Goal Recognition through Reinforcement Learning

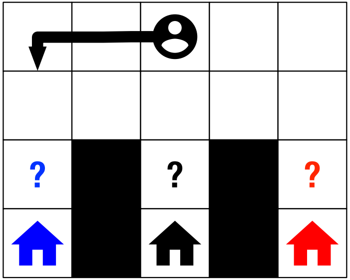
Prof. Felipe Meneguzzi†

†University of Aberdeen, Scotland, UK
felipe.meneguzzi@abdn.ac.uk
‡Plus a ton of other people I acknowledge at the end

Melbourne, May 2024

Table of Contents

- **Goal Recognition** is the task of recognizing agents' goal that explains a sequence of observations of its actions;
    - Related to plan recognition, i.e. recognizing a *top-level* action
    - A specific form of the problem of abduction



Motivation

Why?

- Most GR approaches rely on specifications of the dynamics of the agent in the environment when pursuing a goal. This implies a series of assumptions:
    - Mathematically precise environment specification
    - Actor and observer "share" this specification
    - "Well-behaved" noise and partial observability
- There are several limitations to this process:
    - Cost of Domain Description.
    - Noise Susceptibility.

Talk Outline

- A formalisation of Goal Recognition amenable to ML
- Recent approaches to Goal Recognition using Reinforcement Learning Algorithms
  - GRAQL
  - That which shall remain nameless
- Discussion of future prospects of RL-driven GR

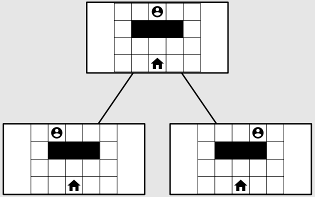
Table of Contents

Definition (Planning Task)

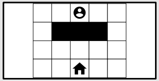
A planning task $\Pi = \langle \Xi, s_0, G \rangle$ is a tuple composed of a domain definition Ξ, an initial state s_0, and a goal state specification G. A solution to a planning task is a plan or policy π that reaches a goal state G starting from the initial state s_0 by following the transitions defined in the domain definition Ξ.

Planning problems have three key ingredients

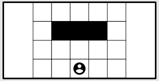
Domain Description



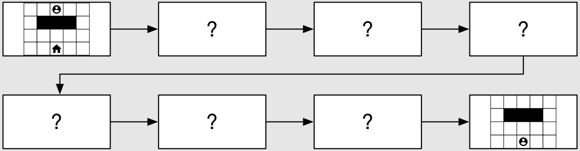
Initial State



Goal State

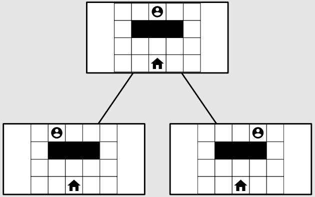


Solution

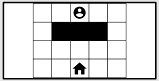


Planning problems have three key ingredients

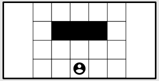
Domain Description



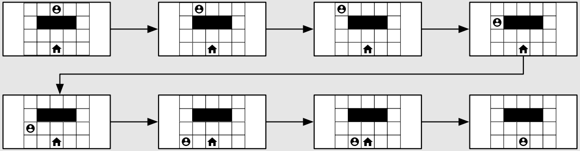
Initial State



Goal State



Solution

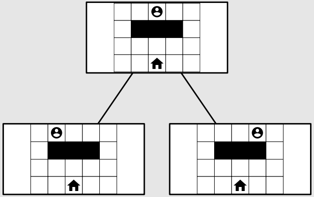


Definition (Goal Recognition Task)

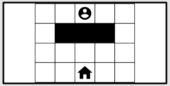
A goal recognition task $\Pi^{\Omega}_{\mathcal{G}} = \langle \Xi, s_0, \mathcal{G}, \Omega \rangle$ is a tuple composed of a domain definition Ξ, an initial state s_0, a set of goal hypotheses $\mathcal{G}$, and a sequence of observations Ω.

Goal/Plan Recognition problems have **four** key ingredients

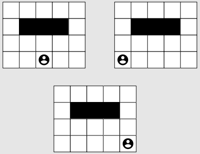
Domain Description



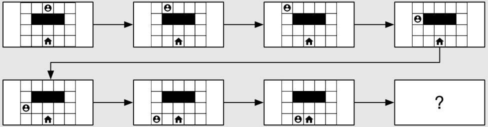
Initial State



Goal **Hypotheses**

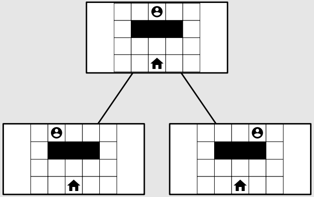


Observations

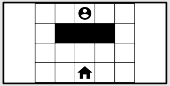


Goal/Plan Recognition problems have **four** key ingredients

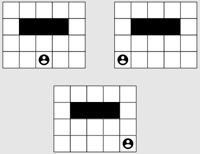
Domain Description



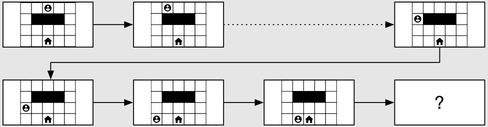
Initial State



Goal **Hypotheses**



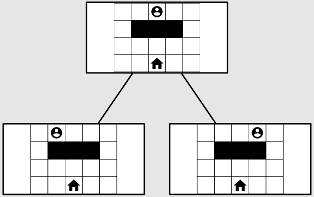
Observations



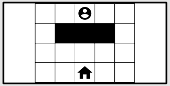
Background
Goal Recognition

Goal/Plan Recognition problems have **four** key ingredients

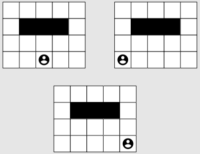
Domain Description



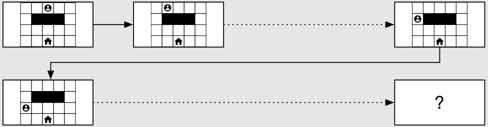
Initial State



Goal **Hypotheses**

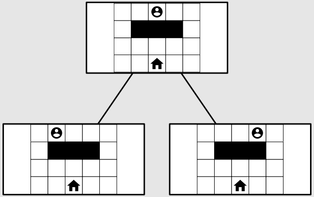


Observations

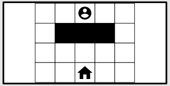


Goal/Plan Recognition problems have **four** key ingredients

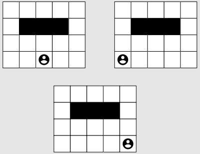
Domain Description



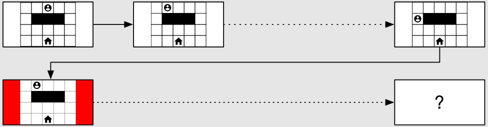
Initial State



Goal **Hypotheses**

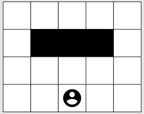


Observations



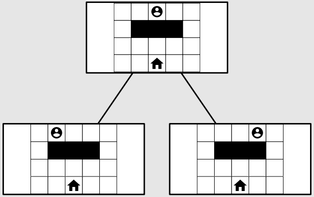
Solution

Correct Goal

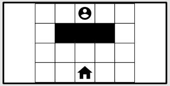


Goal/Plan Recognition problems have **four** key ingredients

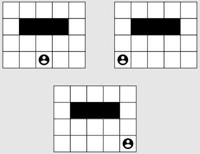
Domain Description



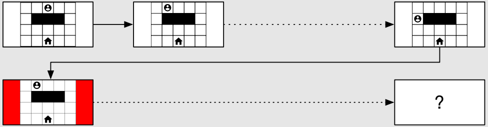
Initial State



Goal **Hypotheses**

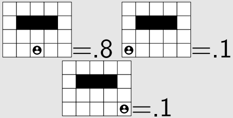


Observations



Solution

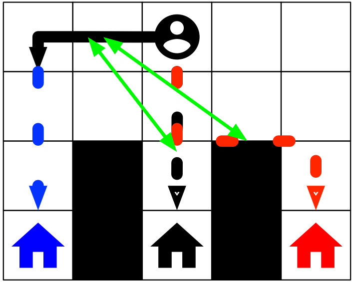
Probability Distribution



Goal Recognition using Planning Domains

Ramirez and Geffner (2009 and 2010)

- First approaches to goal recognition: Plan Recognition as Planning (PRAP)
- Probabilistic model aims to compute $P(G \mid O)$
- Following Bayes Rule $P(G \mid O) = \alpha P(O \mid G) P(G)$
- Given $P(G)$ as a prior, key bottleneck is computing $P(O \mid G)$



- Compute $P(O \mid G)$ in terms of a cost difference $c(G, O) - c(G, \bar{O})$
- Costs **two planner calls per goal hypothesis**

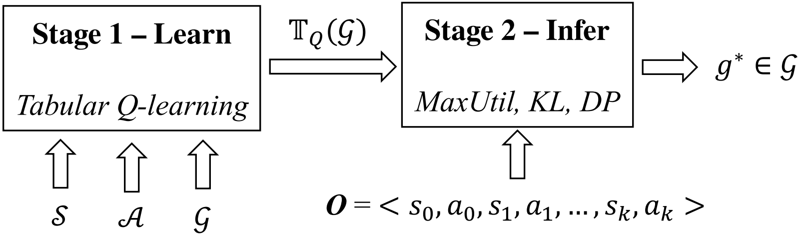
Table of Contents

Goal Recognition Problem (new)

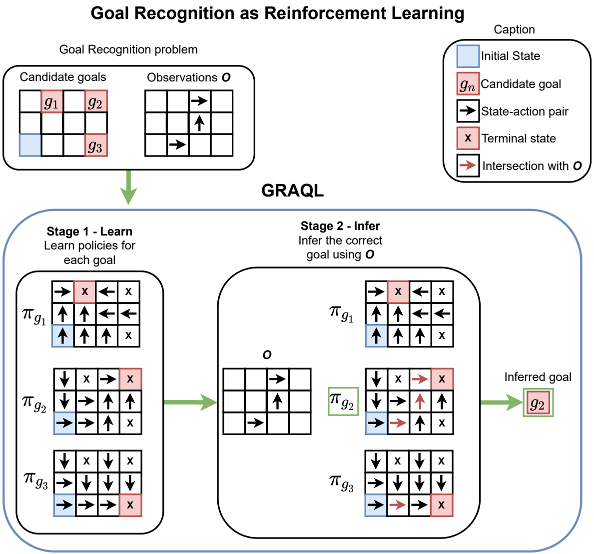
Definition (Goal Recognition Problem)

Given a domain theory $\mathbb{T}_Q(\mathcal{G})$ or $\mathbb{T}_\pi(\mathcal{G})$ and a sequence of observations Ω, output a goal $g \in \mathcal{G}$ that Ω **explains**.

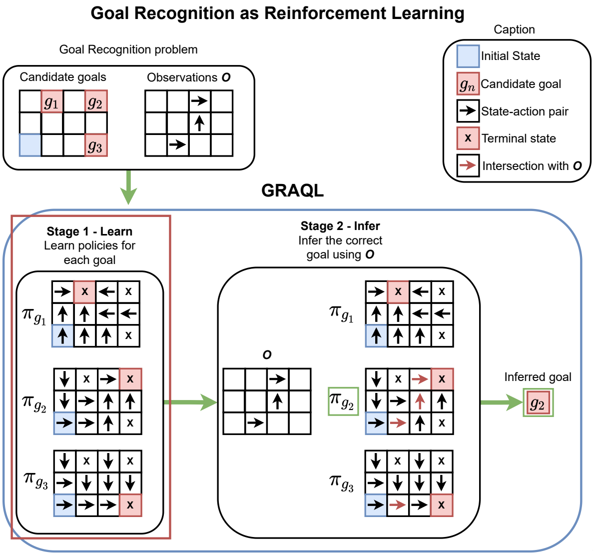
GR as RL framework



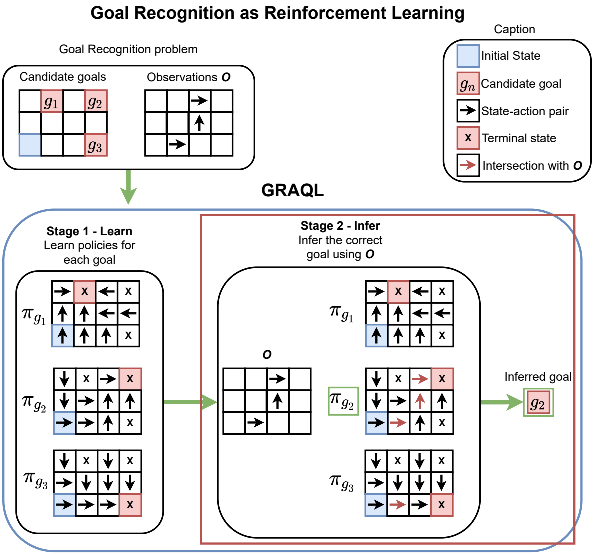
GR as RL example 1



GR as RL example 2



GR as RL example 3



GRAQL - Learning stage

GRAQL provides a first implementation for this framework.

- We use off-the-shelf Q-learning algorithms[1].
- Our goal is to learn informative domain theory with minimal effort.
- Reward for reaching the goal is 100, and 0 otherwise, and the discount factor is 0.9.
- Exploration is ϵ-greedy with linearly decaying values.

[1] `github.com/aimacode/aima-python/blob/master/reinforcement_learning.py`

GRAQL - Inference stage

$$G^* = \underset{g \in \mathcal{G}}{\arg\min}\, \textsc{Distance}(Q_g, \Omega)$$

Three distinct *distances*[2] inspired by three common RL measures:

1. MaxUtil,
2. KL-divergence,
3. Divergence Point.

[2] Not actually metrics

MaxUtil is an accumulation of the utilities collected from the observed trajectory.

$$MaxUtil(Q_g, \Omega) = -\sum_{i \in |\Omega|} Q_g(s_i, a_i) \quad (1)$$

KL-Divergence is a measure for the divergence between two distributions, so we construct two policies, π_g and π_Ω for Q_g and Ω respectively.

$$KL(Q_g, \Omega) = \mathrm{D}_{\mathrm{KL}}(\pi_g \,||\, \pi_\Omega) = \sum_{i \in |\Omega|} \pi_g(a_i \mid s_i) \log \frac{\pi_g(a_i \mid s_i)}{\pi_\Omega(a_i \mid s_i)} \tag{2}$$

Divergence Point (DP) is a measure adapted from Macke et al[3], where given a trajectory Ω and a policy π, it is defined as the minimal point in time in which the action taken by Ω has zero probability to be chosen by π.

$$DP(Q_g, \Omega) = -min\{t \mid \pi_g(a_{t-1} \mid s_{t-1}) \leq \delta\} \tag{3}$$

[3] William Macke, Reuth Mirsky, and Peter Stone. “Expected Value of Communication for Planning in Ad Hoc Teamwork”. In: *Proceedings of the 35th Conference on Artificial Intelligence (AAAI)*. Virtual Conference, Feb. 2021.

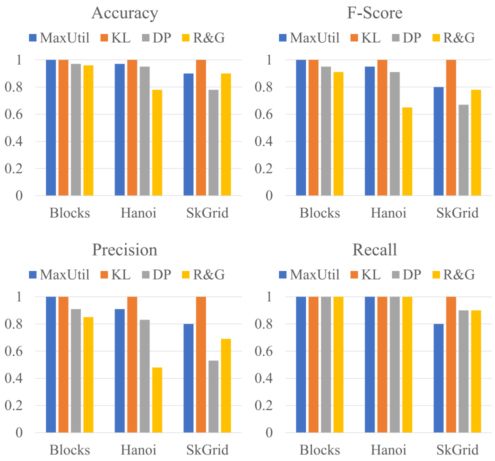
We use three domains from the PDDLGym library for their similarity with commonly used GR evaluation domains:

1. Blocks,
2. Hanoi,
3. SkGrid (which resembles common GR navigation domains with obstacles)

- For each domain, we generate **10** GR problems with **4** candidate goals. We manually choose ambiguous goals.
- Each problem has 7 variants, including partial and noise observations. We have 5 variants with varying degrees of observability (10%, 30%, 50%, 70%, and full observability), and 2 variants that include noise observations with varying degrees of observability (50% and full observability).
- Our test set includes 210 GR problems, which we compare with R&G

Results

Full Observability



		Accuracy				Precision				Recall				F-Score			
O	Domain	MU	KL	DP	RG	MU	KL	DP	RG	MU	KL	DP	RG	MU	KL	DP	RG
0.5	Blocks	**0.95**	0.62	0.93	0.84	**0.95**	0.33	0.77	0.56	0.90	0.50	**1.00**	**1.00**	**0.90**	0.40	0.87	0.71
	Hanoi	**0.97**	0.90	0.93	0.68	**0.91**	0.80	0.77	0.38	**1.00**	0.80	**1.00**	**1.00**	**0.95**	0.80	0.87	0.56
	SkGrid	0.75	0.75	0.57	**0.88**	0.50	0.50	0.35	**0.64**	0.50	0.50	0.80	**0.90**	0.50	0.50	0.48	**0.75**
1.0	Blocks	**1.00**	**1.00**	0.95	0.96	**1.00**	**1.00**	0.83	0.83	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.91	0.91
	Hanoi	**1.00**	**0.95**	0.90	0.78	**1.00**	0.90	0.71	0.48	**1.00**	0.90	**1.00**	**1.00**	**1.00**	0.90	0.83	0.65
	SkGrid	0.85	**0.95**	0.65	0.90	0.70	**0.90**	0.40	0.69	0.70	**0.90**	0.80	**0.90**	0.70	**0.90**	0.53	0.78
Avg	Blocks	**0.97**	0.81	0.94	0.90	**0.97**	0.60	0.80	0.70	0.95	0.75	**1.00**	**1.00**	**0.95**	0.67	0.89	0.81
	Hanoi	**0.99**	0.93	0.91	0.73	**0.95**	0.85	0.74	0.43	**1.00**	0.85	**1.00**	**1.00**	**0.98**	0.85	0.85	0.61
	SkGrid	0.80	0.85	0.61	**0.89**	0.60	**0.70**	0.37	0.67	0.60	0.70	0.80	**0.90**	0.60	0.70	0.51	**0.77**

Table of Contents

Value Function Approximation

- So far we have represented value function by a lookup table
  - Every state s has an entry $V(s)$
  - Or every state-action pair s, a has an entry $Q(s, a)$
- Problem with large MDPs:
  - There are too many states and/or actions to store in memory
  - It is too slow to learn the value of each state individually
- Solution for large MDPs:
  - Estimate value function with *function approximation*

$$\hat{v}(s, \boldsymbol{w}) \approx v_\pi(s)$$
$$\text{or } \hat{q}(s, a, \boldsymbol{w}) \approx q_\pi(s, a)$$

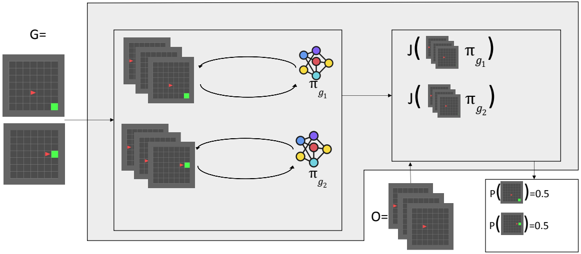
  - Generalize from seen states to unseen states
  - Update parameter $\boldsymbol{w}$ using MC or TD learning

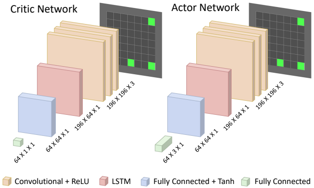
Catchy name for agent architecture

Goal recognition using function approximation

Catchy name for agent architecture

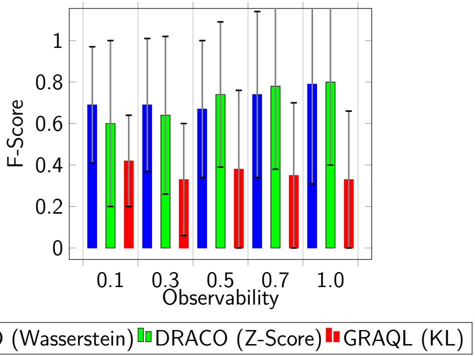
- We adapted our algorithm to use function approximators:
  - Actor-Critic learning
  - Different distance metrics suitable for continuous domains
- Comparison of observations using:
  - Wasserstein distance
  - Z-Score function





Panda-gym

Performance in Panda-Gym



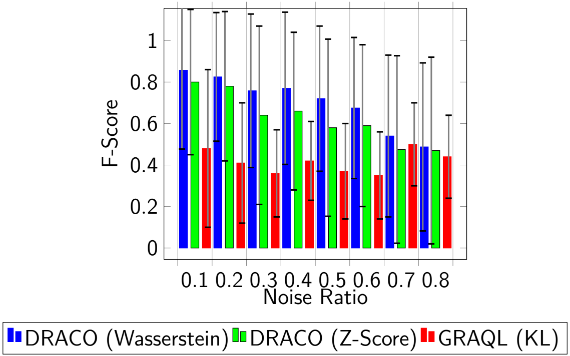


Table of Contents

Amir and Chang 2008[4]; Amado et al. 2019[5]; Asai and Muise 2020[6]; Juba, Le, and Stern 2021[7]

[4] Eyal Amir and Allen Chang. "Learning partially observable deterministic action models". In: *Journal of Artificial Intelligence Research* 33 (2008), pp. 349–402.

[5] Leonardo Amado et al. "Goal recognition in latent space". In: *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2018, pp. 1–8.

[6] Masataro Asai and Christian Muise. "Learning Neural-Symbolic Descriptive Planning Models via Cube-Space Priors: The Voyage Home (to STRIPS)". In: *CoRR* abs/2004.12850 (2020). arXiv: 2004.12850. URL: https://arxiv.org/abs/2004.12850.

[7] Brendan Juba, Hai S. Le, and Roni Stern. "Safe Learning of Lifted Action Models". In: *International Conference on Principles of Knowledge Representation and Reasoning (KR)*. 2021.

- Inverse reinforcement learning (IRL): **Zeng et al 2018**[8].
- Other metric-based GR: **Masters and Sardina 2017**[9]**; Mirsky et al. 2019**[10]

[8] Yunxiu Zeng et al. "Inverse Reinforcement Learning Based Human Behavior Modeling for Goal Recognition in Dynamic Local Network Interdiction.". In: *AAAI Workshops*. 2018, pp. 646–653.

[9] Peta Masters and Sebastian Sardina. "Cost-based goal recognition for path-planning". In: *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*. 2017.

[10] Reuth Mirsky et al. "New goal recognition algorithms using attack graphs". In: *International Symposium on Cyber Security Cryptography and Machine Learning*. Springer. 2019, pp. 260–278.

Table of Contents

What next

Future work

This is part of a larger research agenda, we still make too many assumptions:

- No explicit prior, but we could consider it in various ways
- No null hypothesis (goals are mutually exclusive, and exhaustively enumerated)
- Keyhole settings ignore strategic behaviour in both agents

Future directions for research:

- Incorporating priors (explicitly or otherwise)
  - Reconstruct the reward function with IRL
  - Learn policies via *Imitation Learning* or *Learning from Observation*
- Learning more generic policies/reward functions:
  - Goal Conditioned policies
  - Reward Machines
- Game theoretical settings

Credits

Not my genius

- Reuth Mirsky and Ben Nageris (Bar-Ilan University)
- Leonardo Amado (University of Aberdeen)
- Ramon Pereira (University of Manchester)
- Mor Vered (Monash University)
- Miquel Ramirez (University of Melbourne)
- Nir Oren (University of Aberdeen)
- André Pereira (UFRGS)
- João Paulo Aires (PUCRS)
- Maurício Magnaguagno (PUCRS)
- Juarez Monteiro (SICREDI)
- Roger Granada (Unico)