Learning Domain-Independent Heuristics for Grounded and Lifted Planning

Dillon Z. Chen[1,2], Sylvie Thiébaux[1,2], Felipe Trevizan[2]

[1]LAAS-CNRS, Université de Toulouse, [2]Australian National University
{dillon.chen, sylvie.thiebaux}@laas.fr
felipe.trevizan@anu.edu.au

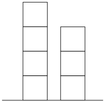
AAAI 2024

What are we doing: learning for planning

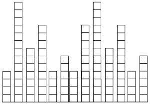
Learn policies/heuristics that generalise

- to problems of **larger size**
- domain-dependent learning; e.g.

train small Blocksworld

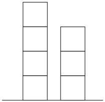


test large Blocksworld

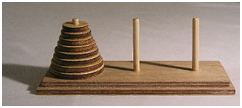


- to problems from **different domains**; e.g.
- domain-independent learning; e.g.

train Blocksworld



test Towers of Hanoi



What are we not doing:

Reinforcement Learning (RL)

- sample inefficient
- does not exploit model structure
- poor generalisation and scaling to larger problems

Large Language Models (LLMs)

- not reasoning on logic; memorise word semantics
- no correctness guarantees
- poor generalisation and scaling to larger problems

Prerequisites

AI Planning

- find a sequence of executable actions that achieve a goal
- requires long range reasoning over very large state space
- makes use of predicate logic

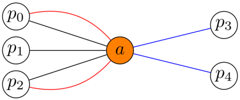
Graph Neural Networks (GNNs)

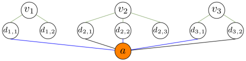
- message passing paradigm
- allow for arbitrary input graphs with fixed feature dimension
- we focus on Message Passing Neural Networks (MPNNs)

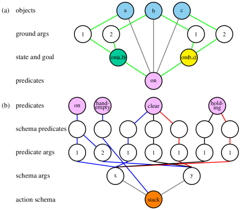
New Contributions

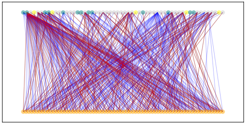
1. *representation*: domain-independent planning graphs
2. *theory*: what heuristics can we learn?
3. *implementation*: GOOSE planner
4. *experiments*: state-of-the-art domain-dependent and -independent learning results

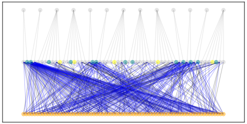
1. New domain-independent planning graphs

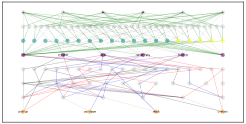








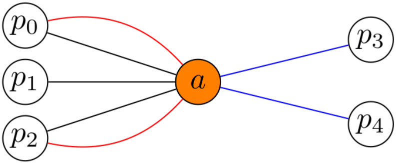


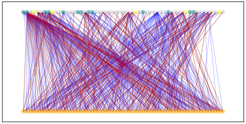


- graph representations of planning tasks → input into GNN

STRIPS Learning Graph (SLG)

STRIPS planning task: $\langle P, A, s_0, G \rangle$

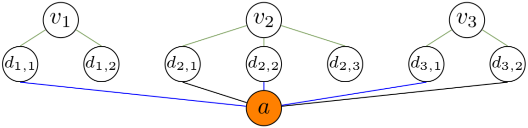


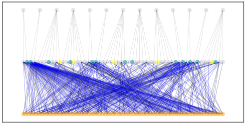


- nodes: propositions + actions
- features: node type + presence of proposition in s_0 or G
- edges: pre - add - del
- learning version of STRIPS PDG [Shleyfman et al., AAAI-15]

Finite domain representation Learning Graph (FLG)

FDR planning task: $\langle \mathcal{V}, A, s_0, G \rangle$

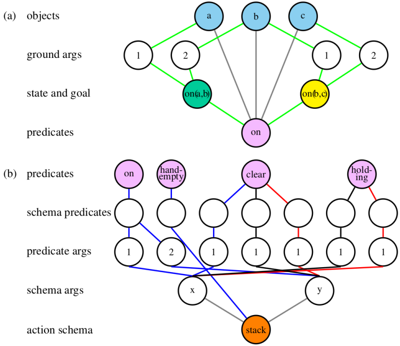


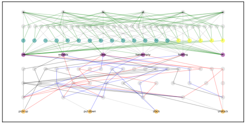


- nodes: variables + domain values + actions
- features: node type + value in s_0 and G
- edges: values, pre - effect
- learning version of FDR PDG [Pochter et al., AAAI-11]

Lifted Learning Graph (LLG)

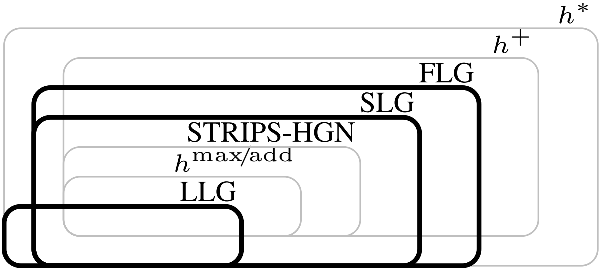
lifted planning task: $\langle \mathcal{P}, \mathcal{O}, \mathcal{A}, s_0, G \rangle$





- graphs encode action schemata instead of actions
- only propositions are those in s_0 and G
- node features and edges encode position of objects in the predicate arguments

2. Theoretical results: what heuristics can they learn?



- expressivity analysis of GNNs operating on planning graphs
- domain-*independent* heuristics we can(not) learn
- proof techniques applicable to other learning for planning architectures e.g. (LLM, RL)

2a. Positive results

Theorem

MPNNs operating on grounded graphs (SLG and FLG) are more expressive than STRIPS-HGN [Shen et al., ICAPS-20]

- Proof idea: STRIPS-HGN do not encode delete effects

Theorem

MPNNs operating on grounded graphs can learn h^{add} and h^{max}

- Proof idea: encode Value Iteration into MPNNs + approximation theorem
- practicality? not much

2b. Negative results

Theorem

MPNNs operating on lifted graphs (LLG) cannot learn h^{add}, h^{max}, h^{+} and h^{*}

- Proof idea: counterexample
- a pair of planning tasks with different heuristic values but appear the same to MPNNs operating on their LLG representation
- <u>thus, “scaling” your NN architecture is pointless</u>

Theorem

MPNNs operating on grounded graphs cannot learn h^{+} and h^{*} nor any approximation

- Proof idea: class of counterexamples

Not all hope is lost

- possible to learn h^* for subclasses of planning tasks [1]
- do not need perfect predictions
- can still perform well on GBFS with inaccurate heuristics

[1] Ståhlberg, S., Bonet, B., Geffner, H. (2022). Learning General Optimal Policies with Graph Neural Networks: Expressive Power, Transparency, and Limits. In *ICAPS*.

3. GOOSE architecture

1. states converted to graphs
   - one of SLG, FLG, LLG
2. graphs fed into a GNN with learned parameters
   - RGCN [Schlichtkrull et al., ESWC-18] for edge-labelled graphs
3. GPU batch evaluate *only*[1] successor states
   - backend search in Fast Downward implementation of GBFS

Code at `https://github.com/DillonZChen/goose`

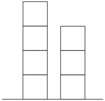
[1] Doing more is suboptimal and is made worse with lazy evaluation GBFS.

4. Experiments: Learning paradigms

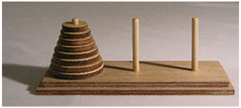
Domain-Independent Learning [Shen et al., ICAPS-20]

- **do not** train on evaluation domain
- learn to solve arbitrary planning problems; “zero shot learning”

train



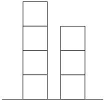
test



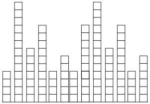
Domain-Dependent Learning

- train on **very small tasks from** the evaluation domain
- learn to solve specific planning problems

train



test



Baselines

- blind: breadth first search
- h^{FF}: GBFS with the h^{FF} heuristic
- HGN: STRIPS-HGN trained in domain-*dependent* fashion

4a. Domain-Independent Learning

- train on tasks *not from* evaluation domain
- training: {IPC benchmarks} \ {evaluation domains}
- testing: number of objects[2] from 15-100

	baselines			GOOSE		
	blind	h^{FF}	HGN	SLG	FLG	LLG
blocks (90)	-	**19**	-	9	8	6
ferry (90)	-	**90**	-	28	22	2
gripper (18)	1	**18**	5	5	3	9
n-puzzle (50)	-	**36**	-	6	3	-
sokoban (90)	74	**90**	10	45	40	15
spanner (90)	-	-	-	-	-	-
visitall (90)	-	6	25	16	**41**	-
visitsome (90)	3	26	33	**73**	65	15

hyperparameters: 8 GNN layers, mean aggr.

[2] except *n*-puzzle and Sokoban

4b. Domain-Dependent Learning

- train on tasks *from the same* evaluation domain
- training: number of objects[3] from 2-10
- testing: number of objects[3] from 15-100

	baselines			GOOSE		
	blind	h^{FF}	HGN	SLG	FLG	LLG
blocks (90)	-	19	-	-	6	**62**
ferry (90)	-	**90**	-	32	33	88
gripper (18)	1	**18**	5	9	6	**18**
n-puzzle (50)	-	**36**	-	10	10	-
sokoban (90)	74	**90**	10	31	29	34
spanner (90)	-	-	-	-	-	**60**
visitall (90)	-	6	25	46	**50**	44
visitsome (90)	3	26	33	**72**	39	65

hyperparameters: 8 GNN layers, mean aggr.

[3] except *n*-puzzle and Sokoban

4c. IPC 2023 Learning Track results

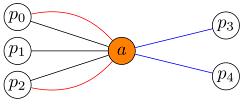
- domain-dependent learning
- planners:
  - h^{FF}: classical planner
  - GOOSE: deep learning
  - WL-GOOSE [2]: classical ML

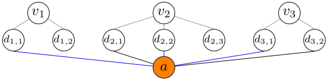
Domain	h^{FF}	GOOSE	WL-GOOSE
blocksworld	28	63.0	**77**
childsnack	26	23.2	**30**
ferry	68	70.0	**76**
floortile	**12**	0.0	2
miconic	**90**	88.6	**90**
rovers	34	25.6	**37**
satellite	**65**	31.0	57
sokoban	36	33.0	**38**
spanner	30	46.4	**<u>74</u>**
transport	**41**	32.4	32
sum coverage	430	413.2	**513**
sum IPC score	393.5	391.0	**471.2**

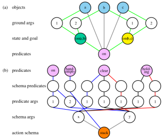
[2] Chen, D. Z., Trevizan, F., Thiébaux, S. (2024). Return to Tradition: Learning Reliable Heuristics with Classical Machine Learning. In *ICAPS*.

Theoretical and SOTA experimental results in learning heuristics for domain-dependent and -independent planning

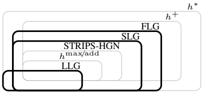
1. New graph representations of planning tasks for learning







2. Theoretical Results



3. GOOSE

	baselines			domain-dep.			domain-ind.		
	blind	h^{FF}	HGN	SLG	FLG	LLG	SLG	FLG	LLG
blocks (90)	-	19	-	-	6	**62**	9	8	6
ferry (90)	-	**90**	-	32	33	88	28	22	2
gripper (18)	1	**18**	5	9	6	**18**	5	3	9
n-puzzle (50)	-	**36**	-	10	10	-	6	3	-
sokoban (90)	74	**90**	10	31	29	34	45	40	15
spanner (90)	-	-	-	-	-	**60**	-	-	-
visitall (90)	-	6	25	46	**50**	44	16	41	-
visitsome (90)	3	26	33	72	39	65	**73**	65	15

4. State-of-the-art Results

Domain	h^{FF}	GOOSE	WL-GOOSE
blocksworld	28	63.0	**77**
childsnack	26	23.2	**30**
ferry	68	70.0	**76**
floortile	**12**	0.0	2
miconic	**90**	88.6	**90**
rovers	34	25.6	**37**
satellite	**65**	31.0	57
sokoban	36	33.0	**38**
spanner	30	46.4	**74**
transport	**41**	32.4	32
sum coverage	430	413.2	**513**
sum IPC score	393.5	391.0	**471.2**

Poster 639

Code at `https://github.com/DillonZChen/goose`

Thanks! Questions?