One Model, Any CSP: Graph Neural Networks as Fast Global Search Heuristics for Constraint Satisfaction

Jan Tönshoff, Jakob Lindner, Berke Kisin, Martin Grohe

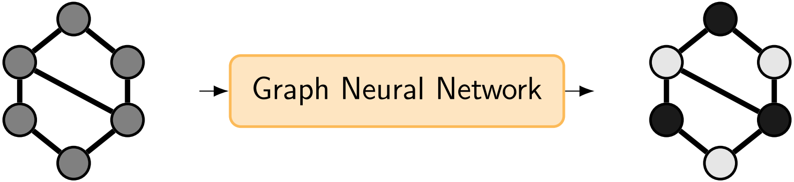
RWTH Aachen

toenshoff@informatik.rwth-aachen.de

October 28, 2022

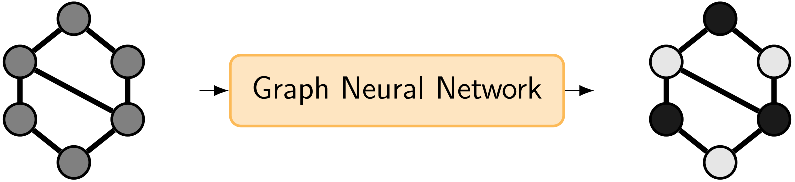
Neural Combinatorial Optimization

Learn heuristics for combinatorial optimization with Graph Neural Networks:



Neural Combinatorial Optimization

Learn heuristics for combinatorial optimization with Graph Neural Networks:

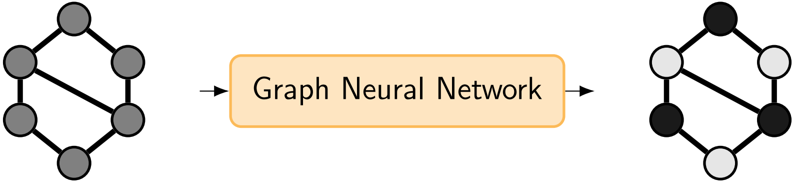


Pros

- Learn novel algorithms from scratch.

Neural Combinatorial Optimization

Learn heuristics for combinatorial optimization with Graph Neural Networks:

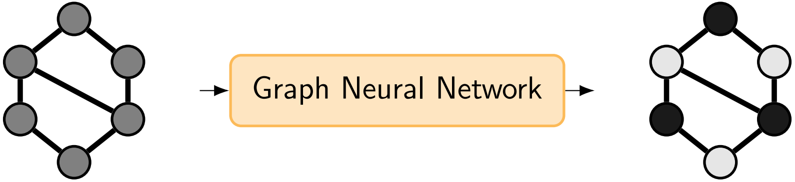


Pros

- Learn novel algorithms from scratch.
- Data-driven fine-tuning for specific input distributions.

Neural Combinatorial Optimization

Learn heuristics for combinatorial optimization with Graph Neural Networks:



Pros

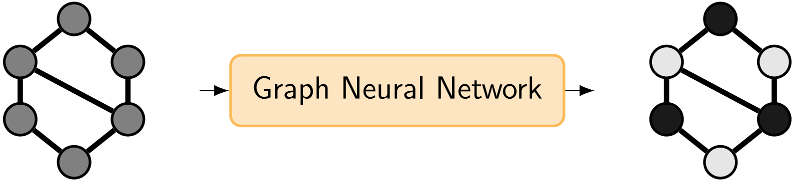
- Learn novel algorithms from scratch.
- Data-driven fine-tuning for specific input distributions.

Cons

- Computationally expensive.

Neural Combinatorial Optimization

Learn heuristics for combinatorial optimization with Graph Neural Networks:



Pros

- Learn novel algorithms from scratch.
- Data-driven fine-tuning for specific input distributions.

Cons

- Computationally expensive.
- Problem specific graph reductions, architectures and training procedures.

Constraint Satisfaction Problems

Constraint Satisfaction Problems

CSP-Instance: $\mathcal{I} = (\mathcal{X}, \mathcal{C}, \mathcal{D})$

Constraint Satisfaction Problems

CSP-Instance: $\mathcal{I} = (\mathcal{X}, \mathcal{C}, \mathcal{D})$

- Variables $\mathcal{X} = \{X_1, \ldots, X_n\}$ with finite domains $\mathcal{D} = \{\mathcal{D}(X_1), \ldots, \mathcal{D}(X_n)\}$

Constraint Satisfaction Problems

CSP-Instance: $\mathcal{I} = (\mathcal{X}, \mathcal{C}, \mathcal{D})$

- Variables $\mathcal{X} = \{X_1, \ldots, X_n\}$ with finite domains $\mathcal{D} = \{\mathcal{D}(X_1), \ldots, \mathcal{D}(X_n)\}$
- Constraints $\mathcal{C} = \{C_1, \ldots, C_m\}$ of the form $C = (s^C, R^C)$:

Constraint Satisfaction Problems

CSP-Instance: $\mathcal{I} = (\mathcal{X}, \mathcal{C}, \mathcal{D})$

- Variables $\mathcal{X} = \{X_1, \ldots, X_n\}$ with finite domains $\mathcal{D} = \{\mathcal{D}(X_1), \ldots, \mathcal{D}(X_n)\}$
- Constraints $\mathcal{C} = \{C_1, \ldots, C_m\}$ of the form $C = (s^C, R^C)$:

$$s^C = (X_1^C, \ldots, X_\ell^C)$$

$$R^C \subseteq \mathcal{D}(X_1^C) \times \cdots \times \mathcal{D}(X_\ell^C)$$

Constraint Satisfaction Problems

CSP-Instance: $\mathcal{I} = (\mathcal{X}, \mathcal{C}, \mathcal{D})$

- Variables $\mathcal{X} = \{X_1, \dots, X_n\}$ with finite domains $\mathcal{D} = \{\mathcal{D}(X_1), \dots, \mathcal{D}(X_n)\}$
- Constraints $\mathcal{C} = \{C_1, \dots, C_m\}$ of the form $C = (s^C, R^C)$:

$$s^C = (X_1^C, \dots, X_\ell^C)$$

$$R^C \subseteq \mathcal{D}(X_1^C) \times \cdots \times \mathcal{D}(X_\ell^C)$$

Variable assignment α: $\alpha(X) \in \mathcal{D}(X)$

Constraint Satisfaction Problems

CSP-Instance: $\mathcal{I} = (\mathcal{X}, \mathcal{C}, \mathcal{D})$

- Variables $\mathcal{X} = \{X_1, \dots, X_n\}$ with finite domains $\mathcal{D} = \{\mathcal{D}(X_1), \dots, \mathcal{D}(X_n)\}$
- Constraints $\mathcal{C} = \{C_1, \dots, C_m\}$ of the form $C = (s^C, R^C)$:

$$s^C = (X_1^C, \dots, X_\ell^C)$$

$$R^C \subseteq \mathcal{D}(X_1^C) \times \dots \times \mathcal{D}(X_\ell^C)$$

Variable assignment α: $\alpha(X) \in \mathcal{D}(X)$

$$\alpha \models C \iff (\alpha(X_1^C), \dots, \alpha(X_\ell^C)) \in R^C$$

Constraint Satisfaction Problems

Quality of assignment α for instance $\mathcal{I} = (\mathcal{X}, \mathcal{C}, \mathcal{D})$:

$$Q_{\mathcal{I}}(\alpha) = \frac{|\{C \in \mathcal{C} \; : \; \alpha \models C\}|}{|\mathcal{C}|}$$

Constraint Satisfaction Problems

Quality of assignment α for instance $\mathcal{I} = (\mathcal{X}, \mathcal{C}, \mathcal{D})$:

$$Q_{\mathcal{I}}(\alpha) = \frac{|\{C \in \mathcal{C} \ : \ \alpha \models C\}|}{|\mathcal{C}|}$$

Decision problem for $\mathcal{I}$:

$$\exists \alpha : Q_{\mathcal{I}}(\alpha) = 1?$$

Constraint Satisfaction Problems

Quality of assignment α for instance $\mathcal{I} = (\mathcal{X}, \mathcal{C}, \mathcal{D})$:

$$Q_{\mathcal{I}}(\alpha) = \frac{|\{C \in \mathcal{C} \ : \ \alpha \models C\}|}{|\mathcal{C}|}$$

Decision problem for $\mathcal{I}$:

$$\exists \alpha : Q_{\mathcal{I}}(\alpha) = 1?$$

Maximization problem for $\mathcal{I}$:

$$\alpha^* = \operatorname*{argmax}_{\alpha} Q_{\mathcal{I}}(\alpha)$$

Constraint Satisfaction Problems

Boolean SAT formula $f = (X_1 \vee X_2) \wedge (\neg X_1 \vee X_3 \vee X_2)$.

Constraint Satisfaction Problems

Boolean SAT formula $f = (X_1 \vee X_2) \wedge (\neg X_1 \vee X_3 \vee X_2)$.

Equivalent CSP $\mathcal{I} = (\mathcal{X}, \mathcal{C}, \mathcal{D})$:

$$\mathcal{X} = \{X_1, X_2, X_3\}, \quad \mathcal{D}(X_i) = \{0, 1\}$$

Constraint Satisfaction Problems

Boolean SAT formula $f = (X_1 \vee X_2) \wedge (\neg X_1 \vee X_3 \vee X_2)$.

Equivalent CSP $\mathcal{I} = (\mathcal{X}, \mathcal{C}, \mathcal{D})$:

$$\mathcal{X} = \{X_1, X_2, X_3\}, \quad \mathcal{D}(X_i) = \{0, 1\}$$

$$C_1 = ((X_1, X_2), \{0, 1\}^2 \setminus (0, 0))$$

Constraint Satisfaction Problems

Boolean SAT formula $f = (X_1 \vee X_2) \wedge (\neg X_1 \vee X_3 \vee X_2)$.

Equivalent CSP $\mathcal{I} = (\mathcal{X}, \mathcal{C}, \mathcal{D})$:

$$\mathcal{X} = \{X_1, X_2, X_3\}, \quad \mathcal{D}(X_i) = \{0, 1\}$$

$$C_1 = ((X_1, X_2), \{0, 1\}^2 \setminus (0, 0))$$

$$C_2 = ((X_1, X_3, X_2), \{0, 1\}^3 \setminus (1, 0, 0))$$

Constraint Satisfaction Problems

Graph Colouring instance (G, k) with $G = (V, E)$ and k colours.

Constraint Satisfaction Problems

Graph Colouring instance (G, k) with $G = (V, E)$ and k colours.

Equivalent CSP $\mathcal{I} = (\mathcal{X}, \mathcal{C}, \mathcal{D})$:

$$\mathcal{X} = V, \quad \mathcal{D}(X) = \{1, \dots, k\}$$

Constraint Satisfaction Problems

Graph Colouring instance (G, k) with $G = (V, E)$ and k colours.

Equivalent CSP $\mathcal{I} = (\mathcal{X}, \mathcal{C}, \mathcal{D})$:

$$\mathcal{X} = V, \quad \mathcal{D}(X) = \{1, \dots, k\}$$

$$\mathcal{C} = \left\{ ((u, v), R_{\neq}) \; : \; \forall uv \in E \right\}$$

RLSAT

Learning local search for SAT with GNNs (Yolcu and Póczos, 2019):

RLSAT

Learning local search for SAT with GNNs (Yolcu and Póczos, 2019):

$$(X_1 \vee X_2) \wedge (\neg X_1 \vee X_3 \vee X_2)$$

$$(X_1, X_2, X_3) \mapsto (1, 0, 0)$$

Instance + Assignment

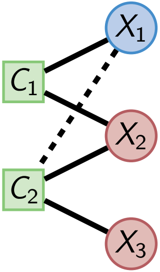
RLSAT

Learning local search for SAT with GNNs (Yolcu and Póczos, 2019):

$$(X_1 \lor X_2) \land (\neg X_1 \lor X_3 \lor X_2)$$

$$(X_1, X_2, X_3) \mapsto (1, 0, 0)$$

$\longrightarrow$



Instance + Assignment

Graph

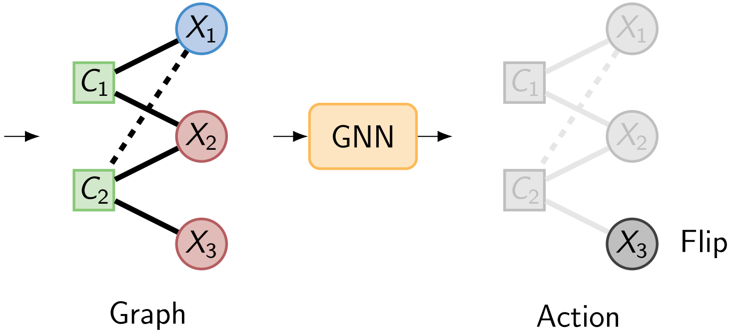
RLSAT

Learning local search for SAT with GNNs (Yolcu and Póczos, 2019):

$$(X_1 \vee X_2) \wedge (\neg X_1 \vee X_3 \vee X_2)$$

$$(X_1, X_2, X_3) \mapsto (1, 0, 0)$$

Instance + Assignment



Related Work

Overview: (Cappart et al., 2021)

SAT:

- RLSAT (Yolcu and Póczos, 2019)
- PDP (Amizadeh et al., 2019)

MaxCut:

- S2V (Khalil et al., 2017)
- ECO-DQN (Barrett et al., 2020)
- ECORD (Barrett et al., 2022)

Binary CSPs:

- RUNCSP (Tönshoff et al., 2021)

ANYCSP

ANYCSP: **A**re **N**eural Networks great heuristics? **Y**es, for **CSP**s!

ANYCSP

ANYCSP: **A**re **N**eural Networks great heuristics? **Y**es, for **CSP**s!

Objectives:

- Design unified graph representation and GNN architecture for **all** CSPs.

ANYCSP

ANYCSP: **A**re **N**eural Networks great heuristics? **Y**es, for **CSP**s!

Objectives:

- Design unified graph representation and GNN architecture for **all** CSPs.
- No restrictions to domain size or relations.

ANYCSP

ANYCSP: **A**re **N**eural Networks great heuristics? **Y**es, for **CSP**s!

Objectives:

- Design unified graph representation and GNN architecture for **all** CSPs.
- No restrictions to domain size or relations.
- Trained unsupervised with reinforcement learning.

ANYCSP

ANYCSP: **A**re **N**eural Networks great heuristics? **Y**es, for **CSP**s!

Objectives:

- Design unified graph representation and GNN architecture for **all** CSPs.
- No restrictions to domain size or relations.
- Trained unsupervised with reinforcement learning.
- Utilizes a global search action space.

Constraint Value Graph

CSP Instance $\mathcal{I}$:

$\mathcal{X} = \{X, Y, Z\}$

$D_X = \{1, 2, 3\}$
$D_Y = \{1, 2\}$
$D_Z = \{1, 2\}$

$C_1 : X \leq Y$
$C_2 : Y \neq Z$

Constraint Value Graph

CSP Instance $\mathcal{I}$:

$\mathcal{X} = \{X, Y, Z\}$

$D_X = \{1, 2, 3\}$
$D_Y = \{1, 2\}$
$D_Z = \{1, 2\}$

$C_1 : X \leq Y$
$C_2 : Y \neq Z$

Assignment $\alpha = (2, 1, 2)$

Constraint Value Graph

CSP Instance $\mathcal{I}$:

$$G(\mathcal{I}, \alpha) = (V, E, L_{\mathcal{D}}, L_{\mathcal{C}})$$

$\mathcal{X} = \{X, Y, Z\}$

$D_X = \{1, 2, 3\}$
$D_Y = \{1, 2\}$
$D_Z = \{1, 2\}$

$C_1 : X \leq Y$
$C_2 : Y \neq Z$

Assignment $\alpha = (2, 1, 2)$

Constraint Value Graph

CSP Instance $\mathcal{I}$:

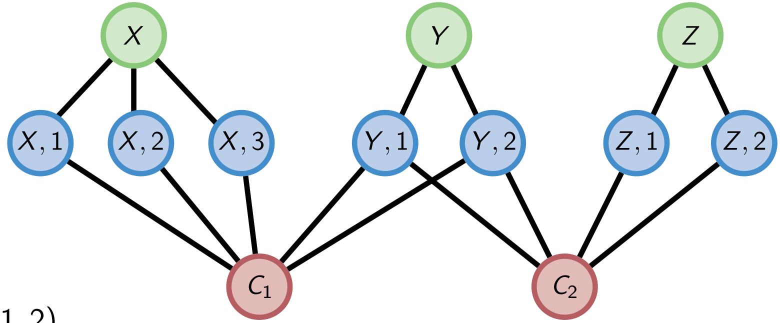
$\mathcal{X} = \{X, Y, Z\}$

$D_X = \{1, 2, 3\}$
$D_Y = \{1, 2\}$
$D_Z = \{1, 2\}$

$C_1 : X \leq Y$
$C_2 : Y \neq Z$

Assignment $\alpha = (2, 1, 2)$

$G(\mathcal{I}, \alpha) = (V, E, L_{\mathcal{D}}, L_{\mathcal{C}})$



Constraint Value Graph

CSP Instance $\mathcal{I}$:

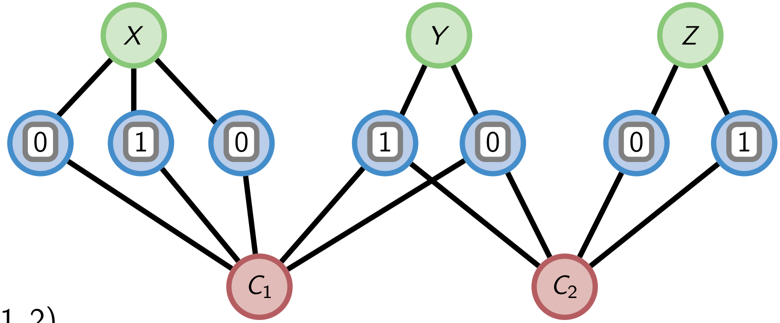
$\mathcal{X} = \{X, Y, Z\}$

$D_X = \{1, 2, 3\}$
$D_Y = \{1, 2\}$
$D_Z = \{1, 2\}$

$C_1 : X \leq Y$
$C_2 : Y \neq Z$

Assignment $\alpha = (2, 1, 2)$

$G(\mathcal{I}, \alpha) = (V, E, L_{\mathcal{D}}, L_{\mathcal{C}})$



$L_{\mathcal{D}}(v) = 1 \iff \alpha(X_v) = v$

Constraint Value Graph

CSP Instance $\mathcal{I}$:

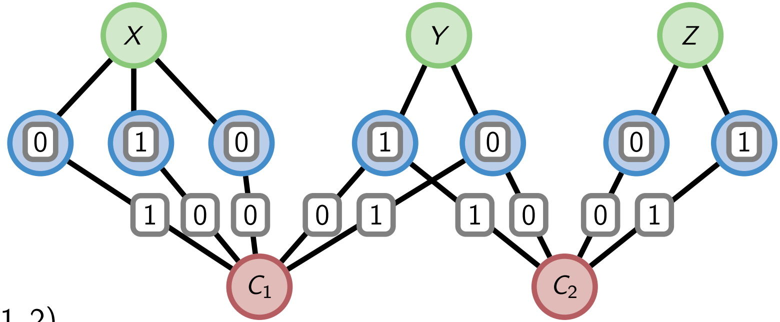
$\mathcal{X} = \{X, Y, Z\}$

$D_X = \{1, 2, 3\}$
$D_Y = \{1, 2\}$
$D_Z = \{1, 2\}$

$C_1 : X \leq Y$
$C_2 : Y \neq Z$

Assignment $\alpha = (2, 1, 2)$

$G(\mathcal{I}, \alpha) = (V, E, L_{\mathcal{D}}, L_{\mathcal{C}})$



$L_{\mathcal{C}}(v, C) = 1 \iff \alpha[X_v = v] \models C$

Constraint Value Graph

CSP Instance $\mathcal{I}$:

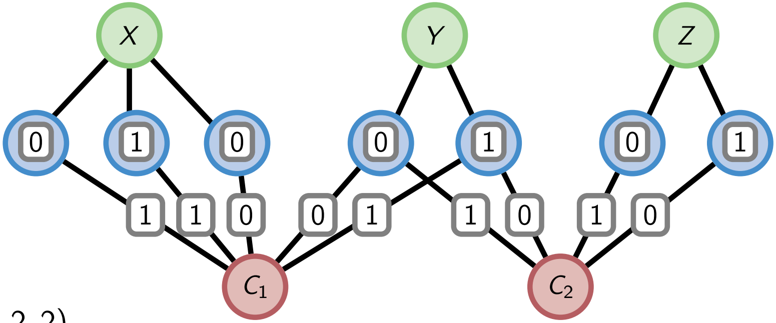
$\mathcal{X} = \{X, Y, Z\}$

$D_X = \{1, 2, 3\}$
$D_Y = \{1, 2\}$
$D_Z = \{1, 2\}$

$C_1 : X \leq Y$
$C_2 : Y \neq Z$

Assignment $\alpha^{(1)} = (2, 2, 2)$

$G(\mathcal{I}, \alpha^{(1)}) = (V, E, L_{\mathcal{D}}, L_{\mathcal{C}})$



$$L_{\mathcal{C}}(v, C) = 1 \iff \alpha[X_v = v] \models C$$

Constraint Value Graph

CSP Instance $\mathcal{I}$:

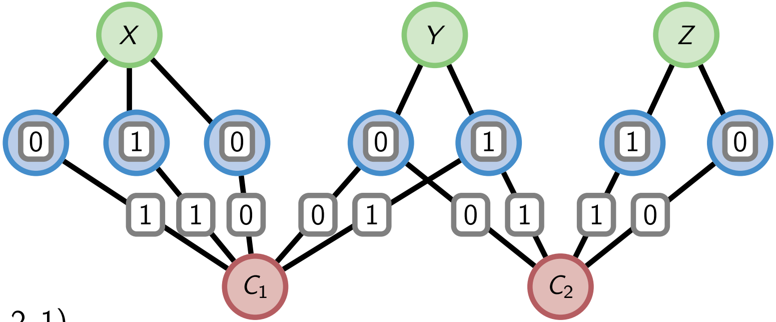
$\mathcal{X} = \{X, Y, Z\}$

$D_X = \{1, 2, 3\}$
$D_Y = \{1, 2\}$
$D_Z = \{1, 2\}$

$C_1 : X \leq Y$
$C_2 : Y \neq Z$

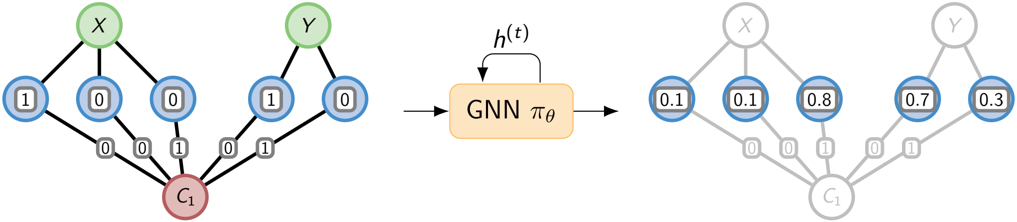
Assignment $\alpha^{(2)} = (2, 2, 1)$

$G(\mathcal{I}, \alpha^{(2)}) = (V, E, L_{\mathcal{D}}, L_{\mathcal{C}})$

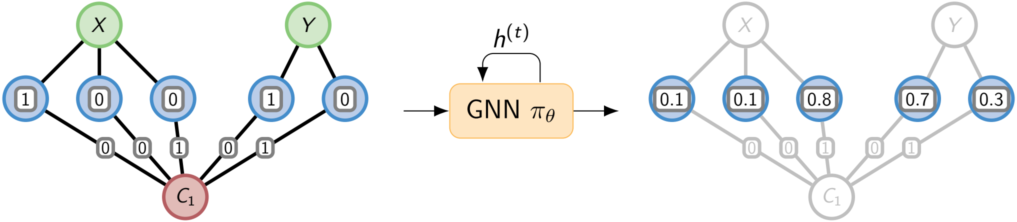


$L_{\mathcal{C}}(v, C) = 1 \iff \alpha[X_v = v] \models C$

Policy GNN

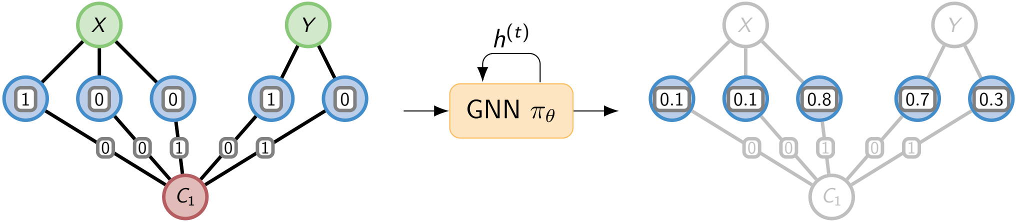


Policy GNN



Our GNN π_θ is a trainable stochastic global search policy:

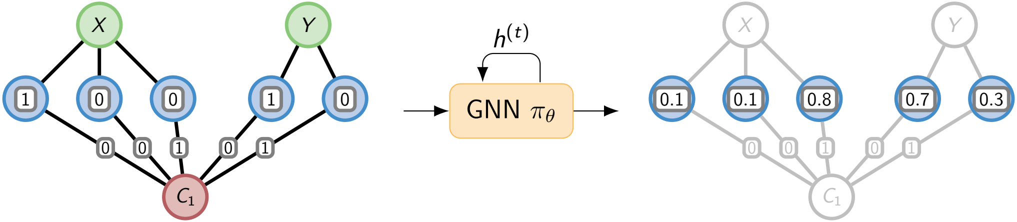
Policy GNN



Our GNN π_θ is a trainable stochastic global search policy:

- Input: $G(\mathcal{I}, \alpha^{(t)})$, recurrent states $h^{(t)} : \dot{\mathcal{D}} \rightarrow \mathbb{R}^d$.

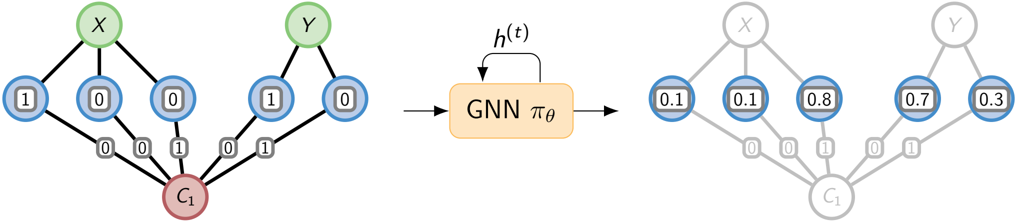
Policy GNN



Our GNN π_θ is a trainable stochastic global search policy:

- Input: $G(\mathcal{I}, \alpha^{(t)})$, recurrent states $h^{(t)} : \dot{\mathcal{D}} \rightarrow \mathbb{R}^d$.
- Output: Soft assignment $\varphi^{(t+1)} : \dot{\mathcal{D}} \rightarrow [0,1]$

Policy GNN



Our GNN π_θ is a trainable stochastic global search policy:

- Input: $G(\mathcal{I}, \alpha^{(t)})$, recurrent states $h^{(t)} : \dot{\mathcal{D}} \rightarrow \mathbb{R}^d$.
- Output: Soft assignment $\varphi^{(t+1)} : \dot{\mathcal{D}} \rightarrow [0, 1]$
- Next assignment: $\alpha^{(t+1)} \sim \varphi^{(t+1)}$

Stochastic Global Search

Input: CSP instance $\mathcal{I}$, number of iterations T.

Stochastic Global Search

Input: CSP instance $\mathcal{I}$, number of iterations T.

$$\alpha^{(0)}$$

$$h^{(0)}$$

Stochastic Global Search

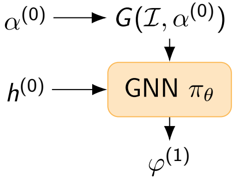
Input: CSP instance $\mathcal{I}$, number of iterations T.

$$\alpha^{(0)} \longrightarrow G(\mathcal{I}, \alpha^{(0)})$$

$$h^{(0)}$$

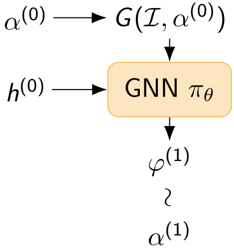
Stochastic Global Search

Input: CSP instance $\mathcal{I}$, number of iterations T.



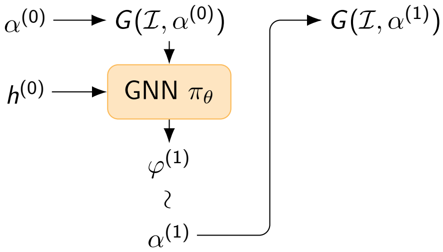
Stochastic Global Search

Input: CSP instance $\mathcal{I}$, number of iterations T.



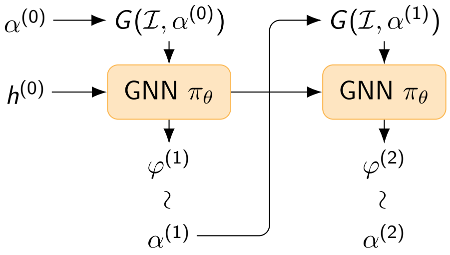
Stochastic Global Search

Input: CSP instance $\mathcal{I}$, number of iterations T.



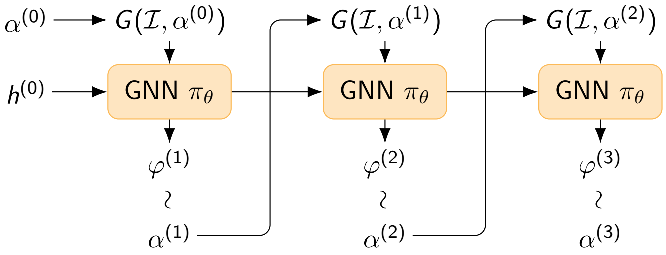
Stochastic Global Search

Input: CSP instance $\mathcal{I}$, number of iterations T.



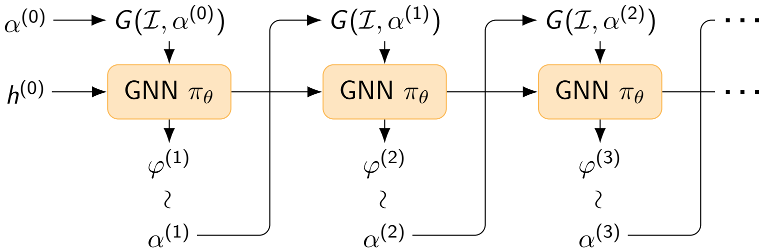
Stochastic Global Search

Input: CSP instance $\mathcal{I}$, number of iterations T.



Stochastic Global Search

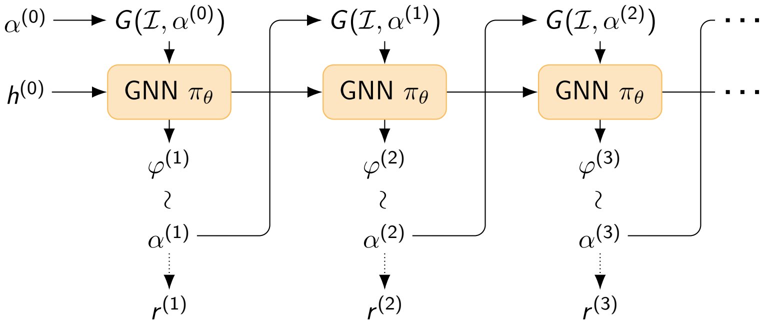
Input: CSP instance $\mathcal{I}$, number of iterations T.



Output: Sequence of assignments $\boldsymbol{\alpha} = \alpha^{(1)}, \dots, \alpha^{(T)}$.

Stochastic Global Search

Input: CSP instance $\mathcal{I}$, number of iterations T.



Output: Sequence of assignments $\boldsymbol{\alpha} = \alpha^{(1)}, \ldots, \alpha^{(T)}$.

Rewarding Improvements

Rewarding Improvements

Highest quality achieved before iteration t:

$$q^{(t)} = \max_{0 \leq t' < t} Q_{\mathcal{I}}(\alpha^{(t')}) \tag{1}$$

Rewarding Improvements

Highest quality achieved before iteration t:

$$q^{(t)} = \max_{0 \leq t' < t} Q_{\mathcal{I}}(\alpha^{(t')}) \tag{1}$$

Reward in iteration t:

$$r^{(t)} = \begin{cases} \\ \\ \end{cases} \tag{2}$$

Rewarding Improvements

Highest quality achieved before iteration t:

$$q^{(t)} = \max_{0 \leq t' < t} Q_{\mathcal{I}}(\alpha^{(t')}) \tag{1}$$

Reward in iteration t:

$$r^{(t)} = \begin{cases} 0 & \text{if } Q_{\mathcal{I}}(\alpha^{(t)}) \leq q^{(t)} \\ & \end{cases} \tag{2}$$

Rewarding Improvements

Highest quality achieved before iteration t:

$$q^{(t)} = \max_{0 \leq t' < t} Q_{\mathcal{I}}(\alpha^{(t')}) \tag{1}$$

Reward in iteration t:

$$r^{(t)} = \begin{cases} 0 & \text{if } Q_{\mathcal{I}}(\alpha^{(t)}) \leq q^{(t)} \\ Q_{\mathcal{I}}(\alpha^{(t)}) - q^{(t)} & \text{if } Q_{\mathcal{I}}(\alpha^{(t)}) > q^{(t)} \end{cases} \tag{2}$$

Rewarding Improvements

Highest quality achieved before iteration t:

$$q^{(t)} = \max_{0 \leq t' < t} Q_{\mathcal{I}}(\alpha^{(t')}) \tag{1}$$

Reward in iteration t:

$$r^{(t)} = \begin{cases} 0 & \text{if } Q_{\mathcal{I}}(\alpha^{(t)}) \leq q^{(t)} \\ Q_{\mathcal{I}}(\alpha^{(t)}) - q^{(t)} & \text{if } Q_{\mathcal{I}}(\alpha^{(t)}) > q^{(t)} \end{cases} \tag{2}$$

For the total reward after T iterations we observe:

$$\sum_{t=1}^{T} r^{(t)} = q^{(T+1)} - Q_{\mathcal{I}}(\alpha^{(0)}) \tag{3}$$

Training

Assume training distribution of CSP instances Ω.

Training

Assume training distribution of CSP instances Ω. Objective:

$$\theta^* = \arg\max_\theta \mathop{\mathbf{E}}_{\substack{\mathcal{I}\sim\Omega \\ \boldsymbol{\alpha}\sim\pi_\theta(\mathcal{I})}} \left[\sum_{t=1}^{T} \lambda^{t-1} r^{(t)} \right] \tag{4}$$

Training

Assume training distribution of CSP instances Ω. Objective:

$$\theta^* = \arg\max_{\theta} \mathop{\mathbf{E}}_{\substack{\mathcal{I}\sim\Omega \\ \boldsymbol{\alpha}\sim\pi_\theta(\mathcal{I})}} \left[\sum_{t=1}^{T} \lambda^{t-1} r^{(t)} \right] \tag{4}$$

We use REINFORCE (Williams, 1992) to learn network parameters θ with SGD:

$$\mathcal{L}(\mathcal{I}, \boldsymbol{\alpha}, \boldsymbol{\varphi}_\theta) = -\sum_{t=1}^{T} G_t \log \mathbf{P}(\alpha^{(t)} | \varphi_\theta^{(t)}), \qquad G_t = \sum_{k=t}^{T} \lambda^{k-t} r^{(k)} \tag{5}$$

Training

Note that our action space $A = \mathcal{D}(X_1) \times \cdots \times \mathcal{D}(X_n)$ is exponentially large!

Training

Note that our action space $A = \mathcal{D}(X_1) \times \cdots \times \mathcal{D}(X_n)$ is exponentially large!

However, we only use efficient parallelizable operations:

$$\alpha^{(t)} \sim \varphi^{(t)} \tag{6}$$

$$\log \mathbf{P}(\alpha^{(t)} | \varphi^{(t)}) = \sum_{X} \log \varphi^{(t)}(\alpha^{(t)}(X)) \tag{7}$$

Training

Note that our action space $A = \mathcal{D}(X_1) \times \cdots \times \mathcal{D}(X_n)$ is exponentially large!

However, we only use efficient parallelizable operations:

$$\alpha^{(t)} \sim \varphi^{(t)} \tag{6}$$

$$\log \mathbf{P}(\alpha^{(t)} | \varphi^{(t)}) = \sum_{X} \log \varphi^{(t)}(\alpha^{(t)}(X)) \tag{7}$$

Policy Gradient methods allow us to handle very large action spaces.

Experiments

Experiments

We train heuristics for the following CSPs:

- Model RB ($\Omega_{\mathbf{RB}}$)
- Graph Coloring ($\Omega_{\mathbf{COL}}$)
- MaxCut ($\Omega_{\mathbf{MCUT}}$)
- 3-SAT ($\Omega_{\mathbf{3SAT}}$)
- Max-k-SAT ($\Omega_{\mathbf{MSAT}}$)

Experiments

We train heuristics for the following CSPs:

- Model RB ($\Omega_{\mathbf{RB}}$)
- Graph Coloring ($\Omega_{\mathbf{COL}}$)
- MaxCut ($\Omega_{\mathbf{MCUT}}$)
- 3-SAT ($\Omega_{\mathbf{3SAT}}$)
- Max-k-SAT ($\Omega_{\mathbf{MSAT}}$)

Training Setup:

- Generated random instances with $|\mathcal{X}| \leq 100$.

Experiments

We train heuristics for the following CSPs:

- Model RB ($\Omega_{\mathbf{RB}}$)
- Graph Coloring ($\Omega_{\mathbf{COL}}$)
- MaxCut ($\Omega_{\mathbf{MCUT}}$)
- 3-SAT ($\Omega_{\mathbf{3SAT}}$)
- Max-k-SAT ($\Omega_{\mathbf{MSAT}}$)

Training Setup:

- Generated random instances with $|\mathcal{X}| \leq 100$.
- $T_{\text{train}} = 40$ search iterations.

Experiments

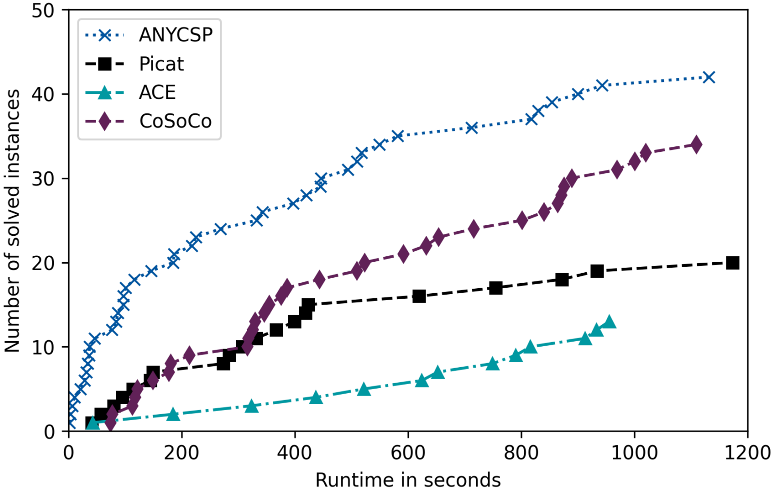
We train heuristics for the following CSPs:

- Model RB ($\Omega_{\mathbf{RB}}$)
- Graph Coloring ($\Omega_{\mathbf{COL}}$)
- MaxCut ($\Omega_{\mathbf{MCUT}}$)
- 3-SAT ($\Omega_{\mathbf{3SAT}}$)
- Max-k-SAT ($\Omega_{\mathbf{MSAT}}$)

Training Setup:

- Generated random instances with $|\mathcal{X}| \leq 100$.
- $T_{\text{train}} = 40$ search iterations.
- Train for 500K steps of SGD ($\sim$48h) and batch size 25.

Model RB Benchmarks



MaxCut

Instances: (Unweighted) GSet graphs

Metric: Mean absolute deviation from known optimum cut value.

Method	$\|V\|$=800	$\|V\|$=1K	$\|V\|$=2K	$|V|\geq$3K
Greedy	411.44	359.11	737.00	774.25
SDP	245.44	229.22	-	-
RUNCSP	185.89	156.56	357.33	401.00
ECO-DQN	65.11	54.67	157.00	428.25
ECORD	8.67	8.78	39.22	187.75
ANYCSP	**1.22**	**2.44**	**13.11**	**51.63**

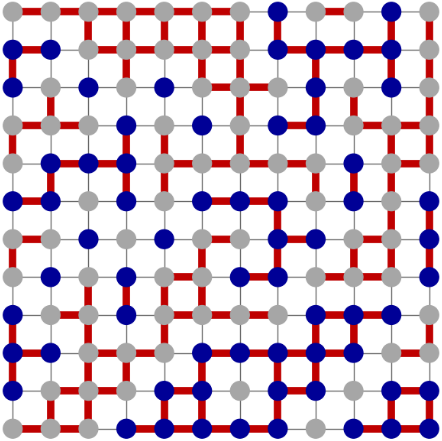
Graph Colouring

Instances: Structured k-Colouring instances ($4 \leq k \leq 73$, $|V| \leq 2000$, $|E| \leq 20000$)

Metric: Number of optimally coloured graphs.

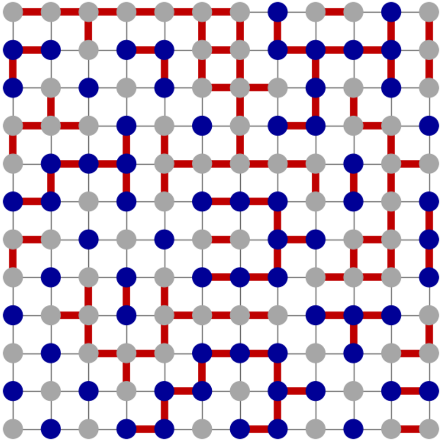
METHOD	$COL_{<10}$	$COL_{\geq 10}$
RUNCSP	33	-
COSOCO	49	33
PICAT	49	38
GREEDY	16	15
DSATUR	38	28
HYBRIDEA	**50**	**40**
ANYCSP	**50**	**40**

Graph Colouring



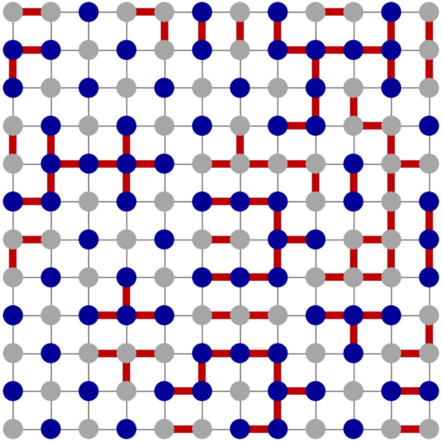
$t = 0$

Graph Colouring



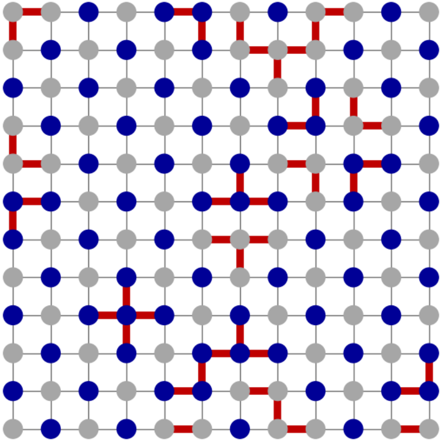
$t = 1$

Graph Colouring



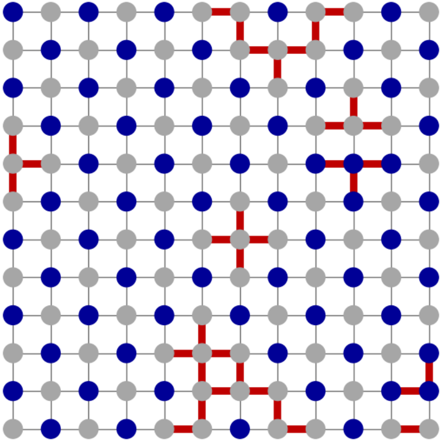
$t = 2$

Graph Colouring



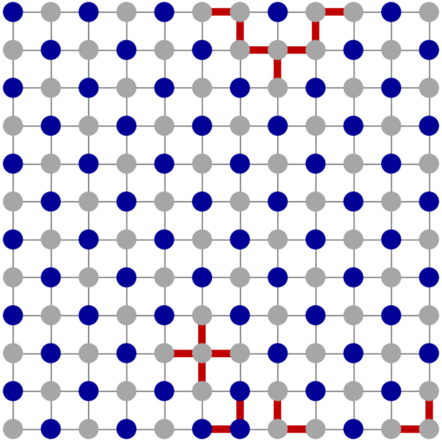
$t = 3$

Graph Colouring



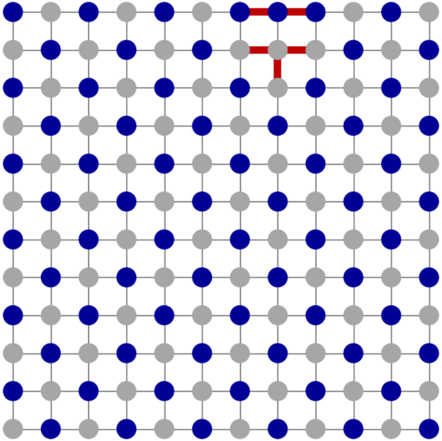
$t = 4$

Graph Colouring



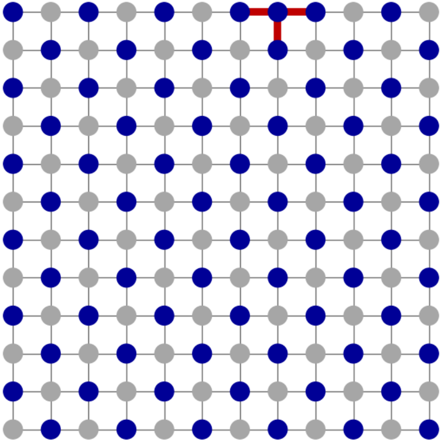
$t = 5$

Graph Colouring



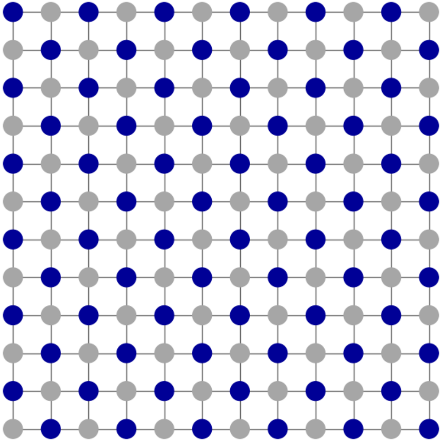
$t = 6$

Graph Colouring



$t = 7$

Graph Colouring



$t = 8$

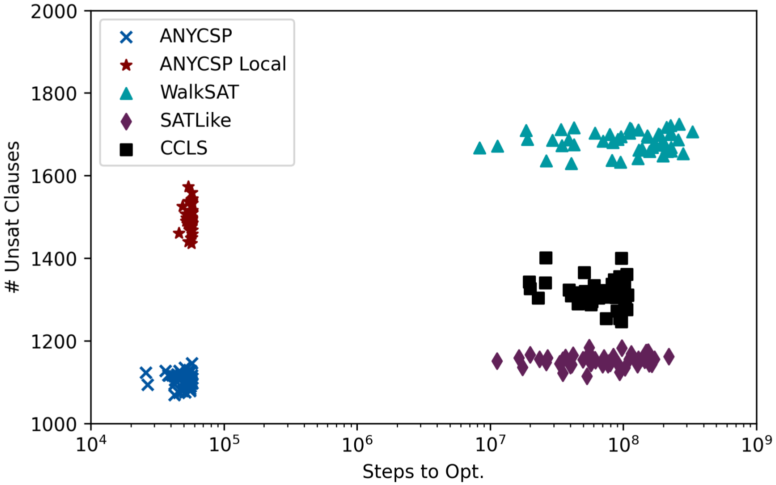
SAT

Instances: Random 3SAT Instances from SATLIB.

Metric: Number of satisfied instances.

Method	SL50	SL100	SL150	SL200	SL250
RLSAT	**100**	87	67	27	12
PDP	93	79	72	57	61
WalkSAT	**100**	**100**	97	93	87
ProbSAT	**100**	**100**	97	87	92
ANYCSP	**100**	**100**	**100**	**97**	**99**

Max-5-SAT



2000
1800
1600
1400
1200
1000
Unsat Clauses
10^4
10^5
10^6
10^7
10^8
10^9
Steps to Opt.
ANYCSP
ANYCSP Local
WalkSAT
SATLike
CCLS

Conclusion

ANYCSP:

- Constraint Value Graphs: A generic and compact representation for CSPs.
- REINFORCE applied to exponential action spaces.

Conclusion

ANYCSP:

- Constraint Value Graphs: A generic and compact representation for CSPs.
- REINFORCE applied to exponential action spaces.

Empirical Observations:

- CSP heuristics can be obtained purely through data-driven training.
- GNNs parameterize a powerful and versatile class of global search heuristic.

References I

Emre Yolcu and Barnabás Póczos. Learning local search heuristics for boolean satisfiability. *Advances in Neural Information Processing Systems*, 32, 2019.

Quentin Cappart, Didier Chételat, Elias B. Khalil, Andrea Lodi, Christopher Morris, and Petar Veličković. Combinatorial optimization and reasoning with graph neural networks. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4348–4355. International Joint Conferences on Artificial Intelligence Organization, 8 2021. doi: 10.24963/ijcai.2021/595. URL https://doi.org/10.24963/ijcai.2021/595. Survey Track.

Saeed Amizadeh, Sergiy Matusevych, and Markus Weimer. Pdp: A general neural framework for learning constraint satisfaction solvers. *arXiv preprint arXiv:1903.01969*, 2019.

References II

Elias Khalil, Hanjun Dai, Yuyu Zhang, Bistra Dilkina, and Le Song. Learning combinatorial optimization algorithms over graphs. *Advances in neural information processing systems*, 30, 2017.

Thomas Barrett, William Clements, Jakob Foerster, and Alex Lvovsky. Exploratory combinatorial optimization with reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3243–3250, 2020.

Thomas D Barrett, Christopher WF Parsonson, and Alexandre Laterre. Learning to solve combinatorial graph partitioning problems via efficient exploration. *arXiv preprint arXiv:2205.14105*, 2022.

Jan Tönshoff, Martin Ritzert, Hinrikus Wolf, and Martin Grohe. Graph neural networks for maximum constraint satisfaction. *Frontiers in Artificial Intelligence*, 3, 2021. ISSN 2624-8212. doi: 10.3389/frai.2020.580607. URL https://www.frontiersin.org/article/10.3389/frai.2020.580607.

References III

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

Max-k-SAT

Instances: CNF formulas with 10K variables and 75K-300K clauses.

Metric: Mean number of unsatisfied clauses.

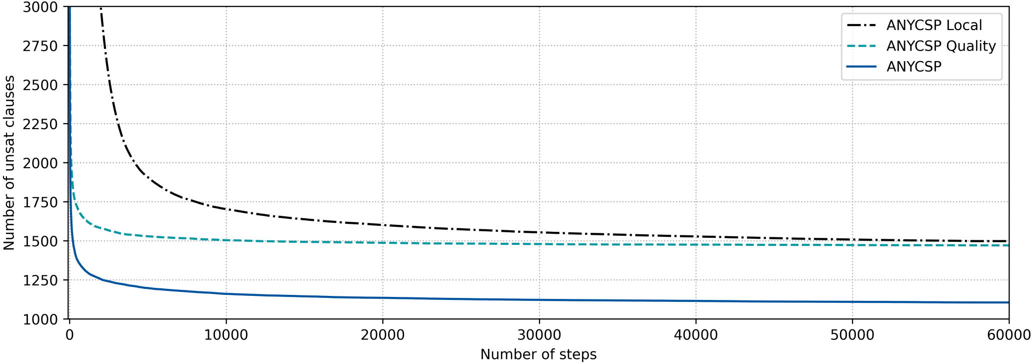
Method	3CNF	4CNF	5CNF
WalkSAT	2145.28	1556.68	1685.10
CCLS	1567.24	1323.14	1315.96
SATLike	1595.86	1188.56	1152.88
ANYCSP	**1537.46**	**1126.44**	**1103.14**

Cross-Comparison

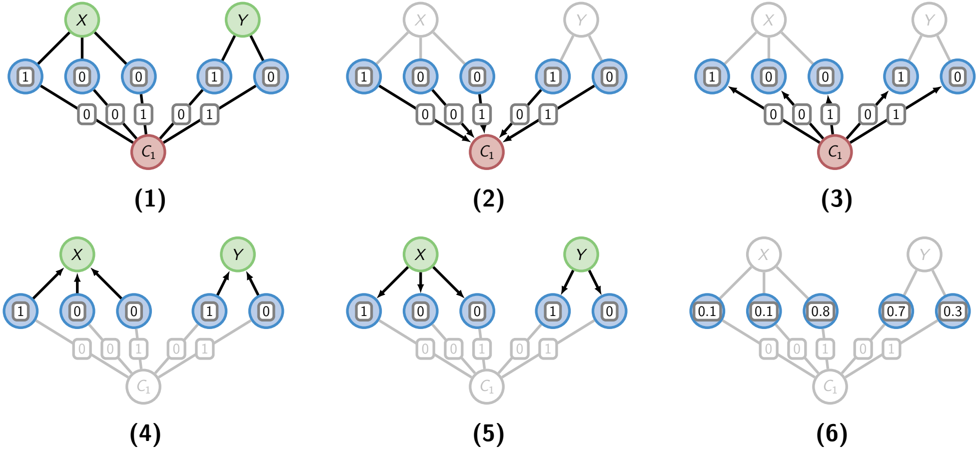
Training Distribution Ω vs Test CSPs:

Ω	RB50	$\text{COL}_{<10}$	Gset800	SL250	MAX-5-CNF
$\Omega_{\textbf{RB}}$	**42**	**50**	655.56	98	6192.18
$\Omega_{\textbf{COL}}$	15	**50**	868.22	96	5076.16
$\Omega_{\textbf{MCUT}}$	0	0	**1.22**	0	9048.64
$\Omega_{\textbf{3SAT}}$	0	19	1213.11	**99**	5001.72
$\Omega_{\textbf{MSAT}}$	0	15	1217.67	66	**1103.14**

Ablation


ANYCSP Local
ANYCSP Quality
ANYCSP
Number of unsat clauses
Number of steps
3000
2750
2500
2250
2000
1750
1500
1250
1000
0
10000
20000
30000
40000
50000
60000

π_θ: Message Passing Scheme



Markov Decision Process

The formal Markov Decision Process of ANYCSP:

- State at time t: $s^{(t)} = (G(\mathcal{I}, \alpha^{(t)}), q^{(t)})$, with $q^{(t)} = \max_{t'<t} Q_{\mathcal{I}}(\alpha^{(t')})$.
- Initial assignment $\alpha^{(0)}$ is drawn uniformly, $q^{(0)} = 0$.
- Action space $\mathcal{A}$: Set of all assignments of $\mathcal{I}$.
- Transition function: $(s^{(t)}, \alpha^{(t+1)}) \mapsto (G(\mathcal{I}, \alpha^{(t+1)}), \max\{q^{(t)}, Q_{\mathcal{I}}(\alpha^{(t)})\})$.
- Reward: $r^{(t)} = \max\{0, Q_{\mathcal{I}}(\alpha^{(t)}) - q^{(t)}\}$