Maximum Independent Set: Self-Training through Dynamic Programming

Lorenzo Brusca*, Lars C.P.M. Quaedvlieg*, Stratis Skoulakis*, Grigorios G. Chrysos, Volkan Cevher

Laboratory for Information and Inference Systems (LIONS)
École Polytechnique Fédérale de Lausanne (EPFL)

Thirty-seventh Conference on Neural Information Processing Systems 2023

Recap - Tackling MIS with ML & DP

- Maximum Independent Set (MIS) is an NP-hard problem in combinatorial optimization.
- Machine Learning (ML) models have shown promise in addressing NP-hard problems [1, 2, 3].
- Dynamic Programming (DP) offers an algorithmic framework by combining solutions of sub-instances.
- We explore the synergy between DP and ML to develop a learning-based algorithm for MIS.
- Our approach involves employing a Graph Neural Network (GNN) to navigate through the DP tree.

Algorithmic structure

Divide: Pick random node $v \quad V$ from G, create $G/N(v), G/\{v\}$ based on the theoretical assumption:

$$G_0 = G/N(v),\ G_1 = G/\{v\} \quad \rightarrow \quad |\mathrm{MIS}(G)| = \max(|\mathrm{MIS}(G_0)|, |\mathrm{MIS}(G_1)|). \tag{1}$$

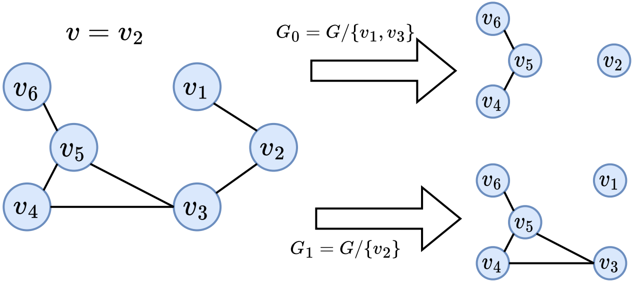


Figure: Generation of graphs G_0, G_1 given that node $v = v_2$ has been randomly picked.

Branch: GNN model (denoted as *comparator* (CMP_θ)) estimates the higher MIS:

$$\mathrm{CMP}_\theta(G_0, G_1) \quad \mathbb{I}\left[|\mathrm{MIS}(G_0)| < |\mathrm{MIS}(G_1)|\right]. \tag{2}$$

Self-training pipeline

Assumption

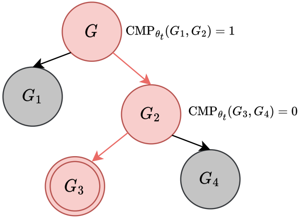
A ***consistent*** *comparator corresponds to an* ***optimal*** *algorithm. Thus, the goal of training is to find the set of parameters* $\theta^\star \quad \Theta$ *such that for all* $G_0, G_1 \quad$ *G:*

$$\mathrm{CMP}_{\theta^\star}(G_0, G_1) = 0 \text{ if and only if } \mathsf{E}\left[\left|A^{\mathrm{CMP}_{\theta^\star}}(G_0)\right|\right] \geq \mathsf{E}\left[\left|A^{\mathrm{CMP}_{\theta^\star}}(G_1)\right|\right].$$

.

Datasamples generation process: **recursion tree**, **roll-out**, and **pairing**:

1. Recursion tree: Run $\mathcal{A}^{\mathrm{CMP}_{\theta_t}}(G)$



2. Roll-out: Estimate the MIS

$$\begin{aligned}
EV_{G_1} &= \max(|\mathcal{A}^{\mathrm{CMP}_{\theta_t}}(G_1)|^{(1)}, \ldots, |\mathcal{A}^{\mathrm{CMP}_{\theta_t}}(G_1)|^{(K)}),\\
EV_{G_2} &= \max(|\mathcal{A}^{\mathrm{CMP}_{\theta_t}}(G_2)|^{(1)}, \ldots, |\mathcal{A}^{\mathrm{CMP}_{\theta_t}}(G_2)|^{(K)}),\\
EV_{G_3} &= \max(|\mathcal{A}^{\mathrm{CMP}_{\theta_t}}(G_3)|^{(1)}, \ldots, |\mathcal{A}^{\mathrm{CMP}_{\theta_t}}(G_3)|^{(K)}),\\
EV_{G_4} &= \max(|\mathcal{A}^{\mathrm{CMP}_{\theta_t}}(G_4)|^{(1)}, \ldots, |\mathcal{A}^{\mathrm{CMP}_{\theta_t}}(G_4)|^{(K)}).
\end{aligned}$$

3. Pairing: Pair every graph

$$\begin{gathered}
\langle (G_1, G_2), \mathbb{I}[EV_{G_1} < EV_{G_2}] \rangle,\\
\langle (G_1, G_3), \mathbb{I}[EV_{G_1} < EV_{G_3}] \rangle,\\
\vdots\\
\langle (G_3, G_4), \mathbb{I}[EV_{G_3} < EV_{G_4}] \rangle.
\end{gathered}$$

Figure: Generation of datasamples with recursion tree, roll-out and pairing steps

Main results

Our Model Performance:

- Outperforms all prior Deep-Learning-Based (DLB) methods across datasets.

Table: Test set results. A higher number corresponds to a better performance. The best results are in bold.

Method (↓) Dataset (→)	RB	COLLAB	TWITTER	SPECIAL
Our model	**0.836**	**0.990**	**0.977**	**0.996**
Best DLB method	0.813	0.978	0.972	0.946

- Generalizes well in out-of-distribution structures, indicating robust pattern extraction.

Open direction

How can the interplay between Dynamic Programming and self-training techniques pave the way for new deep-learning-oriented approaches for demanding Combinatorial Optimization problems?

Acknowledgements

Thank you for joining the presentation.
Looking forward to seeing you in our poster.
Great Hall & Hall B1+B2 #634, Thu 14 Dec 10:45 a.m. — 12:45 p.m. CST.

Scan the QR code to download the paper!
GitHub: `https://github.com/LIONS-EPFL/dynamic-MIS`.

References I

[1] Nikolaos Karalias and Andreas Loukas.
Erdos goes neural: an unsupervised learning framework for combinatorial optimization on graphs.
Advances in Neural Information Processing Systems, 33:6659–6672, 2020.
(Cited on page 2.)

[2] Jan Toenshoff, Martin Ritzert, Hinrikus Wolf, and Martin Grohe.
Run-csp: unsupervised learning of message passing networks for binary constraint satisfaction problems.
CoRR, abs/1909.08387, 2019.
(Cited on page 2.)

[3] Haoyu Wang and Pan Li.
Unsupervised learning for combinatorial optimization needs meta-learning.
arXiv preprint arXiv:2301.03116, 2023.
(Cited on page 2.)