flips. The tensor train is updated using the canonical Density Matrix Renormalization Group (DMRG) method, which iteratively minimizes the system's energy via multiple sweeps across the spin chain. We benchmark the performance of the algorithm using QUBO instances of diverse origin, type, and size, and observe that it reliably identifies the global minimum, not merely near-optimal solutions. We first demonstrate it is capable of solving intermediate-level Sudoku puzzles that involve over 200 Ising spins with long-range couplings dictated by constraint satisfaction. We then apply the algorithm to successfully solve MaxCut problems from the Biq Mac library with up to 251 nodes and $3,265$ edges. We discuss the advantages of this quantum-inspired approach compared to qubits-based quantum algorithms and comment on its potential for industrial-scale applications.

Keywords: combinatorial optimization, Ising spin glass, dense and long-range interactions, quantum-inspired algorithms, tensor networks, tensor train, Sudoku puzzles, MaxCut problems

Acknowledgement

1. Max B. Zhao (M. Z.) would like to thank two teachers from Thomas Jefferson High School for Science and Technology who are not directly involved in but provided inspiration for this project: Mr. Mark Hannum for teaching quantum mechanics, spin Hamiltonians, and quantum algorithms in Electrodynamics class, and Mr. Paul Kosek for the assignment of implementing backtracking algorithms for Sudoku puzzles in Artificial Intelligence class. Thanks to these teachers, M. Z. was able to cross-fertilize ideas from both classes and think about quantum advantages in solving NP-hard optimization problems.

2. M. Z. is indebted to his research supervisor/mentor Dr. Fei Li of George Mason University for guidance in navigating the vast and rapidly evolving landscape of quantum computation. Dr. Li introduced M. Z. to computational complexity theory, Quadratic Unconstrained Binary Optimization (QUBO) problems, and physics of the Ising spin glass. He also assured M. Z. that the language (the jargons of computational quantum physics) barrier is penetrable, and the underlying ideas are intuitive and fun. The discussions with Dr. Li helped M. Z. set the goal of this research project: inventing a new quantum-inspired algorithm for solving QUBO problems.

3. **Contributions:** M. Z. improved the QUBO formulation of Sudoku, conceived and designed the new quantum-inspired algorithm based on sweeping and discrete driving of the Ising spin glass Hamiltonian, wrote the code and carried out the numerical experiments on Sudoku puzzles and Max-Cut instances, conducted a quantitative and qualitative analysis of the data and results, and created the report; Dr. Li provided overall guidance regarding the feasibility of quantum-inspired approaches, offered feedback on algorithm design and data analysis, and encouraged the completion of this work by explaining its potential impact to a wide range of applications (e.g. vehicle routing and quadratic assignment problems).

Commitments on Academic Honesty and Integrity

We hereby declare that we

1. are fully committed to the principle of honesty, integrity and fair play throughout the competition.
2. actually perform the research work ourselves and thus truly understand the content of the work.
3. observe the common standard of academic integrity adopted by most journals and degree theses.
4. have declared all the assistance and contribution we have received from any personnel, agency, institution, etc. for the research work.
5. undertake to avoid getting in touch with assessment panel members in a way that may lead to direct or indirect conflict of interest.
6. undertake to avoid any interaction with assessment panel members that would undermine the neutrality of the panel member and fairness of the assessment process.
7. observe the safety regulations of the laboratory(ies) where we conduct the experiment(s), if applicable.
8. observe all rules and regulations of the competition.
9. agree that the decision of YHSA is final in all matters related to the competition.

We understand and agree that failure to honour the above commitments may lead to disqualification from the competition and/or removal of reward, if applicable; that any unethical deeds, if found, will be disclosed to the school principal of team member(s) and relevant parties if deemed necessary; and that the decision of YHSA is final and no appeal will be accepted.

X_____
Name of team member: Max B. Zhao

X_____
Name of supervising teacher: Fei Li

Table of Contents

1 Background: optimization in the quantum era

The ubiquitous optimization challenge. When making important decisions, we often hope to find the best option out of a discrete set of feasible solutions that satisfy certain constraints. This familiar task is known mathematically as combinatorial optimization, which is fundamentally important yet intrinsically hard [1]. A classical example is the traveling salesman problem: formulated by physicist and mathematician William Rowan Hamilton in the 19th century, the problem continues to influence computational complexity theory today [2]. Combinatorial optimization arises in almost all sectors of industry and commerce such as logistics, supply chain, vehicle routing, microchip design, drug discovery, and portfolio management. A broad subset of combinatorial optimization problems can be formulated as Quadratic Unconstrained Binary Optimization (QUBO) problems [3, 4], which fit under a compact mathematical representation:

$$\min f(\{x_i\}) = \sum_{i,j=1}^{N} x_i Q_{i,j} x_j \qquad\qquad x_i \in \{0,1\}$$

Here $\{x_i\}$ is a set of binary variables, Q is a symmetric matrix, and the objective is to minimize the cost function f. Despite its simple appearance, the QUBO problem is NP-hard [5], meaning that no classical algorithm can efficiently solve all instances as the problem size N increases [2]. As a result, many industrial-scale QUBO problems pose significant computational challenges [4]. Classical heuristic algorithms typically settle for near-optimal solutions, leaving the global minimum unknown or inconclusive.

Quantum mechanics to the rescue. In recent decades, combinatorial optimization has also become *a physics problem* as the field of quantum information and computation matures. There is a growing interest in inventing the next-generation QUBO solvers based on quantum algorithms to overcome the large-N barrier. A common strategy is to encode the cost function into a cost Hamiltonian describing a carefully prepared quantum system, which is then manipulated by taking advantage of quantum superposition, tunneling, and other non-classical effects before the results are read out by measurements [6]. More specifically, a QUBO problem can be mapped to an Ising spin Hamiltonian denoted by H_z [7, 8] where the interactions between spins are specified by the matrix elements Q_{ij} and the cost function f corresponds to the eigenvalues of H_z. After the mapping, finding the solution of the optimization problems becomes equivalent to finding the ground state of the Hamiltonian H_z. There are several ways to do this, for example by quantum annealing [9, 10, 11] or by an iterative procedure called Quantum Approximate Optimization Algorithm (QAOA) [12, 13]. These algorithms must run on quantum hardwares that typically contain an array of qubits plus control and readout components.

Reality check from hardware limitations. While quantum algorithms such as quantum annealing and QAOA hold great promise to surpass classical optimization algorithms, their performance is currently limited by the hardware. For example, in the state-of-the-art Advantage annealer from D-Wave, each qubit can couple to at most 15 nearby qubits [14], thus many QUBO problems with denser or longer-range connectivity between qubits become incompatible. QAOA, on the other hand, is restricted by the qubit coherence times and quantum circuit depth to small systems with up to the order of 40 qubits [15, 16]. Thus there is still a very long way to go before these hardware-based quantum optimization algorithms reach their full potential to solve industrial-scale optimization problems. The reality of hardware constraints begs the question, can we gain quantum advantage without the need of qubits? The answer is, perhaps counterintuitively, yes.

The emerging field of quantum-inspired algorithms. Here the term "quantum-inspired" describes approaches that simulate the quantum behaviors – such as superposition, tunneling and entanglement – of given Hamiltonians on classical computers [17]. It is well known that finding the exact solution of interacting quantum systems usually demand compute growing exponentially with the problem size N. But for certain classes of spin Hamiltonians, the low-energy quantum states can be efficiently expressed as tensor networks and solved within a time that is some polynomial of N [18, 19]. Tensor network is essentially a graphical language to depict, compute, and understand interacting quantum systems. One of its key ideas is to

decompose complicated quantum objects with many legs (high rank tensors) into smaller building blocks with only a few legs (lower rank tensors such as matrices, see the illustration in Section 3). Manipulating these objects then amounts to playing "*quantum lego*" [20] – even high school students can do it. In recent years, tensor networks have become a powerful and versatile tool not only in quantum physics and chemistry but also in machine learning and combinatorial optimization.

Difficulty in finding the ground state. Once a QUBO problem is mapped to a spin Hamiltonian, one can try to solve the physics problem of finding its ground state by tensor network calculations. Such quantum-inspired approach, recently explored in [21, 22], has the potential to solve larger optimization problems beyond the reach of present day qubits-based quantum algorithms. It must be emphasized, however, that finding the ground state of the spin Hamiltonians derived from QUBO remains a hard problem. These Hamiltonians tend to be "glassy" and feature dense, long-range interactions similar to Ising spin glasses [23]. In contrast, the interactions in most well understood spin models are uniform and local, e.g. only between nearest neighbors. In order to reach the ground state, the authors of [21, 22] update the tensor network states iteratively using imaginary-time evolution discretized in small time steps. This brute-force scheme requires heavy computation, because for each interaction term between two far-apart spins in the Hamiltonian, the time-evolution operation at every step involves a long sequence of swap operations involving all the spins that lie in between, slowing the calculation down to a crawl. Presently, it is not clear whether the global minimum for hundreds of spins can be reached in this manner. Can we invent a new, much more efficient, quantum-inspired algorithm to solve QUBO problems?

The achievements of this work. In this research project, we design a novel quantum-inspired algorithm and demonstrate that it successfully solves a wide variety of QUBO problems, e.g. Sudoku puzzles and MaxCut puzzles with up to 251 nodes and $3,265$ edges. We present a technical description of the algorithm in Section 3. Roughly speaking, we represent the many-spin wave function using a tensor train, and set up a discrete driving schedule with multiple stops. At each stop, the train is swept back and forth by a procedure bearing the esoteric name of Density Matrix Renormalization Group (DMRG), so each car (tensor) of the train is updated to reflect the interaction between the spins. The final stop of the train reveals the ground state and the corresponding solution of the QUBO problem can be recovered. The following list highlights the capabilities of the algorithm, dubbed "Tensor Train Express":

- It runs on a laptop, and does not need qubits or expensive quantum machines.
- It easily handles hundreds of spins, exceeding the capabilities of the state-of-the-art QAOA.
- It reaches the ground state, not just local minima, of Ising spin glass with dense, long-range couplings.
- For fun, it solves Sudoku puzzles from The New York Times, which have defied quantum annealing.
- For work, it solves a suite of MaxCut instances of varied types and sizes from the Biq Mac library.
- Discrete driving is significantly faster than continuous driving used in quantum annealers.
- The sweeping update is more efficient than costly imaginary time evolution.
- The explicit knowledge of spin wave functions elucidates why and how the algorithm works.

The algorithm's performance is impressive and surprising, given the prevailing belief that to gain the quantum advantage, one must possess more and better qubits. It was naive to ask "Can quantum mechanics help solve the Sudoku in the today's Times even though as a high school student I have got no qubits to play around?" But in hindsight, my ignorance about the "conventional wisdom" freed me to think independently about the spin physics behind combinatorial optimization and to play with the "*quantum lego*" toolbox to manipulate the spin wave functions.

Key innovation. While all individual ingredients of our algorithm are previously known, the key innovation leading to the success of our algorithm lies in the fusion between a discrete driving schedule and

the DMRG sweeping procedure. For example, the usage of a driver is common in quantum annealing and QAOA, albeit typically in conjunction with a continuous driving process. Similarly, tensor train and DMRG are widely used to study one-dimensional quantum systems, but mostly with local interactions. Prior to this work, neither DMRG nor discrete driving has been applied to QUBO, and there is no reliable solver for Ising spin glass with inhomogeneity and long-range interactions. This work fills these gaps.

2 Sudoku as a system of interacting spins

Sudoku is a widely enjoyed logic-based number puzzle played on a 9×9 grid, which is further divided into nine smaller 3×3 sub-grids. The goal of the game is to fill the empty cells with digits from 1 to 9 in such a way that each row, each column, and every 3×3 block contains all nine digits exactly once, without repetition. The puzzle begins with a set of prefilled cells, known as *clues*. Puzzles with fewer clues tend to be harder, but they can be solved efficiently by classical algorithms such as backtracking or stimulated annealing. Why care about these puzzles? Things become more interesting when the game is generalized to an $n^2 \times n^2$ grid, where n is the size of each block ($n = 3$ for the standard Sudoku). This generalization elevates Sudoku to a compelling example of computational complexity: it is NP-hard [24], implying that the problem becomes computationally intractable for classical algorithms as n increases, and Another Solution Problem (ASP) complete, meaning that even deciding whether a given instance has more than one solution is computationally hard [25]. Thus, Sudoku represents a class of difficult problems for which quantum or quantum-inspired algorithms are needed as n increases.

In this section, we formulate Sudoku as a quadratic optimization problem, and then map it to a physical system of interacting spins. The procedure serves three purposes. (1) It firmly places Sudoku under the big umbrella of QUBO problems. As a result of their structural parallels, a quantum-inspired algorithm designed for solving Sudoku can be readily adapted to tackle a wide range of combinatorial optimization challenges. (2) The resulting QUBO problems turn out hard to solve and beyond the reach of quantum annealing [26] or QAOA. In fact, they represent the worst-case scenario where the cost function is entirely due to constraints, leading to non-local couplings among many binary variables. But the validity of the solution is almost trivial to verify - only a series of checks are needed. For these two reasons, Sudoku is an exacting benchmark to assess the quality of QUBO solvers: the algorithm better find the exact ground state. Merely reducing the energy to near-optimal values will violate some of the constraints, leaving the board a glaringly embarrassing mess! (3) The corresponding spin models are characterized by long-range Ising couplings and an inhomogeneous on-site magnetic field. These features will directly influence our algorithm design.

The cost function. Our QUBO formulation follows the approach outlined by Mücke [26], but our choice of the constraints and cost function differs in key ways. Let $z_{i,j,k}$ be a binary variable that takes the value 1 if the cell in the ith row and jth column of the Sudoku grid is assigned the number k, and 0 otherwise, where $i, j, k = 1, 2, \ldots, 9$. The rules of Sudoku can then be encoded through the following four sets of constraints:

$$\sum_{k} z_{i,j,k} = 1, \quad \forall i, j \tag{1}$$

$$\sum_{i} z_{i,j,k} = 1, \quad \forall j, k \tag{2}$$

$$\sum_{j} z_{i,j,k} = 1, \quad \forall i, k \tag{3}$$

$$\sum_{(i,j)\in b} z_{i,j,k} = 1, \quad \forall b, k \tag{4}$$

The first constraint (1) enforces that each cell (i, j) must be assigned exactly one number. The second constraint (2) ensures that within each column j a given number k appears exactly once, and the third

constraint (3) similarly requires each number k appears once in each row i. The fourth constraint (4) sums over all cells (i, j) within the same block b to enforce that each number k occurs only once per block.

Following standard practice [3], we convert these constraints into penalty terms and include them in the cost function. For instance, the penalty corresponding to constraint (1) is given by

$$P_1 = \lambda \sum_{i,j} \Big(\sum_k z_{i,j,k} - 1 \Big)^2 = \lambda \sum_{i,j} \Big(1 - \sum_k z_{i,j,k}^2 + \sum_k \sum_{k' \neq k} z_{i,j,k} z_{i,j,k'} \Big) \tag{5}$$

where we used the property $x = x^2$ for a binary variable x. Besides the constant, the other terms within the parentheses in (5) are quadratic in z, and the off-diagonal coupling is dense – each $z_{i,j,k}$ is coupled to every $z_{i,j,k' \neq k}$. The other penalties, P_2 through P_4, can be applied in a similar manner for constraints (2) through (4). The cost function we seek to minimize is simply the sum

$$f(\{z_{i,j,k}\}) = P_1 + P_2 + P_3 + P_4 \tag{6}$$

Here, we assigned the same penalty weight $\lambda > 0$ to all four constraint terms, departing from [26]. The binary variables $z_{i,j,k}$ can then be organized into a single vector Z with its elements defined by $Z_{81i+9j+k} = z_{i,j,k}$. Then the cost function f assumes the standard QUBO form

$$f\left(\{Z_m\}\right) = \lambda \Big(c_1 + \sum_{m,n=1}^{9^3} Z_m Q_{m,n} Z_n \Big) \tag{7}$$

where the constant $c_1 = 4 \times 9^2$, and the QUBO matrix Q is constructed by collecting both the diagonal and off-diagonal couplings among the elements of Z. The QUBO formulation of Sudoku is not efficient and yields a large number of binary variables. The cost function arises solely from constraints. Clearly, the global minimum of f is 0, as any valid solution satisfies all constraints so all four penalty terms vanish. Due to our uniform choice of penalty structure, the cost function f is proportional to λ. From this point forward, we measure f in units of λ, effectively setting $\lambda = 1$ so that it drops out of the formulation.

Clue propagation and clamping. The clues provided by the puzzle fix the values of certain variables $z_{i,j,k}$, significantly reducing the number of free variables from the original 9^3. This reduction process is known as *clamping*, as described by Mücke [26]. Let N_c denote the number of clues (pre-filled cells) given. For each cell (i^*, j^*) with a given value k^*, we first set $z_{i^*,j^*,k^*} = 1$, and then propagate this clue by applying the constraints (1) through (4), setting

$$z_{i^*,j^*,k \neq k^*} = 0 \tag{8}$$

$$z_{i \neq i^*,j^*,k^*} = 0 \tag{9}$$

$$z_{i^*,j \neq j^*,k^*} = 0 \tag{10}$$

$$z_{(i,j) \in b^*,k^*} = 0 \tag{11}$$

where $(i, j) \in b^*$ denotes all other cells within the same block that contains (i^*, j^*). After all clues have been applied and the affected z-values are "clamped" to either 1 or 0, we collect the remaining N_s free binary variables and organize them into a new vector X. The QUBO problem (7) then reduces to

$$f\left(\{X_m\}\right) = c_1 + c_2 + \sum_{m,n=1}^{N_s} X_m J_{m,n} X_n \tag{12}$$

Corresponding to the reduction $Q \to J$, the reduced QUBO matrix J is derived from the original matrix Q via the transformation

$$J = U^\top Q U \tag{13}$$

where the transformation matrix U and the constant c_2 can be found in [26]. For an intermediate-level Sudoku puzzles with 24 to 26 clues, the reduced QUBO problems typically have N_s exceeding 200.

Mapping to Ising spin glass. Every QUBO problem can be mapped to an Ising-type classical spin model. For each binary variable X_m, we define a corresponding Ising spin-z variable

$$S_m^z = X_m - \frac{1}{2} \tag{14}$$

which takes the value $+\frac{1}{2}$ (referred to as spin-up, $\uparrow$) when $X_m = 1$, and $-\frac{1}{2}$ (spin-down, $\downarrow$) when $X_m = 0$. This change of variables allows the cost function f to be reinterpreted as the Hamiltonian, i.e., the energy function, of N_s spins

$$H_z = f\left(\{X_m\}\right) = c_1 + c_2 + c_3 + \sum_{m\neq n}^{N_s} S_m^z J_{m,n} S_n^z + \sum_{m=1}^{N_s} h_m^z S_m^z \tag{15}$$

where the constant term c_3 resulting from the change of variables is defined by

$$c_3 = \frac{1}{4}\sum_{m=1}^{N_s} J_{m,m} + \frac{1}{4}\sum_{m,n=1}^{N_s} J_{m,n} \tag{16}$$

In addition to the Ising interaction $J_{m,n}$ that couple spin m and spin n, the Hamiltonian H_z also includes an inhomogeneous (i.e., m-dependent) magnetic field in the z-direction that couples linearly to S_m^z

$$h_m^z = \frac{1}{2}\sum_{n=1}^{N_s} \left(J_{m,n} + J_{n,m}\right) \tag{17}$$

To visualize this model, it is useful to imagine the spins are arranged along a fictitious one-dimensional chain, with sites indexed by m. At each site m, the local magnetic field takes the value h_m^z, and spins at sites m and n interact with strength $J_{m,n}$. We refer to Hamiltonian (15) derived from Sudoku loosely as an Ising spin glass. Strictly speaking, the term "spin glass" describes models where the couplings $J_{m,n}$ are random. One example is the celebrated Sherrington-Kirkpatrick model [27]:

$$H_{SK} = h^z \sum_m S_m^z + \sum_{m\neq n} J_{m,n} S_m^z S_n^z \tag{18}$$

which played an influential role in the early history of neural networks and complex systems. In H_{SK}, the magnetic field h^z is a constant across all sites, while the couplings $J_{m,n}$ are allowed for every pair of spins m and n with values randomly drawn from a Gaussian distribution. In contrast, in H_z the J matrix is determined by the game's rules and clues, and is therefore non-random.

Solving the Sudoku puzzle is now equivalent to finding the ground state of the Hamiltonian (15) – that is, identifying the spin configuration that minimizes the energy of the N_s interacting spins in a magnetic field. A correct solution corresponds to zero energy and features exactly $(81 - N_c)$ spins pointing up. While the spin formulation offers a new perspective, it does not simplify the problem: finding the ground state of an Ising spin glass remains NP-hard. In fact, recently Lucas showed that many NP-complete and NP-hard problems can be formulated as Ising spin glass-type Hamiltonians [7]. The Ising formulation is appealing because it opens the door to developing quantum and quantum-inspired algorithms. The "trick" lies in promoting an Ising spin glass to a quantum model.

Promotion to a quantum spin model. We generalize H_z by adding a transverse magnetic field h^x in the x-direction that couples to the spin-x operator S^x, leading to the total Hamiltonian

$$H_{\text{total}} = aH_x + bH_z \tag{19}$$

Here, a and b are real control parameters to be specified later. We allow the transverse field h^x to be site dependent,

$$H_x = \sum_m h_m^x S_m^x \tag{20}$$

The spin operators are defined in the standard way. In the eigenbasis of S^z, they take the matrix form

$$S^z = \frac{1}{2}\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad S^x = \frac{1}{2}\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{21}$$

To simply the notation, we set the reduced Planck constant $\hbar = 1$ throughout.

To summarize, we have promoted the classical spin model (15) to a quantum spin model (19). We call the term H_x the driver Hamiltonian because the transverse field drives the spins to precess and flip. In other words, it induces quantum tunneling between the spin-up and spin-down states, $|\uparrow\rangle \leftrightarrow |\downarrow\rangle$. Consequently, the spin can be in the superposition state such as

$$|\pm\rangle = \frac{1}{\sqrt{2}}\left(|\uparrow\rangle \pm |\downarrow\rangle\right) \tag{22}$$

Furthermore two spins can become entangled, which was impossible in H_z alone.

Characterizing the Ising spin glass. To figure out how the driver Hamiltonian H_x can be leveraged to solve the spin problem, it is essential to characterize the properties of the local fields h_m^z and the coupling matrix $J_{m,n}$. We first notice that for Sudoku-derived spin systems, the longitudinal magnetic field h^z fluctuates significantly from site to site. Figure 1 shows the site-dependent values h_m^z for three intermediate-level Sudoku puzzles published in The New York Times on January 2 (diamonds), January 12 (circles), and January 14 (pluses), 2025. In all cases, the field exhibits highly irregular, large-amplitude variations. According to the term $\sum_m h_m^z S_m^z$ in H_z, spins at sites with large positive h_m^z favor pointing downward (i.e., $S_m^z = -\frac{1}{2}$), while those at sites with smaller or negative h_m^z prefer to point upward. This "rugged" landscape of h^z, as illustrated in Figure 1, leads to spin configurations that may appear random, markedly different from the ordered ferromagnetic or antiferromagnetic patterns seen in simpler spin models.

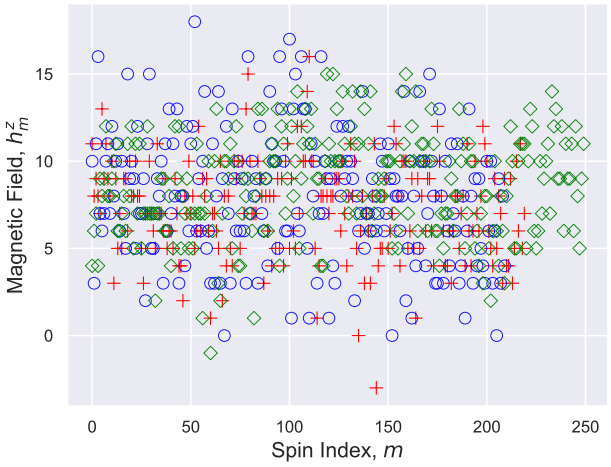


Figure 1: The spin representations of three Sudoku puzzles published in The New York Times on January 2 (diamond), January 12 (circle), and January 14 (plus), 2025, are shown. The mean and standard deviation of the magnetic field h_m^z are as follows: 8.7 ± 3.0 for the January 2 puzzle (diamond), 8.0 ± 3.7 for January 12 (circle), and 7.3 ± 2.7 for January 14 (plus).

Next, let us examine the spin-spin interactions $J_{m,n}$. Figure 2 shows the Ising couplings between the spins in the January 12 Sudoku puzzle. Each spin is represented by a dot, arranged along an elliptical track for clearer visualization. A line connecting spins m and $n \neq m$ indicates a nonzero coupling $J_{m,n}$. Many interactions are clearly long-ranged, extending beyond close neighbors. Closer inspection reveals

that there are a total of $1,261$ connections among the $N_s = 210$ spins, and all couplings are of equal strength, with $J_{m,n} = 2$. A positive coupling constant J corresponds to antiferromagnetic interactions, which energetically favor opposing spin orientations. However, satisfying the competing preferences of all these couplings simultaneously is generally impossible, a phenomenon known as frustration [28]. These two factors – long-range interactions and frustration – greatly complicate the spin optimization problem. Our next section presents a quantum-inspired algorithm that overcomes these difficulties.

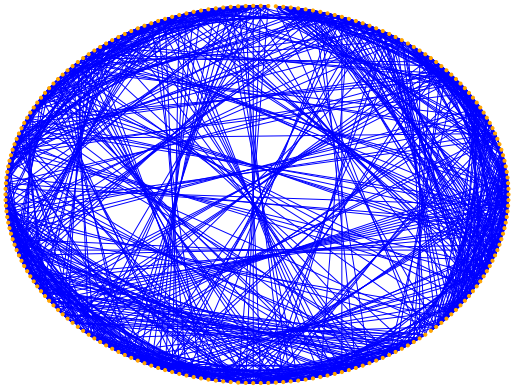


Figure 2: The dense, long-range couplings among the 210 spins for the January 12, 2025 Sudoku puzzle. Each line represents an Ising coupling $J_{m,n} = 2$, which favors spin-m and spin-n being antiparallel. Each spin can point either up or down. Finding the lowest-energy spin configuration corresponds to solving the Sudoku puzzle.

3 Tensor Train Express: the quantum-inspired algorithm

Driving schedule. Before presenting the algorithm, we briefly introduce its three main ingredients. The first is driving, a notion borrowed from quantum annealing [9, 10, 11], a popular heuristic approach to QUBO. An annealer is engineered to follow a time-dependent Hamiltonian of the form

$$\mathcal{H}(t) = s(t)\mathcal{H}_0 + (1 - s(t))\,\mathcal{H}_1 \tag{23}$$

where t is time measured in some proper unit and the annealing schedule $s(t)$ is a continuous function with boundary conditions $s(t = 0) = 1$ and $s(t = 1) = 0$. At $t = 0$, the initial Hamiltonian $\mathcal{H}(t = 0) = \mathcal{H}_0$ has a known ground state $|\psi_0\rangle$ that can be prepared experimentally. Our goal is to find the ground state $|\psi_1\rangle$ of the QUBO Hamiltonian $\mathcal{H}_1$. Provided that $s(t)$ varies at a sufficiently slow rate, the system's state will evolve smoothly from $|\psi_0\rangle$ to $|\psi_1\rangle$, so that the final state $|\psi_1\rangle$ can be measured to extract the QUBO solution. For our problem, we can set $\mathcal{H}_0 = H_x$, $\mathcal{H}_1 = H_z$, and choose the initial state as

$$|\psi_0\rangle = |-\rangle^{N_s} = \bigotimes_{m=1}^{N_s} \frac{1}{\sqrt{2}} \left(|\uparrow\rangle_m - |\downarrow\rangle_m\right) \tag{24}$$

In practice, the success of quantum annealing delicately hinges on several factors. First, during the time evolution, if the energy gap (the difference between the ground state and the first excited state) of $\mathcal{H}(t)$ becomes very small, the time required for quantum annealing could be exponentially large. For a finite annealing time, the final state will no longer be the ground state of $\mathcal{H}_1$ so the algorithm fails. Second, a faithful implementation of $\mathcal{H}(t)$ requires full connectivity between the spins. For example, the dense

connection pattern in Figure 2 differs from D-Wave's Pegasus graph, where each qubit (spin) is only coupled to 15 other qubits in a periodic pattern [14]. Consequently, many QUBO problems must undergo an additional process called embedding to fit into the annealer's hardware [14]. Third, accessing the quantum annealers often comes with cost or restrictions on the running time, making them impractical for many users with limited resources.

These limitations motivate us to develop a quantum-inspired algorithm for QUBO problems. The term "quantum-inspired" is used in line with the review article [17] to describe algorithms that draw inspiration from quantum computation and information techniques but operate on classical data (e.g., from Sudoku or MaxCut problems) and run on classical computers rather than quantum hardware. Specifically, we map the classical problem in equation (15) to a quantum spin problem in equation (19) and simulate the quantum spin system classically using its tensor network representation [18]. We shall see that the quantum-inspired approach can circumvent some of the drawbacks of qubits-based quantum algorithms.

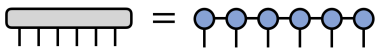
Tensor train. Tensor networks provide an intuitive and powerful language to do quantum mechanics. One of the most well-understood tensor networks is the tensor train, also known as Matrix Product State (MPS) [18, 19]. In our context, the quantum state of an N-spin system can be expanded as

$$|\psi_N\rangle = \sum_{s_i} T^{s_1,s_2,\ldots,s_N} |s_1, s_2, \ldots, s_N\rangle \tag{25}$$

where $|s_1, s_2, \ldots, s_N\rangle$ represents the product state in which the i-th spin is in the eigenstate $|s_i\rangle$ with eigenvalue $s_i = \pm\frac{1}{2}$, and $T^{s_1,s_2,\ldots,s_N}$ is a rank-N tensor with N free indices. For large N, directly manipulating T becomes prohibitively expensive, as the number of its components, 2^N, grows exponentially. The idea of tensor train decomposition is to express T as the product of a "train" of lower-rank tensors denoted by A^{s_i}

$$T^{s_1,s_2,\ldots,s_N} = \sum_{\alpha_i} A^{s_1}_{\alpha_1} A^{s_2}_{\alpha_1,\alpha_2} A^{s_3}_{\alpha_2,\alpha_3} \ldots A^{s_N}_{\alpha_{N-1}} \tag{26}$$

Here each A^{s_i} is a matrix, with the two lower indices being row and column number, respectively. (Note that A^{s_1} and A^{s_N} are special, as they only have one lower index and correspond to row and column matrices, respectively.) The lower indices, $\alpha_i = 1, 2, \ldots, D$, are summed over and the integer D is called the bond dimension. Tensor networks are pleasant to work with because all the cumbersome equations and operations can be represented graphically. For instance, (26) becomes



The many-spin wave function T can be visualized as a train consisting of N cars (carriages), with each car corresponding to a matrix A with dimension D [29],

For readers who prefer the Dirac notation, consider the following example with $N = 4$ and $D = 2$ [30],

$$|W\rangle = |\uparrow\uparrow\uparrow\downarrow\rangle + |\uparrow\uparrow\downarrow\uparrow\rangle + |\uparrow\downarrow\uparrow\uparrow\rangle + |\downarrow\uparrow\uparrow\uparrow\rangle = \begin{pmatrix} |\uparrow\rangle & |\downarrow\rangle \end{pmatrix} \begin{pmatrix} |\uparrow\rangle & |\downarrow\rangle \\ 0 & |\uparrow\rangle \end{pmatrix} \begin{pmatrix} |\uparrow\rangle & |\downarrow\rangle \\ 0 & |\uparrow\rangle \end{pmatrix} \begin{pmatrix} |\downarrow\rangle \\ |\uparrow\rangle \end{pmatrix} \tag{27}$$

This W state of four spins is constructed from the product of four matrices, i.e., it is a train with four cars of size 2. The tensor train representation in equation (26) is exact, provided that D is sufficiently large. In practice, a finite value of D can be chosen so that the tensor train serves as a good approximation to T, treating all matrix elements as variational parameters. A major advantage of the tensor train representation is that the total number of parameters scales as ND^2, a significant reduction from 2^N.

Sweeping. The ground states of many one-dimensional quantum systems can be found accurately in the tensor train representation using an algorithm called Density Matrix Renormalization Group (DMRG) [31, 32]. DMRG can be viewed as an iterative variational procedure that optimizes the tensors by "sweeping" through the train multiple times to progressively lower the energy. The procedure starts with an initial set of A^{s_i} and focuses on one bond, i.e., two neighboring cars A^{s_i} and $A^{s_{i+1}}$ (this subsystem is then described by a density matrix), at a time. The components of these two matrices are adjusted by applying the Hamiltonian and treating the rest of the train as an effective environment [32]. Once these two cars are updated ("renormalized"), the algorithm moves on to the next bond, sweeping from left to right and then back. The number of sweeps required to converge to the ground state depends on the Hamiltonian, the initial tensors, and parameters such as D. DMRG's time complexity is ND^3. We choose DMRG as the engine of our QUBO solver because it is numerically stable and available in several open-source libraries. In addition, as we mentioned earlier, a direct approach based on imaginary-time evolution is too slow for Hamiltonians with dense, long-range couplings.

The final piece of the algorithm design is to set a driving schedule for the tensor train. Instead of the traditional, slow, continuous annealing schedule $s(t)$, we adopt a discrete driving schedule consisting of M stops that interpolate between the starting point H_x and the destination H_z,

$$\boxed{H_1 = H_x} \;\rightarrow\; \boxed{H_2} \;\rightarrow\; \cdots \;\rightarrow\; \boxed{H_{M-1}} \;\rightarrow\; \boxed{H_M = H_z}$$

The Hamiltonian at the i-th stop is given by

$$H_i = a_i H_x + b_i H_z, \quad i = 1, 2, \ldots, M, \tag{28}$$

where the driving parameters a_i and b_i control the mixing between the problem Hamiltonian H_z and the driving term H_x at the i-th stop. The set $\{a_i, b_i\}$ specifies a discrete driving path with $a_1 = 1, b_1 = 0$ and $a_M = 0, b_M = 1$. Our train is an express because it makes only a few stops (e.g. $M = 5$ or 10) and it does not spend a lot of time at each stop, thanks to the efficiency of the DMRG sweeps.

Now we are ready to put everything together. The algorithm proceeds as follows.

Algorithm 1 Tensor Train Express

choose an initial state $|\psi_0\rangle$ as a tensor train
for $i = 1$ to M **do**
  update the tensor train wave function for H_i from $|\psi_{i-1}\rangle$ to $|\psi_i\rangle$ by DMRG sweeps,

$$|\psi_{i-1}\rangle \xrightarrow{\text{DMRG sweeps for } H_i} |\psi_i\rangle$$

end for
return $|\psi_M\rangle$

Once the algorithm succeeds in finding the ground state of the Ising spin glass H_z, the corresponding spin configuration, which encodes the QUBO solution, can be computed from $|\psi_M\rangle$. We shall show in Section 4 and Section 5 that the combination of tensor train, DMRG sweeps, and a discrete driving schedule yields a more tractable and efficient approach for QUBO problems.

Why a direct approach fails. One might wonder why we don't simply ask DMRG to find the ground state of H_z directly, thereby bypassing all the preceding steps up to $i = M-1$. The reason is that for DMRG to work effectively, a good guess for the starting state is usually required. While DMRG is an excellent tool for finding the ground state of one-dimensional gapped Hamiltonians with local interactions, it can easily get trapped in local minima when applied to glassy or long-range Hamiltonians such as H_z. In our algorithm, all the preceding steps are designed to successively steer the tensors along a discrete path

$$|\psi_0\rangle \;\rightarrow\; |\psi_1\rangle \;\rightarrow\; \cdots \;\rightarrow\; |\psi_M\rangle$$

At each step, the DMRG sweeps bring the tensors close to the respective ground state of the intermediate Hamiltonians with increasing weight on H_z, so that $|\psi_{M-1}\rangle$ is sufficiently close to the ground state of H_z.

4 Solving Sudoku puzzles

In this section, we test the performance of our algorithm by applying it to solve the Sudoku puzzles published daily by The New York Times [33]. These puzzles are archived and readily accessible online [34]. Table 1 presents a few examples. Each row lists a puzzle, uniquely identified by its date of publication and assigned difficulty level, where "m" stands for intermediate and "h" for hard. Also displayed for each puzzle are the number of clues (N_c), the number of up spins in the correct solution ($N_\uparrow$), the total number of spins (N_s) in the corresponding Ising spin glass H_z, and the constant energy shift $c_1 + c_2 + c_3$ in H_z (see (15)). A successful run of the algorithm should drive the energy down to zero. For verification, we take the converged spin configuration, translate it back to the X variables, then to the full QUBO vector Z, and print the completed Sudoku board to ensure that all numbers fit correctly and none of the constraints are violated.

Puzzle date	Level	N_c	$N_\uparrow$	N_s	$c_1 + c_2 + c_3$
January 8, 2024	h	24	57	211	368.5
December 10, 2024	m	26	55	200	359.0
January 2, 2025	m	22	59	250	520.5
January 8, 2025	m	23	58	232	461.5
January 12, 2025	m	27	54	210	426.5
January 14, 2025	m	23	58	232	457.0
January 15, 2025	m	25	56	210	380.0

Table 1: A few examples of Sudoku puzzles solved by the quantum-inspired algorithm, in chronological order according to their publication date in The New York Times. These puzzles vary in the number of clues (N_c), total number of spins (N_s), and other parameters; see the main text for details.

Exhibit A, Jan. 8, 2024. We discuss a few examples to illustrate how the algorithm parameters are chosen for the calculations. For the January 8, 2024 puzzle considered in [26] and listed in the first row of the table, we start from the state (24), which is an equal-weight superposition of all S^z basis states, and use a homogeneous transverse field $h_m^x = 0.7$ in the driver Hamiltonian H_x. The driving schedule is linear:

$$b_i = \frac{i}{M}, \qquad a_i = 1 - b_i, \qquad M = 10.$$

For DMRG, we set the maximum bond dimension $D = 60$ and perform $N_{sw} = 5$ sweeps at each stop i. With these settings, the proposed algorithm successfully solves the puzzle. A DMRG sweep may take 0.1 to 8 seconds (depending on which H_i and $|\psi_i\rangle$). And the algorithm spends most of its time on the preceding steps $i < M$ during which the tensor train wave function remains a big object with $D = 60$ and the computation is expensive. Only in the last step ($i = M$) is the problem Hamiltonian H_z considered. This final stage is efficient and fast: the tensors typically converge within three DMRG sweeps, and the bond dimension drops quickly from 60 to 1, meaning the ground state becomes a product state, as expected. The quick convergence is credited to the proximity of $|\psi_{M-1}\rangle$ to the ground state $|\psi_M\rangle$ as a result of steering.

The choice of initial state and the driving schedule used here resemble those in quantum annealing. This is intentional so we can take advantage of superposition and quantum tunneling (spin flips). However, the similarity ends there. Our algorithm does not follow any adiabatic time evolution, so the "trajectory" of the quantum state from $|\psi_0\rangle$ to $|\psi_M\rangle$ is fundamentally different. The algorithm is largely insensitive to the choice of $|\psi_0\rangle$, as long as it serves as a reasonable starting point for the DMRG solution of H_1, which is predominantly H_x with only a small admixture of H_z (note in this case $H_1 \neq H_x$).

To demonstrate the flexibility of the algorithm, we take the same puzzle but start from a random tensor train state with bond dimension 3 and choose $D = 20$ for all subsequent steps. The puzzle is solved successfully at a much faster pace, with each sweep taking about 0.8 seconds. As shown in Figure 3, the energy (black dots) steadily drops to zero. To track the spins' behavior at each step, we define the total spin-z operator measured from its ground-state value $\left(N_\uparrow - \frac{N_s}{2}\right)$,

$$S^z = \sum_{m=1}^{N_s} S^z_m - \left(N_\uparrow - \frac{N_s}{2}\right) \tag{29}$$

and evaluate its expectation value at the i-th state $|\psi_i\rangle$. The red crosses in Figure 3 show the expectation values of S^z (magnified by a factor of 4 to be clearly visible). Figure 3 also includes the expectation values (blue circles) of the total spin-x operator defined by

$$S^x = \sum_{m=1}^{N_s} S^x_m \tag{30}$$

As the weight of H_x is decreased and the weight of H_z is increased during the driving process, S_x gradually approaches zero, while S_z converges to its ground state value. An interesting period is the interval between driving steps 2 and 4 when the spins transition from being predominantly aligned along the x-axis to aligning along the z-axis. The inflection point at this transition appears crucial for the success of the algorithm.

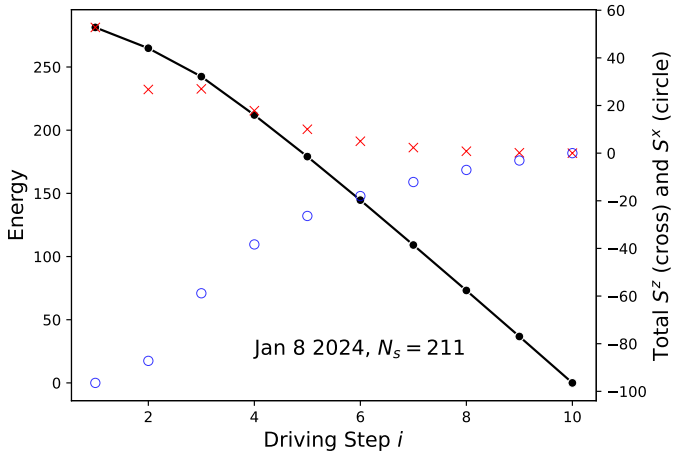


Figure 3: How the January 8, 2024 puzzle gets solved: As the driving field h_x is reduced, the energy gradually drops to zero. The total spin along the x-direction vanishes, while the total spin along the z-direction approaches its ground state value. The bond dimension is $D = 20$ and the driving field is set to $h_x = 0.75$.

If the primary concern is the time it takes to solve the 9×9 Sudoku puzzles, our algorithm pales in comparison to classical algorithms that build on logic and exploit the structure of the problem. In contrast, the QUBO approach to Sudoku puzzles, as noted earlier, is not efficient and resembles more of a blind search. It is treated simply as another QUBO matrix for the sake of benchmarking, and our solution strategy does not care at all about the meaning of the original constraints: instead, we focus on ability to solve the broad class of QUBO problems in general. However, as the board size increases, the algorithm is expected to outperform conventional methods since the calculation scales as $N_s D^3 M$.

Exhibit B, Jan. 15, 2025. Other puzzles listed in the table are solved in a similar manner. Since they differ in N_s, h^z_i, and $J_{i,j}$, directly copying the algorithm parameters from a previously successful example

may not work. Occasionally, the final state might become trapped in a local minimum, with energy higher than zero (indicating that some constraints are violated). In such cases, the solution is simple: restart from a different $|\psi_0\rangle$. A useful strategy to "shake" the system loose to escape local minima is to tune the driving term H_x, for example, by adding random fluctuations to the transverse field,

$$h_m^x = h_x + \eta_m \tag{31}$$

where h_x is the homogeneous part and η_m is a random variable at site m drawn from a uniform distribution within the interval $(-\eta, \eta)$. For example, in solving the January 15, 2025 puzzle in the last row of the table, we set $D = 20$, $h_x = 1.3$, and $\eta = 0.2$ at each driving step. The algorithm once again succeeds in 10 steps.

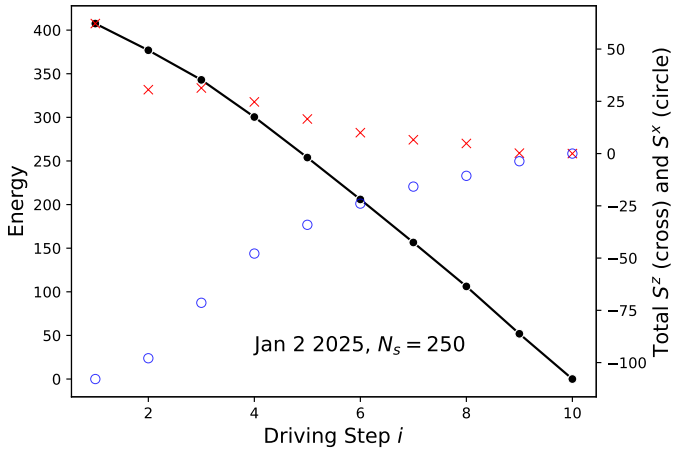


Figure 4: The solution process for another puzzle with fewer clues and more spins: $D = 20$, $h_x = 1$.

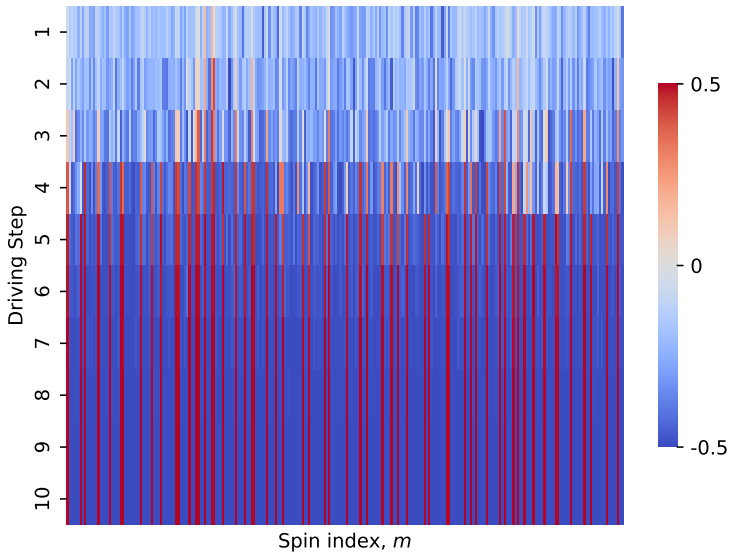


Figure 5: The evolution of the 250 spins during the solution process of the January 2, 2025 puzzle. S_m^z is color-coded between red (spin-up) and blue (spin-down).

Exhibit C, Jan. 2, 2025. Within Table 1, the January 2, 2025 puzzle has the least number of clues

and the maximum number of spins, $N_s = 250$. The broad features of its solution process can be appreciated from Figure 4, which is comparable to our earlier example in Figure 3. In both cases, the energy decreases with i, but not exactly in a linear fashion. A more detailed picture of the solution process is provided by the "heat map" in Figure 5. Here, the horizontal axis represents the spin index $m = 1$ to 250, going from left to right. The vertical axis corresponds to the driving step $i = 1$ to $M = 10$, going from top to bottom. On each row, shown in false color, are the expectation values of the spin-z operator, S^z_m, computed at the corresponding step. We observe that in the initial steps, due to the h_x field, S^z_m appears quite random. By step 4, the spins begin to settle into their "correct" states, i.e., the expectation values of S^z_m align with the final solution. The wave function at this stage still has a large $D = 20$, and the energy remains relatively high. In the last step, after only three DMRG sweeps, the correct solution emerges as a product state, with red and blue colors representing $|\uparrow\rangle$ and $|\downarrow\rangle$, respectively.

This rapid drop in D during the last step of the algorithm is in sharp contrasts with conventional DMRG workflows, where D is often increased during the calculation. This is an indication that DMRG is now being used in an unconventional, creative manner. To double check the solution, we read out the spin configuration, convert it back first to the QUBO vector X, and then to the original binary variables in Z, and print the result on the Sudoku board (with the clues shown in blue). Here is what we get:

2	8	1	3	5	4	6	7	9
9	7	3	8	1	6	5	2	4
5	6	4	7	9	2	8	1	3
1	3	9	4	6	8	2	5	7
8	2	5	9	7	3	1	4	6
6	4	7	5	2	1	9	3	8
4	9	2	6	3	5	7	8	1
3	5	6	1	8	7	4	9	2
7	1	8	2	4	9	3	6	5

This solution is valid yet differs from the answer given in [34]. Our algorithm has independently discovered a new solution!

The benefit of knowing the wave function. Compared to qubit-based quantum algorithms where measurement collapses the quantum state, a major advantage of the quantum-inspired approach is its direct access to the many-spin wave function obtained from tensor network calculations. The wave function offers the ultimate insight into the spin system under driving. As shown in our case studies from Figures 3 to Figure 5, the detailed diagnostic data on the solution process translate into an accurate understanding of the role played by each parameter, which is critical for tuning algorithm parameters.

5 Solving MaxCut problems

In this section, we apply the algorithm to MaxCut, one of the most well-known problems in combinatorial optimization [35]. The goals are twofold. First, by testing it on new problem instances, we demonstrate the versatility of the algorithm and its capability to solve QUBO problems with varying characteristics. Second, by analyzing its performance across different problem sizes and types, we illustrate how the algorithm can be fine-tuned to optimize its performance for specific instances.

MaxCut as Ising spin glass. The MaxCut problem is defined on an undirected graph $G = (V, E)$, where V represents a set of vertices and E represents a set of edges. An edge connecting vertex i and vertex j is denoted by $(i, j) \in E$ and carries a weight $w_{i,j}$. The objective is to partition V into two sets, $V = A \cup \bar{A}$ with $A \cap \bar{A} = \emptyset$, such that the weighted sum of the edges connecting A to $\bar{A}$ (called the "cut") is maximized. This problem is NP-hard, and its connection to spin glass Hamiltonians has long been recognized [35, 36]. To cast the problem into QUBO form, let us define the binary variable $x_i = 1$ if vertex i belongs to set A, and $x_i = 0$ otherwise. The edge (i, j) is part of the cut if and only if $x_i \neq x_j$, i.e., $(x_i - x_j)^2 = 1$. Therefore, the

objective is to maximize the cut value $\sum_{(i,j)\in E} w_{i,j}(x_i - x_j)^2$, or equivalently, to minimize the cost function

$$\min f(\{x_i\}) = \sum_{(i,j)\in E} w_{i,j}(2x_i x_j - x_i^2 - x_j^2) \tag{32}$$

It is easy to cast f into the standard QUBO form $f = \sum_{i,j} x_i Q_{i,j} x_j$ and map the MaxCut problem to an Ising spin glass Hamiltonian similar to (15), but with $c_1 = c_2 = 0$. In the present case, the cost function depends on the weight matrix w, and there is no penalty from constraints. Hence this setup here is almost the opposite of the Sudoku case, where the cost function is entirely derived from the constraints. Despite these differences, both problem types ultimately lead to spin Hamiltonians of the general form (15), which we solve by adding a driver term H_x and designing an appropriate driving schedule.

We evaluate the performance of our algorithm using MaxCut instances drawn from the Biq Mac Library. All instances used in this section are available for download at [37] or [38]. We will refer to each instance by its file name, which contains N_V, the number of vertices. Note that the complexity of the problem depends also on N_E, the number of edges, as well as the values of $w_{i,j}$. In what follows, we discuss three batches, **I** to **III**, of testing cases with increasing difficulty. Each batch has a distinctive distribution of edge weights.

Instance	E_0	M	h_x	η	D
pm1s_80.0	-79	5	1	0	30
pm1s_80.1	-69	5	1	0	30
pm1s_80.2	-67	5	1	0.3	30
pm1s_80.3	-66	5	1	0	30
pm1s_80.4	-69	5	1	0.3	30
pm1s_80.5	-66	20	1	0.3	30
pm1s_80.6	-71	5	1	0	30
pm1s_80.7	-69	5	1	0	30
pm1s_80.8	-68	5	1	0	30
pm1s_80.9	-67	5	1	0	30

Instance	E_0	M	h_x	η	D
pm1s_100.0	-127	5	1	0.3	30
pm1s_100.1	-126	5	1	0.3	30
pm1s_100.2	-125	10	1	0.3	30
pm1s_100.3	-111	10	1	0.3	30
pm1s_100.4	-128	5	1	0.3	30
pm1s_100.5	-125	10	1	0	60
pm1s_100.6	-122	10	1	0.3	30
pm1s_100.7	-112	20	1	0.3	60
pm1s_100.8	-120	10	1	0.3	30
pm1s_100.9	-127	5	1	0.3	30

Table 2: Twenty weighted MaxCut instances and the parameters used to solve them. Each instance is identified by its file name in the Biq Mac Library. The edge weights are either 1 or -1. For the first ten instances, $N_V = 80$ and $N_E = 316$, and for the next ten instances, $N_V = 100$ and $N_E = 495$. The algorithm successfully finds E_0, the ground state energy of the spin model (the MaxCut value is simply $-E_0$), except for the case pm1s_100.5.

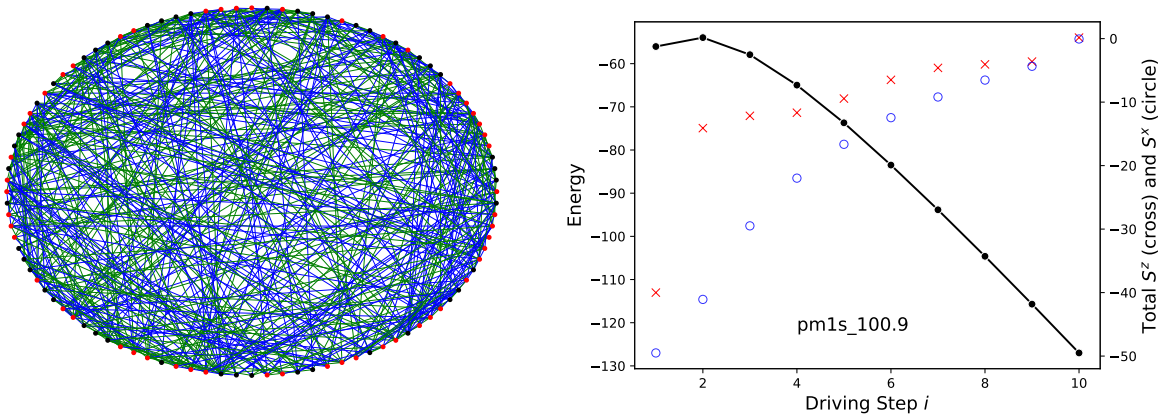


Figure 6: Solving the weighted MaxCut instance pm1s_100.9 from the Biq Mac Library. Left: graph representation of the problem. The edges carry weights of 1 (in green) or -1 (in blue). The solution is indicated by the node color: nodes in black belong to set A, while the rest of the nodes (in red) belong to set $\bar{A}$. Right: the energy (dot), expectation values of S^z (cross, measured from its ground state value and magnified by 10 times) and S^x (circle) during the driving process.

Batch I, listed in Table 2. These are weighted MaxCut problems where the edge weight $w_{i,j}$ is set to either 1 or -1. An example is illustrated in the left panel of Figure 6, featuring 100 nodes and 495 edges. The edges are color-coded: green for $w_{i,j} = 1$ and blue for $w_{i,j} = -1$. It is helpful to compare Figure 6 with the Sudoku example in Figure 2, where the edge weight is constant. In the MaxCut case, the edges are not only denser but also carry signs. In the spin language, this means there are more couplings among the Ising spins, and the coupling can be either ferromagnetic or antiferromagnetic. Also, the magnetic field along the z-axis for each spin m vanishes in this case, i.e., $h_m^z = 0$. Despite these differences, our algorithm successfully solves all instances listed in Table 2, yielding the correct MaxCut values, with one exception: the instance named pm1s_100.5. In this case, the lowest energy obtained, $E_0 = -125$, is above the value of -128 quoted in [37, 38]. To alert the reader, the E_0 value is underscored to indicate that this case remains to be solved. The parameters used in each instance (number of driving steps M, transverse field h_x and its random fluctuations η, and bond dimension D) are listed in Table 2. Five DMRG sweeps are used for each driving step, with one exception: for pm1s_100.6, 10 sweeps are used to improve accuracy.

Comparing the rows of Table 2 reveals some useful strategies for solving new QUBO problems. Within each group of MaxCut instances of similar size, some instances (e.g., pm1s_80.5 and pm1s_100.7) are more challenging to optimize. It is also generally harder to reach the ground state for instances with larger N_V and N_E. To tackle these harder cases, increasing the driving steps M and shaking the system with random fluctuations η often prove effective. Sometimes, increasing the bond dimension (e.g., $D = 60$ for pm1s_100.7) is necessary. Typically, these tuning decisions are made by monitoring the convergence of energy during the driving process (example shown in the right panel of Figure 6 for the instance pm1s_100.9).

Instance	E_0	M	N_{sw}	D	Instance	E_0	M	N_{sw}	D
g05_60.0	-536	10	5	30	g05_100.0	-1430	20	5	40
g05_60.1	-532	10	5	40	g05_100.1	-1425	20	5	40
g05_60.2	-529	10	5	30	g05_100.2	-1432	20	5	40
g05_60.3	-538	10	5	30	g05_100.3	-1423	10	5	30
g05_60.4	-527	10	5	30	g05_100.4	-1440	20	5	40
g05_60.5	-533	10	5	30	g05_100.5	-1435	10	5	30
g05_60.6	-531	10	5	30	g05_100.6	-1434	10	10	40
g05_60.7	-535	10	5	60	g05_100.7	-1431	10	10	40
g05_60.8	-530	10	5	30	g05_100.8	-1432	10	10	60
g05_60.9	-533	20	5	40	g05_100.9	-1430	10	10	40

Table 3: Twenty unweighted MaxCut instances from the Biq Mac Library with dense edges. The first ten cases have $N_V = 60$ and $N_E = 885$, while the remaining cases have $N_V = 100$ and $N_E = 2,475$. All edge weights are set to 1. The algorithm successfully solves all instances except g05_100.3 and g05_100.5, where the E_0 values should be $-1,424$ and $-1,436$, respectively [37, 38]. Here, N_{sw} refers to the number of DMRG sweeps per driving step, with $h_x = 1$ and $\eta = 0.3$.

Batch II, listed in Table 3. The second batch contains unweighted MaxCut problems where all the edges share the same weight of 1 but connections are much more dense when compared to the first batch. For instance, the problem shown in Figure 7 has the same number of nodes as in Figure 6, but the number of edges has increased fivefold to 2475. The spin-spin coupling is so dense that it may seem hopeless for tensor train and DMRG, as they are typically known to work well with spin Hamiltonians involving local interactions. Amazingly, the algorithm continues to yield correct MaxCut values with only a moderate increase in the bond dimension D, despite the steep rise in N_E. The heat map of S_z^m in Figure 7 provides further insight into the solution process for the instance g05_100.4. Regarding the choice of algorithm parameters, we observe similar trends noted earlier from the first batch in Table 2: for larger N_V and N_E, we need to increase the number of driving steps or the number of DMRG sweeps to steer the system toward the ground state. Occasionally, the bond dimension has to be increased (e.g., $D = 60$ for g05_60.7 and g05_100.8). Note that there are two cases in Table 3 (with the E_0 values underscored) where the algorithm came very close to the ground state but did not quite reach it.

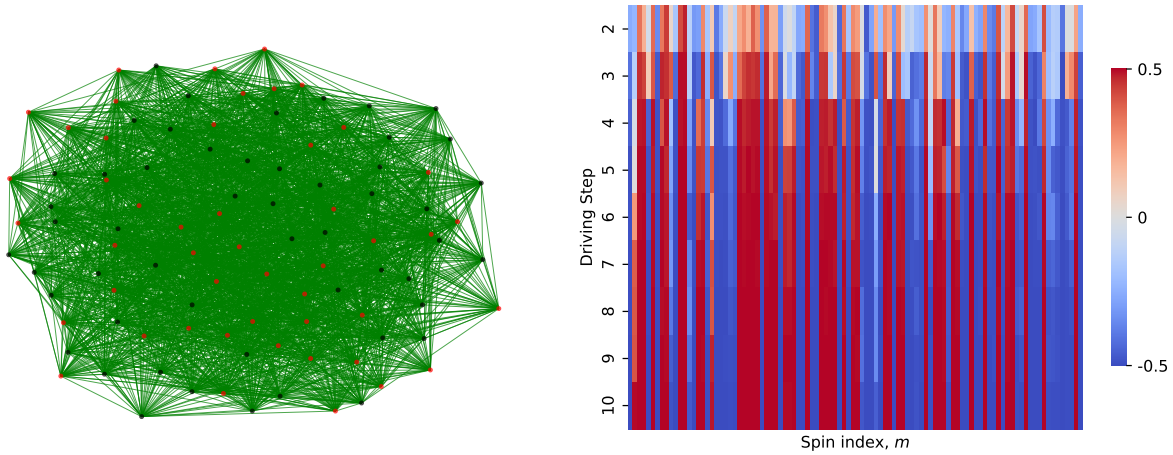


Figure 7: Solving the unweighted MaxCut instance g05_100.4 from the Biq Mac library. Left: graph representation of the problem. All edges (green lines) carry equal weight of 1. The solution found by our algorithm is color-coded: black nodes belong to set A, while red nodes belong to set $\bar{A}$. Right: the expectation value of S_z^m during the driving process as the spins gradually settle into their ground state orientation (red represents spin up, blue represents spin down). Only driving steps 2 to 10 are shown to enhance the color contrast.

Instance	E_0	N_V	N_E	h_x	η	N_{sw}
bqp250-2	-44810	251	3285	0.05	0	10
bqp250-4	-41274	251	3397	0.05	0	10
bqp250-6	-41014	251	3433	0.30	0	5
bqp250-8	-35726	251	3265	0.05	0	10
bqp250-10	-40442	251	3294	1.00	0.3	5

Table 4: The algorithm parameters used to successfully solve five MaxCut instances from the Biq Mac Library with 251 vertices and over 3200 edges, each with integer weights; $D = 30$, $M = 10$.

Batch III, listed in Table 4. The third batch of instances includes even larger MaxCut problems with $N_V = 251$ and N_E exceeding 3200. The coupling between the nodes is so dense that these instances are hard to graph or visualize. To make the matter even worse, in contrast to the examples discussed earlier, the edge weights in these instances vary over a wide range. As an example, the edge weight distribution of the instance bqp250-10 is shown in Figure 8. While there are a few outliers, the majority of $w_{i,j}$ fall within the range from -100 to 100, regardless of $|i-j|$, the distance between nodes. Other instances listed in Table 4 follow a similar distribution for $w_{i,j}$. As a result, the numerical scale of the QUBO matrix and the energy scale of the spin-spin interaction differ significantly from previous examples. To make the algorithm less sensitive to the overall numerical scale of $w_{i,j}$, we rescale the problem by dividing the matrix $w_{i,j}$ by the maximum absolute value of its elements. After this, $|w_{i,j}|$ becomes typically much smaller than 1. Consequently, the value of h_x is smaller compared to previous examples, as shown in the fifth column of Table 4. The choice of h_x is not unique. The right panel of Figure 8 compares the energy for three different values of h_x during the driving process. All three configurations lead to the correct ground state, demonstrating the robustness of the algorithm, even with the significant increase in N_E and N_V.

Performance evaluation. We find the proposed quantum-inspired algorithm has passed the bar of being able to solve a wide variety of MaxCut problems of different sizes and types. Our interest in MaxCut is partly motivated by the fact that all QUBO problems can be formulated as MaxCut problems. Thus, the key takeaway from this section is that our quantum-inspired algorithm, the Tensor Train Express, is versatile

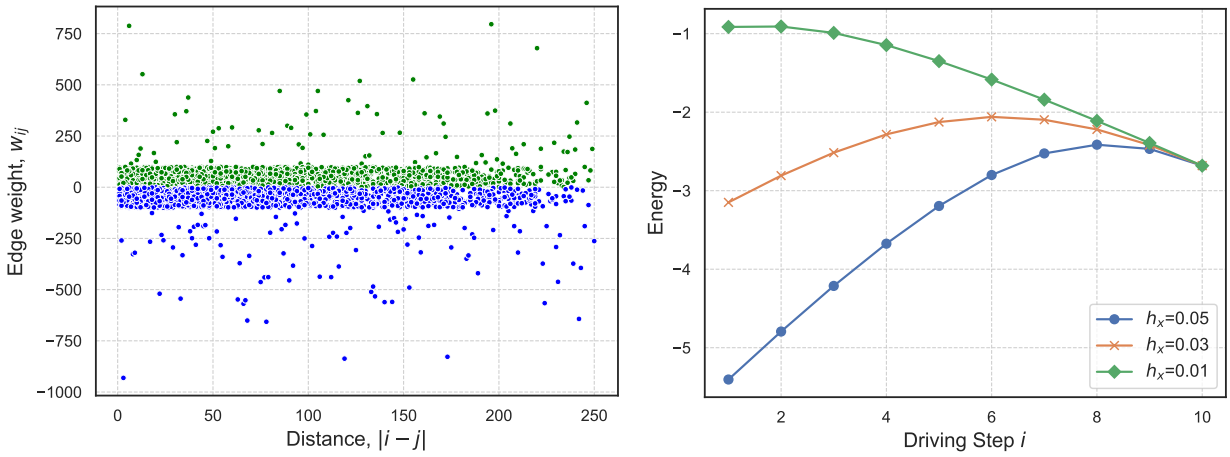


Figure 8: Solving MaxCut instance bqp250-8 with 251 nodes and 3265 edges. Left: the distribution of the edge weights $w_{i,j}$, sorted by $|i-j|$, the distance between the nodes. Most weights lie within -100 to 100. Right: the energy during the driving process for three different values of h_x. All three choices converge to the ground state, yielding the correct MaxCut value.

and continues to perform well in solving QUBO problems of larger sizes with denser connections.

6 A bright future for tensor trains

To conclude this research project, we place our work in a slightly broader context, reiterate our unique contributions, and then offer an outlook for future work.

Eagle versus tensor. The broader context of our work can be illuminated by a story titled "A moving target for quantum advantage" featured in the *Physics* magazine [39] and not directly related to optimization. Back in 2023, the experimental results on two-dimensional quantum spin systems carried out on IBM's 127-qubit Eagle quantum processor were believed to have achieved quantum advantage – i.e., performing "accurate computations at a scale beyond brute-force classical simulation" [40]. However, soon after, improved tensor network calculations on classical computers produced more accurate results than the Eagle [41]. This interesting episode tells us that the power of quantum-inspired approach based on tensor network states should not be underestimated. In the near term, until a great improvement in quantum hardware, quantum-inspired algorithms can bridge the gap by solving middle- to large-scale problems. An emerging trend is to cultivate a "symbiotic relationship" [39] between "the three musketeers" of classical, quantum, and quantum-inspired algorithms where they coexist, compete and complement each other. For example, tensor networks have been proposed to replace parts of the QAOA [42] or quantum annealing [43] algorithms. Beyond optimization, several tensor-train based solvers of nonlinear partial differential equations were demonstrated in the last year [44, 45, 46, 47]. Tensor trains and additionally quantum-inspired algorithms in general show great future promise.

Novelty. To our best knowledge, QUBO problems with 250 variables and 3200 couplings have not been solved by qubits-based quantum algorithms (we could not afford to test them on quantum annealers). Previous works on quantum-inspired optimization algorithms focused on imaginary-time evolution to tackle the problem Hamiltonian H_z directly [21, 22]. The amount of computing resources required to follow this route is beyond our means. We were forced to invent a poor man's workaround which requires only laptop computers and gets the job done in a few minutes to an hour. Our algorithm, the Tensor Train Express, achieves these goals, and also differs significantly from previous work in two ways. First, we avoid a direct attack on H_z. Instead we flank the enemy by going through a multi-stop drive sequence to steer the tensor

train states using a series of intermediate Hamiltonians. Second, at each driving stage, we call upon DMRG to sweep and update the tensors using the intermediate Hamiltonians, which turns out to be surprisingly efficient given the presence of many long-range interactions. Individually, these ideas are nothing new in quantum physics. But they are combined in an unconventional way and applied to solve interesting open problems in this research project. In Sections 4 and 5, we have presented conclusive evidence that the algorithm can successfully solve a variety of Sudoku puzzles and MaxCut problems.

Outlook. There are three directions to generalize this work: to scale up, to go practical, and to dig deeper. (1) We have intentionally kept a cap on the problem size so that the test runs remain computationally inexpensive. In future work, we plan to test the algorithm on much larger problems, for example MaxCut instances in the Gset with over 800 vertices [48]. For these large problems, we envision different trials with varied initial states or driving parameters running in parallel on high-performance computing clusters for days or weeks. (2) There is a long list of problems that can be cast into the QUBO form [7, 49], including practical applications such as vehicle routing. Optimization is not just about cutting cost and maximizing profit, it is also important to many branches of engineering and physical sciences, e.g. in robotics and protein folding. Thus, quantum-inspired algorithms may find applications in rather unexpected places. (3) Our algorithm did not succeed for every Sudoku puzzle fed in. We have witnessed that some problems are stubborn and refuse to lower the energy further. Very similar phenomena also occur in quantum annealing, so we are not surprised. Why are some problems harder, do we need to redesign the algorithm or resort to different tensor networks, and what sets the fundamental limit of tensor-based quantum inspired algorithms? Addressing these questions would be a good direction for future work.

7 Code and Demonstration

The algorithm and all experiments were implemented using Julia 1.10.5. Calculations involving spin Hamiltonians, tensor trains, and DMRG call the open-source library ITensors version 0.6.23 built by Matthew Fishman, Steven R. White, and E. Miles Stoudenmire [50]. A segment of the code is shown in Figure 9. Data are collected on a laptop equipped with an Apple M3 chip. Figure 1 to Figure 8 were generated by python scripts that import the numpy, matplotlib, seaborn, and networkx library. This research report was prepared using LaTeX.

Project code is available at the Github repository

https : //github.com/mzhao1599/qudoku

which contains the code, input data, and two demonstration videos of the Julia program running:

- Sudoku-demo.mp4.
- MaxCut-demo.mp4.

```
sites = siteinds("S=1/2",M)

# compute the onsite magnetic fields for reduced qubo, to be used for h1
ha = zeros(M)
for i in eachindex(ha)
    slice1 = Q[i,i+1:M]
    slice2 = Q[1:i-1,i]
    ha[i] = Q[i,i] + 0.5*( sum(slice1) + sum(slice2) )
end

# set up h1, the problem Hamiltonian, as OpSum
h1 = OpSum()
for i=1:M
    h1 += (ha[i], "Sz", i)
    for j in i+1:M
        h1 += (Q[i,j], "Sz", i, "Sz", j)
    end
end

## random initial tensor train state
psi = random_mps(sites; linkdims=7)
println("Starting from a random MPS:")

## another choice, initial state is a product state, all spins point along x, |+> or |->
#psi = MPS(sites, ["-" for n in 1:M] )
#println("Starting from a |--->:")
```

```
numsteps = 20
bond_dimension = 60
numsweeps = 10
hx_const = 1.25
hx_random_scale = 0.05

del = 1.0/numsteps
en_list = zeros(Float64, numsteps)
idx = 1
# loop over driving step
for sp in range(start=1-del,step=-del,length=numsteps)
    println("step sp  = ", sp)

    ## set up the transverse Hamiltonian, constant term + added randomness
    rd = hx_random_scale*randn(M)
    h2 = OpSum()
    for i=1:M
        h2 += (hx_const+rd[i], "Sx", i)
    end

    # the total Hamiltonian, a mixture of h1 and h2
    H = MPO( (1.0-sp)*h1 + sp*h2, sites)

    # call dmrg
    energy,psi = dmrg(H,psi;nsweeps=numsweeps,maxdim=bond_dimension,noise=0.003)
    # save the energy
    en_list[idx] = (energy + en_shift)
    idx += 1
end
```

Figure 9: A code snippet showing how to set up the Hamiltonians, specify the initial tensor train state, organize the driving schedule, and call the DMRG sweeps.

8 References

[1] Bernhard H Korte and Jens Vygen. *Combinatorial optimization*. Springer, 2011.

[2] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to algorithms*. MIT press, 2022.

[3] Fred Glover, Gary Kochenberger, Rick Hennig, and Yu Du. Quantum bridge analytics I: a tutorial on formulating and using QUBO models. *Annals of Operations Research*, 314(1):141–183, 2022.

[4] Abraham P Punnen, editor. *The quadratic unconstrained binary optimization problem*. Springer Cham, 2022.

[5] F Barahona. On the computational complexity of Ising spin glass models. *Journal of Physics A: Mathematical and General*, 15(10):3241, 1982.

[6] Amira Abbas, Andris Ambainis, Brandon Augustino, Andreas Bärtschi, Harry Buhrman, Carleton Coffrin, Giorgio Cortiana, Vedran Dunjko, Daniel J. Egger, Bruce G. Elmegreen, Nicola Franco, Filippo Fratini, Bryce Fuller, Julien Gacon, Constantin Gonciulea, Sander Gribling, Swati Gupta, Stuart Hadfield, Raoul Heese, Gerhard Kircher, Thomas Kleinert, Thorsten Koch, Georgios Korpas, Steve Lenk, Jakub Marecek, Vanio Markov, Guglielmo Mazzola, Stefano Mensa, Naeimeh Mohseni, Giacomo Nannicini, Corey O'Meara, Elena Peña Tapia, Sebastian Pokutta, Manuel Proissl, Patrick Rebentrost, Emre Sahin, Benjamin C. B. Symons, Sabine Tornow, Víctor Valls, Stefan Woerner, Mira L. Wolf-Bauwens, Jon Yard, Sheir Yarkoni, Dirk Zechiel, Sergiy Zhuk, and Christa Zoufal. Challenges and opportunities in quantum optimization. *Nature Reviews Physics*, 6(12):718–735, 2024.

[7] Andrew Lucas. Ising formulations of many NP problems. *Frontiers in Physics*, 2, 2014.

[8] Naeimeh Mohseni, Peter L. McMahon, and Tim Byrnes. Ising machines as hardware solvers of combinatorial optimization problems. *Nature Reviews Physics*, 4(6):363–379, 2022.

[9] Aleta Berk Finnila, Maria A Gomez, C Sebenik, Catherine Stenson, and Jimmie D Doll. Quantum annealing: A new method for minimizing multidimensional functions. *Chemical Physics Letters*, 219(5-6):343–348, 1994.

[10] Tadashi Kadowaki and Hidetoshi Nishimori. Quantum annealing in the transverse Ising model. *Phys. Rev. E*, 58:5355–5363, Nov 1998.

[11] Arnab Das and Bikas K. Chakrabarti. Colloquium: Quantum annealing and analog quantum computation. *Rev. Mod. Phys.*, 80:1061–1081, Sep 2008.

[12] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*, 2014.

[13] Stuart Hadfield, Zhihui Wang, Bryan O'gorman, Eleanor G Rieffel, Davide Venturelli, and Rupak Biswas. From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. *Algorithms*, 12(2):34, 2019.

[14] The D-Wave Advantage System: An Overview. `https://www.dwavesys.com/media/3xvdipcn/14-1058a-a_advantage_processor_overview.pdf`.

[15] Guido Pagano, Aniruddha Bapat, Patrick Becker, Katherine S Collins, Arinjoy De, Paul W Hess, Harvey B Kaplan, Antonis Kyprianidis, Wen Lin Tan, Christopher Baldwin, et al. Quantum approximate optimization of the long-range Ising model with a trapped-ion quantum simulator. *Proceedings of the National Academy of Sciences*, 117(41):25396–25401, 2020.

[16] Kostas Blekos, Dean Brand, Andrea Ceschini, Chiao-Hui Chou, Rui-Hao Li, Komal Pandya, and Alessandro Summer. A review on quantum approximate optimization algorithm and its variants. *Physics Reports*, 1068:1–66, 2024.

[17] Larry Huynh, Jin Hong, Ajmal Mian, Hajime Suzuki, Yanqiu Wu, and Seyit Camtepe. Quantum-inspired machine learning: a survey. *arXiv preprint arXiv:2308.11269*, 2023.

[18] Frank Verstraete, Valentin Murg, and J Ignacio Cirac. Matrix product states, projected entangled pair states, and variational renormalization group methods for quantum spin systems. *Advances in Physics*, 57(2):143–224, 2008.

[19] Román Orús. A practical introduction to tensor networks: Matrix product states and projected entangled pair states. *Annals of Physics*, 349:117–158, 2014.

[20] Jacob Biamonte and Ville Bergholm. Tensor networks in a nutshell. *arXiv preprint arXiv:1708.00006*, 2017.

[21] Samuel Mugel, Carlos Kuchkovsky, Escolástico Sánchez, Samuel Fernández-Lorenzo, Jorge Luis-Hita, Enrique Lizaso, and Román Orús. Dynamic portfolio optimization with real datasets using quantum processors and quantum-inspired tensor networks. *Phys. Rev. Res.*, 4:013006, Jan 2022.

[22] Siddhartha Patra, Sukhbinder Singh, and Román Orús. Projected entangled pair states with flexible geometry. *Phys. Rev. Res.*, 7:L012002, Jan 2025.

[23] G. E. Santoro, R. Martonak, E. Tosatti, and R. Car. Theory of quantum annealing of an Ising spin glass. *Science*, 295(5564):2427–2430, 2002.

[24] Takayuki Yato and Takahiro Seta. Complexity and completeness of finding another solution and its application to puzzles. *IEICE transactions on fundamentals of electronics, communications and computer sciences*, 86(5):1052–1060, 2003.

[25] Nobushida Ueda and Tadaaki Nagao. NP-completeness results for NONOGRAM via parsimonious reductions. *Technical Report TR96-0008, Department of Computer Science, Tokyo Institute of Technology*, 1996.

[26] Sascha Mücke. A simple QUBO formulation of sudoku. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO '24 Companion, pages 1958–1962, New York, NY, USA, 2024. Association for Computing Machinery.

[27] David Sherrington and Scott Kirkpatrick. Solvable model of a spin-glass. *Phys. Rev. Lett.*, 35:1792–1796, Dec 1975.

[28] J Vannimenus and G Toulouse. Theory of the frustration effect. II. Ising spins on a square lattice. *Journal of Physics C: Solid State Physics*, 10(18):L537, 1977.

[29] Figure modified from source `https://pngtree.com/freepng/coloring-train-vector_8856582.html`.

[30] Gregory M. Crosswhite and Dave Bacon. Finite automata for caching in matrix product algorithms. *Phys. Rev. A*, 78:012356, Jul 2008.

[31] Steven R. White. Density matrix formulation for quantum renormalization groups. *Phys. Rev. Lett.*, 69:2863–2866, Nov 1992.

[32] U. Schollwöck. The density-matrix renormalization group. *Rev. Mod. Phys.*, 77:259–315, Apr 2005.

[33] `https://www.nytimes.com/puzzles/sudoku`.

[34] https://nytsudoku.net.

[35] Richard M. Karp. *Reducibility among Combinatorial Problems*, pages 85–103. Springer US, Boston, MA, 1972.

[36] Francisco Barahona, Martin Grötschel, Michael Jünger, and Gerhard Reinelt. An application of combinatorial optimization to statistical physics and circuit layout design. *Operations Research*, 36(3):493–513, 1988.

[37] https://biqmac.aau.at/.

[38] http://bqp.cs.uni-bonn.de/library/html/instances.html.

[39] Michael Schirber. A moving target for quantum advantage. *Physics*, 17:13, 2024.

[40] Youngseok Kim, Andrew Eddins, Sajant Anand, Ken Xuan Wei, Ewout van den Berg, Sami Rosenblatt, Hasan Nayfeh, Yantao Wu, Michael Zaletel, Kristan Temme, and Abhinav Kandala. Evidence for the utility of quantum computing before fault tolerance. *Nature*, 618(7965):500–505, 2023.

[41] Joseph Tindall, Matthew Fishman, E. Miles Stoudenmire, and Dries Sels. Efficient tensor network simulation of IBM's Eagle kicked Ising experiment. *PRX Quantum*, 5:010308, Jan 2024.

[42] Michael Streif and Martin Leib. Training the quantum approximate optimization algorithm without access to a quantum processing unit. *Quantum Science and Technology*, 5(3):034008, 2020.

[43] Ilia A. Luchnikov, Egor S. Tiunov, Tobias Haug, and Leandro Aolita. Large-scale quantum annealing simulation with tensor networks and belief propagation. *arXiv preprint arXiv:2409.12240*, 2024.

[44] Ryan J. J. Connor, Callum W. Duncan, and Andrew J. Daley. Tensor network methods for the Gross-Pitaevskii equation on fine grids. *arXiv preprint arXiv:2507.01149*, 2025.

[45] Aleix Bou-Comas, Marcin Płodzień, Luca Tagliacozzo, and Juan José García-Ripoll. Quantics Tensor Train for solving Gross-Pitaevskii equation. *arXiv preprint arXiv:2507.03134*, 2025.

[46] Marcel Niedermeier, Adrien Moulinas, Thibaud Louvet, Jose L. Lado, and Xavier Waintal. Solving the Gross-Pitaevskii equation on multiple different scales using the quantics tensor train representation. *arXiv preprint arXiv:2507.04262*, 2025.

[47] Qian-Can Chen, I-Kang Liu, Jheng-Wei Li, and Chia-Min Chung. Solving the Gross-Pitaevskii Equation with Quantic Tensor Trains: Ground States and Nonlinear Dynamics. *arXiv preprint arXiv:2507.04279*, 2025.

[48] https://web.stanford.edu/~yyye/yyye/Gset/.

[49] https://blog.xa0.de/post/List-of-QUBO-formulations/.

[50] Matthew Fishman, Steven R. White, and E. Miles Stoudenmire. The ITensor Software Library for Tensor Network Calculations. *SciPost Phys. Codebases*, 4, 2022.