RELIC: Reinforcement Learning Based Ising Optimization via Graph Compression

Duy Do Le[1] Truong-Son Hy[2] Thang N. Dinh[3]

IEEE QCRL 2025 Workshop

Presenter: Phuong Cao Cong[3]

[1]Independent Researcher
[2]The University of Alabama at Birmingham
[3]Virginia Commonwealth University
tndinh@vcu.edu

Agenda

Introduction

Method

Experiments

Introduction

Ising Model

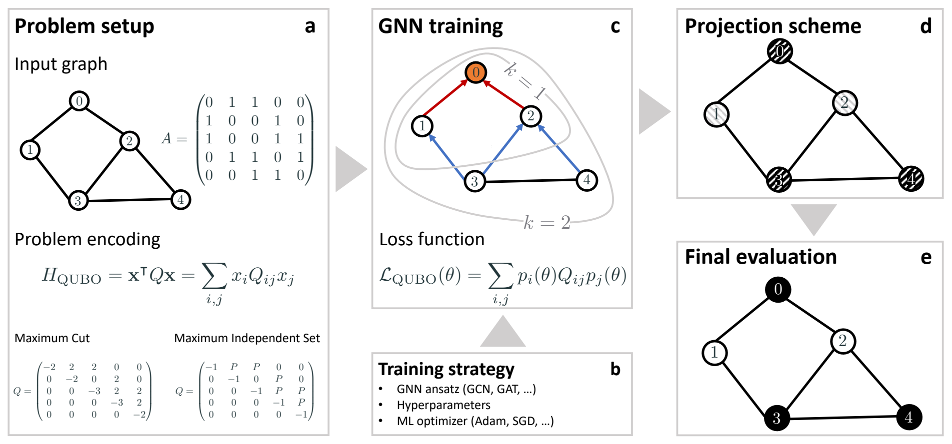
$$E(s) = \sum_{(i,j)\in E} J_{ij}\, s_i s_j + \sum_i h_i s_i$$

- **Problem:** minimize energy over spin configurations $s_i \in \{-1, +1\}$
- A PI-encoding framework for most NP-complete combinatorial optimization problems
  - Traveling Salesman
  - Portfolio optimization
  - Molecular docking
  - Next-G MIMO Processing

Ising Solvers

- **Exact methods**: GUROBI, CPLEX, MOSEK
  - Exponential-time worst case due to the NP-hardness
- **Classical heuristics**: Simulated Annealing, Tabu Search, Genetic Algorithms
- **Quantum & Quantum-inspired solvers**: Quantum annealing (QA), Gate-based QAOA, Digital Annealers
- **Neural combinatorial solvers**: Learn *heuristics* and or *policies* that generalize across instances using Deep Learning
  - **Problem-specific solvers**:
    - Pointer Networks [TSP, VRP]: NeurIPS 2015 – SL/RL sequence models
    - Policy gradient to learn heuristics [TSP, Knapsack] (Bello et al., ICLR 2017)
    - Transformer-based RL solvers [TSP, VRP] (Kool et al., ICLR 2019)
  - **General Ising/QUBO Solver**:
    - Physical-inspired GNN (PI-GNN) (Schuetz et al. Nature Machine Intel. 22)

Physics-Inspired Graph Neural Network



PI-GNN approach for solving QUBO. Schuetz et al. Machine Intelligence 2022

Expensive Inference time to find node embedding!

Method

Our Approach: RELIC - Reinforcement Learning for Ising Compression

- Neural Combinatorial Solver for (general) Ising Model
- Policy gradient (REINFORCE) to learn optimal edge contraction
- For an n-size Ising, a sequence of $n-1$ edge contractions induces a solution
- Policy trained via REINFORCE with *inexpensive* **final rewards**
    - **unsupervised learning**
- Generalizes to unseen graph topologies of various sizes

Compression Actions

Merge (aligned spins at ground states): contract (u, v), rewire neighbors of v:

$$\begin{aligned} J_{uw} &\leftarrow J_{uw} + J_{vw}, \quad \forall w \in \mathcal{N}(v) \setminus \{u\} \\ \text{offset} &\leftarrow \text{offset} + J_{uv} \end{aligned}$$

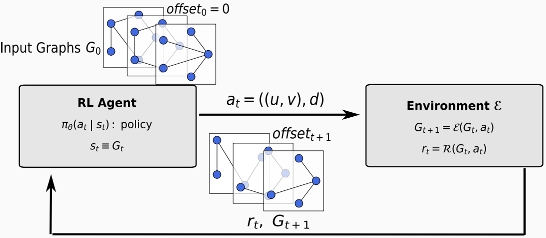
Flip-Merge (anti-aligned spins at ground states): negate v's couplings, then merge:

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

then apply as above.

- Preserve energy (up to an offset).

RL Formulation



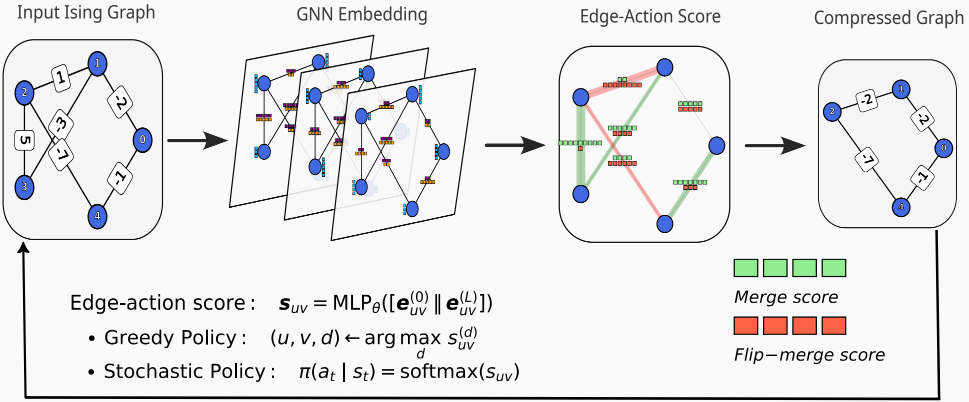
State s_t: current graph $G_t = (V_t, E_t)$
Actions $a_t = ((u, v), d)$: choose edge (u, v) and action $d \in \{\textit{merge}, \textit{flip-merge}\}$.

Transition: apply contraction, update G_{t+1} and $\texttt{offset}_{t+1}$.
Horizon: $T = |V_0| - 1$ for full compression.
Reward:

$$
\begin{aligned}
r_T &= -\left(\texttt{offset}_T + E_{\text{final}} - b\right) \\
&= -\left(\texttt{offset}_T - b\right),
\end{aligned}
$$

with baseline b for variance reduction and, $E_{\text{final}} = 0$ as fully-compressed into a node.

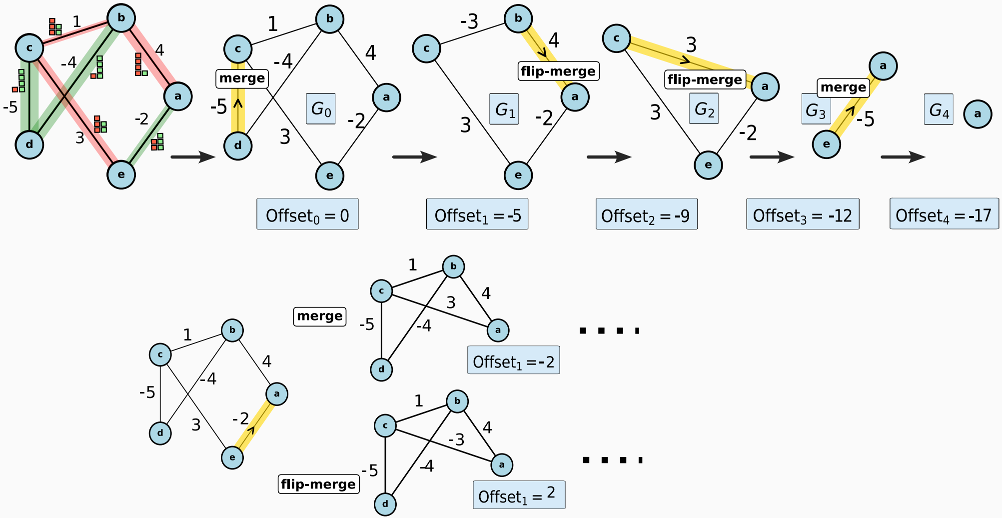
Policy Architecture



Edge-action score : $\boldsymbol{s}_{uv} = \mathrm{MLP}_\theta([\boldsymbol{e}_{uv}^{(0)} \,\|\, \boldsymbol{e}_{uv}^{(L)}])$

- Greedy Policy : $(u, v, d) \leftarrow \arg\max_d s_{uv}^{(d)}$
- Stochastic Policy : $\pi(a_t \mid s_t) = \mathrm{softmax}(s_{uv})$

Policy Architecture



Training Details

Objective (policy gradient):

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}\Big[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t)\ (r_T - b_t)\Big].$$

Baselines: greedy rollout; stochastic average of multiple rollouts.

Exploration: ε-greedy schedule.

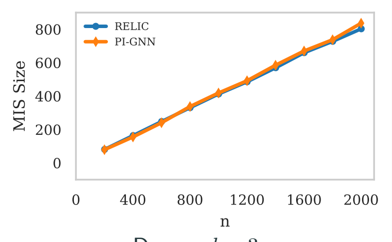
Stabilization: reward normalization, gradient clipping, mini-batch updates.

Experiments

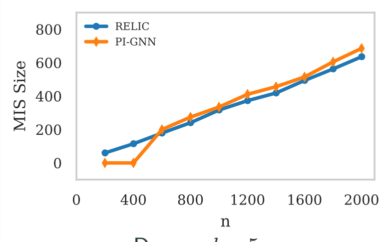
Experimental Setup

Training data	5,000 graphs, $n = 25$ (ER, BA, WS), $J_{ij} \sim \mathcal{U}[-5, 5]$, $h_i = 0$
Policy	3 MP layers, edge-action readout
Tasks	Ising generated from Maximmum Independent Set (MIS) [**?**]
Metrics	MIS size (higher is better), runtime scaling
Hardware	NVIDIA RTX 3060
Method	RELIC and PI-GNN

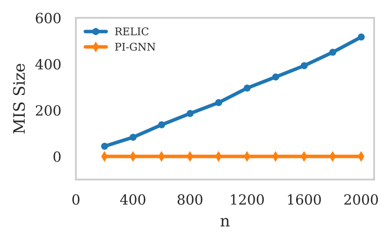
Results: MIS Quality



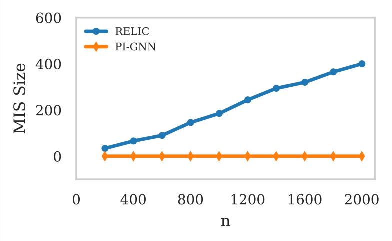
Degree $d = 3$



Degree $d = 5$

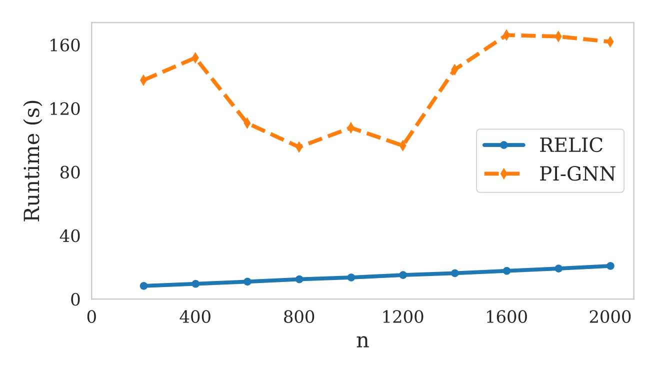


Degree $d = 7$



Degree $d = 9$

Results: Runtime Scaling



RELIC exhibits **linear** scaling and large speedups vs. PI-GNN.

Summary

- Contribution:
    - Neural Combinatorial Solver to solve (general) Ising model
    - Generalize well to Ising instances of large scale
- Limitation and future work
    - Benchmark on Ising from various combinatorial optimization problems
    - Include comparisons with more classical solvers

THANKS!