Knowledge-based Reasoning and Learning under Partial Observability in Ad Hoc Teamwork

Dodampegama, Hasra; Sridharan, Mohan

DOI:
10.1017/S1471068423000091

Document Version
Publisher's PDF, also known as Version of record

Citation for published version (Harvard):
Dodampegama, H & Sridharan, M 2023, 'Knowledge-based Reasoning and Learning under Partial Observability in Ad Hoc Teamwork', *Theory and Practice of Logic Programming*, pp. 1-19.
https://doi.org/10.1017/S1471068423000091

TPLP: Page 1–19.

doi:10.1017/S1471068423000091

Knowledge-based Reasoning and Learning under Partial Observability in Ad Hoc Teamwork

HASRA DODAMPEGAMA and MOHAN SRIDHARAN
Intelligent Robotics Lab, School of Computer Science, University of Birmingham, UK
(*e-mails:* hhd968@student.bham.ac.uk, m.sridharan@bham.ac.uk)

submitted 02 June 2023; accepted 05 June 2023

Abstract

Ad hoc teamwork (AHT) refers to the problem of enabling an agent to collaborate with teammates without prior coordination. State of the art methods in AHT are *data-driven*, using a large labeled dataset of prior observations to model the behavior of other agent *types* and to determine the ad hoc agent's behavior. These methods are computationally expensive, lack transparency, and make it difficult to adapt to previously unseen changes. Our recent work introduced an architecture that determined an ad hoc agent's behavior based on non-monotonic logical reasoning with prior commonsense domain knowledge and models learned from limited examples to predict the behavior of other agents. This paper describes KAT, a knowledge-driven architecture for AHT that substantially expands our prior architecture's capabilities to support: (a) online selection, adaptation, and learning of the behavior prediction models; and (b) collaboration with teammates in the presence of partial observability and limited communication. We illustrate and experimentally evaluate KAT's capabilities in two simulated benchmark domains for multiagent collaboration: Fort Attack and Half Field Offense. We show that KAT's performance is better than a purely knowledge-driven baseline, and comparable with or better than a state of the art data-driven baseline, particularly in the presence of limited training data, partial observability, and changes in team composition.

KEYWORDS: knowledge representation, non-monotonic logical reasoning, ecological rationality, ad hoc teamwork, applications of logic programming.

1 Introduction

Ad hoc teamwork (AHT) is the challenge of enabling an agent (called the *ad hoc agent*) to collaborate with previously unknown teammates toward a shared goal (Stone *et al.* 2010). As motivating examples, consider the simulated multiagent domain *Fort Attack* (FA, Figure 1a), where a team of guards has to protect a fort from a team of attackers (Deka and Sycara 2021), and the *Half Field Offense* domain (HFO, Figure 1d), where a team of offense agents has to score a goal against a team of defenders that includes a goalkeeper (Hausknecht *et al.* 2016). Agents in these domains have limited knowledge of each other's capabilities, no prior experience of working as a team, limited ability to observe the environment (Figure 1b), and limited bandwidth for communication. Such

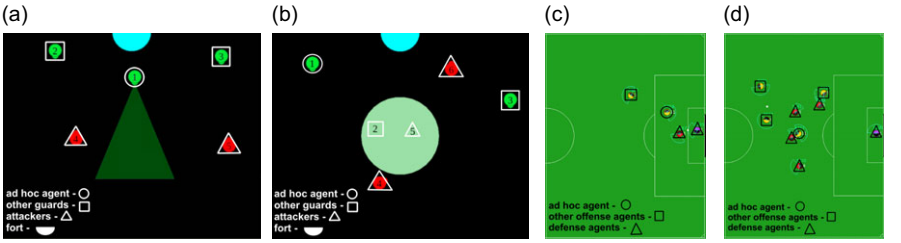


Fig. 1. Screenshots: (a, b) fort attack environment; (c, d) half-field offense environment.

scenarios are representative of multiple practical application domains such as disaster rescue and surveillance.

The state of the art in AHT has transitioned from the use of predetermined policies for selecting actions in specific states to the use of a key "data-driven" component. This component uses probabilistic or deep network methods to model the behavior (i.e., action choice in specific states) of other agents or agent types, and to optimize the ad hoc agent's behavior. These methods use a long history of prior experiences in different scenarios, and of the interactions with these agent types, as training examples. It is difficult to obtain such training examples in complex domains, and computationally expensive to build the necessary models or to revise them in response to new situations. At the same time, just reasoning with prior knowledge will not allow the ad hoc agent to accurately anticipate the behavior of other agents and it is not possible to encode comprehensive knowledge about all possible situations. In a departure from existing work, we pursue a *cognitive systems* approach, which recognizes that AHT jointly poses representation, reasoning, and learning challenges, and seeks to leverage the complementary strengths of knowledge-based reasoning and data-driven learning from limited examples. Specifically, our knowledge-driven AHT architecture (KAT) builds on knowledge representation (KR) tools to support:

1. Non-monotonic logical reasoning with prior commonsense domain knowledge and rapidly-learned predictive models of other agents' behaviors;
2. Use of reasoning and observations to trigger the selection and adaptation of relevant agent behavior models, and the learning of new models as needed; and
3. Use of reasoning to guide collaboration with teammates under partial observability.

In this paper, we build on and significantly extend our recent work that demonstrated just the first capability (listed above) in the FA domain (Dodampegama and Sridharan 2023a). We use Answer Set Prolog (ASP) for non-monotonic logical reasoning, and heuristic methods based on ecological rationality principles (Gigerenzer 2020) for rapidly learning and revising agents' behavior models. We evaluate KAT's capabilities in the FA domain and the more complex HFO domain. We demonstrate that KAT's performance is better than that of just the non-monotonic logical reasoning component, and is comparable or better than state of the art data-driven methods, particularly in the presence of partial observability and changes in team composition.

https://doi.org/10.1017/S1471068423000091 Published online by Cambridge University Press

2 Related work

Methods for AHT have been developed under different names and in different communities over many years, as described in a recent survey (Mirsky *et al.* 2022). Early work used specific protocols ('plays') to define how an agent should behave in different scenarios (states) (Bowling and McCracken 2005). Subsequent work used sample-based methods such as Upper Confidence bounds for Trees (UCT) (Barrett *et al.* 2013), or combined UCT with methods that learned models from historical data for online planning (Wu *et al.* 2011). More recent methods have included a key data-driven component, using probabilistic, deep network, and reinforcement learning (RL)-based methods to learn action (behavior) choice policies for different *types* of agents based on a long history of prior observations of similar agents or situations (Barrett *et al.* 2017; Rahman *et al.* 2021). For example, RL methods have been used to choose the most useful policy (from a set of learned policies) to control the ad hoc agent in each situation (Barrett *et al.* 2017), or to consider predictions from learned policies when selecting an ad hoc agent's actions for different types of agents (Santos *et al.* 2021). Attention-based deep neural networks have been used to jointly learn policies for different agent types (Chen *et al.* 2020) and for different team compositions (Rahman *et al.* 2021). Other work has combined sampling strategies with learning methods to optimize performance (Zand *et al.* 2022). There has also been work on using deep networks to learn sequential and hierarchical models that are combined with approximate belief inference methods to achieve teamwork under ad hoc settings (Zintgraf *et al.* 2021).

Researchers have explored different communication strategies for AHT. Examples include a multiagent, multi-armed bandit formulation to broadcast messages to teammates at a cost (Barrett *et al.* 2017), and a heuristic method to assess the cost and value of different queries to be considered for communication (Macke *et al.* 2021). These methods, similar to the data-driven methods for AHT discussed above, require considerable resources (e.g., computation, training examples), build opaque models, and make it difficult to adapt to changes in team composition.

There has been considerable research in developing action languages and logics for single- and multiagent domains. This includes action language $\mathcal{A}$ for an agent computing cooperative actions in multiagent domains (Son and Sakama 2010), and action language $\mathcal{C}$ for modeling benchmark multiagent domains with minimal extensions (Baral *et al.* 2010). Action language $\mathcal{B}$ has also been combined with Prolog and ASP to implement a distributed multiagent planning system that supports communication in a team of collaborative agents (Son *et al.* 2010). More recent work has used $\mathcal{B}$ for planning in single agents and multiagent teams, including a distributed approach based on negotiations for noncooperative or partially collaborative agents (Son and Balduccini 2018). To model realistic interactions and revise the domain knowledge of agents, researchers have introduced specific action types, for example, world altering, sensing, and communication actions (Baral *et al.* 2010). Recent work has represented these action types in action language m$\mathcal{A}*$ while also supporting epistemic planning and dynamic awareness of action occurrences (Baral *et al.* 2022). These studies have demonstrated the expressive power and reasoning capabilities that logics in general, and non-monotonic logics such as ASP in particular, provide in the context of multiagent systems. Our work draws on these findings to address the reasoning and learning challenges faced by an ad hoc agent that

https://doi.org/10.1017/S1471068423000091 Published online by Cambridge University Press

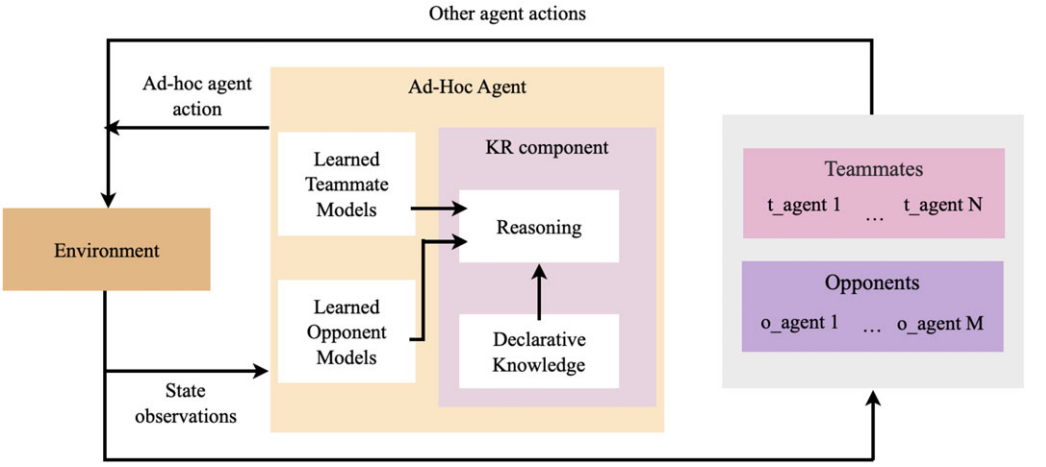


Fig. 2. Our KAT architecture combines complementary strengths of knowledge-based and data-driven heuristic reasoning and learning.

has to collaborate with teammates without any prior coordination, and under conditions of partial observability and limited communication.

3 Architecture

Figure 2 provides an overview of our KAT architecture. KAT enables an ad hoc agent to perform non-monotonic logical reasoning with prior commonsense domain knowledge, and with incrementally learned behavior models of teammate and opponent agents. At each step, valid observations of the domain state are available to all the agents. Each agent uses these observations to independently determine and execute its individual actions in the domain. The components of KAT are described below using the following two example domains.

Example Domain 1: Fort attack (FA). Three guards are protecting a fort from three attackers. One guard is the ad hoc agent that can adapt to changes in the domain and team composition. An episode ends if: (a) guards manage to protect the fort for a period of time; (b) all members of a team are eliminated; or (c) an attacker reaches the fort.

At each step, each agent can move in one of the four cardinal directions with a particular velocity, turn clockwise or anticlockwise, do nothing, or shoot to kill any agent of the opposing team that is within its shooting range. The environment provides four types of built-in policies for guards and attackers (see Section 4.1). The original FA domain is fully observable, that is, each agent knows the state of other agents at each step. We simulate partial observability by creating a "*forest*" in Figure 1b; any agent in this region is hidden from others.

Example Domain 2: Half field offense (HFO). This simulated 2D soccer domain is a complex benchmark for multiagent systems and AHT (Hausknecht *et al.* 2016). Each game (i.e., episode) is essentially played in one half of the field. The ad hoc agent is one of the members of the offense team that seeks to score a goal against agents in the team defending the goal. An episode ends when the: (a) offense team scores a goal; (b) ball

https://doi.org/10.1017/S1471068423000091 Published online by Cambridge University Press

leaves field; (c) defense team captures the ball; or (d) maximum episode length (500) is reached.

There are two versions of the HFO domain: (i) *limited*: with two offense agents and two defense agents (including the goalkeeper); and (ii) *full*: with four offense agents and five defense agents (including the goalkeeper). Similar to prior AHT methods, agents other than the ad hoc agent are selected from teams created in the RoboCup 2D simulation league competitions. Specifically, other offense team agents are based on the binary files of five teams: *helios, gliders, cyrus, axiom, aut.* For defenders, we use *agent2D* agents, whose policy was derived from *helios*. The strategies of these agent types were trained using data-driven (probabilistic, deep, reinforcement) learning methods. HFO supports two state space abstractions: low and high; we use the high-level features. In addition, there are three abstractions of the action space: primitive, mid-level, and high-level; we use a combination of mid-level and high-level actions. This choice of representation was made to facilitate comparison with existing work.

Prior commonsense knowledge in these two domains includes relational descriptions of some domain attributes (e.g., safe regions), agent attributes (e.g., location), default statements, and axioms governing change in the domain, for example, an agent can only move to a location nearby, only shoot others within its range (FA), and only score a goal from a certain angle (HFO). Specific examples of this knowledge are provided later in this section. Although this knowledge may need to be revised over time in response to changes in the domain, we do not explore knowledge acquisition and revision in this paper; for related work by others in our group, please see the papers by Sridharan and Mota (2023) and Sridharan and Meadows (2018).

3.1 Knowledge representation and reasoning

In KAT, the transition diagrams of any domains are described in an extension of the action language $\mathcal{AL}_d$ (Gelfond and Inclezan 2013). Action languages are formal models of parts of natural language that are used to describe the transition diagrams of any given dynamic domain. The domain representation comprises a system description $\mathcal{D}$, a collection of statements of $\mathcal{AL}_d$, and a history $\mathcal{H}$. $\mathcal{D}$ has a sorted signature Σ which consists of *actions*, *statics*, that is, domain attributes whose values cannot be changed, and *fluents*, that is, domain attributes whose values can be changed by actions. For example, Σ in the HFO domain includes basic sorts such as *ad_hoc_agent*, *external_agent*, *agent*, *offense_agent*, *defense_agent*, *x_val*, *y_val*, and sort *step* for temporal reasoning. Sorts are organized hierarchically, with some sorts, for example, *offense_agent* and *defense_agent*, being subsorts of others, for example, *external_agent*. Statics in Σ are relations such as $next\_to(x\_val, y\_val, x\_val, y\_val)$ that encode the relative arrangement of locations (in the HFO domain). The fluents in Σ include *inertial* fluents that obey inertia laws and can be changed by actions, and *defined* fluents that do not obey inertia laws and are not changed directly by actions. For example, inertial fluents in the HFO domain include

$$\begin{aligned} &loc(ad\_hoc\_agent, x\_val, y\_val) \\ &ball\_loc(x\_val, y\_val) \\ &has\_ball(agent), \end{aligned} \tag{1}$$

https://doi.org/10.1017/S1471068423000091 Published online by Cambridge University Press

which describe the location of the ad hoc agent, location of the ball, and the agent that has control of the ball; the value of these attributes changes as a direct consequence of executing specific actions. Defined fluents of the HFO domain include

$$\begin{aligned} &agent\_loc(external\_agent, x\_val, y\_val) \\ &defense\_close(agent, defense\_agent) \\ &far\_from\_goal(ad\_hoc\_agent), \end{aligned} \tag{2}$$

which encode the location of the external (i.e., non-ad hoc) agents, and describe whether a defense agent is too close to another agent, and whether the ad hoc agent is far from the goal. Note that the ad hoc agent has no control over the value of these fluents, although the hoc agent's actions can influence the value of these fluents. Next, actions in the HFO domain include

$$\begin{aligned} &move(ad\_hoc\_agent, x\_val, y\_val) \\ &kick\_goal(ad\_hoc\_agent) \\ &dribble(ad\_hoc\_agent, x\_val, y\_val) \\ &pass(ad\_hoc\_agent, offense\_agent), \end{aligned} \tag{3}$$

which state the ad hoc agent's ability to move to a location, kick the ball toward the goal, dribble the ball to a location, and pass the ball to a teammate. Next, axioms in $\mathcal{D}$ describe domain dynamics using elements in Σ in three types of statements: causal laws, state constraints, and executability conditions. For the HFO domain, this includes statements such as:

$$move(R, X, Y) \textbf{ causes } loc(R, X, Y), \tag{4a}$$

$$dribble(R, X, Y) \textbf{ causes } ball\_loc(X, Y), \tag{4b}$$

$$\neg has\_ball(A1) \textbf{ if } has\_ball(A2),\ A1 \neq A2, \tag{4c}$$

$$\textbf{impossible } shoot(R) \textbf{ if } far\_from\_goal(R). \tag{4d}$$

where Statements 4(a-b) are causal laws that specify that moving and dribbling change the ad hoc agent's and ball's location (respectively) to the desired location. Statement 4(c) is a state constraint that implies that only one agent can control the ball at any time. Statement 4(d) is an executability condition that prevents the consideration of a shooting action (during planning) if the ad hoc agent is far from the goal. Finally, the history $\mathcal{H}$ is a record of observations of fluents at particular time steps, that is, $obs(fluent, boolean, step)$, and of action execution at particular time steps, that is, $hpd(action, step)$. It also includes initial state defaults, that is, statements in the initial state that are believed to be true in all but a few exceptional circumstances, for example, the following $\mathcal{AL}_d$ statement implies that attackers in the FA domain usually spread and attack the fort:

$$\textbf{initial default } spread\_attack(X) \textbf{ if } attacker(X). \tag{5}$$

To enable an ad hoc agent to reason with prior knowledge, the domain description in $\mathcal{AL}_d$ is automatically translated to program $\Pi(\mathcal{D}, \mathcal{H})$ in CR-Prolog (Balduccini and Gelfond 2003), an extension to ASP that supports consistency restoring (CR) rules. ASP is based on stable model semantics and represents constructs difficult to express in classical logic

https://doi.org/10.1017/S1471068423000091 Published online by Cambridge University Press

formalisms. It encodes concepts such as *default negation* and *epistemic disjunction*, and supports non-monotonic reasoning; this ability to revise previously held conclusions is essential in AHT. $\Pi(\mathcal{D},\mathcal{H})$ incorporates the relation $holds(fluent, step)$ to state that a particular fluent is true at a given step, and $occurs(action, step)$ to state that a particular action occurs in a plan at a given step. It includes the signature and axioms of $\mathcal{D}$, inertia axioms, reality checks, closed world assumptions for defined fluents and actions, observations, actions, and defaults from $\mathcal{H}$, and a CR rule for every default allowing the agent to assume that the default's conclusion is false in order to restore consistency under exceptional circumstances. For example, the CR-Prolog statement:

$$\neg spread\_attack(X) \stackrel{+}{\leftarrow} attacker(X), \tag{6}$$

allows the ad hoc agent to consider the rare situation of attackers mounting a frontal attack. Furthermore, it includes helper axioms, for example, to define goals and drive diagnosis. Reasoning tasks such as planning, diagnosis, and inference are then reduced to computing answer sets of Π. The ad hoc agent may need to prioritize different goals at different times, for example, score a goal when it has control of the ball, and position itself at a suitable location otherwise:

$$\begin{aligned} &goal(I) \leftarrow holds(scored\_goal, I) \\ &goal(I) \leftarrow holds(loc(ad\_hoc\_agent, X, Y), I). \end{aligned} \tag{7}$$

A suitable goal is selected and included in $\Pi(\mathcal{D},\mathcal{H})$ automatically at run-time. In addition, heuristics are encoded to direct the search for plans, for example, the following statements:

$$\begin{aligned} &total(S) \leftarrow S = sum\{C, A : occurs(A, I), cost(A, C)\} \\ &\#minimize\{S@p, S : total(S)\}. \end{aligned} \tag{8}$$

encourage the ad hoc agent to select actions that will minimize the total cost when computing action sequences to achieve a particular goal. We use the SPARC system (Balai *et al.* 2013) to write and solve CR-Prolog programs. For computational efficiency, these examples programs build on a refinement-based architecture that represents and reasons with knowledge at two tightly coupled resolutions, with a fine-resolution description ($\mathcal{D}_F$) defined as a refinement of a coarse-resolution description ($\mathcal{D}_C$). For example, the available space in the FA domain and HFO domain is organized into abstract regions in $\mathcal{D}_C$, with each region being refined in $\mathcal{D}_F$ into small grids that are components of this region. $\mathcal{D}_C$ and $\mathcal{D}_F$ are formally coupled through component relations and bridge axioms such as:

$$\begin{aligned} &loc^*(A, Rg) \ \textbf{if} \ \ loc(R, X, Y), \ component(X, Y, Rg) \\ &next\_to^*(Rg_2, Rg_1) \ \textbf{if} \ \ next\_to^*(Rg_1, Rg_2), \end{aligned} \tag{9}$$

where location (X, Y) is in region Rg and superscript "*" refers to relations in $\mathcal{D}_C$. This coupling between the descriptions enables the ad hoc agent to automatically choose the relevant part of the relevant description based on the goal or abstract action, and to transfer relevant information between the descriptions. Example programs are in our repository (Dodampegama and Sridharan 2023b); details of the refinement-based architecture are in the paper by Sridharan *et al.* (2019).

https://doi.org/10.1017/S1471068423000091 Published online by Cambridge University Press

Table 1. *Attributes considered for models of other agents' behavior in FA domain. Number of attributes represent the* number of variables *in each attribute times the* number of agents

Description of attribute	Number
x, y position of agent	12
Distance from agent to center of field	6
Agents' polar angle with center of field	6
Orientation of the agent	6
Distance from agent to fort	6
Distance to nearest attacker from fort	1
Number of attackers not alive	1
Previous action of the agent	1

Table 2. *Attributes for models of teammates and defense agents' behavior in HFO domain. Number of attributes represent the* number of variables *in each attribute times the* number of agents

Description of attribute	Number	Description of attribute	Number
x position of agent	4	x position of agent	4
y position of agent	4	y position of agent	4
Goal opening angle	2	x position of the ball	1
Proximity to the nearest opponent	2	y position of the ball	1
x position of the ball	1		
y position of the ball	1		

3.2 Agent models and model selection

Since reasoning with just prior domain knowledge can lead to poor team performance under AHT settings (see Section 4.2), KAT enables the ad hoc agent to also reason with models that predict (i.e., anticipate) the action choices of other agents. State of the methods attempt to optimize performance in different (known or potential) situations by learning models offline from many (e.g., hundred thousands or millions of) examples. It is intractable to obtain such labeled examples of different situations in complex domains, and the learned models are truly useful only if they can be learned and revised rapidly during run-time to account for previously unknown situations. KAT thus chooses relevant attributes for models that can be: (a) learned from limited (e.g., 10,000) training examples acquired from simple hand-crafted policies (e.g., spread and shoot in FA, pass when possible in HFO); and (b) revised rapidly during run-time to provide reasonable accuracy. Tables 1 and 2 list the identified attributes in the FA and HFO domain respectively.

Similar to our recent work (Dodampegama and Sridharan 2023a), the attributes are identified and the predictive models are learned using the *Ecological Rationality* (ER) approach, which draws on insights from human cognition, Herb Simon's definition of *Bounded Rationality*, and an algorithmic model of heuristics (Gigerenzer 2020; Gigerenzer and Gaissmaier 2011). ER focuses on decision making under true uncertainty (e.g., in

https://doi.org/10.1017/S1471068423000091 Published online by Cambridge University Press

open worlds), characterizes behavior as a joint function of internal (cognitive) processes and the environment, and focuses on *satisficing* based on differences between observed and predicted behavior. Also, heuristic methods are viewed as a strategy to ignore part of the information in order to make decisions more quickly, frugally, and/or accurately than complex methods. In addition, it advocates the use of an adaptive toolbox of classes of heuristics (e.g., one-reason, sequential search, lexicographic), and comparative out-of-sample testing to identify heuristics that best leverage the target domain's structure. This approach has provided good performance in many applications (Gigerenzer 2016).

Specifically, in KAT, ER principles such as abstraction and refinement, and statistical attribute selection methods, are applied to the set of 10,000 samples to identify the key attributes and their representation in Tables 1 and 2; these define behavior in the FA domain and HFO domain respectively. The coarse- and fine-resolution representation described in Section 3.1 is an example of the principle of refinement. In addition to the choice of features, the characteristic factors of AHT, for example, the need to make rapid decisions under resource constraints and respond to dynamic changes with limited examples, are matched with the toolbox of heuristics to identify and use an ensemble of "fast and frugal" (FF) decision trees to learn the behavior prediction models for each type of agent. Each FF tree in an ensemble focuses on one valid action, provides a binary class label, and has the number of leaves limited by the number of attributes (Katsikopoulos *et al.* 2021). Figure 3 shows an example of a FF tree learned (as part of the corresponding ensemble) for a guard agent (Figure 3a) and an attacker agent (Figure 3b) in the FA domain.

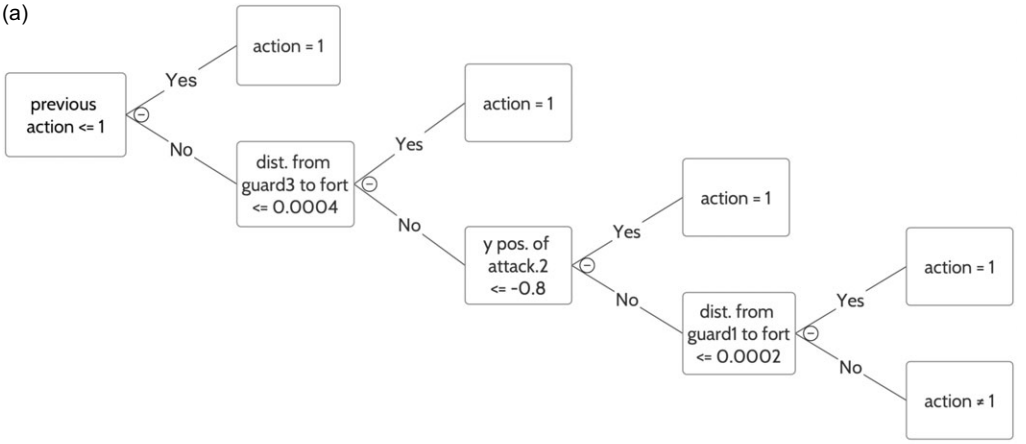
The ad hoc agent's teammates and opponents may include different types of agents whose behavior may change over time. *Unlike our prior work that used static prediction models, we enabled the ad hoc agent to respond to such changes by automatically revising the current model, switching to a relevant model, or learning new models.* Existing models are revised by changing the parameters of the FF trees, and Algorithm 1 is an example of our approach for selecting a suitable model in the context of predicting the pose (i.e., position and orientation) of agents. Specifically, the ad hoc agent periodically compares the existing models' predictions with the observed action choices of each agent (teammate, opponent) over a sliding window of domain state and the agents' action choices; in Algorithm 1, this window is of size 1 (Lines 4–5). Also, a graded strategy is used to compute the error, penalizing differences in orientation less than differences in position (Lines 6–7). The model whose predictions best match the observations is selected for subsequent use and revision (Line 10, Algorithm 1). Note that if none of the models provide a good match over multiple steps, this acts as a trigger to learn a new model.

3.3 Partial observability and communication

In practical AHT domains, any single agent cannot observe the entire domain and communication is a scarce resource. To explore the interplay between partial observability and communication, we modified the original domains. Specifically, in the FA domain, we introduced a *forest* region where attackers can hide from the view of the two guards other than ad hoc agent and secretly approach the fort—see Figure 1b. The ad hoc agent has visibility of the forest region; it can decide when to communicate with its teammates, for example, when: (a) one or more attackers are hidden in the forest; and (b) one of the

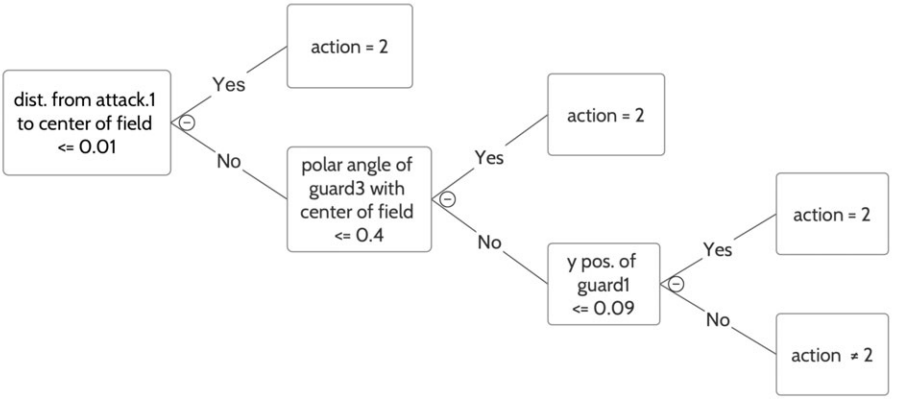
https://doi.org/10.1017/S1471068423000091 Published online by Cambridge University Press

(a)



One FF tree in the ensemble for a guard in the FA domain.

(b)



One FF tree in the ensemble for an attacker in the FA domain.

Fig. 3. Examples of FF trees for a guard and an attacker in the FA domain.

other guards is closer to the hidden attacker(s) than it. The associated reasoning can be encoded using statements such as:

$$holds(shoots(G, AA), I+1) \leftarrow occurs(communicate(AHA, G, AA), I), \tag{10a}$$

$$holds(in\_forest(AA), I) \leftarrow holds(agent\_loc(AA, X, Y), I),\ forest(X, Y),\ not\ \ holds(shot(AA), I), \tag{10b}$$

$$-occurs(communicate(AHA, G, AA), I) \leftarrow\ not\ holds(in\_range(G, AA), I). \tag{10c}$$

where Statement 10(c) encodes that communication is used only when a hidden attacker is within the range of a teammate (i.e., guard agent); Statement 10(b) defines when an attacker is hidden; and Statement 10(a) describes the ad hoc agent's belief that a teammate receiving information about a hidden attacker will shoot it, although the teammate acts independently and may choose to ignore this information. If there are multiple guards satisfying these conditions, the ad hoc agent may only communicate with the guard closest to the hidden attacker(s).

In the HFO domain, we represent partial observability in an indirect manner using the built-in ability to limit each agent's perception to a specific viewing cone relative to

https://doi.org/10.1017/S1471068423000091 Published online by Cambridge University Press

Algorithm 1: Model Selection

Input: $\mathcal{A}$: other agents; $\mathcal{M}$: subset of behavior models; $\{a_{act}\}, \{a_{pred}\}$: actual and predicted action choices of each agent in current round of the game; *scores*: initial values (100) assigned to each agent-model combination.

Output: model: selected model for each agent.

```
for i = 0 to A do
    for m = 0 to M do
        if a_pred[i, m] ≠ a_act[i] then
            l_act, o_act ← actual_pose(a_act)
            l_pred, o_pred ← predicted_pose(a_pred)
            penalty ← abs(l_act − l_pred) + abs(o_act − o_pred)/10
            scores[i, m] = scores[i, m] − penalty
        end
    end
    model[i] = select_model(M, scores[i, *])
end
```

the agent. Specifically, each agent is only able to sense objects (e.g., other agents, ball) within its viewing cone; objects outside its viewing cone are not visible. Given this use of built-in functions, we added some helper axioms to ensure that the ad hoc agent only reasoned with visible objects; no additional communication action was implemented.

4 Experimental setup and results

We experimentally evaluated three hypotheses about KAT's capabilities:

- **H1:** KAT's performance is comparable or better than state of the art baselines in different scenarios while requiring much less training;
- **H2:** KAT enables adaptation to unforeseen changes in the type and number of other agents (teammates and opponents); and
- **H3:** KAT supports adaptation to partial observability with limited communication capabilities.

We evaluated aspects of H1 and H2 in both domains (FA, HFO) under full observability. For H3, we considered partial observability in both domains, and explored limited communication in the FA domain. Each game (i.e., episode) in the FA domain had three guards and three attackers, with our ad hoc agent replacing one of the guards. In HFO domain, each game (i.e., episode) had two offense and two defense players (including one goalkeeper) in the limited version; and four offense and five defense players (including one goalkeeper) in the full version. Our ad hoc agent replaced one of the offense agents in the HFO domain. In the FA domain, the key performance measure was the win percentage of the guards team. In the HFO domain, the key performance measure was the fraction of games in which the offense team scored a goal. In both domains, we also measured the accuracy of the predictive models. Further details of the experiments and the associated baselines are provided below.

https://doi.org/10.1017/S1471068423000091 Published online by Cambridge University Press

4.1 Experimental setup

In the **FA domain**, we used two kinds of policies for the agents other than our ad hoc agent: *hand-crafted policies* and *built-in policies*. Hand-crafted policies were constructed as simple strategies that produce basic behavior. Built-in policies were provided with the domain; they are based on graph neural networks trained using many labeled examples.

Hand-Crafted Policies.

- **Policy1:** guards stay near the fort and shoot attackers who spread and approach.
- **Policy2:** guards and attackers spread and shoot their opponents.

Built-in Policies.

- **Policy220:** guards stay in front of the fort and shoot continuously as attackers approach.
- **Policy650:** guards try to block the fort; attackers try to sneak in from all sides.
- **Policy1240:** guards spread and shoot the attackers; attackers sneak in from all sides.
- **Policy1600:** guards are willing to move away from the fort; some attackers approach the fort and shoot to distract the guards while others try to sneak in.

The ad hoc agent was evaluated in two experiments: **Exp1**, in which other agents followed the hand-crafted policies; and **Exp2**, in which other agents followed the built-in policies. As stated earlier, the ad hoc agent learned behavior models in the form of FF trees from 10,000 state-action observations obtained by running the hand-crafted policies. It was not provided any prior experience or models of the built-in policies.

Our previous work documented the accuracy of a basic AHT architecture that reasoned with some domain knowledge and static behavior prediction models in the FA domain (Dodampegama and Sridharan 2023a). In this paper, the focus is on evaluating the ability to select, revise, and learn the relevant predictive models, and adapt to partial observability. For the former, each agent other than our ad hoc agent was assigned a policy selected randomly from the available policies (described above). The baselines for this experiment were:

- **Base1:** other agents followed a random mix of hand-crafted policies. The ad hoc agent did not revise the learned behavior models or use the model selection algorithm.
- **Base2:** other agents followed a random mix of hand-crafted policies. The ad hoc agent used a model selection algorithm without a graded strategy to compare the predicted and actual actions, that is, fixed penalty assigned for action mismatch in Line 6 of Algorithm 1.
- **Base3:** other agents followed a random mix of built-in policies. The ad hoc agent did not revise the learned behavior models or use the model selection algorithm.
- **Base4:** other agents followed a random mix of built-in policies. The ad hoc agent used the model selection algorithm without a graded strategy to compare predicted and actual actions, that is, fixed penalty assigned for action mismatch in Line 6 of Algorithm 1.

The baselines for evaluating partial observability and communication were as follows:

https://doi.org/10.1017/S1471068423000091 Published online by Cambridge University Press

- **Base5:** in **Exp1**, other agents followed hand-crafted policies and ad hoc agent did not use any communication actions.
- **Base6:** in **Exp2**, other agents followed built-in policies and the ad hoc agent did not use any communication actions.

Recall that KAT allows the use of communication actions (when needed) under conditions or partial observability. Also, each experiment described above (in FA domain) involved 150 episodes and results were tested for statistical significance.

In the **HFO domain**, we used six external agent teams from the 2013 RoboCup simulation competition to create the ad hoc agent's teammates and opponents. Five teams were used to create offense agents: *helios, gliders, cyrus, axiom and aut*; agents of the defense team were based on the *agent2d* team. Similar to the initial phase in the FA domain, we deployed the existing agent teams in the HFO domain and collected observations of states before and after each transition in the episode. Since the actions of other agents are not directly observable, they were computed from the observed state transitions. To evaluate the ability to learn from limited data, we only used data from 300 episodes for each type of agent to create the tree-based models for behavior prediction, which were then revised (as needed) and used by the ad hoc agent during reasoning.

We first compared KAT's performance with a baseline that only used non-monotonic logical reasoning with prior knowledge but without any behavior prediction models (**Exp3**), that is, the ad hoc agent was unable to anticipate the actions of other agents. Next, we evaluated KAT's performance with each built-in external team, that is, all offense agents other than the ad hoc agent were based on one randomly selected external team in each episode. In **Exp4**, we measured performance in the limited version, that is, two offense players (including ad hoc agent) against two defense agents (including goalkeeper). In **Exp5**, we measured performance in the full version, that is, four offense players (including ad hoc agent) played against five defense agents (including goalkeeper). In **Exp6** and **Exp7**, we evaluated performance under partial observability in the limited and full versions respectively. As the baselines for **Exp4-Exp5**, we used recent (state of the art) AHT methods: PPAS (Santos *et al.* 2021), and PLASTIC (Barrett *et al.* 2017). These methods considered the same external agent teams mentioned above, allowing us to compare our results with the results reported in their papers. For **Exp6-Exp7**, we used the external agent teams as baselines. We conducted 1000 episodes for each experiment described above, and tested the results for statistical significance.

4.2 Experiment results

We begin with the results of experiments in the **FA domain**. First, Table 3 summarizes the results of using our model selection algorithm in **Exp1**. When the other agents followed the hand-crafted policies and the model selection mechanism was not used by the ad hoc agent (**Base1**), the team of guards had the lowest winning percentage. When the ad hoc agent used the model selection algorithm with a fixed penalty assigned for any mismatch between predicted and actual actions (**Base2**), the performance of the team of guards improved. When the ad hoc agent used KAT's model selection method (Algorithm 1), the winning percentage of the team of guards was substantially higher

https://doi.org/10.1017/S1471068423000091 Published online by Cambridge University Press

Table 3. *Wins (%) for guards with hand-crafted policies in FA domain* (**Exp1**). *Model adaptation improves performance*

Experiment	Win %
Without model selection (**Base1**)	63
When using direct comparison (**Base2**)	68
With model selection algorithm (**KAT**)	73

Table 4. *Wins (%) for guards with built-in policies in FA domain* (**Exp2**). *Model adaptation improves performance*

Experiment	Win %
Without model selection (**Base3**)	47
When using direct comparison (**Base4**)	45
With model selection algorithm (**KAT**)	55

Table 5. *Wins (%) for guards with hand-crafted policies in FA domain* (**Exp1**). *Communication addresses partial observability*

Policy	With Comm. (%)	Without Comm. (%, **Base5**)
Policy1	73	58
Policy2	19	8

than the other two options. These results demonstrated that *KAT's adaptive selection of the behavior prediction models improved performance.*

Next, the results of **Exp2** are summarized in Table 4. We observed that KAT enabled the ad hoc agent to adapt to previously unseen teammates and opponents that used the FA domain's built-in policies, based on the model selection algorithm and the online revision of the behavior models learned from the hand-crafted policies. KAT provided the best performance compared with not using any model adaptation or selection (**Base3**), and when model selection assigned a fixed penalty for action mismatch (**Base4**). These results and Table 3 support **H1** and **H2**.

The results from **Exp1** under partial observability, with and without communication (**Base5**), are summarized in Table 5. Recall that the other agents used the FA domain's hand-crafted policies in this experiment. When the communication actions were enabled for the ad hoc (guard) agent, the winning percentage of the team of guards was substantially higher than the winning percentage of the team of guards when they could not use the communication actions. **Policy2** was a particularly challenging scenario (because both guards and attackers can shoot), which justified the lower (overall) winning percentage.

https://doi.org/10.1017/S1471068423000091 Published online by Cambridge University Press

Table 6. *Wins (%) for guards with built-in policies in FA domain* (**Exp2**). *Communication addresses partial observability*

Policy	With Comm. (%)	Without Comm. (%, **Base6**)
Policy220	79	85
Policy650	42	41
Policy1240	46	43
Policy1600	18	17

Table 7. *Fraction of goals scored (i.e., games won) by the offense team in HFO domain with and without the learned behavior prediction models* (**Exp3**). *Reasoning with prior domain knowledge but without the behavior prediction models has a negative impact on performance*

Version	KAT (%)	Logical Reasoner (%)
Limited (2v2)	79	67
Full (4v5)	30	26

Next, the results from **Exp2** under partial observability, with and without communication (**Base6**), are summarized in Table 6. Recall that the other agents used the FA domain's built-in policies. We observed that when the guards (other than the ad hoc agent) followed policies 650, 1240, or 1600, the winning percentage of the team of guards was comparable or higher when communication actions were enabled compared with when these actions were not available (**Base6**). With Policy 220, the performance of the team of guards was slightly worse when the communication actions were enabled. However, unlike the other policies, Policy 220 results in the guards spreading themselves in-front of the fort and shooting continuously. Under these circumstances, partial observability and communication strategies were not important factors in determining the outcome of the corresponding episodes. These results support hypothesis **H3**.

We next describe the results from the **HFO domain**. Table 7 summarizes results of **Exp3**, which compared KAT's performance with a baseline that had the ad hoc agent only reasoning with prior knowledge, that is, without any learned models predicting the behavior of other agents. With KAT, the fraction of goals scored by the offense team was significantly higher than with the baseline. These results emphasized the importance of learning and using the behavior prediction models, and indicated that leveraging the interplay between representation, reasoning, and learning leads to improved performance, which supports hypothesis **H1**.

Next, the prediction accuracy of the learned behavior models created for the limited version (**Exp4**) and full version (**Exp5**) of the HFO domain are summarized in Tables 8 and 9, respectively. Recall that these behavior models were learned for the agents other than the ad hoc agent using data from 300 episodes (for each external agent type). This translated to orders of magnitude fewer training samples than the few

https://doi.org/10.1017/S1471068423000091 Published online by Cambridge University Press

Table 8. *Prediction accuracy of the learned agent behavior models in limited (2v2) version of the HFO domain* (**Exp4**)

Agent type	Accuracy (%)
Helios	78.2
Gliders	83.2
Cyrus	69.5
Aut	72.4
Axiom	76.2
Agent2D	79.8

Table 9. *Prediction accuracy of the learned agent behavior models in full (4v5) version of the HFO domain* (**Exp5**)

Agent type	Accuracy (%)
Helios	86.0
Gliders	66.4
Cyrus	77.6
Aut	67.7
Axiom	73.6
Agent2D	71.9

Table 10. *Fraction of goals scored (i.e., games won) by the offense team in HFO domain in the limited version* (*2v2,* **Exp4**)*and full version* (*4v5,* **Exp5**). *KAT's performance comparable with the baselines in the limited version and much better than the baselines in the full version*

Version	KAT (%)	PPAS (%)	PLASTIC (%)
Limited (2v2)	79	80	80
Full (4v5)	30	20	20

hundred thousand training samples used by state of the art data-driven methods that do not reason with domain knowledge. The prediction accuracy varied over a range for the different agent types. Although the accuracy values were not very high, the models could be learned and revised quickly during run time; also, these models resulted in good performance when the ad hoc agent also reasoned with prior knowledge.

The results of **Exp4** and **Exp5** comparing KAT's performance with the state of the art baselines for the HFO domain (PPAS, PLASTIC) are summarized in Table 10. Recall that these data-driven baselines required orders of magnitude more training examples and did not support reasoning with prior domain knowledge. The fraction of goals scored (i.e., games won) by the team of offense agents including our ad hoc agent was comparable with the goals scored by the baselines for the limited version, and substantially better

https://doi.org/10.1017/S1471068423000091 Published online by Cambridge University Press

Table 11. *Goals scored (i.e., games won) by offense team in HFO domain under partial observability* (**Exp6**, **Exp7**). *KAT's performance comparable with baseline that had no ad hoc agents in the team but used training datasets that were orders of magnitude larger*

Version	KAT (%)	Original Team (%)
Limited (2v2)	71	76
Full (4v5)	18	20

than goals scored by the baselines for the full version. These results strongly support hypotheses **H1** and **H2**.

The results of evaluating KAT under partial observability (in HFO domain) are summarized in Table 11 compared with teams of external agent types without any ad hoc agent. Although the results indicate that KAT's performance was slightly lower than the baseline teams without any ad hoc agents, the difference was not significant and mainly due to noise (e.g., in the perceived angle to the goal during certain episodes). The ability to provide performance comparable with teams whose training datasets were orders of magnitude larger strongly supports hypothesis **H3**.

In addition to the experimental results documented above, videos of experimental trials, including trials involving unexpected changes in the number and type of other agents, are provided in support of the hypotheses in our open-source repository (Dodampegama and Sridharan 2023b).

5 Conclusions

Ad hoc teamwork (AHT) refers to the problem of enabling an agent to collaborate with others without any prior coordination. This problem is representative of many practical multiagent collaboration applications. State of the art AHT methods are data-driven, requiring a large labeled dataset of prior observations to learn offline models that predict the behavior of other agents (or agent types) and determine the ad hoc agent's behavior. This paper described KAT, a knowledge-driven AHT architecture that supports non-monotonic logical reasoning with prior commonsense domain knowledge and predictive models of other agents' behaviors that are learned and revised rapidly online using heuristic methods. KAT leverages KR tools and the interplay between reasoning and learning to automate the online selection and revision of the behavior prediction models, and to guide collaboration and communication under partial observability and changes in team composition. Experimental results in two benchmark simulated domains, Fort Attack and Half Field Offense, demonstrated that KAT's performance is better than that of just the non-monotonic logical reasoning component, and is comparable or better than state of the art data-driven methods that require much larger training datasets, provide opaque models, and do not support rapid adaptation to previously unseen situations.

Our architecture open up multiple directions for further research. For example, we will investigate the introduction of multiple ad hoc agents in the benchmark domains used in this paper and in other complex multiagent collaboration domains. We will also continue to explore the benefits of leveraging the interplay between reasoning and learning for AHT in teams of many more agents, including on physical robots collaborating

https://doi.org/10.1017/S1471068423000091 Published online by Cambridge University Press

with humans. In addition, we will build on other work in our group (Sridharan and Mota 2023; Sridharan and Meadows 2018) to demonstrate the ad hoc agent's ability to learn previously unknown domain knowledge. Furthermore, we will build on our recent work (Dodampegama and Sridharan 2023a) and the work of others in our group (Mota *et al.* 2021) to enable the ad hoc agent to provide relational descriptions as explanations of its decisions and beliefs in response to different kinds of questions.

Acknowledgments

This work was supported in part by the U.S. Office of Naval Research Award N00014-20-1-2390. All conclusions are those of the authors alone.

References

Balai, E., Gelfond, M. and Zhang, Y. 2013. Towards answer set programming with sorts. In *International Conference on Logic Programming and Nonmonotonic Reasoning*.

Balduccini, M. and Gelfond, M. 2003. Logic programs with consistency-restoring rules. In *AAAI Spring Symposium on Logical Formalization of Commonsense Reasoning*.

Baral, C., Gelfond, G., Pontelli, E. and Son, T. C. 2022. An action language for multi-agent domains. *Artificial Intelligence 302*, 103601.

Baral, C., Gelfond, G., Son, T. C. and Pontelli, E. 2010. Using answer set programming to model multi-agent scenarios involving agents' knowledge about other's knowledge. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, Vol. 1. 259–266.

Baral, C., Son, T. C. and Pontelli, E. 2010. Reasoning about multi-agent domains using action language $\mathcal{C}$: A preliminary study. In *Computational Logic in Multi-Agent Systems*, J. Dix, M. Fisher, and P. Novák, Eds. Springer Berlin Heidelberg, 46–63.

Barrett, S., Rosenfeld, A., Kraus, S. and Stone, P. 2017. Making friends on the fly: Cooperating with new teammates. *Artificial Intelligence 242*, 132–171.

Barrett, S., Stone, P., Kraus, S. and Rosenfeld, A. 2013. Teamwork with limited knowledge of teammates. In *AAAI Conference on Artificial Intelligence*, Vol. 27, 102–108.

Bowling, M. and McCracken, P. 2005. Coordination and adaptation in impromptu teams. In *National Conference on Artificial Intelligence*, 53–58.

Chen, S., Andrejczuk, E., Cao, Z. and Zhang, J. 2020. AATEAM: Achieving the ad hoc teamwork by employing the attention mechanism. In *AAAI Conference on Artificial Intelligence*, 7095–7102.

Deka, A. and Sycara, K. 2021. Natural emergence of heterogeneous strategies in artificially intelligent competitive teams. In *Advances in Swarm Intelligence*, Y. Tan and Y. Shi, Eds. Springer International Publishing, Cham, 13–25.

Dodampegama, H. and Sridharan, M. 2023a. Back to the future: Toward a hybrid architecture for ad hoc teamwork. In *AAAI Conference on Artificial Intelligence*.

Dodampegama, H. and Sridharan, M. 2023b. Code. https://github.com/hharithaki/KAT.

Gelfond, M. and Inclezan, D. 2013. Some properties of system descriptions of AL_d. *Applied Non-Classical Logics, Special Issue on Equilibrium Logic and ASP 23,* 1–2, 105–120.

Gigerenzer, G. 2016. *Towards a Rational Theory of Heuristics*. Palgrave Macmillan UK, London, 34–59.

Gigerenzer, G. 2020. What is bounded rationality? In *Routledge Handbook of Bounded Rationality*. Routledge.

Gigerenzer, G. and Gaissmaier, W. 2011. Heuristic decision making. *Annual Review of Psychology 62*, 451–482.

Hausknecht, M., Mupparaju, P., Subramanian, S., Kalyanakrishnan, S. and Stone, P. 2016. Half field offense: An environment for multiagent learning and ad hoc teamwork. In *AAMAS Adaptive Learning Agents Workshop*.

Katsikopoulos, K., Simsek, O., Buckmann, M. and Gigerenzer, G. 2021. *Classification in the Wild: The Science and Art of Transparent Decision Making*. MIT Press.

Macke, W., Mirsky, R. and Stone, P. 2021. Expected value of communication for planning in ad hoc teamwork. In *AAAI Conference on Artificial Intelligence*, 11290–11298.

Mirsky, R., Carlucho, I., Rahman, A., Fosong, E., Macke, W., Sridharan, M., Stone, P. and Albrecht, S. 2022. A survey of ad hoc teamwork: Definitions, methods, and open problems. In *European Conference on Multiagent Systems*.

Mota, T., Sridharan, M., and Leonardis, A. 2021. Integrated commonsense reasoning and deep learning for transparent decision making in robotics. *Springer Nature CS 2,* 242.

Rahman, M. A., Hopner, N., Christianos, F. and Albrecht, S. V. 2021. Towards open ad hoc teamwork using graph-based policy learning. In *International Conference on Machine Learning*, 8776–8786.

Santos, P. M., Ribeiro, J. G., Sardinha, A. and Melo, F. S. 2021. Ad hoc teamwork in the presence of non-stationary teammates. In *Progress in Artificial Intelligence*, G. Marreiros, F. S. Melo, N. Lau, H. Lopes Cardoso, and L. P. Reis, Eds. Springer International, 648–660.

Son, T. and Balduccini, M. 2018. Answer set planning in single- and multi-agent environments. *Künstliche Intelligenz 32*.

Son, T. C., Pontelli, E. and Nguyen, N.-H. 2010. Planning for multiagent using asp-prolog. In *Computational Logic in Multi-Agent Systems*, J. Dix, M. Fisher, and P. Novák, Eds. Springer Berlin Heidelberg, 1–21.

Son, T. C. and Sakama, C. 2010. Reasoning and planning with cooperative actions or multi-agents using answer set programming. In *Declarative Agent Languages and Technologies VII*. Lecture Notes in Computer Science, vol. 5948. Springer Berlin Heidelberg, 208–227.

Sridharan, M., Gelfond, M., Zhang, S. and Wyatt, J. 2019. REBA: A refinement-based architecture for knowledge representation and reasoning in robotics. *Journal of Artificial Intelligence Research 65*, 87–180.

Sridharan, M. and Meadows, B. 2018. Knowledge representation and interactive learning of domain knowledge for human-robot collaboration. *Advances in Cognitive Systems 7*, 77–96.

Sridharan, M. and Mota, T. 2023. Towards Combining Commonsense Reasoning and Knowledge Acquisition to Guide Deep Learning. *Autonomous Agents and Multi-Agent Systems 37,* 4.

Stone, P., Kaminka, G., Kraus, S. and Rosenschein, J. 2010. Ad Hoc Autonomous Agent Teams: Collaboration without Pre-Coordination. In *AAAI Conference on Artificial Intelligence*, 1504–1509.

Wu, F., Zilberstein, S. and Chen, X. 2011. Online planning for ad hoc autonomous agent teams. In *International Joint Conference on Artificial Intelligence*, 439–445.

Zand, J., Parker-Holder, J. and Roberts, S. J. 2022. On-the-fly strategy adaptation for ad-hoc agent coordination. In *International Conference on Autonomous Agents and Multiagent Systems*, 1771–1773.

Zintgraf, L., Devlin, S., Ciosek, K., Whiteson, S. and Hofmann, K. 2021. Deep interactive Bayesian reinforcement learning via meta-learning. In *International Conference on Autonomous Agents and Multiagent Systems*.

https://doi.org/10.1017/S1471068423000091 Published online by Cambridge University Press