Beyond Goal Recognition: A Reinforcement Learning-based Approach to Inferring Agent Behaviour

Sheryl Mantik
RMIT University
Melbourne, Australia
sheryl.mantik@student.rmit.edu.au

Michael Dann
viewa.com
Melbourne, Australia
michaeltdann@gmail.com

Minyi Li
jahan.ai
Melbourne, Australia
liminyi0709@gmail.com

Huong Ha
RMIT University
Melbourne, Australia
huong.ha@rmit.edu.au

Julie Porteous
RMIT University
Melbourne, Australia
julie.porteous@rmit.edu.au

ABSTRACT

Goal recognition (GR) involves inferring an agent's goals based on observed actions. In addition to goals, however, in various cases it may be useful to infer additional agent attributes, such as preferences, beliefs, and ability level, so as to gain deeper insights into the agent's decision-making process. Recent advances in GR have incorporated Reinforcement Learning (RL), which provides greater practicality and adaptability, especially in stochastic environments. This adaptability creates the opportunity to extend RL-based frameworks beyond goal recognition. In this work, we build upon a recent RL-based GR framework to propose a generalised approach capable of inferring a wider range of agent attributes. By integrating these attributes within the problem formulation, we demonstrate how off-the-shelf RL techniques can be applied to infer them effectively. Our results show that this extended framework accurately distinguishes fine-grained differences in agent attributes across diverse scenarios. Moreover, we show that recognising these additional attributes can in turn improve goal recognition accuracy.

KEYWORDS

Agent Modelling; Reinforcement Learning; Goal Recognition; Behavioural Attributes

ACM Reference Format:
Sheryl Mantik, Michael Dann, Minyi Li, Huong Ha, and Julie Porteous. 2025. Beyond Goal Recognition: A Reinforcement Learning-based Approach to Inferring Agent Behaviour. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 9 pages.

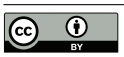
1 INTRODUCTION

Goal Recognition (GR) refers to the task of a *recogniser* inferring the goal of an *agent*, whether human or robotic, based on its observable trajectory within a given environment [8, 15, 19, 23]. Earlier works in GR have relied on the assumption of a complete domain model, which is a specification of the environment and possible actions in certain conditions [19, 21]. However, constructing accurate domain theories is often costly and impractical in real-world applications that involve dynamic and uncertain environments, limiting the effectiveness of model-based GR [3]. Recently, there has been a shift towards data-driven or learning-based frameworks. In particular, Reinforcement Learning (RL) has been used to approximate domain theories through learnt policies, thereby alleviating the need to construct an explicit domain specification [2]. RL's flexibility in learning from interactions with the environment makes it a more scalable and adaptable framework. This work focuses on extending RL-based frameworks beyond traditional GR to infer a broader set of agent attributes, enabling a more comprehensive understanding of agent behaviour in dynamic and uncertain environments.

While GR focuses primarily on goal inference, we argue that RL-based frameworks have the potential to infer a wider variety of attributes. In many domains, decision-making is influenced by factors beyond goals, such as preferences, beliefs, knowledge of the environment, and physical or cognitive abilities [1]. In some cases, knowing the goal alone may not be sufficient to accurately predict an agent's future behaviour. By inferring additional attributes, a recogniser can gain deeper insights into the agent's decision-making process. A more holistic understanding of these attributes can provide practical benefits across a range of applications.

For instance, a robotic assistant might perform tasks such as fetching objects. While the robot's goal is clear—to retrieve the object—the path it chooses could be influenced by various factors. These include the robot's preferences, such as favouring smoother surfaces over rough terrain to reduce wear and tear or conserve energy. Its decision-making might also be shaped by its beliefs about the environment; for example, if the robot has incomplete knowledge due to partial observability of certain areas, it might avoid them, believing they contain obstacles or hazards. Additionally, its ability level, such as a low battery affecting its capacity to lift heavier objects or move quickly, could lead the robot to take a more energy-efficient route or postpone certain actions until recharged. Inferring these attributes—preferences, beliefs, and ability levels—can help improve understanding of its behaviours.

Recognising that goal inference alone does not fully capture an agent's attributes, we propose a generalised RL-based framework that also infers preferences, beliefs, and ability levels. By incorporating these attributes, our framework allows for a more nuanced



Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

understanding of agents and their decision-making processes. Moreover, this approach leverages off-the-shelf RL techniques, making it both scalable and adaptable to a wide range of domains.

The remainder of this paper is structured as follows. Section 2 provides a detailed overview of recent advances in RL-based GR and formulates the GR problem as an RL problem. Section 3 introduces our generalised framework and explains how various agent attributes are modelled within the problem formulation. In Section 4, we describe our experimental setup and the diverse scenarios used to evaluate the framework. In Section 5, we highlight related work that shares similar aims of inferring agent attributes. Finally, in Section 6, we conclude the paper and outline potential future directions for extending this work.

2 PRELIMINARIES: GOAL RECOGNITION AS REINFORCEMENT LEARNING

Goal recognition (GR) involves inferring an agent's goal based on its observed behaviour within an environment [8, 15, 19, 23]. Traditionally, GR relies on model-based approaches, where domain models are explicitly defined [16]. However, creating such models is often impractical in real-world scenarios due to the need for domain expertise and the nature of the environment being dynamic and uncertain. Recently, model-free GR methods have emerged, leveraging machine learning or data-driven approaches to infer goals directly from observed actions without requiring detailed domain knowledge [3].

In this work, we focus on model-free GR approaches that leverage RL techniques to acquire domain models. This RL-based approach was first proposed by [2], where the domain model is formulated as an RL task. Formally, the learning task of acquiring a domain model is specified as a Markov Decision Process (MDP) in Definition 2.1.

Definition 2.1 (Markov Decision Process). A Markov Decision Process is a tuple $\mathcal{M} = \langle S, A, T, \gamma, R \rangle$ where S is a set of states, A is a set of actions, $T(s, a, s')$ is a transition function, where it defines the probability of transitioning to a new state s' given the current state s and action a, γ is a discount factor, $R(s)$ is a reward function.

The RL problem involves learning a policy $\pi(a|s)$ for an MDP, which is a function that defines the probability of the agent taking action $a \in A$ in state $s \in S$. An RL policy π can be derived from a Q-function, $Q(s, a)$ — the expected return for taking action a in state s then acting greedily in subsequent steps. These components—policies and Q-functions—are essential in capturing an agent's behaviour within an environment. To formally represent the agent's interaction with its environment, we define a domain model $\mathfrak{D}$, which encapsulates either a set of Q-functions or policies that govern the agent's actions in different states.

Definition 2.2 (Domain Model). Domain model $\mathfrak{D}_\rho$ is a tuple of $\langle S, A, \rho \rangle$ where S is a set of states s, A is a set of actions a, and ρ is either a set of Q-functions $\mathcal{Q} = \{Q_g\}_{g \in \mathcal{G}}$, or a set of policies $\Pi = \{\pi_g\}_{g \in \mathcal{G}}$.

For each policy-based domain model $\mathfrak{D}_\pi$, a utility-based domain model $\mathfrak{D}_\mathcal{Q}$ can be derived using the softmax function for each g. The softmax function is commonly used to map Q-values to a probabilistic policy, as shown in Equation 1,

$$\pi(a|s) = \frac{Q(s, a)}{\sum_{a' \in A} Q(s, a')}. \tag{1}$$

With the defined domain model, we can now define the GR problem in the context of RL. Using the RL-based framework, we model GR as the problem of recognising which policy or Q-function best explains the observed behaviour, given the defined domain model $\mathfrak{D}_\rho$. The RL-based GR problem can be structured as follows:

Definition 2.3 (Goal Recognition as Reinforcement Learning). A Goal Recognition as Reinforcement Learning is a tuple $GR = \langle \mathfrak{D}_\rho, \mathcal{G}, O, Prob \rangle$, where $\mathfrak{D}_\rho$ is a domain model, $\mathcal{G}$ is a set of goals g, O is a set of observations $\langle o_1, o_2, ..., o_n \rangle$, and $Prob$ a probability distribution over $\mathcal{G}$.

The RL-based framework consists of two main stages: offline training and online inference. During offline training, the framework approximates Q-functions using off-the-shelf RL techniques. Before approximating the Q-functions, the reward function $R(s)$ is defined as part of the MDP, where each goal g produces a positive reward when achieved. Specifically, the reward function $R(s)$ assigns a positive value when the agent achieves goal g, and zero otherwise. In [2], a tabular Q-learning approach is applied for discrete domains. More recently, extensions to continuous domains have been made using Deep Reinforcement Learning (DRL) [12]. A detailed discussion of our framework's adaptation for offline training is provided in Section 3.2.

During online inference, distance measures are used to compare the observed trajectory with the learnt Q-functions. Specifically, the goal g is evaluated by comparing the Q-functions for each goal $Q_g(s, a)$ with the observed actions. The work in [2] uses the Kullback-Leibler Divergence (DKL) measure. For DKL, two types of policies are constructed:

First, a softmax policy for each specified goal, π_g, is derived from the learnt Q-functions Q_g using Equation 1. Second, a policy π_O is constructed for the observed trajectory O, where $\pi_O(a|s_i) = 1$ when $a = a_i$, and $\pi_O(a|s_i) = 0$ when $a \neq a_i$.

Formally, DKL is described in Equation 2, which measures the distance between the two probability distributions, π_g and π_O,

$$D_{KL}(\pi_O || \pi_g) = \sum_{i \in |O|} \pi_O(a_i|s_i) log \frac{\pi_O(a_i|s_i)}{\pi_g(a_i|s_i)}. \tag{2}$$

The work in [2] also explores other commonly used RL distance measures, such as MaxUtil and Divergence Point. We focus on DKL because it generally performs better than these other measures based on [2]. A more detailed discussion of our framework's adaptation for online inference is provided in Section 3.3.

3 EXTENDING GOAL RECOGNITION TO INFER OTHER AGENT ATTRIBUTES

In this section, we introduce our recognition framework for inferring other agent attributes. Specifically, we define this task as the Attribute Recognition (AR) problem, where the goal is to infer a specified attribute of an agent based on its observations. Formally, the AR problem can be defined as follows:

Definition 3.1 (Attribute Recognition). Attribute Recognition is a tuple $AR = \langle \mathfrak{D}, \mathcal{A}, O, Prob \rangle$ which consists of:

- $\mathfrak{D}$ is the domain model,
- $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2 \times ... \times \mathcal{A}_n$ is the Cartesian product of sets of attribute types $\mathcal{A}_1, \mathcal{A}_2, ..., \mathcal{A}_n$. Each $\mathcal{A}_i$ represents a type of attribute, and an attribute $\alpha \in \mathcal{A}$ is an n-tuple $\alpha = (\alpha_1, \ldots, \alpha_n)$ where $\alpha_i \in \mathcal{A}_i$,
- O is a set of observations $\langle o_1, o_2, ..., o_n \rangle$,
- $Prob$ as probability distribution over $\mathcal{A}$.

Similar to RL-based GR approaches, we use a standard MDP to formulate the domain model $\mathfrak{D}$ from Definition 2.2 as the input to the AR problem. For each attribute $\alpha \in \mathcal{A}$, we generate a corresponding set of Q-functions or policies. More specifically, RL-based AR can be viewed as the task of inferring which Q-function $Q_\alpha(s, a)$ best explains the observation O. When the attribute set $\mathcal{A}$ is defined as goals G, this problem becomes equivalent to the GR problem in Definition 2.3. We present this problem definition as a general abstraction into which various attribute types can be incorporated.

In the remainder of this section, we introduce our Attribute Recognition (AR) framework, which consists of three main stages. In Section 3.1, we describe how we construct the MDP to model the agent's attributes. Sections 3.2 and 3.3 cover the offline training and online inference stages, respectively, which build upon previous RL-based GR frameworks. In earlier RL-based GR frameworks, the first stage (attribute modelling) was integrated into the second stage (offline training) as highlighted in Section 2. However, we aim to demonstrate that by separating these stages and explicitly defining the attributes within the MDP, we can train policies tailored to each attribute specification while still being able to infer them accurately.

3.1 Modelling Agent Attributes in Markov Decision Process

We explore several attributes that are useful for coordination, as highlighted in our motivation in Section 1, such as ability level, preferences, and beliefs. In this section, we demonstrate how to incorporate these attributes into our problem formulation within an MDP from Definition 2.1.

3.1.1 Ability Level. We define ability level as an attribute that reflects how capable an agent is of completing a given task. To model this, we assign a probability that the agent will succeed in the task. A common approach to measuring an agent's performance is the Elo rating system [10], which was originally developed for chess and has since been widely adopted in various games.

We adapt the Elo rating system to capture an agent's ability to complete a task based on the difficulty of the task, rather than comparing two agents. Mathematically, given an agent's ability rating λ and the task difficulty rating δ, the difference between these ratings serves as a predictor of the agent's likelihood of success. The probability that the agent will solve the task is given by Equation 3. In the standard Elo system, the scaling factor k is set to 400.

$$E = \frac{1}{1 + 10^{(\delta - \lambda)/k}} \tag{3}$$

We introduce two new parameters—agent ability level λ and task difficulty rating δ—to extend the MDP defined in Definition 2.1. Formally, we propose an Ability-Based Markov Decision Process (MDP_{AB}) as an extension to the standard MDP, defined as follows:

Definition 3.2 (Ability-based Markov Decision Process). An Ability-Based Markov Decision Process is a tuple $\hat{\mathcal{M}} = \langle \hat{S}, A, \hat{T}, \gamma, \hat{R}, \lambda, \delta \rangle$ that extends a base MDP, where $\hat{S}$ is a set of states, A is a set of actions, $\hat{T}(s, a, s')$ is a transition function, γ is a discount factor, $\hat{R}(s)$ is a reward function, λ is the ability level of an agent, $\delta(s, a)$ is a difficulty rating function of state-action pairs.

In this formulation, we treat a state-action pair in MDP_{AB} as a task. The difficulty of the task is represented by $\delta(s, a)$, which indicates how challenging it is for an agent to perform action a in state s. A higher value of λ corresponds for agent with higher ability, while a higher value of $\delta(s, a)$ indicates a more difficult task.

The set of states $\hat{S}$ is extended with the addition of a special failure state, s_{fail}, such that $\hat{S} = S \cup s_{\text{fail}}$. Consequently, the reward function $\hat{R}(s)$ must also be extended to account for the reward associated with reaching s_{fail}.

The transition function $\hat{T}(s, a, s')$ defines the probability of transitioning from state s to state s', and it is influenced by the agent's ability λ and the task difficulty $\delta(s, a)$. We define $\hat{T}(s, a, s')$ as a conditional function:

$$\hat{T}(s, a, s') = \begin{cases} \frac{1}{1+10^{(\delta(s,a)-\lambda)/k}} * T(s, a, s'), & \text{if } s' \neq s_{\text{fail}} \\ \frac{1}{1+10^{(\lambda-\delta(s,a))/k}}, & \text{if } s' = s_{\text{fail}} \end{cases} \tag{4}$$

The probabilities in $\hat{T}(s, a, s')$ depend on whether the agent transitions to the failure state s_{fail} or to any other state. In both cases, the transition probability is calculated using the agent's ability λ and the task difficulty $\delta(s, a)$, following the Elo rating formula in Equation 3. The two possible cases are outlined as follows:

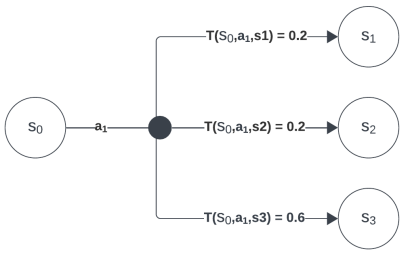
- If $s' \neq s_{\text{fail}}$, we compute the probability of the agent successfully completing the task and multiply it by the transition function $T(s, a, s')$ from the standard MDP (as defined in Definition 2.1).
- If $s' = s_{\text{fail}}$, we compute the probability of the agent failing the task.

An illustration of this extended transition function $\hat{T}(s, a, s')$ is provided in Figure 1, where it depicts the transitions from state s_0 to possible subsequent states s_1, s_2, s_3, or the failure state s_{fail} after executing action a_1.

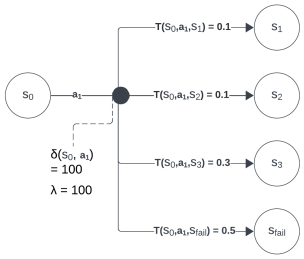
During training, the agent with a low ability level will learn that attempting certain tasks has a lower probability of success compared to agents with a high ability level. This results in different policies for agents with high ability versus those with low ability.

3.1.2 Preferences. We define preferences as an attribute that describes how much an agent favours a particular outcome over another. An outcome refers to a specific state that the agent can achieve as a consequence of its actions within the environment. For example, an outcome might be the collection of an item, reaching a certain location, or completing a task.

Although preferences are similar to goals, they differ in important ways. A goal implies a firm intention to achieve a specific outcome, meaning the agent is focused on reaching that particular result. In contrast, preferences describe the degree to which an agent favours one outcome over others, allowing more flexibility. An agent might prefer certain outcomes over others, but this does



(a) Transition function of a standard MDP



(b) Transition function of MDP_{AB}

Figure 1: Transition function adaptation for ability level.

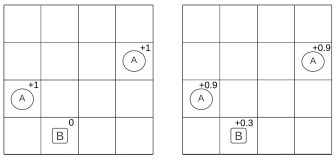


Figure 2: Reward weighting of Goal A (left) vs. Preference over A: 0.7 and B: 0.3 (right).

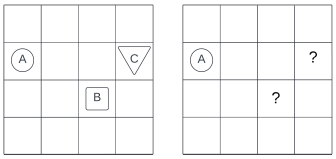


Figure 3: Fully observable state with distinct items (left) vs. Partial observable state with indistinguishable items (right).

not necessarily imply the agent is exclusively focused on a single goal.

For this work, we use a simplified definition of preference to illustrate how they can be integrated into the framework. We will discuss potential extensions in Section 6.

To incorporate preferences into an MDP, we build on how RL-based GR approaches define goals. In RL-based approach, the reward function $R(s)$ expresses an agent's goal by assigning a positive reward when achieved, as discussed in Section 2. For example, assume an agent operates in a grid world with two items, A and B in Figure 2. In the given domain, if the agent's goal is collect item A, $R(s)$ will assign a positive reward (e.g., +1) when the agent successfully collects item A. However, $R(s)$ will assign 0 when the agent collects item B. We view this as focusing on a single outcome and assign a positive reward when the specific outcome is achieved.

When modelling preferences, we can extend this view of the reward function to express agent's preferences over possible outcomes. Instead of focusing on a single outcome, we consider multiple outcomes and use the reward function $R(s)$ to assign numerical values that represent the strength of the agent's preference for each outcome. A higher reward indicates a stronger preference for that outcome. For example, in Figure 2, if the agent has a stronger preference for item A than for item B, this can be modelled by reward function $R(S)$ assigning 0.9 for when collecting item A and 0.3 when collecting item B, proportional to the agent's preference.

During training, the agent with stronger preferences will prioritise achieving the preferred outcomes, while agents with weaker preferences will place less emphasis on these outcomes. This will result in different policies for agents with strong preferences versus those with weak preferences.

3.1.3 Belief. We define belief as an agent's knowledge about the state of a domain or environment. Agents may not always have complete knowledge of the true state of the world and often operate under partial observability. To address this, agents maintain a belief about the state of the world, which serves as a probabilistic estimate of the environment. This estimate informs the agent's decision-making process. As with preferences, we are using a simplified definition of beliefs, with possible extensions (see Section 6).

To represent belief within an MDP, we update the state representation S to account for partial observability. Specifically, we modify certain states so that outcomes with different rewards may appear indistinguishable to the agent.

To illustrate this, consider a grid world domain as shown in Figure 3. An agent with full observability can distinguish between all items A, B, and C. However, we modify the state representation for an agent with partial observability so that items B and C, which have conflicting rewards (e.g., -1 for B and +1 for C), appear indistinct to the agent. During training, the agent with partial observability will learn that collecting these indistinguishable items carries a 50% probability of yielding a good outcome. This would result in different policies for agents with full observability versus those with partial observability.

3.2 Offline Training: Learning Domain Model

The offline learning stage focuses on building the domain model by learning Q-functions. For each known instance of an attribute $\alpha = (\alpha_1, ..., \alpha_n) \in \mathcal{A}_1 \times ... \times \mathcal{A}_n$, we formulate an MDP that models these attributes, which is used to train the Q-functions $Q_\alpha(s, a)$. Learning the Q-functions $Q_\alpha(s, a)$ can be accomplished using various off-the-shelf RL techniques, which will be demonstrated in our experiments in Section 4. As the number of attribute instances α increases, the number of Q-functions to be learnt also grows. We address this scalability challenge in Section 4.3, where we discuss an extension of the offline training process to manage a large set of attributes.

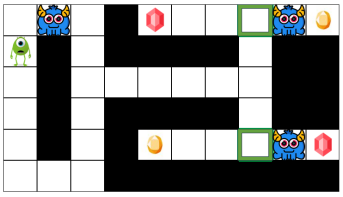


Figure 4: Treasure Domain

3.3 Online Attribute Inference

After learning the Q-functions, we can begin inferring agent attributes based on observations. During online execution, for each step i in the trajectory, the recogniser captures an observation of a state-action pair $o_i = \langle s_i, a_i \rangle$. The recogniser then compares the learnt Q-functions for each attribute $Q_\alpha(s_i, a_i)$ with the observed state-action pair o_i using a distance measure.

Initially, we adopt the DKL measure, as shown in Equation 2, following the approach from [2]. However, DKL only provides a summation of scores at each step i of the observation O, without yielding a probability distribution over the attributes.

To address this limitation, we propose using Bayesian Inference (BI) as a distance measure. This method is adapted from [20], which was extensively used for planning-based GR [16] and, to our knowledge, this has not been explored in RL-based GR [3]. BI allows us to calculate a probability distribution over the attributes directly and incorporates prior knowledge about attribute probabilities. Formally, we define the BI method to compute the posterior probability distribution over attributes α given an observation $o_i \in O$, as shown in Equation 5.

$$P(\alpha|o_i) = \frac{P(o_i|\alpha)P(\alpha)}{P(o_i)}. \tag{5}$$

4 EXPERIMENTS

To evaluate our Attribute Recognition (AR) framework, we conducted experiments in two deterministic domains: the Treasure domain (Figure 4), which we created, and the Craft World domain (Figure 5), adapted from [9]. Both domains require agents to collect items but differ in complexity. The Treasure domain illustrates how an agent's behaviour is influenced by attributes beyond its goal, while the more complex Craft World domain allows for a wider range of attributes.

We assessed our AR framework's performance in three key areas. In Section 4.1, we evaluated whether AR framework could infer behaviour more effectively when considering multiple attributes, compared to only inferring goals, which in turn enhance GR performance. In Section 4.2, we tested AR framework's ability to infer a single attribute in isolation. In Section 4.3, we revisited the scalability issue discussed in Section 3, to determine whether AR framework could infer attributes using a single Q-function.[1]

4.1 Enhancing Goal Recognition Performance

In this section, we aim to evaluate whether our proposed AR framework can enhance GR performance by inferring both goal and ability levels. We use the Treasure domain (Figure 4), in which the green agent must collect either a red or yellow gem. There are two paths to each gem: a riskier but faster one, and a safer, obstacle-free one (with blue monsters as obstacles). The agent's ability level determines the probability of successfully passing these obstacles.

λ	AR	$g=0,\lambda=100$	$g=0,\lambda=500$	$g=1,\lambda=100$	$g=1,\lambda=500$
100	AR_λ	**110.288**	114.798	113.905	113.538
	$AR_{\lambda_{\text{low}}}$	**110.288**	n/a	113.905	n/a
	$AR_{\lambda_{\text{high}}}$	n/a	114.798	n/a	**113.538**
500	AR_λ	87.053	**82.202**	85.203	82.865
	$AR_{\lambda_{\text{low}}}$	87.053	n/a	**85.203**	n/a
	$AR_{\lambda_{\text{high}}}$	n/a	**82.202**	n/a	82.865

Table 1: Average DKL scores[2] for (g, λ) predictions over 1000 trajectories. The bold values indicate AR inference's most confident estimates on (g, λ).

λ	AR	$g=0,\lambda=100$	$g=0,\lambda=500$	$g=1,\lambda=100$	$g=1,\lambda=500$
100	AR_λ	**0.786**	0.026	0.096	0.093
	$AR_{\lambda_{\text{low}}}$	**0.895**	n/a	0.105	n/a
	$AR_{\lambda_{\text{high}}}$	n/a	0.237	n/a	**0.763**
500	AR_λ	0.106	**0.392**	0.273	0.229
	$AR_{\lambda_{\text{low}}}$	0.296	n/a	**0.704**	n/a
	$AR_{\lambda_{\text{high}}}$	n/a	**0.638**	n/a	0.362

Table 2: Average BI Probabilities[3] for (g, λ) predictions over 1000 trajectories. The bold values indicate AR inference's most confident estimates on (g, λ).

In our AR framework, the MDP is extended to handle ability level using MDP_{AB} (Definition 3.2). For each action, we apply the transition function $\hat{T}(s, a, s')$ with probabilities from Equation 4. Cells near obstacles are assigned a difficulty of $\delta(s, a) = 100$. We define two ability levels using Elo ratings: $\lambda_{\text{low}} = 100$ (low) and $\lambda_{\text{high}} = 500$ (high). For successful transitions ($s' \neq s_{\text{fail}}$), the transition probability is the agent's Elo rating multiplied by 1. For agents near obstacles about to attempt passing the obstacle, those with low ability have a transition probability $\hat{T}(s, a, s') = 0.5$, while high-ability agents have $\hat{T}(s, a, s') = 0.9$. The difficulty rating $\delta(s, a)$ for adjacent obstacle pathways is uniform, and other state-action pairs have $\delta(s, a) = -\infty$, giving a pass-through probability of 1. s_{fail} is defined as a terminal state when the agent fails to pass the monster.

We implemented three recognisers: AR_λ (inferring both goal G and ability level Λ), and baselines $AR_{\lambda_{\text{low}}}$ and $AR_{\lambda_{\text{high}}}$, which only infer goals. These baselines represent the assumption in GR approaches that do not account for other attributes (in this case, varying ability levels). Each recogniser must infer 2 goals ($g = 0$, $g = 1$) and 2 ability levels ($\lambda = 100$, $\lambda = 500$). We selected two evaluation points (marked with green borders in Figure 5) to assess predictions using distance measures, where the agent's trajectory provides the most information for inference.

For this simple domain, we use Q-value iteration [22] with 100 iterations for training these recognisers. For AR_λ, four policies were generated for each goal and ability level (g, λ) combination. For $AR_{\lambda_{\text{low}}}$ and $AR_{\lambda_{\text{high}}}$, two policies were generated for each g based on the corresponding λ.

[1] Code is available at https://github.com/sh-ryl/AAMAS25_AR

[2] Lower DKL scores indicate greater confidence.

[3] Higher BI probabilities indicate greater confidence.

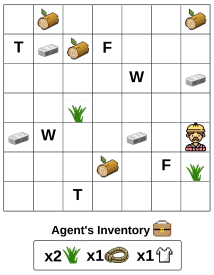


Figure 5: Craft World Domain

Item	Ingredients	Craft Location
Axe	Iron, stick	Toolshed
Bed	Grass, plank	Workbench
Bridge	Iron, wood	Factory
Cloth	Grass	Factory
Plank	Wood	Toolshed
Rope	Grass	Toolshed
Stick	Wood	Workbench

Table 3: Item recipes in Craft World domain.

4.1.1 Results. All results are averaged over 1000 trajectories on the evaluation point. For brevity, we only present the performance of our AR framework of recognising agent's with $g = 0$ as ground truth, as the results of recognising $g = 1$ are symmetrical.

Our recognisers' prediction results based on the distance measure, DKL scores are shown in Table 1, and BI probabilities in Table 2, which show the performance of the attribute recognisers (AR_λ, $AR_{\lambda_{\text{low}}}$, and $AR_{\lambda_{\text{high}}}$) in predicting the agent's attributes $\{g, \lambda\}$. Lower scores for DKL and higher probabilities for BI means that the recogniser is more confident with the prediction of attributes $\{g, \lambda\}$. For agent with low ability level $\lambda = 100$, both AR_λ and $AR_{\lambda_{\text{low}}}$ achieved highest confidence and correctly infers goal 0, while $AR_{\lambda_{\text{high}}}$ incorrectly infers goal 1. Similarly, for agent with high ability level $\lambda = 500$, both AR_λ and $AR_{\lambda_{\text{high}}}$ achieved highest confidence and correctly infers goal 1, while $AR_{\lambda_{\text{low}}}$ incorrectly infers goal 0.

In summary, we can see that $AR_{\lambda_{\text{low}}}$ and $AR_{\lambda_{\text{high}}}$ perform well when the agent possesses an ability level that it expected but suffers when the agent possesses an ability level that it does not have a basis of. Our main approach, AR_λ, performs generally well on inferring both low and high ability levels. These results establish that modelling other attributes can enhance GR performance.

4.2 Individual Attribute Inference

In this section, we evaluate the AR framework's ability to infer individual agent attributes—ability levels, preferences, and beliefs —independently. We use a single-agent version of Craft World adapted from [9], where agents move, craft, and collect items based on recipes (Table 3). Raw materials (grass, wood, iron) are collected from the environment, and episodes terminate after 100 timesteps.

For each attribute, we create specific scenarios following the MDP modifications in Section 3.1. We trained two sets of policies using DQN [17]: one for the AR recognisers and one for the agents. Results, shown in Figures 6, 7, and 8, display the BI probabilities averaged over 1000 episodes. Each column represents an agent

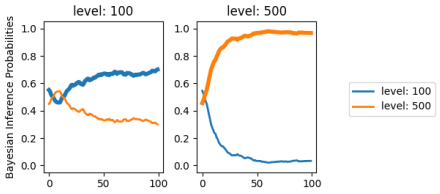


Figure 6: BI probabilities[3] for *ability level*. Each column represents an agent's policy trained w.r.t. a ground truth attribute. The bold line represents the AR inference on the ground truth attribute.

executing a specific policy, with the BI probabilities indicating the framework's confidence in a particular policy that reflects a specific attribute value. Higher probabilities mean that our framework is more confident of a specific policy which represent an attribute. The bolded line represents AR inference on the ground truth attribute. The DKL sum results are omitted as they align with the BI findings.

Each attribute's MDP modifications and AR framework performance are discussed below.

Ability Level. For this scenario, the agent collects an axe or a bridge. The axe is harder to make, requiring two crafting actions, while the bridge only requires one. The reward function assigns +1 for the axe and +0.7 for the bridge, with an additional +0.2 incentive for collecting the necessary materials. The aim is to produce policies that prioritise the axe when agents have higher-ability, while lower-ability agents focus on the bridge.

We set the crafting difficulty for all items set at $\delta(s, a) = 100$, and $\hat{T}(s, a, s') = 1$ for move and collect actions. Ability levels are set to $\Lambda = \{100, 500\}$, and s_{fail} represents a failed crafting attempt, scattering materials.

Figure 6 shows that for the lower ability agent (100), our framework would first infer the agent as high-ability agent within the first 20 timesteps, and infers the agent as low-ability agent afterwards with the probability stabilising around 0.7 by the end of the episode. Meanwhile, for the higher ability agent (500), our framework infers it accurately and the probability reaches nearly 1.0 within the first 30 timesteps. These results are likely due to the fact that, in the early timesteps, agents with both ability levels perform similar actions, such as collecting wood, which does not immediately reveal their skill level. However, as the episode progresses and the low ability agent attempts tasks such as crafting a bridge, the framework gathers more distinguishing information, allowing it to more accurately infer the agent's ability level. These results show that the AR framework effectively distinguishes between ability levels.

Preference. In this scenario, the agent collects cloth and stick, with five different reward weightings: (0.1,0.9), (0.3,0.7), (0.5,0.5), (0.7,0.3) and (0.9,0.1), summing to 1 to allow easy comparison of the agent's relative preferences between the two items. These combinations represent varying levels of preference, from strong (0.9,0.1) and (0.1,0.9) to balanced (0.5,0.5) and weaker preferences (0.3,0.7) and (0.7,0.3). A separate Q-function was trained for each weighting.

Figure 7 shows that agent with strong preferences (first and fifth columns), our framework infers it correctly, and converges to high confidence within the first 30 timesteps. Similarly for agent

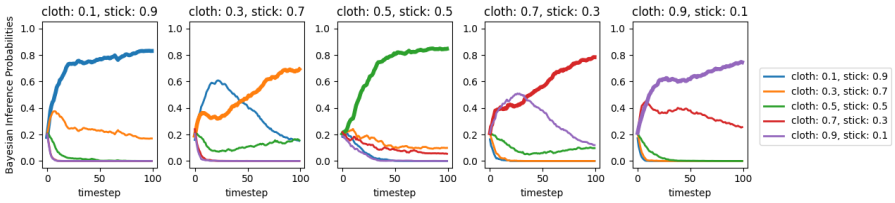


Figure 7: BI probabilities[3] for *preferences*. Each column represents an agent's policy trained w.r.t. a ground truth attribute. The bold line represents the AR inference on the ground truth attribute.

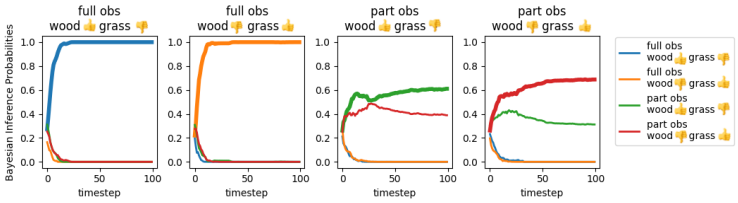


Figure 8: BI probabilities[3] for *beliefs*. Each column represents an agent's policy trained w.r.t. a ground truth attribute. The bold line represents the AR inference on the ground truth attribute.

with balanced preferences (third column), our framework infers it correctly, indicating the framework is capable of recognising the agent's neutrality. The agent with weaker preferences (second and fourth columns) takes longer to infer, as the agent prioritises the item with the slightly higher reward first. This is shown by the first 30 steps where our framework infers that the agent might have a strong preference. As timesteps progress, the agent gradually reveals its slight preference, and our framework adjusts its inference to agent having weaker preferences around timestep 50.

Overall, the AR framework effectively infers preferences: quicker, confident predictions for strong or balanced preferences; and slightly delayed but accurate predictions for weaker preferences.

Belief. In this scenario, the reward weighting is set to 0.7 for iron and either 1 or -1 for wood and grass. We designed two types of agents: one with full observability and one with partial observability, where wood and grass are indistinguishable. The goal is to train policies that account for these differences in perception.

Figure 8 shows that for fully observable agents, our framework's inference converges quickly with high confidence, as it can correctly identify wood and grass within the first 20 timesteps. This behaviour is similar to the strong preference scenarios, where the framework confidently infers actions due to the clarity of the observations.

In the case of partially observable agents, the results indicate that our framework is able to infer the agent's inability to distinguish between wood and grass. This is reflected in the rapid decline in probabilities for fully observable policies. However, the confidence in partially observable policies does not necessarily indicate framework's accuracy in identifying whether wood or grass is good. Instead, the results stem from slight differences in trained policies for partial observability. These policies behave differently when wood is good versus when grass is good, even though the items are indistinguishable to the agent. Theoretically, the BI probabilities for the red and green lines should hover around 0.5, as the agent's actions should reflect the uncertainty between the two items. The divergence in probabilities is a result of these minor variations in the trained policies rather than the uncertainty of the environment.

Overall, the results show the AR framework's ability to adapt its inference process based on the agent's level of observability.

4.3 Enabling Scalability

We revisit the scalability issue highlighted in Section 3.2. Training a separate neural network for each possible instance of an attribute is not scalable. For example, in our preference experiment in Section 4.2, we trained five different policies for five different weight combinations. However, these combinations do not cover every possible preference, and training additional policies would be time-consuming. This challenge becomes even more difficult when dealing with multiple attribute types. Therefore, in this experiment, we aim to determine whether our AR framework can scale to handle fine-grained instances of attributes.

To address this issue, we propose a training stage that can learn a single Q-function capable of generalising across different instances of the attribute α. Instead of only taking state and action as inputs, we modify the Q-function to also take α as an additional parameter. The modified Q-function can be defined as $\hat{Q}(s, a, \alpha)$, representing the expected rewards for future steps when taking action a in state s, given attribute α. This modification allows the Q-function to generalise across different attributes α.

For the experiment, we use the same domain as Section 4.2, focusing on inferring the preference attribute between cloth and stick. We use DQN [17] to approximate the Q-functions and set randomised reward weightings (e.g. 0.35 for cloth and 0.65 for stick) that sum to 1 for each training episode.

During online inference, we use the following reward weighting combinations for the recogniser: (0.1, 0.9), (0.2, 0.8), (0.3, 0.7), (0.4, 0.6), (0.5, 0.5), (0.6, 0.4), (0.7, 0.3), (0.8, 0.2), (0.9, 0.1). We test two types of agents: one using the modified Q-function $\hat{Q}(s, a, \alpha)$, one using the standard Q-function $Q_\alpha(s, a)$. Both agent executes in the environment using this Q-functions reflecting the following weight combinations: (0.1, 0.9), (0.3, 0.7), (0.5, 0.5), (0.7, 0.3), (0.9, 0.1).

4.3.1 Results. Figures 9 and 10 show that our framework, when using the updated $\hat{Q}(s, a, \alpha)$, performs well in inferring agent preferences. When compared to agents trained with the modified Q-function $\hat{Q}(s, a, \alpha)$, the framework accurately distinguishes all preferences. For agents using the standard Q-function $Q(s, a)$, performance is slightly worse, particularly for weak preferences, though the correct preference weights still rank in the top three inferences.

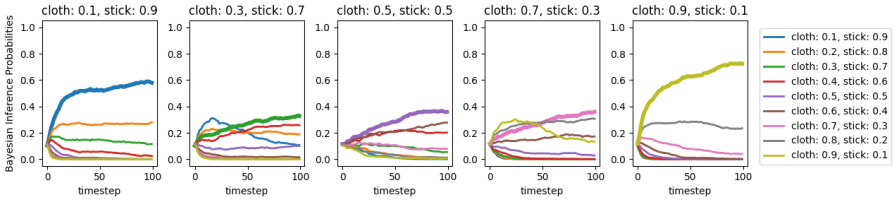


Figure 9: BI probabilities[3] for preferences with AR framework trained using the modified Q-function. Each column represents an agent's policy trained with *modified* Q-function w.r.t. a ground truth attribute.

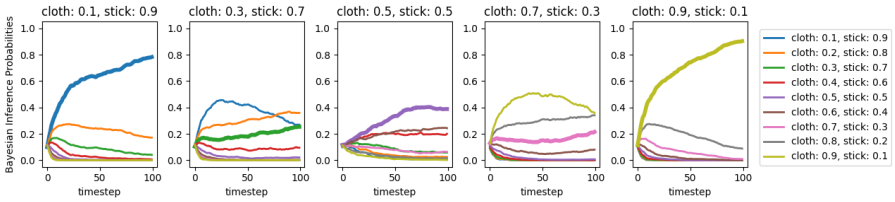


Figure 10: BI probabilities[3] for preferences with AR framework trained using the modified Q-function. Each column represents an agent's policy trained with *standard* Q-function w.r.t. a ground truth attribute.

Overall, these results demonstrate that the AR framework with the modified Q-function $\hat{Q}(s, a, \alpha)$ offers promising improvements in the efficiency of the offline training stage and shows strong potential for scalability as problem complexity increases.

5 RELATED WORK

Beyond GR, the concept of inferring agent attributes has been explored and applied across various fields. Agent modelling, a broader research area, encompasses a range of approaches aimed at understanding agent behaviour in different contexts [1]. While GR primarily focuses on inferring goals from observed behaviour, other agent modelling approaches often seek to uncover deeper behavioural attributes, such as preferences, beliefs, and intentions [6, 18].

Additionally, a significant body of research in agent modelling has focused on inference in competitive settings, such as Opponent Modelling [13, 27] and Multi-Agent RL [7]. However, our work is distinct in that we model an agent's decision-making based solely on observation, without requiring direct interaction with the agent.

Recent advancements in agent modelling have been inspired by Theory of Mind (ToM) [4], a human cognitive ability to infer another's mental state, including beliefs, desires, and intentions. ToM allows individuals to adopt another's perspective. This concept has motivated research in AI, where efforts have been made to develop a "machine mind", a model structured on the human ToM framework [23]. Many studies have employed techniques like Bayesian inverse planning [5, 6, 11] and inverse reinforcement learning [14, 26] to infer an agent's beliefs and desires by quantifying these attributes.

More recently, approaches have emerged that focus on learning an agent's latent mental state representation from its observable behaviour. This shift reduces the need to explicitly quantify attributes like beliefs and instead enables the model to infer a higher-level understanding of the agent's mental state. Notable contributions include the ToM neural network, which explores the concept of "learning how to learn" through meta-learning techniques [18].

These works possess similar questions and motivations in mind: how can we model an agent in observation and make a robust inference of their "mind" from their actions. Our approach is similar to [5, 6] in a way that we quantify an attribute to help with inference.

6 CONCLUSION AND FUTURE DIRECTIONS

In this work, we introduced a framework for Attribute Recognition (AR), which extends traditional Goal Recognition (GR) to infer other agent attributes such as ability, preferences, and beliefs. Through our RL-based approach, we demonstrated how these attributes can be inferred from observed agent behaviour through modifying the learning task for the RL problem, enabling a more comprehensive understanding of agents behaviour. Our framework was tested in two deterministic environments, the Treasure domain and the Craft World domain, to illustrate its effectiveness in handling both simple and complex behaviours. Our experiments show that inferring these attributes improves the accuracy of GR and highlights the broader potential of our framework in various settings.

Future work could explore different methods for modeling agent attributes in complex multi-agent environments. For example, preference based RL reduces the need for numeric rewards through pairwise comparisons [24, 25]. Overall, the potential for extending this framework to a variety of agent attributes is vast, and further research will be crucial to explore the full range of possibilities.

REFERENCES

[1] Stefano V. Albrecht and Peter Stone. 2018. Autonomous agents modelling other agents: A comprehensive survey and open problems. *Artificial Intelligence* 258 (2018), 66–95.
[2] Leonardo Amado, Reuth Mirsky, and Felipe Meneguzzi. 2022. Goal Recognition as Reinforcement Learning. *Proceedings of the AAAI Conference on Artificial Intelligence* 36, 9 (Jun. 2022), 9644–9651.
[3] Leonardo Amado, Sveta Paster Shainkopf, Ramon Fraga Pereira, Reuth Mirsky, and Felipe Meneguzzi. 2024. A Survey on Model-Free Goal Recognition. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, Kate Larson (Ed.). International Joint Conferences on Artificial Intelligence Organization, 7923–7931. Survey Track.
[4] Janet Wilde Astington and Jennifer M Jenkins. 1995. Theory of mind development and social understanding. *Cognition & Emotion* 9 (1995), 151–165.
[5] Chris L. Baker, Julian Jara-Ettinger, Rebecca Saxe, and Joshua B. Tenenbaum. 2017. Rational quantitative attribution of beliefs, desires and percepts in human mentalizing. *Nature Human Behaviour* 1, 4 (March 2017), 0064.

[6] Chris L. Baker, Rebecca Saxe, and Joshua B. Tenenbaum. 2011. Bayesian Theory of Mind: Modeling Joint Belief-Desire Attribution. *Proceedings of the Annual Meeting of the Cognitive Science Society* 33 (2011).

[7] Lucian Buşoniu, Robert Babuka, and Bart De Schutter. 2008. A Comprehensive Survey of Multiagent Reinforcement Learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 38 (2008), 156–172.

[8] Mattia Chiari, Alfonso Emilio Gerevini, Francesco Percassi, Luca Putelli, Ivan Serina, and Matteo Olivato. 2023. Goal Recognition as a Deep Learning Task: The GRNet Approach. *Proceedings of the International Conference on Automated Planning and Scheduling* 33, 1 (Jul. 2023), 560–568.

[9] Michael Dann, Yuan Yao, Natasha Alechina, Brian Logan, Felipe Meneguzzi, and John Thangarajah. 2023. Multi-Agent Intention Recognition and Progression. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, Edith Elkind (Ed.). International Joint Conferences on Artificial Intelligence Organization, 91–99. Main Track.

[10] Arpad E Elo. 1967. The proposed uscf rating system, its development, theory, and applications. *Chess Life* 22, 8 (1967), 242–247.

[11] Owain Evans, Andreas Stuhlmüller, and Noah D. Goodman. 2016. Learning the preferences of ignorant, inconsistent agents *(AAAI'16)*. AAAI Press, 323–329.

[12] Zihao Fang, Dejun Chen, Yunxiu Zeng, Tao Wang, and Kai Xu. 2023. Real-Time Online Goal Recognition in Continuous Domains via Deep Reinforcement Learning. *Entropy* 25 (2023).

[13] He He, Jordan Boyd-Graber, Kevin Kwok, and Hal Daumé, III. 2016. Opponent Modeling in Deep Reinforcement Learning. In *Proceedings of The 33rd International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 48)*, Maria Florina Balcan and Kilian Q. Weinberger (Eds.). PMLR, New York, New York, USA, 1804–1813.

[14] Julian Jara-Ettinger. 2019. Theory of mind as inverse reinforcement learning. *Current Opinion in Behavioral Sciences* 29 (2019), 105–110.

[15] Henry A. Kautz and James F. Allen. 1986. Generalized plan recognition. In *Proceedings of the Fifth AAAI National Conference on Artificial Intelligence* (Philadelphia, Pennsylvania) *(AAAI'86)*. AAAI Press, 32–37.

[16] Felipe Meneguzzi and Ramon Fraga Pereira. 2021. A Survey on Goal Recognition as Planning. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, Zhi-Hua Zhou (Ed.). International Joint Conferences on Artificial Intelligence Organization, 4524–4532. Survey Track.

[17] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Kirkeby Fidjeland, Georg Ostrovski, Stig Petersen, Charlie Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. *Nature* 518 (2015), 529–533.

[18] Neil Rabinowitz, Frank Perbet, Francis Song, Chiyuan Zhang, S. M. Ali Eslami, and Matthew Botvinick. 2018. Machine Theory of Mind. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, 4218–4227.

[19] Miquel Ramírez and Hector Geffner. 2009. Plan recognition as planning. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence* (Pasadena, California, USA) *(IJCAI'09)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1778–1783.

[20] Miquel Ramírez and Hector Geffner. 2010. Probabilistic plan recognition using off-the-shelf classical planners. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence* (Atlanta, Georgia) *(AAAI'10)*. AAAI Press, 1121–1126.

[21] Shirin Sohrabi, Anton V. Riabov, and Octavian Udrea. 2016. Plan recognition as planning revisited. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence* (New York, New York, USA) *(IJCAI'16)*. AAAI Press, 3258–3264.

[22] Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement Learning: An Introduction* (2 ed.). MIT Press.

[23] Franz A. Van-Horenbeke and Angelika Peer. 2021. Activity, Plan, and Goal Recognition: A Review. *Frontiers in Robotics and AI* 8 (2021).

[24] Christian Wirth, Riad Akrour, Gerhard Neumann, and Johannes Fürnkranz. 2017. A Survey of Preference-Based Reinforcement Learning Methods. *J. Mach. Learn. Res.* 18 (2017), 136:1–136:46.

[25] Christian Wirth, Johannes Fürnkranz, and Gerhard Neumann. 2016. Model-free preference-based reinforcement learning. *Proceedings of the AAAI Conference on Artificial Intelligence* 30, 1 (2016), 2222–2228.

[26] Haochen Wu, Pedro Sequeira, and David V. Pynadath. 2023. Multiagent Inverse Reinforcement Learning via Theory of Mind Reasoning. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems* (London, United Kingdom) *(AAMAS '23)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 708–716.

[27] Yijie Zhang, Roxana Rădulescu, Patrick Mannion, Diederik M. Roijers, and Ann Nowé. 2020. Opponent Modelling for Reinforcement Learning in Multi-Objective Normal Form Games. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems* (Auckland, New Zealand) *(AAMAS '20)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 2080–2082.