GOOSE: Learning Domain-Independent Heuristics

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

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

GenPlan 2023

Motivation: deep learning and planning?

Use DL to learn policies or heuristics that generalise

- to problems of larger size
- to problems from different domains

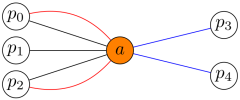
Reduce data and resources needed by DL

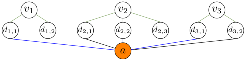
- exploiting planning representations (PDDL planning)
- why relearn what you already know?

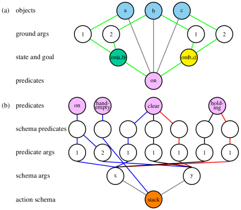
New Contributions

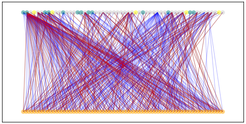
1. domain-independent grounded and lifted planning graphs
2. theoretical results: what heuristics can they learn?
3. implementation: GOOSE planner
4. domain-dependent and -independent learning experiments

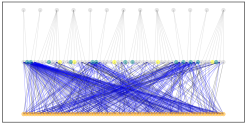
1. New planning graph representations

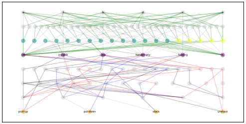






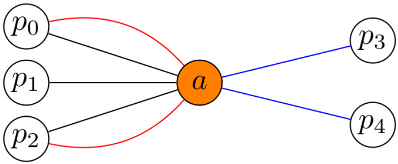


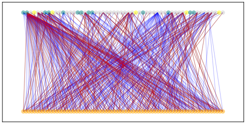




- graph representations of planning tasks = inputs into GNNs

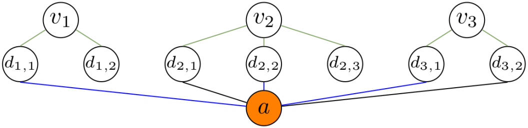
STRIPS Learning Graph (SLG)

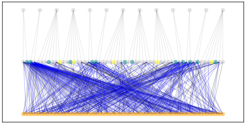




- nodes: propositions + actions
- features: node type + presence of proposition in s_0 or G
- edges: pre - add - del

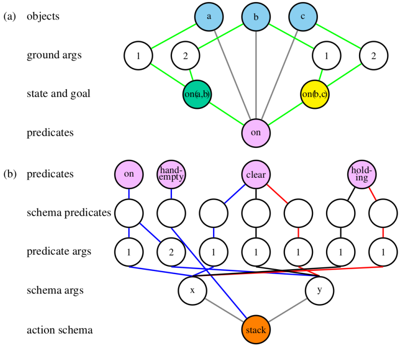
Finite domain representation Learning Graph (FLG)

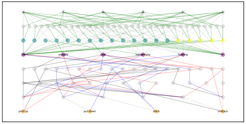




- nodes: variables + domain values + actions
- features: node type + value in s_0 and G
- edges: values, pre - effect

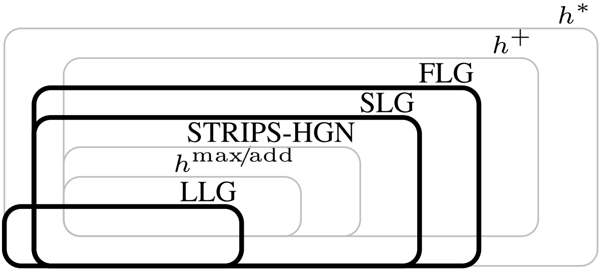
Lifted Learning Graph (LLG)





- 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?



- more expressive than STRIPS-HGN by considering full planning task information

2a. Grounded graphs can learn h^{add} and h^{max}

Theorem

Let $L, B \in \mathbb{N}$, $\mathcal{G} \in \{SLG, FLG\}$, $\varepsilon > 0$ and $h \in \{h^{add}, h^{max}\}$. Then there exists a set of parameters θ for an MPNN $\mathcal{F}_\theta$ such that for all planning tasks Π, if naive dynamic programming for computing h converges within L iterations for Π, and $h(s_0) \leq B$, then we have $|h(s_0) - \mathcal{F}_\theta(\mathcal{G}(\Pi))| < \varepsilon$.

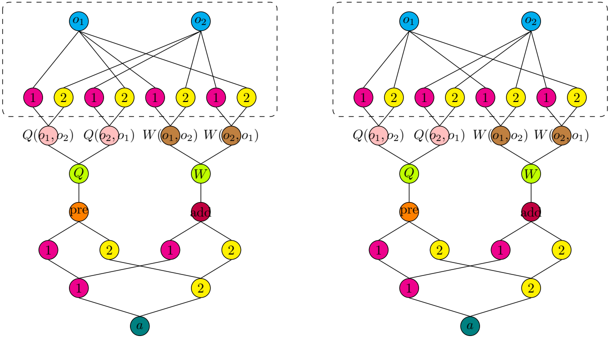
- Proof idea: encode Value Iteration into GNNs + approximation theorem
- practicality? not much

2b. Lifted graphs cannot learn h^{add}, $h^{\max}$, h^{+} and h^{*}

Theorem

Let $h \in \{h^{add}, h^{max}, h^{+}, h^{*}\}$. There exists a pair of planning tasks Π_1 and Π_2 with $h(\Pi_1) \neq h(\Pi_2)$ such that for any set of parameters Θ for an MPNN we have $\mathcal{F}_{\Theta}(LLG(\Pi_1)) = \mathcal{F}_{\Theta}(LLG(\Pi_2))$.

- Proof idea: counterexample



(a) LLG of P_1.

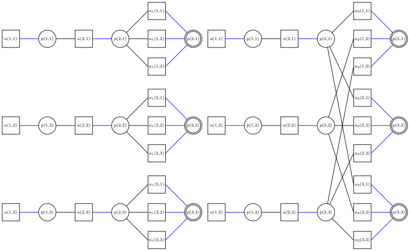
(b) LLG of P_2.

2c. Grounded graphs cannot learn h^+ and h^*, nor any approximation

Theorem

Let $h \in \{h^+, h^*\}$, $\mathcal{G} \in \{SLG, FLG, LLG\}$ and $c > 0$. There exists a pair of planning tasks Π_1 and Π_2 with $h(\Pi_1) \neq h(\Pi_2)$ such that for any set of parameters Θ for an MPNN we do not have $\bigwedge_{i=1,2} |\mathcal{F}_\Theta(\mathcal{G}(\Pi_i)) - h(\Pi_i)| \leq c$. Also, for any set of parameters we do not have $\bigwedge_{i=1,2} |1 - \mathcal{F}_\Theta(\mathcal{G}(\Pi_i))/h(\Pi_i)| \leq c$.

- Proof idea: class of counterexamples



(a) SLG of P_1 for $n = 3$. (b) SLG of P_2 for $n = 3$.

Not all hope is lost

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

3. GOOSE architecture

1. states converted to graphs
2. graphs fed into GNN with learned parameters
3. GPU batch evaluate *only*[1] successor states in GBFS

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

4a. Domain-Independent Learning

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

			GOOSE		
	blind	h^{FF}	SLG	FLG	LLG
blocks (90)	-	**19**	14	14	16
ferry (90)	-	**90**	15	6	12
gripper (18)	1	**18**	4	5	9
n-puzzle (50)	-	**36**	3	3	-
sokoban (90)	74	**90**	48	42	39
spanner (90)	-	-	-	-	**10**
visitall (90)	-	6	**44**	24	38
visitsome (90)	3	26	**37**	14	-

[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

			GOOSE		
	blind	h^{FF}	SLG	FLG	LLG
blocks (90)	-	19	10	11	**29**
ferry (90)	-	**90**	33	33	78
gripper (18)	1	**18**	5	9	**18**
n-puzzle (50)	-	**36**	10	10	1
sokoban (90)	74	**90**	52	56	34
spanner (90)	-	-	-	-	**55**
visitall (90)	-	6	**52**	35	39
visitsome (90)	3	26	**78**	23	3

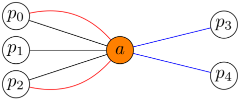
[3] except *n*-puzzle and Sokoban

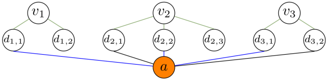
IPC2023 Learning Track Benchmarks

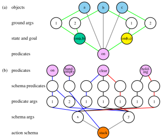
Domain	blind	h^{FF}	GOOSE$_{max}$	GOOSE$_{mean}$	Graph Kernels GenPlan23	Graph Kernels ICAPS24
blocksworld	8	28	49	58	49	**77**
childsnack	9	26	19	20	20	**30**
ferry	10	68	64	72	74	**76**
floortile	2	**12**	-	-	2	2
miconic	30	**90**	**90**	**90**	**90**	**90**
rovers	15	34	25	29	**45**	37
satellite	12	**65**	31	29	37	57
sokoban	27	36	32	33	37	**38**
spanner	30	30	30	33	30	**74**
transport	9	41	38	35	**49**	32
sum	152	430	378	399	433	**513**

GOOSE: Learning Domain-Independent Heuristics

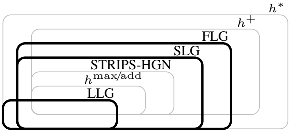
1. New graph representations of planning tasks







2. Theoretical Results



3./4. GOOSE + experiments

	blind	h^{FF}	GOOSE SLG	GOOSE FLG	GOOSE LLG
blocks (90)	-	19	10	11	**29**
ferry (90)	-	**90**	33	33	78
gripper (18)	1	**18**	5	9	**18**
n-puzzle (50)	-	**36**	10	10	1
sokoban (90)	74	**90**	52	56	34
spanner (90)	-	-	-	-	**55**
visitall (90)	-	6	**52**	35	39
visitsome (90)	3	26	**78**	23	3

	blind	h^{FF}	GOOSE SLG	GOOSE FLG	GOOSE LLG
blocks (90)	-	**19**	14	14	16
ferry (90)	-	**90**	15	6	12
gripper (18)	1	**18**	4	5	9
n-puzzle (50)	-	**36**	3	3	-
sokoban (90)	74	**90**	48	42	39
spanner (90)	-	-	-	-	**10**
visitall (90)	-	6	**44**	24	38
visitsome (90)	3	26	**37**	14	-

Domain	blind	h^{FF}	$\text{GOOSE}_{\text{max}}$	$\text{GOOSE}_{\text{mean}}$	Graph Kernels GenPlan23	Graph Kernels ICAPS24
blocksworld	8	28	49	58	49	**77**
childsnack	9	26	19	20	20	**30**
ferry	10	68	64	72	74	**76**
floortile	2	**12**	-	-	2	2
miconic	30	**90**	**90**	**90**	**90**	**90**
rovers	15	34	25	29	**45**	37
satellite	12	**65**	31	29	37	57
sokoban	27	36	32	33	37	**38**
spanner	30	30	30	33	30	**74**
transport	9	41	38	35	**49**	32
sum	152	430	378	399	433	**513**