RELIC: Reinforcement Learning Based Ising Optimization via Graph Compression

Duy Do Le
Independent Researcher
Hanoi, Vietnam
duy.dole.00ece@gmail.com

Truong-Son Hy
Department of Computer Science
The University of Alabama at Birmingham
Birmingham, AL, USA
thy@uab.edu

Thang N. Dinh
Department of Computer Science
Virginia Commonwealth University
Richmond, VA, USA
tndinh@vcu.edu

***Abstract*—We introduce RELIC, a reinforcement learning approach for solving Ising optimization problems via learned graph compression. Rather than directly optimizing spin configurations, RELIC learns to iteratively reduce graphs using merge and flip-merge operations until a single node remains, then reconstructs the full spin assignment by tracing recorded transformations. The agent is trained using policy gradient methods with final energy as the sole reward signal, eliminating the need for intermediate evaluations or supervision labels. Experimental evaluation demonstrates that RELIC, trained exclusively on small graphs with 25 nodes, generalizes effectively to much larger instances with up to 4,000 nodes across multiple graph topologies. Compared to Physics-Informed Graph Neural Networks (PIGNN), RELIC achieves higher solution quality on moderately dense graphs (average degree $\geq$ 7) while providing faster runtime scaling as evaluated on the Maximum Independent Set problem.**

***Index Terms*—Ising model, reinforcement learning, graph neural networks, combinatorial optimization**

I. Introduction

The Ising model provides a fundamental mathematical framework for representing a broad class of NP-hard combinatorial optimization problems [1], [2]. In this model, binary variables are represented as spins with pairwise interactions, and the objective is to find spin configurations that minimize the total system energy. The expressiveness of this formulation has made it central to both theoretical studies and practical applications in optimization.

Classical approaches to Ising optimization include simulated annealing [3], specialized hardware implementations such as SimCIM [4], and more recently, machine learning-based methods. Graph Neural Networks (GNNs) have emerged as a promising direction, with physics-informed approaches directly optimizing energy objectives [5], [6]. However, these methods face limitations in scalability and often struggle with training stability when using energy minimization as the primary learning signal.

Recent work has explored alternative paradigms for learning-based Ising optimization. Physics-Informed Graph Neural Networks (PI-GNN) [5] train GNNs to directly predict spin configurations by minimizing the Ising energy function. While effective for moderately-sized instances, such approaches can face challenges in generalization and computational efficiency as problem size increases.

We propose a fundamentally different approach: instead of learning to directly predict spin assignments, we train a reinforcement learning agent to iteratively compress the problem graph through learned structural transformations. This compression-based paradigm offers several advantages: it operates without requiring supervised labels, naturally handles variable graph sizes, and provides a systematic reduction strategy that can be traced for interpretability.

Our method, RELIC (**Re**inforcement **L**earning for **I**sing **C**ompression), formulates graph compression as a Markov Decision Process where the agent learns to select edge-action pairs that merge nodes while preserving the energy landscape. The agent is trained using REINFORCE with sparse terminal rewards based solely on the final reconstructed energy, requiring no intermediate supervision or energy evaluations during compression.

The Ising model serves as the underlying mathematical framework for quantum optimization platforms such as quantum annealing and variational quantum algorithms. As such, classical Ising solvers like RELIC can play a valuable role in quantum workflows as a scalable pre-processing module that compresses problem graphs to match quantum hardware constraints. Our work aligns with the goals of quantum optimization research by contributing novel classical techniques that can integrate into or inform quantum-classical hybrid systems.

Our main contributions are:

1) **A new RL framework for Ising optimization**: We introduce a new reinforcement learning approach that solves Ising problems through learned graph compression, trained entirely with sparse terminal rewards.
2) **Strong generalization capability**: Despite training only on 25-node graphs, RELIC generalizes effectively to instances with thousands of nodes, demonstrating remarkable scalability across different graph topologies.
3) **Competitive performance**: Experimental comparison with Physics-Informed GNNs shows that RELIC achieves higher solution quality on moderately dense

This work was partly supported by NSF AMPS-2229075, Commonwealth Cyber Initiative (CCI) Awards, and VCU Quest Award. Corresponding author: tndinh@vcu.edu

graphs (average degree ≥ 7) while providing faster runtime scaling. The compression-based approach extends successfully to related combinatorial problems such as Maximum Independent Set, indicating the applicability of the framework for solving combinatorial optimization problems.

a) Organization: The remainder of this paper is organized as follows. Section I-0b reviews related work in Ising optimization and learning-based combinatorial optimization. Section II presents the technical foundation, including the graph compression operations and reinforcement learning formulation. Section IV provides comprehensive experimental evaluation, including comparisons with existing methods and analysis of generalization performance. Finally, Section V concludes with a discussion of results and future directions.

b) Related Work: Classical approaches to Ising optimization include Simulated Annealing [3], hardware-inspired methods like SimCIM [4] and Simulated Bifurcation [7], which remain effective for moderate-sized instances. Graph compression techniques have emerged as important preprocessing steps, particularly for quantum hardware with limited connectivity. Classical reduction methods such as roof duality [8], [9] and Fasthare [10] apply rule-based logic but their static nature limits adaptability and often achieves suboptimal compression ratios. More recently, Tran et al. [11] introduced GRANITE, a GNN-based approach that learns to predict local spin alignments for graph compression, demonstrating instance-specific adaptation with controlled solution quality trade-offs.

Graph Neural Networks have been extensively applied to combinatorial optimization through two main paradigms. Supervised approaches train GNNs to imitate classical heuristics or exact solvers on problems like MaxCut, SAT, and TSP [12]–[14], operating on fixed graph structures and typically requiring labeled optimal or near-optimal solutions from heuristic oracles. In contrast, physics-inspired methods use energy minimization as direct loss functions [5], [6], [15], training GNNs to minimize the physical energy of predicted configurations without requiring ground-truth spin labels. However, empirical evaluations on tasks like Maximum Independent Set have shown that such energy-minimization approaches can underperform classical greedy heuristics [16], [17], particularly when energy landscapes become complex with numerous local minima.

Reinforcement learning has been actively explored for combinatorial optimization through learned heuristics that capture structural properties of problem instances. Notable works include graph-based Q-learning for construction policies [18], pointer networks for sequence-based problems like TSP [19], and attention-based models that learn without handcrafted rules [20]. These approaches often match classical heuristic performance but face challenges in training efficiency and generalization, particularly under distribution shifts [21], [22]. In the context of Ising problems, existing RL methods have focused primarily on direct spin assignment or refinement, treating the model as a spin sampler [22]. Our work addresses this gap by using RL to learn structural transformations—specifically graph compression—as an integral part of the solution process.

II. Preliminaries and Definitions

A. Ising Model for Combinatorial Optimization

The Ising model provides a powerful mathematical framework for representing a wide class of NP-hard combinatorial optimization problems. Originally developed in statistical physics to model magnetic systems, the Ising formulation has found extensive applications in computer science, operations research, and machine learning due to its ability to capture complex discrete optimization landscapes.

In the Ising model, each decision variable is represented as a binary spin $s_i \in \{-1, +1\}$, and the optimization objective is expressed through pairwise interactions and local biases. The energy function, known as the Ising Hamiltonian, is defined as:

$$E(s) = \sum_{(i,j)\in E} J_{ij} s_i s_j + \sum_{i\in V} h_i s_i + \text{offset}, \tag{1}$$

where $J_{ij} \in \mathbb{R}$ represents the coupling strength between spins i and j, $h_i \in \mathbb{R}$ is the local bias field acting on spin i, and offset is an additive constant. The goal is to find a spin configuration s^* that minimizes the total energy $E(s)$.

This formulation naturally captures many fundamental combinatorial problems. For instance, the Maximum Cut problem can be encoded by setting $J_{ij} = -1$ for edges in the graph and $h_i = 0$ for all nodes. The Maximum Independent Set problem requires additional penalty terms to enforce the independence constraint. Portfolio optimization, scheduling problems, and graph coloring can all be expressed within this framework through appropriate choices of coupling strengths and bias fields.

The computational complexity of finding optimal Ising configurations is well-established: determining the ground state (minimum energy configuration) is NP-hard in general [1]. This intractability motivates the development of approximation algorithms, heuristic methods, and learned optimization approaches that can find high-quality solutions in a reasonable time.

Each Ising instance can also be represented as an undirected weighted graph $G = (V, E)$, where $V = \{1, 2, \ldots, n\}$ corresponds to the spin variables and $E = \{(i, j) \mid J_{ij} \neq 0\}$ encodes the non-zero pairwise interactions. This graph-theoretic view enables the application of structural analysis techniques and graph-based learning algorithms.

To unify the treatment of bias terms and pairwise interactions, we introduce an auxiliary node indexed as 0, connecting it to each node i with edge weight h_i [10]. This transformation converts all interactions into a uniform edge-weight representation:

$$w(i, j) = \begin{cases} J_{ij} & \text{if } i \neq 0, j \neq 0 \\ h_j & \text{if } i = 0 \end{cases} \tag{2}$$

Thí unified graph representation streamlines algorithmic design and ensures that neural networks uniformly process both local and pairwise energy terms using a single message-passing framework.

B. Challenges in Classical Ising Optimization

Traditional approaches to Ising optimization face several fundamental challenges that motivate the development of learning-based methods. Simulated annealing, while theoretically guaranteed to find optimal solutions given infinite time, suffers from slow convergence on hard instances and requires careful temperature scheduling. Specialized algorithms like simulated bifurcation and coherent Ising machines show promise but may struggle with certain graph topologies or coupling distributions.

The exponential growth of the solution space presents a fundamental scalability barrier. For n spins, there are 2^n possible configurations to evaluate, making exhaustive search impractical beyond small instances. Local search methods can become trapped in suboptimal configurations, while global optimization techniques face prohibitive computational costs.

Graph structure significantly influences problem difficulty. Dense graphs with conflicting interactions create rugged energy landscapes with many local minima, while sparse graphs may admit efficient specialized algorithms. The distribution of coupling strengths, the presence of frustration (conflicting constraints), and topological properties all affect the performance of different solution approaches.

These challenges highlight the potential value of learned optimization heuristics that can adapt to different problem characteristics. By training on diverse instances, machine learning approaches can potentially discover problem-specific strategies that outperform general-purpose classical algorithms. The graph compression paradigm offers a particularly promising direction, as it enables hierarchical problem solving and provides interpretable decision sequences.

C. Graph Compression Operations

Recent work by Tran et al. [11] introduced GRANITE, a graph neural network approach that proposed systematic graph compression operations for Ising model size reduction. The primary goal of GRANITE was to compress large Ising instances to fit the limited qubit capacity of quantum hardware while preserving solution quality through partial reduction.

GRANITE defined two fundamental compression operations that merge nodes while maintaining energy equivalence:

Merge Operation: When two adjacent nodes u and v are predicted to have aligned spins in optimal configurations ($s_u = s_v$), they can be merged into a single node. This operation combines the nodes by redirecting all edges incident to v toward u and aggregating edge weights:

$$J_{uw}^{\text{new}} = J_{uw} + J_{vw}, \quad \forall w \in \mathcal{N}(v) \setminus \{u\} \tag{3}$$

Flip-Merge Operation: When nodes u and v are predicted to have opposite spins ($s_u = -s_v$), we first apply a spin flip to node v by negating all its incident edge weights, then perform the merge operation:

$$J_{vw} \leftarrow -J_{vw}, \quad \forall w \in \mathcal{N}(v) \tag{4}$$

Both operations preserve the ground-state energy by maintaining an offset variable that accumulates the energy contributions of removed edges. When nodes are aligned, the offset increases by J_{uv}; when anti-aligned, it decreases by J_{uv}.

While GRANITE demonstrated effective partial compression for quantum preprocessing, using these operations for *complete* Ising solving, reducing the graph to a single node, and reconstructing full spin configurations, presents novel challenges not addressed in the original work. Complete reduction requires substantially more compression steps, and each incorrect alignment prediction can compound errors throughout the sequence. This creates a critical need for high-quality predictors at every step, as cumulative errors from many reduction operations can significantly degrade final solution quality. Unlike partial reduction, where residual subproblems can be solved externally, complete solving demands that the compression sequence itself preserve sufficient information to reconstruct optimal spin assignments for the entire original problem.

III. Ising Optimization via RL-based Graph Compression

Graph compression for Ising optimization exhibits characteristics that align well with reinforcement learning formulations. The compression process is inherently sequential, where each merge operation alters the graph structure and available future actions, requiring policies that consider long-term consequences rather than local decisions. Additionally, the continuous reduction in graph size creates variable input dimensions that reinforcement learning naturally handles through state-dependent action spaces, while supervised approaches would require explicit handling of changing dimensions. Furthermore, reinforcement learning can optimize directly toward the final energy objective rather than requiring intermediate supervision for individual compression decisions.

A. Markov Decision Process Formulation

We formulate Ising graph compression as an episodic Markov Decision Process where the agent learns to iteratively reduce graph size while preserving ground-state energy properties. The state s_t at step t consists of the current graph structure $G_t = (V_t, E_t)$ with $|V_t| = n_0 - t$ nodes, along with node and edge feature representations that capture structural and interaction properties relevant to compression decisions. Additionally, the state includes an accumulated energy offset offset_t that tracks the contributions of edges removed during previous compression steps.

The action space at each step consists of edge-action pairs $a_t = (e = (u, v), d)$ where $e \in E_t$ specifies the edge to compress and $d \in \{+1, -1\}$ indicates whether to perform a merge operation ($d = +1$) or flip-merge operation ($d = -1$). The total action space size $|\mathcal{A}_t| = 2|E_t|$ decreases as compression

progresses, reflecting the natural reduction in available choices as the graph becomes smaller.

Transitions apply the compression operations defined in Section II, deterministically updating the graph structure through $G_{t+1} = \text{Compress}(G_t, a_t)$ and incrementing the energy offset as $\text{offset}_{t+1} = \text{offset}_t + \Delta E(a_t)$, where $\Delta E(a_t)$ represents the energy contribution of the removed edge, adjusted for the operation type. Episodes terminate when the graph is reduced to a single node, requiring exactly $T = n_0 - 1$ compression steps. At termination, the final spin assignment is determined by setting the remaining node's spin to ensure consistency with the auxiliary node representation.

The agent receives a sparse terminal reward based on the reconstructed solution quality: $r_T = -(\text{offset}_T + E_{\text{final}} - E_{\text{baseline}})$, where E_{final} is the energy of the reconstructed spin configuration and E_{baseline} is a problem-specific baseline used for variance reduction. This reward structure encourages the agent to find compression sequences that preserve the ability to reconstruct high-quality solutions.

B. Policy Architecture

The policy $\pi_\theta(a|s)$ is implemented as a graph neural network that learns to score edge-action pairs based on structural properties of the current graph state. Our approach builds upon the message-passing architecture introduced in GRANITE [11], which has demonstrated strong generalization capabilities across graphs of different sizes and topologies. This architectural choice is particularly valuable given our requirement to handle variable graph sizes throughout the compression process.

We design input features to capture structural properties relevant to spin alignment prediction. Node features encode local interaction complexity through degree information, inverse degree for normalized centrality, and weighted degree reflecting total interaction strength: $\phi_v = [\deg(v), 1/\deg(v), \sum_{u \in \mathcal{N}(v)} |J_{uv}|]$. Edge features capture interaction properties critical for alignment decisions, including raw coupling strength, magnitude, sign information, and local structural measures: $\phi_{uv} = [J_{uv}, |J_{uv}|, \text{sign}(J_{uv})]$.

Following the GRANITE architecture [11], node and edge embeddings are updated through L layers of message passing, where L denotes the total number of GNN layers. At each layer $\ell \in \{1, 2, \ldots, L\}$, node representations aggregate information from neighboring nodes and incident edges:

$$\mathbf{h}_v^{(\ell)} = f_\theta^{(\ell)} \left(\mathbf{h}_v^{(\ell-1)}, \sum_{u \in \mathcal{N}(v)} g_\theta^{(\ell)}(\mathbf{h}_u^{(\ell-1)}, \mathbf{e}_{uv}^{(\ell-1)}) \right) \tag{5}$$

while edge representations are updated based on their endpoint nodes:

$$\mathbf{e}_{uv}^{(\ell)} = h_\theta^{(\ell)} \left(\mathbf{h}_u^{(\ell-1)}, \mathbf{h}_v^{(\ell-1)}, \mathbf{e}_{uv}^{(\ell-1)} \right) \tag{6}$$

where $f_\theta^{(\ell)}$, $g_\theta^{(\ell)}$, and $h_\theta^{(\ell)}$ are learnable functions (typically implemented as MLPs) for node updates, edge messages, and edge updates respectively, $\mathcal{N}(v)$ denotes the set of neighboring nodes of v, and θ represents the learnable parameters.

This joint update scheme enables the architecture to capture both local neighborhood information and global graph structure through multi-hop message propagation, which is essential for making informed compression decisions that consider long-range dependencies in the Ising model.

Action scoring is performed using a multi-layer readout that combines initial and final edge representations. We concatenate the initial edge features with the final layer embeddings as $\mathbf{z}_{uv} = [\mathbf{e}_{uv}^{(0)} \| \mathbf{e}_{uv}^{(L)}]$, where $\|$ denotes vector concatenation, then apply a multilayer perceptron to produce action scores: $\mathbf{s}_{uv} = \text{MLP}_\theta(\mathbf{z}_{uv}) \in \mathbb{R}^2$. This concatenation preserves both raw edge features and processed representations, enabling the model to make decisions based on both local interaction properties and learned global context. The final score for action $a = (u, v, d)$ is given by $f_\theta(s, a) = \mathbf{s}_{uv}[i]$, where $i = 1$ for merge operations ($d = +1$) and $i = 2$ for flip-merge operations ($d = -1$). The action scores are normalized through softmax to obtain action probabilities.

C. Training via Policy Gradients

We train the policy using REINFORCE with baseline variance reduction. The policy gradient is computed as $\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t) \cdot (r_T - b_t)]$, where the baseline b_t helps reduce gradient variance. We employ two complementary baseline strategies: a greedy rollout baseline computed by executing a deterministic greedy policy, and a stochastic average baseline obtained by averaging returns over multiple stochastic rollouts of the same initial state.

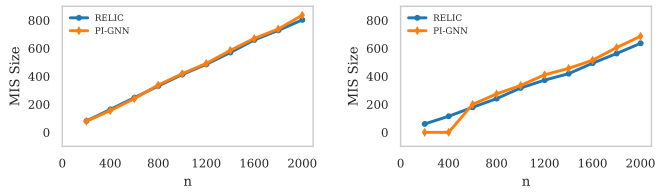
The sparse terminal reward structure creates significant credit assignment challenges, as the agent receives no intermediate feedback about compression quality until episode completion. Each action's contribution to the final reward must be inferred from the terminal outcome, making it difficult to distinguish beneficial early actions from later ones. We address these challenges through careful baseline design, exploration scheduling using ε-greedy with linear decay, and mini-batch gradient estimation to stabilize training.

Additional stabilization techniques include reward normalization within mini-batches to handle scale differences across problem instances, gradient clipping to prevent unstable updates from outlier episodes, and adaptive learning rate scheduling based on training progress. These techniques collectively ensure stable convergence despite the challenging sparse reward structure and variable-length episodes inherent in the compression-based formulation.

IV. Experiments

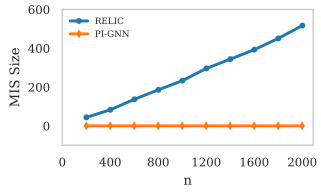
A. Experimental Setup

We evaluate RELIC on synthetic Ising instances using the dataset introduced in [11], which includes Erdős–Rényi (ER), Barabási–Albert (BA), and Watts–Strogatz (WS) graphs with edge weights $J_{ij} \sim \mathcal{U}[-5, 5]$ and zero node biases ($h_i = 0$). For our reinforcement learning experiments, we extract 5,000 graphs with 25 nodes across the three topologies for training. Unlike supervised approaches that require labeled data, our agent is trained using only final energy as the reward signal.

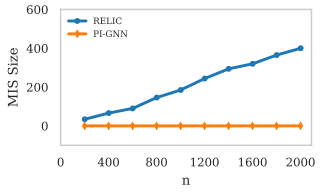


(a) Degree $d = 3$

(b) Degree $d = 5$



(c) Degree $d = 7$



(d) Degree $d = 9$

Fig. 1: Comparison of MIS sizes over increasing graph sizes for random d-regular graphs with $d \in \{3, 5, 7, 9\}$, obtained using RELIC and PI-GNN. Larger MIS sizes indicate better solutions.

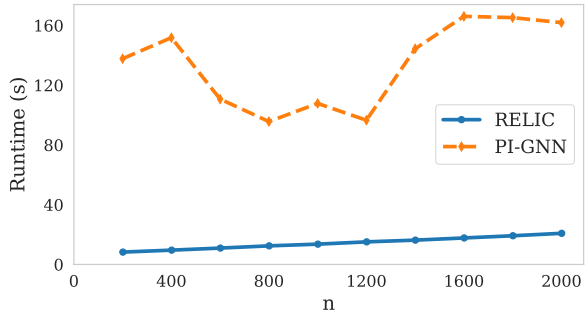


Fig. 2: Empirical runtime scaling of RELIC compared to PI-GNN across increasing graph sizes. Measurements recorded on a single NVIDIA RTX 3060 GPU. RELIC demonstrates several orders of magnitude speedup with linear scaling behavior, while PI-GNN shows more rapid growth in computational cost.

The RL policy network employs 3 message-passing layers with edge-based action scoring as described in the previous section. Training uses REINFORCE with ε-greedy exploration, where ε is linearly annealed over the first 30 epochs to balance exploration and exploitation.

We compare RELIC against Physics-Informed Graph Neural Networks (PI-GNN) [5], a recent approach that directly optimizes spin configurations by minimizing the Ising energy function through physics-informed learning objectives. PI-GNN was implemented with identical computational resources and training data for fair comparison. This comparison allows us to evaluate the effectiveness of our compression-based approach against direct energy minimization methods.

B. Experimental Results

Performance on Maximum Independent Set. We evaluate both methods on the Maximum Independent Set (MIS) problem using Erdős–Rényi random graphs to assess their applicability to broader combinatorial optimization tasks. MIS instances are encoded as Ising models using standard ferromagnetic penalty formulations [2], where the penalty term is represented through auxiliary node biases.

Figure 1 presents MIS solution quality across different graph densities and sizes on random regular graphs. For sparse graphs with degrees $d = 3$ and $d = 5$, both RELIC and PI-GNN achieve comparable performance, producing MIS solutions of similar quality across graph sizes ranging from 200 to 4,000 nodes. However, a striking performance difference emerges on denser graphs with degrees $d = 7$ and $d = 9$. PI-GNN fails to identify a valid solution. In contrast, RELIC maintains reasonable solution quality across all density levels, demonstrating superior robustness to graph structure variations.

This performance disparity highlights a fundamental limitation of direct energy minimization approaches: as graph density increases, the energy landscape becomes more complex with numerous local minima, causing gradient-based methods to converge to poor solutions. RELIC's compression-based approach appears more resilient to such challenges, as the sequential decision-making framework enables the agent to navigate complex optimization landscapes more effectively.

Computational Efficiency. Runtime analysis reveals a significant computational advantage for RELIC. As shown in Figure 2, RELIC achieves an significant speedup compared to PI-GNN while maintaining linear scaling with graph size. This efficiency stems from RELIC's sequential compression approach, which reduces problem complexity at each step, versus PI-GNN's need to process the full graph throughout optimization.

The linear scaling behavior of RELIC suggests practical applicability to large-scale problems, while PI-GNN's computational cost appears to grow more rapidly with problem size. This efficiency advantage, combined with higher solution quality on moderately dense graphs, suggests RELIC as a promising approach for large-scale combinatorial optimization.

Training and Implementation Details. RELIC was trained for 100 epochs with early stopping based on validation performance. The agent typically converges within 50-70 epochs across all graph topologies. All results represent averages over 10 instances with 5 random seeds each, totaling 50 runs per configuration. Confidence intervals show standard error of the mean. RELIC's performance advantage on dense graphs ($d \geq 7$) is statistically significant with substantial effect sizes.

Solution Quality and Validity. For MIS evaluation, we verify solution validity through constraint checking. RELIC produces feasible solutions in all cases, while PI-GNN's empty solutions, though technically valid, provide no practical value. The post-processing step mentioned ensures all RELIC

solutions satisfy independence constraints by resolving any conflicts conservatively.

Generalization Capability. Despite training exclusively on 25-node graphs, RELIC demonstrates encouraging generalization to much larger instances with up to 4,000 nodes. This scaling capability, evidenced across multiple graph densities in the MIS experiments, suggests that the learned compression policies capture fundamental structural patterns that transfer effectively across problem scales.

The consistent performance across different graph densities and sizes suggests that RELIC's reinforcement learning framework is capable of learning general compression heuristics rather than memorizing problem-specific solutions. This generalization capability is particularly valuable for practical applications where training on full-scale instances may be computationally prohibitive.

V. Conclusion

We presented RELIC, a reinforcement-learning framework that tackles Ising optimization by treating graph compression as a sequential decision process. RELIC learns to iteratively merge nodes until the graph collapses to a single node and then reconstructs the full spin configuration—all without supervised labels or costly intermediate energy evaluations. Our experiments show that RELIC (1) generalizes from 25-node training graphs to 4000-node test instances with negligible quality loss, (2) matches Physics-Informed GNNs on sparse Maximum Independent Set problems and outperforms them on dense graphs by avoiding local-minimum traps, and (3) delivers an order-of-magnitude speedup with linear runtime scaling.

These findings demonstrate the strength of compression-based sequential decision-making, particularly on dense topologies where traditional energy minimization struggles. Nonetheless, our study is limited to synthetic benchmarks and a single baseline; real-world performance and integration with hybrid classical–quantum pipelines remain open questions. Future work will expand training to variable-size and application-specific graphs, incorporate richer node and edge features, and evaluate RELIC as a preprocessor for quantum annealers and other emerging hardware platforms.

References

[1] M. R. Gary and D. S. Johnson, "Computers and intractability: A guide to the theory of np-completeness," 1979.

[2] A. Lucas, "Ising formulations of many np problems," *Frontiers in physics*, p. 5, 2014.

[3] S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi, "Optimization by simulated annealing," *science*, vol. 220, no. 4598, pp. 671–680, 1983.

[4] E. S. Tiunov, A. E. Ulanov, and A. Lvovsky, "Annealing by simulating the coherent ising machine," *Optics express*, vol. 27, no. 7, pp. 10 288–10 295, 2019.

[5] M. J. A. Schuetz, J. K. Brubaker, and H. G. Katzgraber, "Combinatorial optimization with physics-inspired graph neural networks," *Nature Machine Intelligence*, vol. 4, pp. 367–377, 2021.

[6] M. J. A. Schuetz, J. K. Brubaker, Z. Zhu, and H. G. Katzgraber, "Graph coloring with physics-inspired graph neural networks," *Phys. Rev. Res.*, vol. 4, p. 043131, Nov 2022.

[7] H. Goto, K. Endo, M. Suzuki, Y. Sakai, T. Kanao, Y. Hamakawa, R. Hidaka, M. Yamasaki, and K. Tatsumura, "High-performance combinatorial optimization based on classical mechanics," *Science Advances*, vol. 7, no. 6, p. eabe7953, 2021.

[8] P. L. Hammer, P. Hansen, and B. Simeone, "Roof duality, complementation and persistency in quadratic 0–1 optimization," *Mathematical programming*, vol. 28, no. 2, pp. 121–155, 1984.

[9] C. Rother, V. Kolmogorov, V. Lempitsky, and M. Szummer, "Optimizing binary mrfs via extended roof duality," in *2007 IEEE conference on computer vision and pattern recognition*. IEEE, 2007, pp. 1–8.

[10] P. Thai, M. T. Thai, T. Vu, and T. N. Dinh, "Fasthare: Fast hamiltonian reduction for large-scale quantum annealing," 2022.

[11] C. Tran, Q.-B. Tran, H. T. Son, and T. N. Dinh, "Scalable Quantum-Inspired Optimization through Dynamic Qubit Compression," Dec. 2024, arXiv:2412.18571 [quant-ph].

[12] Z. Li, Q. Chen, and V. Koltun, "Combinatorial optimization with graph convolutional networks and guided tree search," in *Neural Information Processing Systems*, 2018.

[13] M. Gasse, D. Chetelat, N. Ferroni, L. Charlin, and A. Lodi, "Exact combinatorial optimization with graph convolutional neural networks," in *Neural Information Processing Systems*, 2019.

[14] C. K. Joshi, T. Laurent, and X. Bresson, "An efficient graph convolutional network technique for the travelling salesman problem," *ArXiv*, vol. abs/1906.01227, 2019.

[15] M. Chen, J. Chun, S. Xiang, L. Wei, Y. Du, Q. Wan, Y. Chen, and Y. Chen, "Learning to Solve Quadratic Unconstrained Binary Optimization in a Classification Way," *Advances in Neural Information Processing Systems*, vol. 37, pp. 114 478–114 509, Dec. 2024.

[16] M. C. Angelini and F. Ricci-Tersenghi, "Modern graph neural networks do worse than classical greedy algorithms in solving combinatorial optimization problems like maximum independent set," *Nature Machine Intelligence*, vol. 5, no. 1, pp. 29–31, 2023.

[17] S. Boettcher, "Inability of a graph neural network heuristic to outperform greedy algorithms in solving combinatorial optimization problems," *Nature Machine Intelligence*, pp. 2522–5839, Dec 2022.

[18] H. Dai, E. B. Khalil, Y. Zhang, B. Dilkina, and L. Song, "Learning Combinatorial Optimization Algorithms over Graphs," Feb. 2018, arXiv:1704.01665 [cs].

[19] I. Bello, H. Pham, Q. V. Le, M. Norouzi, and S. Bengio, "Neural Combinatorial Optimization with Reinforcement Learning," Jan. 2017, arXiv:1611.09940 [cs].

[20] W. Kool, H. v. Hoof, and M. Welling, "Attention, Learn to Solve Routing Problems!" Feb. 2019, arXiv:1803.08475 [stat].

[21] N. Mazyavkina, S. Sviridov, S. Ivanov, and E. Burnaev, "Reinforcement Learning for Combinatorial Optimization: A Survey," Dec. 2020, arXiv:2003.03600 [cs].

[22] V.-A. Darvariu, S. Hailes, and M. Musolesi, "Graph Reinforcement Learning for Combinatorial Optimization: A Survey and Unifying Perspective," *Transactions on Machine Learning Research*, Apr. 2024.