TPLP: Page 1–15. © The Author(s), 2023. Published by Cambridge University Press 2023
doi:10.1017/xxxxx

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

HASRA DODAMPEGAMA
University of Birmingham

MOHAN SRIDHARAN
University of Birmingham

submitted xx xx xxxx; revised xx xx xxxx; accepted xx xx xxxx

Abstract

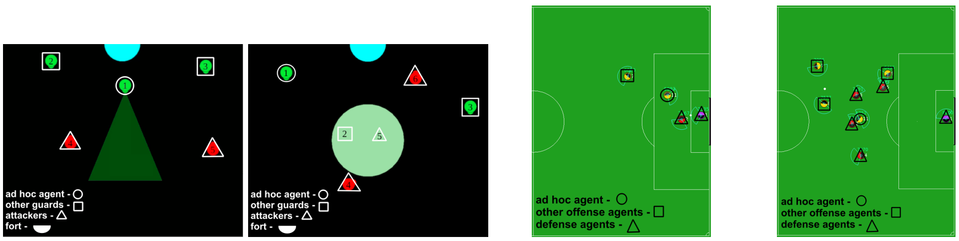
Ad hoc teamwork focuses on enabling an agent to collaborate with teammates without prior coordination. State of the art ad hoc teamwork methods use data-driven methods and a large labeled dataset of prior observations to model the behavior of other agent *types* and to determine the ad hoc agent's behavior. These methods are computationally expensive, lack transparency, and make it difficult to adapt to changes in team composition. Our recent work introduced a proof of concept architecture that determined an ad hoc agent's behavior based on non-monotonic logical reasoning with prior commonsense domain knowledge and predictive models of other agents' behaviour that were learned from limited examples. In this paper, we use KR tools to substantially expand the architecture's capabilities, supporting: (a) online adaptation and choice of learned models of other agents' behavior; and (b) collaboration in the presence of partial observability and limited communication. Experimental evaluation in two different simulated benchmark domains for ad hoc teamwork demonstrates performance comparable or better than state of the art data-driven baselines in both simple and complex scenarios, 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 defenders (Hausknecht et al. 2016). Agents in these domains have limited knowledge of each other's capabilities, no prior experience of working as a team, ability to observe only part of the environment (Figure 1b), and limited communication bandwidth. Such scenarios are representative of practical multiagent 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



(a) Fully observable (b) Partial observable (c) Limited version (d) Full version

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

ad hoc agent's behavior, based on a long history of prior experience. It is difficult to obtain such training examples, and computationally expensive to build the necessary models or to revise them in response to new situations in complex domains. 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 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 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, which provided a proof of concept demonstration of just the first capability 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, even in the presence of partial observability and changes in team composition.

2 Related Work

Methods for AHT have been developed under different names, 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 teammates based on a long history or 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) for each situation (Barrett et al. 2017), or 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), and used deep network-based learned sequential and hierarchical models with approximate belief inference methods (Zintgraf et al. 2021). In addition, researchers have explored different communication strategies for AHT, *e.g.,* a multiagent, multi-armed bandit formulation to broadcast messages to teammates at a cost (Barrett et al. 2017), or heuristic method to assess the cost and value of different queries (Macke et al. 2021). These methods 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. 2010b). 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 non-cooperative 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, *e.g.,* world-altering, sensing, and communication actions (Baral et al. 2010a). 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 that ASP can be used to represent and reason in multiagent domains. Our work draws on these findings to address the reasoning and learning challenges faced by an ad hoc agent that has to collaborate with teammates under conditions of partial observability and limited communication.

3 Architecture

Figure 2 is an overview of our KAT architecture. Our ad hoc agent performs 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, who then independently determine and execute their individual actions in the environment. KAT's components are described using 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 team and domain. 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 direction with a particular velocity, turn clockwise or anticlockwise, do nothing, or shoot to kill any agent in the

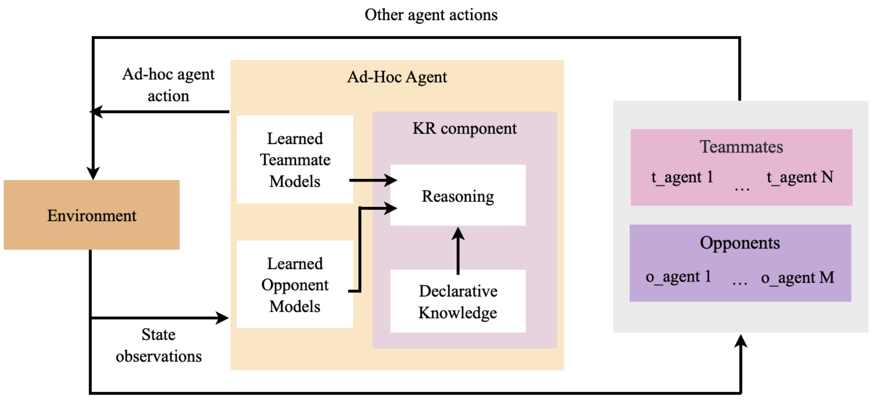


Fig. 2: Our KAT architecture combines complementary strengths of knowledge-based and data-driven heuristic reasoning and learning.

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, *i.e.,* 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). The ad hoc agent is a member of the offense team that seeks to score a goal against a defense team. An episode ends when: (a) offense team scores a goal; (b) ball leaves field; (c) defense team captures the ball; or (d) maximum episode length (500) is reached.

There are two version of the domain: (i) *limited*: two offense agents and two defense agents (including goalkeeper); and (ii) *full*: four offense agents, five defense agents (including goalkeeper). Agents other than the ad hoc agent are selected from teams created in the RoboCup 2D simulation league competitions. Similar to prior AHT methods, other offense team agents can be 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, high; we use the high-level features. 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.

Prior commonsense knowledge in FA and HFO includes relational descriptions of some domain attributes (*e.g.,* safe regions), agent attributes (*e.g.,* location), default statements, and axioms governing change, *e.g.,* an agent can only move to a location nearby, only shoot others within its shooting range (FA), and only score a goal from a certain angle (HFO). This knowledge may need to be revised over time.

3.1 Knowledge Representation and Reasoning

In our architecture, the transition diagrams of any domains are described in an extension of the action language $\mathcal{AL}_d$ (Gelfond and Inclezan 2013). 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*, *i.e.,* attributes whose values cannot be changed, and *fluents*, *i.e.,* 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; some sorts (*e.g., offense_agent*, *defense_agent*,) can be subsorts of others (*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. Σ includes *inertial* fluents that obey inertia laws and can be changed by actions, and *defined* fluents that do not obey inertia laws are not directly changed by actions. Inertial fluents in the HFO domain include:

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

which describe the location of the ad hoc agent, location of the ball, and the agent that has control of the ball. 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. 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 toward the goal, dribble to a location, and pass to a teammate. Axioms in $\mathcal{D}$ describe domain dynamics using elements in Σ in causal laws, state constraints, and executability conditions 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 imply 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 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, history $\mathcal{H}$ is a record of observations of fluents, *i.e.,* $obs(fluent, boolean, step)$, and action executions, *i.e.,* $hpd(action, step)$ at specific time steps. It also includes initial state defaults, *i.e.,* statements that are initially believed to be true in all but a few exceptional circumstances.

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 encodes *default negation* and *epistemic disjunction*, and supports non-monotonic reasoning; this ability to revise previously held conclusions is essential in AHT. $\Pi(\mathcal{D}, \mathcal{H})$ includes 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 inertia axioms, reality check axioms, closed world assumptions for defined fluents and actions, and helper axioms (*e.g.,* 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, *e.g.,* score a goal when it has control of the ball, and position itself otherwise:

$$goal(I) \leftarrow holds(scored\_goal, I). \quad goal(I) \leftarrow holds(loc(ad\_hoc\_agent, X, Y), I). \tag{5}$$

A suitable goal is selected and included at run-time based on current state, and the cost is minimized when computing a plan of actions for a given goal:

$$total(S) \leftarrow S = sum\{C, A : occurs(A, I), cost(A, C)\}. \ \#minimize\{S@p, S : total(S)\}.$$

We use the SPARC system (Balai et al. 2013) to write and solve CR-Prolog programs; examples are in our open source repository (Dodampegama and Sridharan 2023b). For computational efficiency, our programs build on prior work in our group to represent and reason at two tightly-coupled resolutions—see (Sridharan et al. 2019) for details.

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 under different (known) situations by learning models offline from many (*e.g.,* millions of) examples. It is intractable to obtain such labeled examples of different situations in complex domains. KAT thus chooses relevant attributes for models that can be: (a) learned from limited (*e.g.,* 10K) 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), 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.,* open worlds), characterizes behavior as a function of internal (cognitive) processes and environment, and focuses on *satisficing* based on differences between observed and predicted behavior. Also, heuristic methods (*e.g.,* one-reason, lexicographic) 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, experimentally choosing the method that best leverages domain structure. Specifically, KAT enables the ad hoc agent to learn an ensemble of "fast and frugal" (FF) decision trees that predict the behavior of other agents; each tree provides a binary class label and the number of leaves is limited by number of attributes (Katsikopoulos et al. 2021).

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 models, we enabled the ad hoc agent to automatically identify and respond to such changes by revising, switching between, or learning new models.* Existing models are revised by changing the parameters of FF trees, and Algorithm 1 describes an instance of our model selection approach for models predicting the pose (*i.e.,* position and orientation) of agents. Specifically, the ad hoc agent periodically compares the predictions of the existing models

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

Description of attribute	Number	Description of attribute	Number
x, y position of agent	12	distance from agent to fort	6
distance from agent to center of field	6	distance to nearest attacker from fort	1
agents' polar angle with center of field	6	number of attackers not alive	1
orientation of the agent	6	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 *size of attribute* times the *number of agents.*

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

Description of attribute	Number
x position of agent	4
y position of agent	4
x position of the ball	1
y position of the ball	1

with the actual (*i.e.,* observed) action choices of each agent (teammate, opponent) over a sliding window of information about the domain state and the agents' action choices. Also, a graded strategy for computing the error penalizes *e.g.* differences in orientation less than differences in position (Lines 4-6, Algorithm 1). An existing model is used (and revised) or a new one is learned based on the degree of match (Line 10, Algorithm 1).

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 (FA, HFO). Specifically, in the FA domain, we introduced a *forest* region where attackers can hide from the view of the two

Algorithm 1: Model Selection

```
Input: $\mathcal{A}$: other agents; $\mathcal{M}$: subset of behaviour models; $\{a_r\}$: actual action
       choices of agents, $\{a_p\}$: action predictions from behaviour models
1  for $i = 0$ to $\mathcal{A}$ do
2      for $m = 0$ to $\mathcal{M}$ do
3          if $a_p \neq a_r$ then
4              $l_r, o_r \leftarrow$ real_location_orientation($a_r$)
5              $l_p, o_p \leftarrow$ pred_location_orientation($a_p$)
6              $penalty \leftarrow abs(l_r - l_p) + abs(o_r - o_p)/10$
7          end
8          scores = scores - penalty
9      end
10     update_model_scores($\mathcal{M}$, scores)
11 end
```

guards (other than ad hoc agent), giving them an opportunity to secretly approach the fort—Figure 1b. The ad hoc agent has visibility of the forest region; it can decide when to communicate with its teammates, *e.g.,* when: (a) one of more attackers are hidden in the forest; and (b) one of the other guards is closer to the hidden attacker(s) than it. The associated reasoning can be encoded using statements such as:

$$holds(shoots(G, AA), I+1) \leftarrow occurs(communicate(AHA, G, AA), I) \tag{6a}$$

$$holds(in\_forest(AA), I) \leftarrow holds(agent\_loc(AA, X, Y), I),\ forest(X, Y),\ not\ \ holds(shot(AA), I) \tag{6b}$$

$$-occurs(communicate(AHA, G, AA), I) \leftarrow\ not\ holds(in\_range(G, AA), I). \tag{6c}$$

where Statement 6(c) encodes that communication is used only when a hidden attacker is within range of a teammate; Statement 6(b) defines when an attacker is hidden; and Statement 6(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 using the builtin ability to limit each agent's perception to a specific viewing cone, *i.e.,* the agent is only able to sense objects (*i.e.,* agents, ball) within this cone. Given this use of builtin functions, we added some helper axioms that ensured 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 our architecture's capabilities:

H1: Our architecture's performance is comparable or better than state of the art baselines in different scenarios while requiring much less training;

H2: Our architecture enables adaptation to unforeseen changes in the type and number of other agents (teammates and opponents); and

H3: Our architecture 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 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 had two offense and two defense players (including one goalkeeper) in the limited version; and four offense players and five defense agents (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 experiments and baselines are provided below.

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 behaviour. Built-in policies were provided with the domain; they are based on graph neural networks trained using many training examples.

Hand-Crafted Policies.

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

Built-in Policies.

- **Policy220:** guards place themselves in front of the fort and shoot continuously; attackers try to approach the fort.
- **Policy650:** guards try to block the fort; attackers try to sneak in from all sides.
- **Policy1240:** guards spread and shoot the attackers; attackers sneak from all sides.
- **Policy1600:** guards are willing to move 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 built behavior models in the form of FF trees from 10000 state-action observations obtained by running the hand-crafted policies. It did not receive any prior experience or models of the built-in policies.

Our previous work documented the accuracy of static behavior models and the performance of a proof of concept AHT architecture in the FA domain (Dodampegama and Sridharan 2023a). In this paper, we focused on evaluating the ability to select and revise 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; ad hoc agent did not revise behavior models or use the model selection algorithm.
- **Base2:** other agents followed a random mix of hand-crafted policies; ad hoc agent used a model selection algorithm without a graded strategy to compare predicted and actual actions (*i.e.,* binary comparison instead of Line 6 in Algorithm 1).
- **Base3:** other agents followed a random mix of builtin policies; ad hoc agent did not revise models or use the model selection algorithm.
- **Base4:** other agents followed a random mix of built-in policies; ad hoc agent used the model selection algorithm without a graded strategy to compare predicted and actual actions (*i.e.,* binary comparison instead of Line 6 in Algorithm 1).

The baselines for evaluating partial observability and communication were:

- **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.

Each experiment 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 *agent2d* team. Similar to the initial phase in the FA domain, we deployed the existing agent teams in the HFO domain and collected state observations. 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 models for behavior prediction, which were then 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**), *i.e.,* it was unable to anticipate the actions of other agents. Next, we evaluated KAT's performance with each built-in external team, *i.e.,* all offense agents other than the ad hoc agent were based on one external team at a time. In **Exp4**, we measured performance in the limited version, *i.e.,* two offense players (including ad hoc agent) against two defense agents (including goalkeeper). In **Exp5**, we measured performance in the full version, *i.e.,* 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 used the same external agent types for comparison, allowing us to compare our results with those in their papers. For **Exp6-Exp7**, we used the external agent teams as baselines. We conducted 1000 episodes for each experiment, and tested 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 without a graded strategy for comparing predicted and actual actions (**Base2**), the performance of the team of guards improves. When the ad hoc agent used our model selection method (Algorithm 1), the winning percentage of the team of guards is substantially higher. These results demonstrated that *adaptively selecting behavior models improved our architecture's performance.*

Next, the results of **Exp2** are summarized in Table 4. We observed that KAT enabled the ad hoc agent to adapt to unforeseen teammates and opponents that were using the FA domain's built-in policies, based on online revision of the behavior models (*i.e.,* the FF trees) learned from the hand-crafted policies and the model selection algorithm. KAT provided the best performance compared with not using any model adaptation or selection (**Base3**), and when model selection was used without the graded strategy (**Base4**). These results and those in Table 3 support hypotheses **H1** and **H2**.

The results from **Exp1** under partial observability, with and without communication (**Base5**), are summarized in Table 5. 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** is a particularly challenging scenario (both guards and attackers shoot), which justified the lower (overall) winning percentage.

Table 3: Wins (%) for guards with hand-crafted policies (**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 (**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 (**Exp1**). Communication addresses partial observability.

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

Table 6: Wins (%) for team of guards built-in policies (**Exp2**). Communication addresses partial observability.

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

Next, the results from **Exp2** under partial observability, with and without communication (**Base6**) strategies, are summarized in Table 6. Other agents used the FA domain's built-in policies. We observed that the winning percentage of the team of guards when following the policies 650, 1240, and 1600 was comparable or higher when the communication actions were enabled compared with the absence of these actions (**Base6**). With Policy 220, the performance was slightly worse with the communication actions. However, unlike the other policies, Policy 220 results in the guards spreading themselves in-front of the fort and shooting continuously. As a result, partial observability and communication strategies did not contribute significantly to the outcome. These results support **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, *i.e.,* 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. Leveraging the interplay between representation, reasoning, and learning led to this improved performance that supports **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 was much smaller than the number of samples (often a few million) 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.

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 require millions of training examples and do not include knowledge-based reasoning. The fraction of goals scored (and games won) by the team of

Table 7: Fraction of goals scored by the offense team in HFO domain in **Exp3**.

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

Table 8: Prediction accuracy of the learned agent behaviour 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 behaviour 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

offense agents including our ad hoc agent was comparable with the goals scored by the baselines for the limited version, and substantially better than 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 numbers indicated that KAT's performance is slightly lower than teams without any ad hoc agents, the difference is not significant and mainly due to noise (*e.g.,* in perceived angle to the goal). The ability to provide performance comparable with teams whose training datasets are orders of magnitude larger strongly supports **H3**.

Due to space constraints, additional video results, including that of experimental trials involving unexpected changes in the number and type of other agents, are provided 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. 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 models that predict 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. In the future, we will investigate the introduction of multiple ad hoc agents in complex multiagent collaboration

Table 10: Fraction of goals scored by the offense team in HFO domain in the limited version(2v2) and full version(4v5).

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

Table 11: Goals scored by offense team in HFO domain under partial observability. KAT's performance comparable with baseline that had no ad hoc agents in the team.

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

domains. We will also further explore the interplay between reasoning and learning for AHT in teams of many more agents, and for AHT on physical robots collaborating with humans. Furthermore, we will build on other work in our group (Mota et al. 2021; Sridharan and Mota 2023) to demonstrate the ad hoc agent's ability to revise existing knowledge and provide relational descriptions as explanations of its decisions and beliefs.

References

Balai, E., Gelfond, M., and Zhang, Y. Towards Answer Set Programming with Sorts. In *International Conference on Logic Programming and Nonmonotonic Reasoning* 2013.

Balduccini, M. and Gelfond, M. Logic Programs with Consistency-Restoring Rules. In *AAAI Spring Symposium on Logical Formalization of Commonsense Reasoning* 2003.

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. 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* 2010a, volume 1, 259–266.

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

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. Teamwork with limited knowledge of teammates. In *AAAI Conference on Artificial Intelligence* 2013, volume 27, pp. 102–108.

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

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

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

Dodampegama, H. and Sridharan, M. Back to the Future: Toward a Hybrid Architecture for Ad Hoc Teamwork. In *AAAI Conference on Artificial Intelligence* 2023a.

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. What is Bounded Rationality? In *Routledge Handbook of Bounded Rationality* 2020. 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. Half field offense: An environment for multiagent learning and ad hoc teamwork. In *AAMAS Adaptive Learning Agents Workshop* 2016.

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. Expected value of communication for planning in ad hoc teamwork. In *AAAI Conference on Artificial Intelligence* 2021, pp. 11290–11298.

Mirsky, R., Carlucho, I., Rahman, A., Fosong, E., Macke, W., Sridharan, M., Stone, P., and Albrecht, S. A Survey of Ad Hoc Teamwork: Definitions, Methods, and Open Problems. In *European Conference on Multiagent Systems* 2022.

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. Towards open ad hoc teamwork using graph-based policy learning. In *International Conference on Machine Learning* 2021, pp. 8776–8786.

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

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. Planning for multiagent using asp-prolog. In Dix, J., Fisher, M., and Novák, P., editors, *Computational Logic in Multi-Agent Systems* 2010, pp. 1–21. Springer Berlin Heidelberg.

Son, T. C. and Sakama, C. Reasoning and Planning with Cooperative Actions or Multiagents using Answer Set Programming. In *Declarative Agent Languages and Technologies VII* 2010, volume 5948 of *Lecture Notes in Computer Science*, pp. 208–227. Springer Berlin Heidelberg.

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

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

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

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

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