Branch-and-bound tensor networks for exact characterization of classical ground states

Yi-Jia Wang [1,2,*] Xuanzhao Gao [3,*] Pan Zhang [1,2,4,5,†] Feng Pan [6,‡] and Jin-Guo Liu [7,§]

[1]*CAS Key Laboratory for Theoretical Physics, Institute of Theoretical Physics, Chinese Academy of Sciences, Beijing 100190, China*

[2]*School of Physical Sciences, University of Chinese Academy of Sciences, Beijing 100049, China*

[3]*Center for Computational Mathematics, Flatiron Institute, Simons Foundation, New York, New York 10010, USA*

[4]*School of Fundamental Physics and Mathematical Sciences, Hangzhou Institute for Advanced Study, UCAS, Hangzhou 310024, China*

[5]*Beijing Academy of Quantum Information Sciences, Beijing 100193, China*

[6]*Science, Mathematics and Technology Cluster, Singapore University of Technology and Design, 8 Somapah Road, Singapore 487372, Singapore*

[7]*Thrust of Advanced Materials, Function Hub, Hong Kong University of Science and Technology (Guangzhou), Guangzhou 511453, China*

(Received 4 February 2026; revised 11 June 2026; accepted 22 September 2026; published 18 November 2026)

Characterizing the ground-state properties of disordered systems, such as spin glasses and combinatorial optimization problems, is fundamental but computationally challenging. Finding exact ground states and counting their degeneracies are NP-hard and #P-hard, respectively, placing severe limits on exact algorithms. While tensor networks have emerged as a powerful paradigm for these tasks by leveraging massive hardware parallelization, their scalability is fundamentally bottlenecked by exponential memory growth. To overcome this, we introduce the branch-and-bound tensor network (BBTN) method. By integrating adaptive branch-and-bound search with efficient tropical tensor network contraction, BBTN sets new benchmarks for exact computation. It pushes the tractable frontier to previously unreachable scales, enabling exact ground-state counting for $\pm J$ spin glasses up to 65×65 and maximum independent set solving on King's subgraphs up to 100×100, reducing run times required by state-of-the-art solvers from millions of years to hours. Furthermore, it consistently outperforms leading open-source integer-programming solvers and runs up to an order of magnitude faster than commercial solvers in rugged landscapes, establishing a versatile and scalable framework for solving hard problems in statistical physics and combinatorial optimization.

DOI: 10.1103/g76v-wxz7

I. INTRODUCTION

Disordered systems, encompassing both spin glasses [1,2] and combinatorial optimization problems [3], offer critical theoretical insights for understanding complex energy landscapes [4–6], frustration [7,8], and glassy dynamics [9]. They are also fundamental to a wide array of applications across science and engineering, ranging from probabilistic inference [10] to quantum information science [11,12]. In statistical physics, studying the ground-state properties of disordered systems elucidates the nature of low-lying excitations and the theory of replica symmetry breaking [13–16]. Similarly, in combinatorial tasks like the maximum independent set problem [3], understanding the solution-space properties is crucial for determining computational complexity [17,18], identifying theoretical phase transitions [19,20], and defining the fundamental performance limits of practical solvers [3,21].

However, finding exact ground states and counting their degeneracies are NP-hard and #P-hard, respectively [21,22]. Traditional exact solvers, such as branch-and-cut [23–25] and branch-and-bound algorithms [26–31], typically search for single optimal solutions using task parallelism. When used for counting, such tree-search approaches must explicitly enumerate or certify all optimal solutions, so the complexity can explode exponentially with the ground-state degeneracy. Consequently, these traditional approaches fail to leverage the massive data parallelism of modern graphics processing units (GPUs), rendering them intractable for large-scale problems. Meanwhile, approximate methods like Monte Carlo [32] or simulated annealing [33] provide no exactness guarantees and often fail in rugged landscapes.

To overcome these limitations, tensor network methods [34–37] have emerged as a powerful exact framework for tackling core challenges in statistical physics and combinatorial optimization. By exploiting global tensor network structure and massive GPU parallelism [38,39], tensor networks offer significant speedups over traditional serial algorithms. In particular, tropical tensor networks represent the state of the art for exact ground-state counting, reaching 32×32 for two-dimensional lattice spin glasses and $6 \times 6 \times 6$ for three-dimensional lattices [36]. Nevertheless, their scalability faces

*These authors contributed equally to this work.
†Contact author: panzhang@itp.ac.cn
‡Contact author: feng_pan@sutd.edu.sg
§Contact author: jinguoliu@hkust-gz.edu.cn

Published by the American Physical Society under the terms of the Creative Commons Attribution 4.0 International license. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.

a fundamental memory bottleneck, as space complexity scales exponentially with the contraction treewidth [34]. Although the slicing technique [40] can alleviate this memory pressure, it incurs a prohibitive exponential time overhead.

In this work, we introduce the branch-and-bound tensor network (BBTN) method, which integrates branch-and-bound search [26–28,41–43] with tropical tensor network contraction to simultaneously reduce space and time complexity. By adaptively branching on variables and aggressively pruning configurations using local and global bounds [21,44–46], BBTN constructs unbalanced but dramatically smaller search trees. To demonstrate BBTN's efficacy across both physics-inspired and hardware-relevant optimization landscapes, we evaluate it on two paradigmatic benchmarks: the $\pm J$ Ising spin glass [1,2,22] and the maximum independent set problem on King's subgraphs [47–49], which naturally maps to the Rydberg blockade mechanism in neutral atom arrays [50–52]. Across these benchmarks, BBTN significantly extends the boundaries of computational tractability, reaching unprecedented scales. For exact ground-state counting of spin glasses, BBTN extends the accessible system size up to 65×65 on two-dimensional lattices and $8 \times 8 \times 8$ on three-dimensional lattices. For optimization on King's subgraphs, it significantly surpasses the state-of-the-art open-source integer-programming solver SCIP [25] and generally outperforms the commercial solver IBM CPLEX [24], extending the frontier of classical exact algorithms for these challenging optimization problems.

II. TROPICAL TENSOR NETWORKS

We consider the spin glass model defined on a graph $G = (V, E)$ with energy function

$$H(\boldsymbol{\sigma}) = \sum_{(i,j)\in E} -J_{ij}\sigma_i\sigma_j + \sum_{i\in V} -h_i\sigma_i\,, \tag{1}$$

where $\sigma_i \in \{-1, +1\}$ encodes the ith spin, and J_{ij} and h_i denote the coupling constants and external fields, respectively. The ground-state energy $H^* = \min_{\boldsymbol{\sigma}} H(\boldsymbol{\sigma})$ is the zero-temperature limit $H^* = -\lim_{\beta\to\infty} \frac{1}{\beta} \ln Z$ of the partition function $Z = \sum_{\boldsymbol{\sigma}} \prod_{(i,j)\in E} e^{\beta J_{ij}\sigma_i\sigma_j} \prod_{i\in V} e^{\beta h_i\sigma_i}$, where the exponential terms are known as local Boltzmann factors.

Since Z is composed of these local Boltzmann factors combined via sum and product operations, taking this limit effectively replaces standard arithmetic with operations directly on the exponents: Addition becomes a minimum, $-\lim_{\beta\to\infty} \frac{1}{\beta} \ln(e^{-\beta x} + e^{-\beta y}) = \min(x, y)$, and multiplication becomes a sum, $-\lim_{\beta\to\infty} \frac{1}{\beta} \ln(e^{-\beta x} e^{-\beta y}) = x + y$. Consequently, the ordinary $(+, \times)$ algebra acting on the Boltzmann factors is replaced by the *tropical* algebra over the associated energies $x = -\beta^{-1} \ln(\cdot) \in \mathbb{R} \cup \{+\infty\}$, where tropical addition $x \oplus y = \min(x, y)$ selects the lower-energy alternative and tropical multiplication $x \odot y = x + y$ accumulates energy along a configuration [36,37]. Each local Boltzmann factor thereby becomes an energy tensor indexed by its spins (encoded as $s_i = 0, 1 \leftrightarrow \sigma_i = -1, +1$): Each edge (i, j) carries $B^{ij}_{s_i s_j} = -J_{ij}\sigma_i\sigma_j$ and each vertex i carries $w^i_{s_i} = -h_i\sigma_i$. These local tensors, sharing indices associated with common spins, form a tensor network whose tropical contraction yields the ground-state energy:

$$H^* = \bigoplus_{s\in\{0,1\}^{|V|}} \Big[\Big(\bigodot_{(i,j)\in E} B^{ij}_{s_i s_j}\Big) \odot \Big(\bigodot_{i\in V} w^i_{s_i}\Big)\Big]. \tag{2}$$

For exact counting, each tensor entry is promoted from an energy x to a pair (x, n), where n counts how many partial configurations attain that energy; when two terms tie for the minimum their counts are added, so the contraction returns both the ground-state energy and its degeneracy [36,53]; the full hybrid tropical-ordinary algebra is given in the Supplemental Material [54]. For other combinatorial optimization problems, one chooses the tensors $\boldsymbol{B}^{ij}$ and $\boldsymbol{w}^i$ to encode the objective and assigns infinite energy ($(\infty, 0)$ for counting) to forbidden local configurations.

The tensor networks under the various algebras above are collectively denoted by $\mathcal{T}(G)$, with the contraction result $\texttt{contract}(\mathcal{T}(G))$. The computational cost is measured along two dimensions: time complexity (total floating-point operations) and space complexity (memory for the largest intermediate tensor), both of which scale exponentially with the treewidth of the tensor network's line graph [34]. In practice, with a fixed memory budget, the space complexity determines the maximum system size that can be handled.

III. TENSOR NETWORKS WITH SLICING

A standard mitigation technique for the memory bottleneck is *slicing*, which decomposes the original problem into subproblems by iterating over configurations of a selected variable subset:

$$\text{contract}(\mathcal{T}(G)) = \bigoplus_{\boldsymbol{x}\in\{0,1\}^{|V_f|}} (\text{contract}(\mathcal{T}(G)\,|_{s_{V_f}=\boldsymbol{x}})), \tag{3}$$

where $\mathcal{T}(G)\,|_{s_{V_f}=\boldsymbol{x}}$ denotes the sliced tensor network with variables in V_f fixed to configuration $\boldsymbol{x}$. The subset V_f is precomputed by algorithms such as dynamic slicing [55] to maximize space reduction while minimizing $|V_f|$. However, as shown in the left panel of Fig. 1, each additional fixed variable doubles the number of subnetworks with the same topology but different contents, requiring the contraction to be performed $2^{|V_f|}$ times and incurring exponential time overhead. This trade-off renders slicing impractical except for very small slicing sets.

IV. BRANCH-AND-BOUND TENSOR NETWORK

To overcome this bottleneck, we propose BBTN, which integrates branch-and-bound search [46] into tensor network contraction. BBTN retains the basic slicing idea of reducing memory by fixing variables. The difference is that BBTN does not enumerate all assignments of a fixed slicing set. Instead, as illustrated in the right panel of Fig. 1, BBTN exploits two key insights: pruning, which uses bounds to discard branches that cannot affect the final optimum, and nonuniform branching, which fixes different variable subsets in different branches rather than relying on one precomputed slicing set.

At each step, BBTN selects a branching region $R \subseteq V$ and analyzes the local subnetwork $\mathcal{T}_R$. Pruning is applied at two levels. Global pruning uses the best ground-state energy found

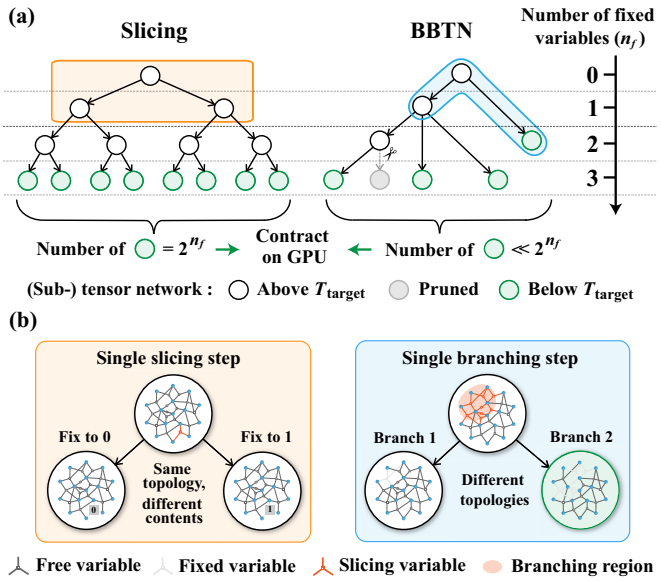


FIG. 1. Schematic comparison of slicing and BBTN methods. (a) Tensor network decomposition process. Slicing (left) exhaustively fixes a precomputed set of n_f variables, generating 2^{n_f} leaf subnetworks satisfying memory constraints (green nodes). BBTN (right) employs a branch-and-bound framework with pruning and online branching, producing an unbalanced tree with substantially fewer leaf subnetworks. Dashed arrows show subnetworks pruned before branching and thus never generated. (b) Single-step comparison. Slicing (left) fixes one variable per step, generating two subnetworks. BBTN (right) examines multiple variables, prunes infeasible or suboptimal configurations, and generates an adaptive number of branches with different subsets of fixed variables.

so far: Any branch whose lower bound is already worse than this value is discarded. Local pruning eliminates assignments inside R that cannot contribute to the global ground state, doing so before any expensive contraction is performed. In particular, for each spin configuration on the boundary of R, only locally optimal interior assignments are retained, and additional inference rules can further discard provably suboptimal assignments [54]. The result is a reduced feasible set $\mathcal{F}(R) \subseteq \{0,1\}^{|R|}$ of local configurations that still need to be evaluated.

A naive branch-and-bound step would create one branch for each configuration in $\mathcal{F}(R)$, equivalently fixing all variables in R. However, such exhaustive branching yields an intractable number of subproblems. BBTN instead groups surviving configurations using partial assignments. We write the resulting branch set as $\mathcal{D}(R) \subseteq \{(R_f, \boldsymbol{x}) \mid R_f \subseteq R,\ \boldsymbol{x} \in \{0,1\}^{|R_f|}\}$, where each branch fixes only the variables in R_f. For exact counting, these branches must be *complete*, so every configuration in $\mathcal{F}(R)$ is included, and *orthogonal*, so no configuration is counted twice. The final answer is obtained by contracting and aggregating over these valid branches.

The branching set is chosen online [46] to reduce the memory bottleneck as much as possible. In conventional branch-and-bound frameworks, the time complexity is assumed to be exponential in the problem size ρ, with the branching factor γ as the base, i.e., $O(\gamma^{\rho})$. Here, since our primary bottleneck is space complexity, we define ρ as the cumulative memory overflow:

$$\rho(\mathcal{T}) = \sum_{\mathbf{T}} \max(0,\ \log_2 |\mathbf{T}| - \log_2 T_{\text{target}}), \tag{4}$$

where $|\mathbf{T}|$ is the tensor size and T_{target} is the memory threshold. For candidate branches $\mathcal{T}_i$, BBTN chooses the rule that minimizes the branching factor γ defined by $\gamma^{\rho(\mathcal{T})} = \sum_i \gamma^{\rho(\mathcal{T}_i)}$. This criterion favors decompositions that reduce memory overflow while creating as few subnetworks as possible.

V. APPLICATIONS

We evaluate BBTN on two representative models: the $\pm J$ Ising spin glass on various graphs, including regular two-dimensional and three-dimensional lattices and irregular real-world networks; and the maximum independent set problem on King's subgraphs, which is a hardware-relevant benchmark for neutral-atom quantum processors. As baselines, we compare BBTN against tropical tensor networks with slicing, the general state-of-the-art tensor-network approach for exact counting [36], whose slicing sets are calculated by dynamic slicing [37,55]. For optimization on King's subgraphs, we select two integer-programming solvers, which represent the state of the art for maximum independent set problems on these graphs [24]: the open-source solver SCIP [25] and the closed-source commercial solver IBM CPLEX [24]. Furthermore, although enumeration-based branch-and-bound methods are not generally competitive for exact counting (with counting results reported only up to $\mathbf{8 \times 8}$ on two-dimensional lattices [56]), they can perform very well in special cases where the coupling pattern or graph structure yields strong bounds. We therefore also include IBM CPLEX `populate` [57].

We first compare the theoretical algorithmic complexity after removing implementation-dependent effects, naturally measured within tensor-network methods by the total time complexity of all tensor contractions. For exact counting on the $L \times L$ two-dimensional lattice, as shown in Fig. 2(a), plain tensor-network contraction exceeds the memory target for $L > 31$ and serves merely as a theoretical baseline with exponentially growing time complexity. Tensor networks with slicing can operate within a given space-complexity target, but the slicing overhead makes their time complexity grow superexponentially with L. In contrast, by detecting the distributional structure of the ground-state manifold, BBTN achieves a substantially smaller total contraction cost and a much slower scaling, computing models up to 65×65, a scale that would require at least 10^6 years under slicing. As shown in Fig. 2(b), this significant time-complexity advantage persists across diverse graph topologies and other counting problems, including maximum independent set.

This computational advantage directly translates to exact optimization tasks. As shown in Fig. 2(c), for maximum independent set problems on King's subgraphs, the time complexity and scaling of BBTN remain vastly superior to slicing. Consequently, BBTN extends the computable system size to 100×100, whereas slicing is limited to at most 60×60.

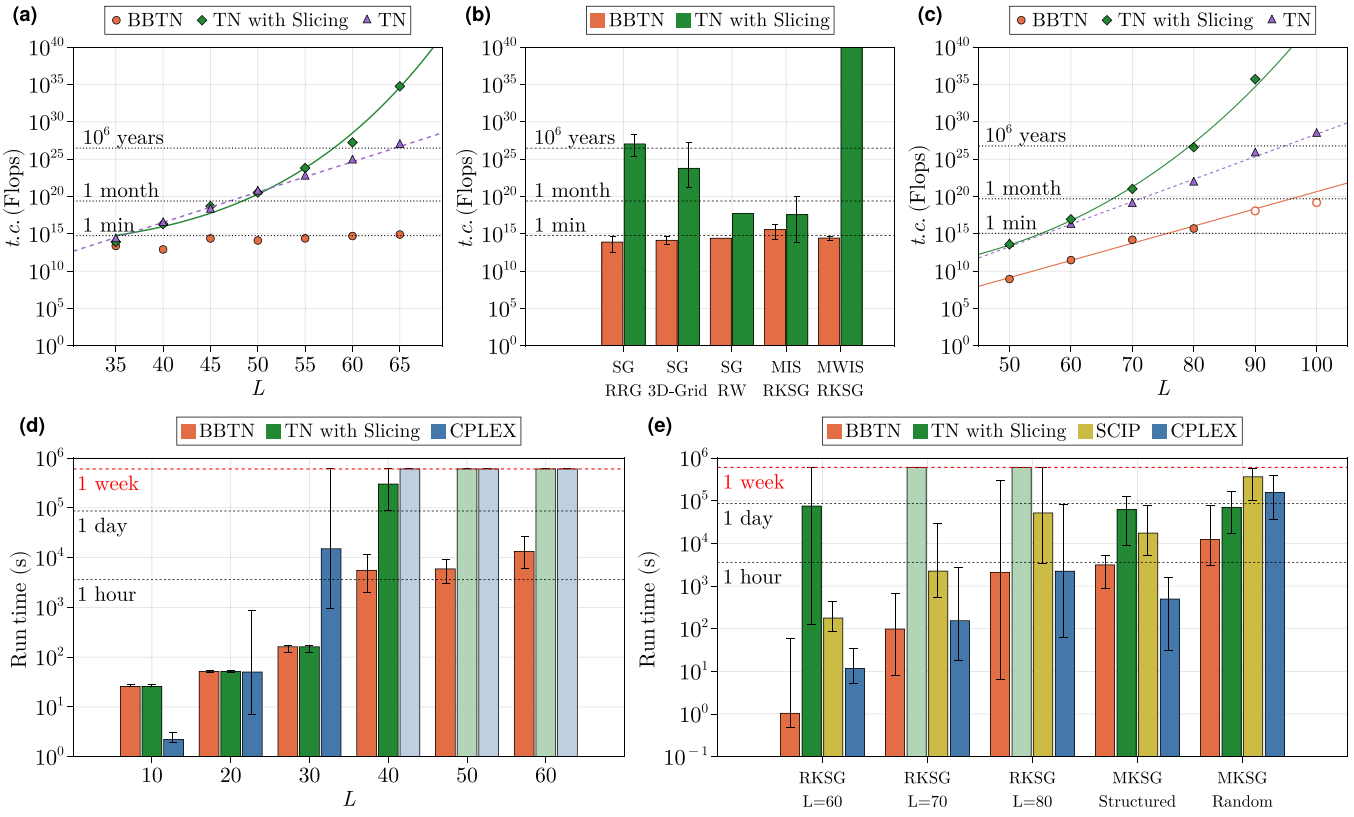


FIG. 2. Performance comparison of BBTN against tropical tensor networks, tropical tensor networks with slicing, and the CPLEX and SCIP solvers, across ground-state counting [panels (a), (b), and (d)] and ground-state optimization [panels (c) and (e)] tasks. (a) Theoretical time complexity, measured in floating-point operations, for counting spin glasses ($J = \pm 1$, $h = 0.5$) on $L \times L$ two-dimensional lattices. (b) Time complexity for counting across five problem classes: spin glasses ($J = \pm 1$, $h = 0.5$) on random regular graphs ($n = 600$, $d = 3$), on an $8 \times 8 \times 8$ three-dimensional lattice, and on the `1shp1561` real-world network (a thermal problem graph) [58], as well as maximum independent set on random King's subgraphs ($L = 60$, filling $= 0.8$), and maximum-weight independent set on random King's subgraphs ($L = 100$, filling $= 0.8$, with weights uniformly sampled from integers 1 to 10). (c) Time complexity for optimization on random King's subgraphs. For BBTN, hollow dots show averages over only the instances solvable in time (4/10 for $L = 90$ and 2/10 for $L = 100$), where the scaling curves are fits to the solid dots. (d) Wall-clock run-time for counting on $L \times L$ two-dimensional lattices. Translucent bars indicate instances exceeding the one-week time limit. (e) Wall-clock run-time for optimization on random and mapped King's subgraphs. Markers and bars are averages over ten instances per setting; error bars span the min-max. The theoretical operation counts in panels (a)–(c) are converted to time using the peak throughput of a single NVIDIA A100 graphics processing unit. This theoretical metric isolates the pure tensor contraction cost, assuming perfect hardware utilization and excluding the classical time spent on the branch-and-bound slicing process.

To establish a fair comparison with non-tensor-network solvers, we also directly evaluate wall-clock run times. For exact counting, as shown in Fig. 2(d), while CPLEX `populate` is highly efficient on the smallest systems, its computational cost grows rapidly with L, limiting its applicability to 30×30 instances. Meanwhile, for $L < 30$, both slicing and BBTN degenerate into plain tensor-network contractions, yielding identical run times. Their performance divergence becomes evident only at larger scales, where severe overhead restricts slicing to at most 40×40 instances. BBTN, in contrast, exhibits significantly shorter computing times and a much gentler scaling curve. Strikingly, BBTN evaluates 60×60 instances in a matter of hours—a task estimated to require million-year-scale run times using either slicing or CPLEX `populate` (see the Supplemental Material [54] (see also Refs. [59–64] therein) for derivations).

For exact optimization, Fig. 2(e) shows that although all methods scale exponentially with L, BBTN consistently demonstrates competitive performance and exceptional robustness. On random King's-subgraph instances—generated by randomly retaining sites of an $L \times L$ King's subgraph with a 0.8 filling rate—BBTN is substantially faster than SCIP and slicing, and generally outperforms CPLEX across the computable range. We further investigate mapped King's-subgraph instances, which encode the integer factorization problem into maximum-weight independent sets [54,65]. Here, the structured and random factorization ensembles create distinct solution landscapes. In the structured landscape, which is relatively straightforward for traditional branch-and-bound searches, BBTN is slower than CPLEX but remains approximately $6\times$ faster than SCIP. Crucially, however, in the extremely rugged random landscape, the tensor-network backend endows BBTN with exceptional robustness, rendering it nearly $10\times$ faster than CPLEX. Collectively, these performance gains across both counting and optimization tasks establish BBTN as a new state of the art for exact combinatorial computation.

VI. DISCUSSION

BBTN's success on spin glass and maximum independent set problems, two representative NP-hard models from statistical physics and combinatorial optimization, demonstrates its immediate applicability to a broader class of NP-hard problems, including dominating set, graph coloring, and k-SAT. Beyond optimization, the framework's ability to handle sparse tensor network contractions allows it to address fundamental tasks such as quantum circuit simulation and quantum error correction. By extending the underlying algebra to generic semirings, BBTN can also characterize complex solution-space properties, such as solution configurations and k-largest spectra. Furthermore, relaxing exactness constraints naturally extends the framework to high-fidelity approximate contractions.

The paradigm established by BBTN can be synergistically integrated with AI-driven and mathematical optimization frameworks to achieve even greater efficacy. Specifically, the current reliance on stepwise optimization of local heuristic measures, which is computationally intensive and lacks global foresight, could be replaced by deep reinforcement learning, Monte Carlo tree search, or automated discovery via large language models [66], which would substitute myopic decision-making with a strategic, global policy. Additionally, integrating advanced techniques from the mathematical programming community, such as branch-and-cut and branch-and-price, offers sophisticated solution-space transformations that complement tensor network contractions.

ACKNOWLEDGMENTS

This work was partially supported by the National Natural Science Foundation of China under Grants No. 12325501, No. 12404568, No. 12047503, and No. 12247104, the National Key R&D Program of China (Grant No. 2024YFB4504004), and the Guangzhou Municipal Science and Technology Project (No. 2024A03J0607). F.P. acknowledges the support from the Ministry of Education Singapore under Grant No. SKI 2021_07_03, the National Research Foundation Singapore under its National Quantum Computing Hub translational Fund No. W24Q3D0002 (NQCH), and A*STAR under its MTC Young Individual Research Grant No. H25-MRG3466. The authors also thank Zisong Shen, Cheng Ye, Yan Mi, and Xiwei Pan for helpful discussions on the manuscript.

The Julia [67] implementation of our algorithm is available in Ref. [68]. For tropical tensor networks and dynamic slicing, we use `GenericTensorNetworks.jl` [37,69]. `SCIP` [25,70] is called via `JuMP.jl` [71,72]. All CPU experiments run on an Intel Xeon Platinum 8268 (2.90 GHz). Optimization experiments run on an NVIDIA A100 and counting experiments on an NVIDIA H100.

DATA AVAILABILITY

The data that support the findings of this article are openly available [68], embargo periods may apply.

[1] K. Binder and A. P. Young, Spin glasses: Experimental facts, theoretical concepts, and open questions, Rev. Mod. Phys. **58**, 801 (1986).

[2] K. H. Fischer and J. A. Hertz, *Spin Glasses* (Cambridge University Press, Cambridge, 1991), Vol. 1.

[3] D.-Z. Du and P. M. Pardalos, *Handbook of Combinatorial Optimization* (Springer US, Boston, MA, 1998), Vol. 3.

[4] Q. Zhou and W. H. Wong, Energy landscape of a spin-glass model: Exploration and characterization, Phys. Rev. E **79**, 051117 (2009).

[5] E. S. Rodrigues and P. M. C. de Oliveira, Spin-glass energy landscape, J. Stat. Phys. **74**, 1265 (1994).

[6] V. Ros and Y. V. Fyodorov, The high-dimensional landscape paradigm: Spin-glasses, and beyond, in *Spin Glass Theory and Far Beyond: Replica Symmetry Breaking After 40 Years*, edited by Patrick Charbonneau, Enzo Marinari, Marc Mézard, Giorgio Parisi, Federico Ricci-Tersenghi, Gabriele Sicuro, and Francesco Zamponi (World Scientific, Singapore, 2023), pp. 95–114.

[7] M. Mézard, G. Parisi, and M. A. Virasoro, Theory of the frustration effect in spin glasses: I, in *Spin Glass Theory and Beyond: An Introduction to the Replica Method and Its Applications* (World Scientific, Singapore, 1987), Vol. 9, p. 99.

[8] S. Kirkpatrick, Frustration and ground-state degeneracy in spin glasses, Phys. Rev. B **16**, 4630 (1977).

[9] L. F. Cugliandolo, Course 7: Dynamics of glassy systems, in *Slow Relaxations and Nonequilibrium Dynamics in Condensed Matter*, edited by J.-L. Barrat, M. Feigelman, J. Kurchan, and J. Dalibard (Springer, Berlin, Heidelberg, 2004), pp. 367–521.

[10] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques* (The MIT Press, Cambridge, MA, 2009).

[11] H. Cao, S. Zhao, D. Feng, Z. Shen, H. Yan, T. Su, W. Sun, H. Xu, F. Pan, H. Yu, and P. Zhang, Exact decoding of quantum error-correcting codes, Phys. Rev. Lett. **134**, 190603 (2025).

[12] A. Dutta, *Quantum Phase Transitions in Transverse Field Spin Models* (Cambridge University Press, Cambridge, 2015).

[13] D. Sherrington and S. Kirkpatrick, Solvable model of a spin-glass, Phys. Rev. Lett. **35**, 1792 (1975).

[14] A. Auffinger, W.-K. Chen, and Q. Zeng, The SK model is infinite step replica symmetry breaking at zero temperature, Commun. Pure Appl. Math. **73**, 921 (2020).

[15] G. Parisi, Infinite number of order parameters for spin-glasses, Phys. Rev. Lett. **43**, 1754 (1979).

[16] M. Mézard, G. Parisi, and M. A. Virasoro, *Spin Glass Theory and Beyond: An Introduction to the Replica Method and Its Applications* (World Scientific, Singapore, 1987), Vol. 9.

[17] D. Gamarnik, E. C. Kızıldağ, W. Perkins, and C. Xu, Algorithms and barriers in the symmetric binary perceptron model, in *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)* (IEEE Computer Society, Los Alamitos, CA, 2022), pp. 576–587.

[18] G. Bresler and B. Huang, The algorithmic phase transition of random k-SAT for low degree polynomials, in *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)* (IEEE Computer Society, Los Alamitos, CA, 2022), pp. 298–309.

[19] F. Krzakała, A. Montanari, F. Ricci-Tersenghi, G. Semerjian, and L. Zdeborová, Gibbs states and the set of solutions of random constraint satisfaction problems, Proc. Natl. Acad. Sci. USA **104**, 10318 (2007).

[20] H. Daudé, M. Mézard, T. Mora, and R. Zecchina, Pairs of SAT-assignments in random Boolean formulæ, Theor. Comput. Sci. **393**, 260 (2008).

[21] C. Moore and S. Mertens, *The Nature of Computation* (Oxford University Press, Oxford, 2011).

[22] F. Barahona, On the computational complexity of Ising spin glass models, J. Phys. A: Math. Gen. **15**, 3241 (1982).

[23] C. De Simone, M. Diehl, M. Jünger, P. Mutzel, G. Reinelt, and G. Rinaldi, Exact ground states of Ising spin glasses: New experimental results with a branch-and-cut algorithm, J. Stat. Phys. **80**, 487 (1995).

[24] R. S. Andrist, M. J. A. Schuetz, P. Minssen, R. Yalovetzky, S. Chakrabarti, D. Herman, N. Kumar, G. Salton, R. Shaydulin, Y. Sun, M. Pistoia, and H. G. Katzgraber, Hardness of the maximum independent set problem on unit-disk graphs and prospects for quantum speedups, Phys. Rev. Res. **5**, 043277 (2023).

[25] T. Achterberg, SCIP: Solving constraint integer programs, Math. Program. Comput. **1**, 1 (2009).

[26] F. V. Fomin, F. Grandoni, and D. Kratsch, Measure and conquer: A simple $O(2^{0.288n})$ independent set algorithm, in *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithm* (Society for Industrial and Applied Mathematics, Philadelphia, PA, 2006), pp. 18–25.

[27] C. M. Li, F. Manyà, and J. Planes, Exploiting unit propagation to compute lower bounds in branch and bound Max-SAT solvers, in *Principles and Practice of Constraint Programming—CP 2005* (Springer, Berlin, 2005), pp. 403–414.

[28] A. Abrame and D. Habet, AHMAXSAT: Description and evaluation of a branch and bound Max-SAT solver, J. Satisf. Boolean Model. Comput. **9**, 89 (2015).

[29] T. Akiba and Y. Iwata, Branch-and-reduce exponential/FPT algorithms in practice: A case study of vertex cover, Theor. Comput. Sci. **609**, 211 (2016).

[30] M. Xiao and H. Nagamochi, Confining sets and avoiding bottleneck cases: A simple maximum independent set algorithm in degree-3 graphs, Theor. Comput. Sci. **469**, 92 (2013).

[31] M. Xiao and H. Nagamochi, Exact algorithms for maximum independent set, Inf. Comput. **255**, 126 (2017).

[32] R. Y. Rubinstein and D. P. Kroese, *Simulation and the Monte Carlo Method* (John Wiley & Sons, Hoboken, NJ, 2016).

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

[34] I. L. Markov and Y. Shi, Simulating quantum computation by contracting tensor networks, SIAM J. Comput. **38**, 963 (2008).

[35] U. Schollwöck, The density-matrix renormalization group in the age of matrix product states, Ann. Phys. **326**, 96 (2011).

[36] J.-G. Liu, L. Wang, and P. Zhang, Tropical tensor network for ground states of spin glasses, Phys. Rev. Lett. **126**, 090506 (2021).

[37] J.-G. Liu, X. Gao, M. Cain, M. D. Lukin, and S.-T. Wang, Computing solution space properties of combinatorial optimization problems via generic tensor networks, SIAM J. Sci. Comput. **45**, A1239 (2023).

[38] M. Roa-Villescas, X. Gao, S. Stuijk, H. Corporaal, and J.-G. Liu, Probabilistic inference in the era of tensor networks and differential programming, Phys. Rev. Res. **6**, 033261 (2024).

[39] M. Roa-Villescas and J.-G. Liu, TensorInference: A Julia package for tensor-based probabilistic inference, J. Open Source Software **8**, 5700 (2023).

[40] J. Gray and S. Kourtis, Hyper-optimized tensor network contraction, Quantum **5**, 410 (2021).

[41] J. Argelich, C. M. Li, F. Manyà, and J. R. Soler, Clause branching in MaxSAT and MinSAT, in *Artificial Intelligence Research and Development*, edited by Zoe Falomir, Karina Gibert, and Enric Plaza (IOS Press, Amsterdam, 2018), pp. 17–26.

[42] D. Eppstein, The traveling salesman problem for cubic graphs, J. Graph Algorithms Appl. **11**, 61 (2007).

[43] G. Carpaneto and P. Toth, Some new branching and bounding criteria for the asymmetric travelling salesman problem, Manage. Sci. **26**, 736 (1980).

[44] T. Achterberg, Constraint integer programming, Ph.D. thesis, Technische Universität Berlin, 2007.

[45] D. R. Morrison, S. H. Jacobson, J. J. Sauppe, and E. C. Sewell, Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning, Discrete Optim. **19**, 79 (2016).

[46] X. Gao, Y.-J. Wang, P. Zhang, and J.-G. Liu, Automated discovery of branching rules with optimal complexity for the maximum independent set problem, arXiv:2412.07685.

[47] H. Pichler, S.-T. Wang, L. Zhou, S. Choi, and M. D. Lukin, Quantum optimization for maximum independent set using Rydberg atom arrays, arXiv:1808.10816.

[48] S. Ebadi *et al.*, Quantum optimization of maximum independent set using Rydberg atom arrays, Science **376**, 1209 (2022).

[49] M. Kim, K. Kim, J. Hwang, E.-G. Moon, and J. Ahn, Rydberg quantum wires for maximum independent set problems, Nat. Phys. **18**, 755 (2022).

[50] L. Bombieri, Z. Zeng, R. Tricarico, R. Lin, S. Notarnicola, M. Cain, M. D. Lukin, and H. Pichler, Quantum adiabatic optimization with Rydberg arrays: Localization phenomena and encoding strategies, PRX Quantum **6**, 020306 (2025).

[51] N.-C. Chiu, E. C. Trapp, J. Guo, M. H. Abobeih, L. M. Stewart, S. Hollerith, P. L. Stroganov, M. Kalinowski, A. A. Geim, S. J. Evered, S. H. Li, X. Lyu, L. M. Peters, D. Bluvstein, T. T. Wang, M. Greiner, V. Vuletić, and M. D. Lukin, Continuous operation of a coherent 3,000-qubit system, Nature (London) **646**, 1075 (2025).

[52] D. Bluvstein *et al.*, A fault-tolerant neutral-atom architecture for universal quantum computation, Nature (London) **649**, 39 (2026).

[53] S. Kourtis, C. Chamon, E. Mucciolo, and A. E. Ruckenstein, Fast counting with tensor networks, SciPost Phys. **7**, 060 (2019).

[54] See Supplemental Material at http://link.aps.org/supplemental/10.1103/g76v-wxz7 for (1) algorithmic details; (2) geometric interpretation of one branching step; (3) exact-counting benchmarks and Chinese-remainder reconstruction; (4) branching-strategy and overhead comparisons; and (5) solution-landscape

analyses, attached to this manuscript, which includes Refs. [59–64].

[55] J. Chen, F. Zhang, C. Huang, M. Newman, and Y. Shi, Classical simulation of intermediate-size quantum circuits, arXiv:1805.01450.

[56] *Computational Complexity and Statistical Physics*, edited by A. Percus, G. Istrate, and C. Moore (Oxford University Press, New York, NY, 2006).

[57] E. Danna, M. Fenelon, Z. Gu, and R. Wunderling, Generating multiple solutions for mixed integer programming problems, in *Integer Programming and Combinatorial Optimization, Lecture Notes in Computer Science*, edited by M. Fischetti and D. P. Williamson (Springer, Berlin, 2007), Vol. 4513, pp. 280–294.

[58] T. A. Davis and Y. Hu, The University of Florida sparse matrix collection, ACM Trans. Math. Software **38**, 1 (2011).

[59] R. P. Brent and P. Zimmermann, *Modern Computer Arithmetic* (Cambridge University Press, Cambridge, UK, 2010).

[60] D. Coppersmith, Finding a small root of a bivariate integer equation; factoring with high bits known, in *International Conference on the Theory and Applications of Cryptographic Techniques* (Springer, Berlin, Heidelberg, 1996), pp. 178–189.

[61] S. Jin, A. Sen, W. Guo, and A. W. Sandvik, Phase transitions in the frustrated Ising model on the square lattice, Phys. Rev. B **87**, 144406 (2013).

[62] G. Kalachev, P. Panteleev, and M.-H. Yung, Multi-tensor contraction for XEB verification of quantum circuits, arXiv:2108.05665.

[63] A. Kalz, A. Honecker, S. Fuchs, and T. Pruschke, Phase diagram of the Ising square lattice with competing interactions, Eur. Phys. J. B **65**, 533 (2008).

[64] J.-G. Liu, J. Wurtz, M.-T. Nguyen, M. D. Lukin, H. Pichler, and S.-T. Wang, Computer-assisted gadget design and problem reduction of unweighted maximum independent set (unpublished).

[65] M.-T. Nguyen, J.-G. Liu, J. Wurtz, M. D. Lukin, S.-T. Wang, and H. Pichler, Quantum optimization with arbitrary connectivity using Rydberg atom arrays, PRX Quantum **4**, 010316 (2023).

[66] Z. Hou, X. Li, Y. Zhang, T. Li, and K. You, LLM4Branch: Large language model for discovering efficient branching policies of integer programs, in *Proceedings of Machine Learning Research* (2026), Vol. 306, pp. 44132–44155.

[67] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, Julia: A fresh approach to numerical computing, SIAM Rev. **59**, 65 (2017).

[68] X. Gao, Y.-J. Wang, and J.-G. Liu, TensorBranching.jl: A Julia package for the branch-and-bound tensor-network method, GitHub, 2025, https://github.com/xuanzhaogao/TensorBranching.jl.

[69] X. Gao, X. Li, and J. Liu, Programming guide for solving constraint satisfaction problems with tensor networks, Chin. Phys. B **34**, 050201 (2025).

[70] T. Achterberg, T. Berthold, T. Koch, and K. Wolter, Constraint integer programming: A new approach to integrate CP and MIP, in *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, edited by Laurent Perron and Michael A. Trick (Springer, Berlin, Heidelberg, 2008), pp. 6–20.

[71] I. Dunning, J. Huchette, and M. Lubin, JuMP: A modeling language for mathematical optimization, SIAM Rev. **59**, 295 (2017).

[72] M. Lubin, O. Dowson, J. D. Garcia, J. Huchette, B. Legat, and J. P. Vielma, JuMP 1.0: Recent improvements to a modeling language for mathematical optimization, Math. Program. Comput. **15**, 581 (2023).