In *Proceedings of the 29th International Joint Conference on Artificial Intelligence* (IJCAI 2020), Yokohama, Japan January 2021

A Penny for Your Thoughts: The Value of Communication in Ad Hoc Teamwork

Reuth Mirsky[1] , **William Macke**[1] , **Andy Wang**[1] , **Harel Yedidsion**[1] and **Peter Stone**[1,2]

[1]Department of Computer Science, The University of Texas at Austin
[2]Sony AI

{reuth, wmacke, andywang6666, harel, pstone}@cs.utexas.edu

Abstract

In ad hoc teamwork, multiple agents need to collaborate without having knowledge about their teammates or their plans a priori. A common assumption in this research area is that the agents cannot communicate. However, just as two random people may speak the same language, autonomous teammates may also happen to share a communication protocol. This paper considers how such a shared protocol can be leveraged, introducing a means to reason about Communication in Ad Hoc Teamwork (CAT). The goal of this work is enabling improved ad hoc teamwork by judiciously leveraging the ability of the team to communicate. We situate our study within a novel CAT scenario, involving tasks with multiple steps, where teammates' plans are unveiled over time. In this context, the paper proposes methods to reason about the timing and value of communication and introduces an algorithm for an ad hoc agent to leverage these methods. Finally, we introduces a new multiagent domain, the tool fetching domain, and we study how varying this domain's properties affects the usefulness of communication. Empirical results show the benefits of explicit reasoning about communication content and timing in ad hoc teamwork.

1 Introduction

Autonomous agents are becoming increasingly capable of solving complex tasks, but encounter many challenges when required to solve such tasks as a team. In many multiagent tasks, the coordination strategy is either learned or decided a priori while assuming full knowledge of the teammates and the task at hand. However, as agents become more robust and diverse, they are more likely to need to cooperate in new situations without the ability to coordinate in advance. As a topical example, service robots might be deployed to assist medical teams in an epidemic outbreak. Such heterogeneous robots might be deployed without any prior coordination about each other's capabilities to assist, but they will only be effective if they are able to work together, and with the medical team, without the need to be explicitly provided with coordination strategies in advance.

This motivation is the basis for *ad hoc teamwork*, which is defined as collaborating with teammates without pre-coordination [Stone *et al.*, 2010; Albrecht and Stone, 2018]. This terminology reflects that the *collaboration* is ad hoc – the ways in which the agents learn, act and interact may be quite principled. There are two main properties that distinguish ad hoc teamwork from other multiagent systems. First, it assumes that all teammates strive to be collaborative [Stone *et al.*, 2010]. Second, the properties of the environment and of the teammates cannot be changed by the ad hoc agent. Its task is to reason and plan under these conditions.

Since agents engaged in ad hoc teamwork are developed independently, it is commonly (and reasonably) not assumed that they can communicate with one another. However, in many real world situations, previously unfamiliar agents may still share a communication protocol. In the medical emergency example, once a robot is deployed, even if it does not know the exact goals or plans of a physician in advance, they may still be able to communicate using visual and audio signals, by choosing a legible plan [Kulkarni *et al.*, 2019], or even by altering the teammates' achievable actions [Bisson *et al.*, 2011]. Barrett et al. [2014] considered a scenario in which either teammates are assumed to share a common communication protocol, or else this assumption can be quickly tested on the fly (e.g. by probing). Their work was the first to investigate an ad hoc agent's reasoning over communication in ad hoc teamwork (CAT). However it was situated in a very restrictive multiagent setting, namely a multi-arm bandit, where each task was a single choice of which arm to pull.

The main contribution of this paper is a detailed study of a much more general CAT scenario, that can model a service robot that fetches different tools for a physician in a hospital [Cakmak and Thomaz, 2012]. The physician would normally prefer to avoid the additional cognitive load of communicating with the robot, but will answer an occasional question from it so that the robot can be a better collaborator. Based on this scenario's core properties, we name it the Sequential One-shot MultiAgent Limited Inquiry CAT scenario, or SOMALI CAT. In SOMALI CAT the agents execute *sequential plans* and only the ad hoc agent can inquire about a teammate's goal. SOMALI CAT was defined to be a broadly representative class of CAT problems. Here we investigate several questions that are common across all CAT problems: (1) **Potential:** What are the cases where communication can

be useful? (2) **Content:** What should be communicated? (3) **Timing:** When should the agent communicate? To answer these questions in the context of SOMALI CAT, we provide measures for the value of communication, which are motivated by the value of information [Howard, 1966]. These measures are utilized in a heuristic algorithm that can reason about query content and timing to understand better the teammate's plan while minimizing the total cost of execution.

As a secondary contribution, our empirical analysis introduces a new multiagent domain, where different CAT properties can make communication more or less useful. In this *Tool Fetching domain*, one agent visits a workstation while the teammate fetches relevant tools for that station.

2 Related Work

We divide relevant research into three areas that involve agents reasoning about other agents: ad hoc teamwork, communicating agents, and modeling other agents.

Ad Hoc Teamwork. Ad hoc teamwork was first introduced as the challenge to create an autonomous agent that is able to efficiently and robustly collaborate with previously unknown teammates on tasks to which they are all individually capable of contributing as team members [Stone *et al.*, 2010]. Subsequent works proposed to model teammates by mapping them to one out of a set of types [Albrecht and Stone, 2017; Ravula *et al.*, 2019]. Our work is similar in the sense that we represent the type of the teammate as its final goal. Wang et al. [2020] recently proposed an Inverse Reinforcement Learning technique to infer teammates' goals on the fly when no communication is available. Barrett et al. [2014] were the first to reason about CAT, in the context of a multi-arm bandit problem. Their work was later extended by Chakraborty et al. [2017] to handle multiple decentralized ad hoc agents. There have been several works on ad hoc agents that attempt to influence the behavior of their teammates [Stone *et al.*, 2013; Agmon and Stone, 2012]. Unlike our work, where the interaction is done using explicit communication, these works assume implicit communication or interaction via action choices of the agents in episodic settings.

Communicating Agents. The concept of communicating agents has been a fertile research area in the context of multiagent systems [Singh, 1998]. What makes ad hoc teamwork with communication different from other works on communication in multiagent systems is that it takes the point of view of a single agent trying to collaborate with teammates with predefined policies: the single agent cannot change how the teammates will respond to the communicated information. In contrast, in general multiagent communication, the agents are typically symmetrical in their ability to reason about one another (distributed decision making). This point of view shifts the challenges from the perspective of protocol design, to the perspective of reasoning about other agents' information states. For the rest of this section, we focus on works that make a similar assumption. Reinforcement learning is a learning approach, where the agent actively tries new things rather than following a fixed policy [Epshteyn *et al.*, 2008]. Various works investigated communication mechanisms to facilitate learning. Several works allow the learning agent to accept critique [Griffith *et al.*, 2013] or advice [Lowe *et al.*, 2017; Torrey and Taylor, 2013; Amir *et al.*, 2016; Kapourchali and Banerjee, 2019; Roth *et al.*, 2006]. In these works, the information is given by a teacher and the other agent passively integrates it. With recent developments in deep learning, several works were proposed for a sub-area of multiagent systems, where agents share information using learned communication protocols [Hernandez-Leal *et al.*, 2019; Mordatch and Abbeel, 2018; Foerster *et al.*, 2016]. These works make several assumptions not used in this work: that the agents can learn new communication skills, and that the agent trains with its teammates before execution.

Modeling other Agents. Another related research area is goal recognition, where an observer is required to recognize the goal of an actor given a sequence of observed actions. A common assumption in goal recognition is the *keyhole* assumption, that the actor is unaware of the observer [Kautz and Allen, 1986], an assumption that does not hold in our work. Several other works broke this assumption as well, either in order to communicate when there is ambiguity [Mirsky *et al.*, 2018], by performing actions in an attempt to assist the other agent [Fern *et al.*, 2007] or disambiguate between hypotheses [Shvo and McIlraith, 2020], or by influencing the other agent to take a specific course of action [Bisson *et al.*, 2011]. These works did not quantify the value of a specific query, nor did they provide an algorithm that can leverage this value for planning in collaboration with the other agent.

3 SOMALI CAT

The rest of this paper focuses on the SOMALI CAT scenario. In this scenario, there are two agents: the ad hoc agent that we design and its teammate. The teammate has a goal in mind (a task to be executed out from a set of possible tasks) and the ad hoc agent needs to assist that teammate, by helping with the task execution. The ad hoc agent is able to ask the teammate about its goal in which case the teammate will incur a cost by always responding. We define SOMALI CAT using the following properties:

Environment. The task performed requires a sequence of actions. It is also an episodic, one-shot task. The environment is static, deterministic, and fully observable.

Teammate. A teammate that is assumed to have perfect knowledge about the environment. It is assumed to plan optimally, given that it is unaware of the ad hoc agent's plans or costs. We cannot change its learning or reasoning capabilities.

Communication Channel. There is one communication channel, where the ad hoc agent can query as an action, and if it does, the teammate will forgo its action to reply truthfully (the communication channel is noiseless).

To explain and exemplify our definitions in the SOMALI CAT problem, we introduce a specific domain: the *tool fetching domain*. This domain is inspired by the crafting environment [Devin *et al.*, 2019], with the main difference being that there are two agents and only one of them can pick up tools. The domain is a discrete-action world. It contains n stationary workstations and one toolbox with k different tools. Each

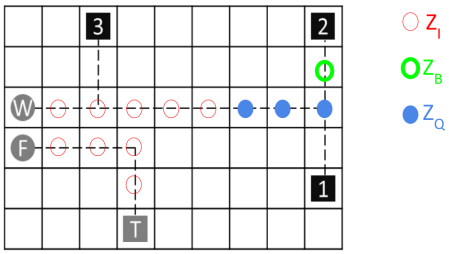

Figure 1: An example of the tool fetching domain. W, F and T are the locations of the worker, fetcher and toolbox respectively. The locations of the workstations are represented by the numbered squares. The circles show which zone each of the paths belong to (See section 5 for more information on the zones). The hollow red circles show the steps that belong only to the Zone of Information. The thick green circles show the steps that only belong to the Zone of Plan Branching. The blue disks show the steps that belong to the Zone of Querying.

workstation requires a unique set of tools to work in – in our implementation, we use $k = n$ and each workstation requires exactly one tool. The two agents in the environment are: an ad hoc agent, the *fetcher*, and a teammate, the *worker*. Tools can only be picked up by the fetcher. The worker's task is to use some workstation, while the fetcher's task is to bring the required tools as quickly as possible to that (initially unknown) workstation.

An agent can change its position during the game by executing one of the following actions: U for moving up, D for moving down, R for moving right, L for moving left, or N for staying put. The Fetcher may also pick up a tool (T) or ask a question (Q). The format of the questions we evaluate are "What is your goal workstation?" and "Is your goal in G'?" where G' is a subset of workstations. These queries were chosen because they can replace any type of query that disambiguates between potential goals (e.g. "Is your goal on the right?", "What are your next k actions?"), as long as we know how to resolve this ambiguity based on the response. A worker must respond immediately (and truthfully) to a question (R), which causes it to lose the opportunity to execute any other action. At each timestep, both agents decide separately upon an action they are interested in performing. A conflict can only occur if the fetcher asks a question (Q), which forces the worker to perform a replying action (R) instead of executing its planned action.

The problem ends successfully when both agents have reached some workstation, and the fetcher holds the relevant tool. The transitions are deterministic, and in our setup only one object can be held at a time – if the fetcher picks up a tool while holding another one, the old one automatically drops in the square of the pickup. Pickup has no effect unless the agent is at the same position as a tool. Figure 1 shows a running example of the tool fetching domain, with the agents at their initial locations.

4 Query Potential

In this section we define the Query Potential, or the maximum gain from querying in a CAT domain relative to that domain's minimal cost without querying. We consider a domain model in CAT to be a set of actions for both the teammate (A_T) and the ad hoc agent (A_A), a set of queries ($Q \subseteq A_A$) and a cost function (C) that maps joint plan execution of the agents (denoted $C(p_A, p_T)$) to $\mathbb{R}$.

We assume that the teammate has a plan (p_T) that is unaffected by the ad hoc agent, but the ad hoc agent's plan is contingent on the teamate's actions (π_A). Given these assumptions, the optimal behavior of the ad hoc agent with respect to C, under the constraint of asking exactly k queries ($Q \subseteq A_A$) can be defined as a function of the teammate's plan and k: ($\pi_A : A_T^* \times \mathbb{N} \rightarrow A_A^*$). This function is the trace of the optimal policy for the ad hoc agent, given its uncertainty over the teammate's plan. Due to this uncertainty, the trace may include actions such as waiting, querying, or backtracking.

Intuitively, the value of querying in a domain is the maximal gain that can be obtained by querying. We therefore now define the query potential of a domain $D = \{A_T, A_A, C\}$ as:

$$P(D) = \max_{p_T \in A_T^*} \frac{C(p_T, \pi_A(p_T, 0)) - \min_{k \geq 1} C(p_T, \pi_A(p_T, k))}{C(p_T, \pi_A(p_T, 0))} \quad (1)$$

If the cost of the optimal plan with and without the best possible query are the same, then the query potential is 0. If there is a query that can improve the total expected cost of the ad hoc agent's execution, then its potential is a positive value, and if all queries can only worsen the total cost, then its potential is negative. It might seem reasonable that if the query potential of a domain is positive, and all queries have the same cost, then querying about all goals right in the beginning of the task is the optimal behavior. However, in the next section we disprove this notion and present upper and lower bounds on the best timing of a query.

5 Query Content and Timing

We now consider the scenario where we want to evaluate the value of a specific query in our environment. Intuitively, the value of a query depends on the time when it is asked. For instance in our example domain, any query that is asked after the worker (the teammate) reaches a station is worthless, since the goal is already clear from the worker's actions. In general, for any plan of the teammate there is some time t where its goal becomes clear from its actions. Keren et al. [2014] formalize this notion as the **Worst Case Distinctiveness** (***wcd***), or the maximal amount of time an optimal plan can be ambiguous between two goals.

Definition 1. *The wcd between goals i and j for an agent, $\boldsymbol{wcd(i,j)}$, is the length of the longest shared prefix of some plans p_i, p_j for that agent that are optimal for goals i and j respectively.*

In this work, we define the wcd between goals i and j for the teammate as $wcd_T(i,j)$ and for the ad hoc agent as $wcd_A(i,j)$. The wcd values for both the teammate and the ad hoc agent in our running example are: $wcd_T(1,2) = 9$, $wcd_T(1,3) = 3$, $wcd_T(2,3) = 5$, $wcd_A(i,j) = 6$ when $i \neq j$. If the worker follows the dashed line to station 1, as shown in Figure 1, then it is impossible to tell that the goal is

not station 2 until a very late stage of the worker's plan execution. The length of this shared prefix is the wcd_T between stations 1 and 2, since any other path will disambiguate between stations 1 and 2 sooner. When the fetcher considers a query that disambiguates between goal g_i and goal g_j (e.g. "is g_i your goal?" or "what is your goal?"), it knows that this information will be revealed without querying after $wcd_T(i,j)$ timesteps, where $wcd_T(i,j)$ is the wcd for disambiguating between goal i and goal j of the teammate, by observing its actions.

Using this insight, we can define the times at which the ad hoc agent will gain information by querying about goals i, j:

Definition 2. *The **Zone of Information** for two goals i, j is $\boldsymbol{Z_I(i,j)} = \{t | t \leq wcd_T(i,j)\}$.*

After $wcd_T(i,j)$, no new information can be gained from querying about g_i versus g_j.

As an additional insight, it is not unlikely that there exists two or more optimal plans for the *ad hoc agent* such that there is significant overlap between the plan executions for g_i and g_j. Let us consider the maximal point in time t such that there exists a common plan prefix of optimal plans for goals g_i and g_j. t is the wcd_A of the ad hoc agent's plans. Consider the running example, where no matter what is the goal of its teammate, the ad hoc agent must first reach the toolbox in order to fetch a tool. This means that the ad hoc agent's actions up until the 6th timestep are the prefix of an optimal plan to either goal.

Given that two optimal plans can be identical up to a given timestep, no new information about the teammate's goal can require a change of plan until after this time has passed. We can then calculate the period of time at which our plan can still be improved without additional expected cost:

Definition 3. *The **Zone of Plan Branching** for two goals i, j is $\boldsymbol{Z_B(i,j)} = \{t | t \geq wcd_A(i,j)\}$.*

As the wcd_A considers the worst-case, it is possible that the teammate reveals its goal before that timestep, hence querying about the teammate's goal before the ad hoc agent is required to commit to a specific goal cannot decrease (and might increase) the total cost of the ad hoc agent's plan execution. As we require both the ability to improve our plan, and the ability to gain information, we can define the period of time where we are able to gain value from querying without committing to a plan that can incur a penalty:

Definition 4. *The **Zone of Querying** of two goals i, j is $\boldsymbol{Z_Q(i,j)} = Z_I(i,j) \cap Z_B(i,j)$.*

Figure 1 shows a representation of each zone in our running example. We mark the locations of the agents at different timesteps, given that the teammate executes the plan with the longest shared prefix between the plan for workstation 1 and the plan for workstation 2.

We can further expand this term to define the zone of querying for a query that disambiguates a set of goals, G', from $G \setminus G'$. In order to define this value, we consider the joint space in which some query can provide useful information without executing an action that is only relevant to goals in one of the sets $G \setminus G'$ or G' and not to goals in the other:

$$Z_Q(G') = \bigcup_{i \in G', j \notin G'} Z_I(i,j) \quad \cap \quad \bigcup_{i \in G', j \notin G'} Z_B(i,j) \tag{2}$$

Intuitively, a query $q_{G'}$ can have a benefit only as long as the teammate's sequence of executed actions is a prefix of two or more optimal plans for goals disambiguated by the query, and the ad hoc agent's next action deviates from an optimal plan for at least one of the goals in G'. This means that the first timestep inside $Z_Q(G')$ is the first timestep at which both (1) the ad hoc agent might execute a wrong action and (2) there might still be some ambiguity regarding the goal of the teammate.

Definition 5. *The **Critical Querying Point (CQP)** about a set of goals $G' \subsetneq G$ is the first timestep inside Z_Q of G'.*

$$CQP(G') = \begin{cases} \underset{t \in Z_Q(G')}{\arg\min}\, t & \text{if } Z_Q(G') \neq \emptyset \\ -1 & \text{otherwise} \end{cases} \tag{3}$$

In our running example, the critical querying points are $CQP(\{1\}) = CQP(\{2\}) = CQP(\{1,3\}) = 6$, and $CQP(\{3\}) = -1$. Next, we propose a heuristic planning algorithm for the ad hoc agent that considers the critical querying points to ask about each subset of goals $G' \subsetneq G$, and chooses the first timestep that is a CQP for some G', and queries "Is your goal in G'?"

6 Query Algorithm

In this section, we present an algorithm for planning the ad hoc agent's actions while reasoning about the zones of querying for a set of goals, in order to determine when to query. This algorithm comprises three parts: (1) Calculating wcd_T and wcd_A for all pairs of goals and keeping these values in a matrix; (2) Calculating Z_Q for multiple goals; and (3) Using this information to determine when and what to query.

Calculating worst case distinctiveness values. In addition to formalizing wcd, Keren et al. [2014] also provide an efficient method for calculating the wcd of 2 goals. Z_Q for any two goals is merely the interval from wcd_A to wcd_T, which can be identified via a matrix lookup.

Calculating Z_Q for multiple goals. Calculating Z_Q for more than two goals is significantly more complicated. First, for a set of goals $G' \subsetneq G$, we calculate $Z_B(G')$ and $Z_I(G')$:

$$Z_B(G') = \{t \mid t \geq \min_{i \in G', j \notin G'} wcd_A[g_i, g_j]\} \tag{4}$$

$$Z_I(G') = \{t \mid t \leq \max_{i \in G', j \notin G'} wcd_T[g_i, g_j]\} \tag{5}$$

Given $Z_B(G')$ and $Z_I(G')$, the value of $Z_Q(G')$ is calculated using Equation 2, by taking the intersection of all unions of Z_B and Z_I for goals i, j such that $i \in G'$ and $j \in G \setminus G'$.

Determining When to Query. As we have shown how to calculate Z_Q for a set of goals, we can now leverage this information to determine when to query. As mentioned above the CQP corresponds to the first point in Z_Q, which is the optimal time to query about a set of goals. In addition, we know

Algorithm 1 Calculate CQP for a set of goals G'

procedure CALCULATE WHEN TO QUERY FOR $G'(G$, the global set of goals, G', the goals to query about)
  $initialize(wcd_T, wcd_A)$
  $Z_B \leftarrow [\min_{i \in G', j \notin G'} wcd_A[g_i, g_j], \infty)$
  $Z_I \leftarrow [0, \max_{i \in G', j \notin G'} wcd_T[g_i, g_j]]$
  $Z_Q \leftarrow Z_B \cap Z_I$
  if $|Z_Q| > 0$ **then**
    return $\min(Z_Q)$
  else
    return -1

that we cannot gain value from a query to disambiguate goals whose $Z_Q = \emptyset$. We finalize this strategy in Algorithm 1. After calculating Z_Q for all G', an informed ad hoc agent will choose to query at the earliest CQP for any G', and will choose a query that disambiguates G' from $G \setminus G'$, such as "is your goal in G'?"[1]. In our domain, the minimal CQP is the same value for all G' such that $CQP(G') \neq -1$.

7 Empirical Results

The purpose of our experiments is twofold: (1) to methodically investigate the effects of query timing, content, and querying strategy on the performance of our example domain; and (2) to empirically demonstrate that the strategies based on Z_Q hold in general for different setups of the domain.

In all experiments, the goal of the worker is defined as the station it plans to visit, and unless stated otherwise, that goal is randomly chosen at the beginning of each simulation episode. We evaluate the total cost of the episode as the timesteps it takes for the ad hoc agent to reach that station.

7.1 Timing Effect on Cost

The first experiment was designed to methodically evaluate the benefit of a query at varying timesteps, by enumerating all possible decisions made by the teammate and the ad hoc agent. The ad hoc agent is allowed to query at most once, and only if there is still ambiguity between goals. If the agent is uncertain about the teammate's goal and is not set to query until some future point, it waits before picking up a tool that might be incorrect (This behavior is the optimal policy for the ad hoc agent given its uncertainty of the teammate's goal).

Figure 2 shows the average and worst case number of timesteps in the ad hoc agent's plan. The top, middle, and lower graphs represent a scenario where the teammate's true goal is station 1, 2 or 3 respectively. The x-axis is the timestep in which the query "what is your goal?" can be asked, and the y-axis shows the average number of timesteps it took the ad hoc agent to complete its plan, where each point is an average across all optimal plans. The results of these graphs together represent all optimal paths that the teammate can follow. The top part of each graph shows Z_Q, Z_B, and Z_I for the query.

As presented earlier, not asking a question at all (point X in the graphs) can potentially have a better value than querying

[1]We conjecture that it is an optimal policy for this type of queries given equiprobable goals, but it is not necessarily optimal otherwise.

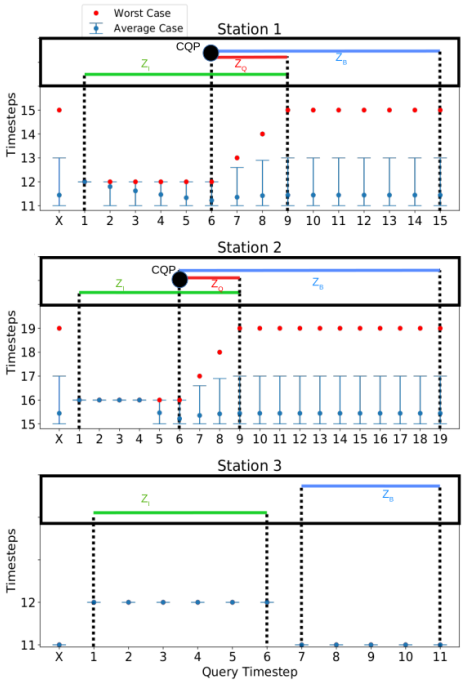

Figure 2: The effect of query timing on the ad hoc agent's plan.

if the teammate executes a plan that disambiguates its goal at an early stage, but the variance of using such a strategy is high and in the worst case can lead to bad values.

In each graph for goal i, up until the $CQP(\{i\})$, we see a plateau in the *worst case* values - the ad hoc agent might waste a timestep on querying, but it will be certain of the teammate's goal by the time it reaches the toolbox and has to make a decision about the tool to pick up. After the CQP the worst-case increases, as the potential number of waiting steps, up until it reaches the end of Z_I where there is no ambiguity about the teammate's goal. When looking at the *average case*, querying at the CQP at timestep 6 has the lowest value. Earlier queries are sometimes redundant when the teammates chooses a path that disambiguates its goal before the CQP. Later queries might cause the ad hoc agent to wait at the toolbox until the goal of the teammate is clear. The bottom graph, where the teammate's goal is 3, is an interesting case since $Z_Q(\{3\}) = \emptyset$, so there is no gain from querying.

7.2 Query Content Effect on Cost

Figure 3 shows another methodical evaluation of all enumerated optimal plans, but instead of varying the timing, we varied the possible content of the queries the ad hoc agent can ask. In this experiment, the ad hoc agent can only ask one question. The x-axis shows the outcome of using different queries - *Never* means that no query is asked, *First(x)* and *CQP(x)* respectively mean that the query that was asked is "Is your goal workstation x?" for $x \in 1, 2, 3$ in the beginning of the episode or at the $CQP\{x\}$. *First(All)* and *CQP(All)* respectively mean that the query was "What is your goal?" in the beginning of the episode or at the $CQP\{G\}$ for all goals. The y-axis shows the average additional number of timesteps

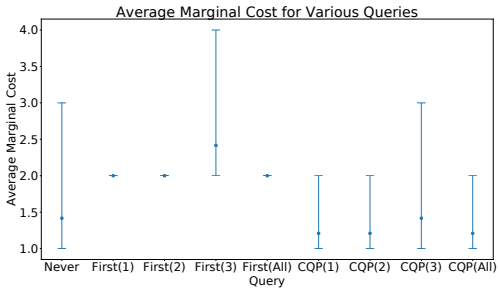

Figure 3: The effect of query content on the ad hoc agent's plan.

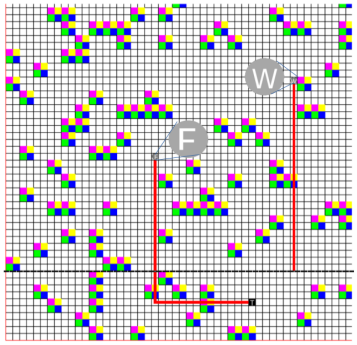

Figure 4: An instance of the domain used in our experiments. The fetcher (agent on the left) travels to the toolbox while the worker (agent on the right) moves downwards. When the fetcher arrives at the toolbox, it already knows the worker is not traveling to any of the stations (shown by the colored squares) above the dashed line since the worker has been moving downwards. Thus the fetcher should only query about stations below the dashed line.

compared to an optimal plan. As seen in this figure, the best strategy is to ask a query that disambiguates goals that share the highest wcd_T values — in our case, a query that disambiguates between workstations 1 and 2 at their CQP: $CQP(1), CQP(2)$, or $CQP(All)$.

7.3 Algorithmic Evaluation

Figure 5 shows an evaluation of our algorithm, that queries in the beginning of the joint Z_Q for any two goals (which in our case is when it reaches the toolbox). This algorithm was compared to the following baselines: *Random* - starts querying at a random timestep; *First* - starts querying at the first timestep; *Never* - never queries. The queries for the *First, Random,* and Z_Q strategies were chosen randomly from "Is your goal in G'?" for $G' \subseteq G$. In addition, we guaranteed that G' consists of half the worker's potential goals to ensure maximum information gain. The query "What is your goal?" is omitted as the marginal cost is constant (the cost of that query) when this query is allowed. We evaluated these approaches on 100 different domain instances, where in each instance we have a 50x50 grid with 400 stations. The stations were clustered in groups of 4 as 2 by 2 squares, with the location of each cluster chosen at random. Figure 4 shows an example of a domain instance. Each domain was tested with a randomly chosen target station, and the worker's plan was

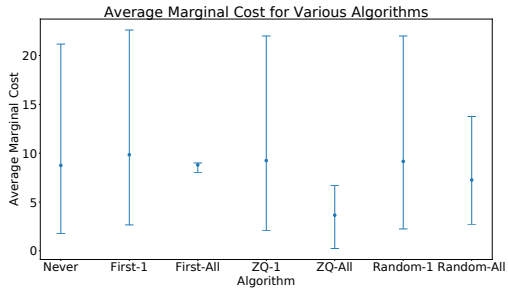

Figure 5: Average performance of different algorithms in 100 randomized tool-fetching simulations.

chosen randomly from the set of optimal plans. Each algorithm in this experiment has two variations: Z_Q*-1, First-1, Random-1* query once, while Z_Q*-All, First-All, Random-All* continue to ask new queries (each with half the remaining possible goals) at every time step after they have started until the teammate's goal is clear. As seen in this figure, the $Z_Q - All$ approach performs best, with an average of 3.65 steps more than a perfect plan (if the ad hoc agent had complete knowledge of the teammate's goal), where the next best approach that is not based on Z_Q takes an average of 7.25 steps more than a perfect plan. The average length of a perfect plan was 65 timesteps. The results for $Z_Q(All)$ were significantly better than from all others (p-values of the t-tests for each algorithm with $Z_Q(All)$ were < 0.01).

8 Conclusions

In this paper, we investigated a type of CAT problem, which we refer to as SOMALI CAT. We defined the querying potential of a domain and presented the boundaries at which querying in a domain can be beneficial. We introduced a testbed for SOMALI CAT, and our empirical results show the benefit of leveraging knowledge about these boundaries both in our running example and on randomly generated setups. This work is a first step towards a thorough investigation of CAT problems. We plan on extending this work to other scenarios inside and outside of the SOMALI CAT context, such as scenarios with stochastic transitions, where the task is repetitive rather than one-shot, or with suboptimal teammates.

Acknowledgements

This work has taken place in the Learning Agents Research Group (LARG) at UT Austin. LARG research is supported in part by NSF (CPS-1739964, IIS-1724157, NRI-1925082), ONR (N00014-18-2243), FLI (RFP2-000), ARO (W911NF-19-2-0333), DARPA, Lockheed Martin, GM, and Bosch. Peter Stone serves as the Executive Director of Sony AI America and receives financial compensation for this work. The terms of this arrangement have been reviewed and approved by the University of Texas at Austin in accordance with its policy on objectivity in research.

References

[Agmon and Stone, 2012] Noa Agmon and Peter Stone. Leading ad hoc agents in joint action settings with multiple teammates. In *AAMAS*, pages 341–348, 2012.

[Albrecht and Stone, 2017] Stefano V Albrecht and Peter Stone. Reasoning about hypothetical agent behaviours and their parameters. In *AAMAS*, pages 547–555, 2017.

[Albrecht and Stone, 2018] Stefano V Albrecht and Peter Stone. Autonomous agents modelling other agents: A comprehensive survey and open problems. *Artificial Intelligence*, 258:66–95, 2018.

[Amir *et al.*, 2016] Ofra Amir, Ece Kamar, Andrey Kolobov, and Barbara Grosz. Interactive teaching strategies for agent training. *IJCAI*, 2016.

[Barrett *et al.*, 2014] Samuel Barrett, Noa Agmon, Noam Hazon, Sarit Kraus, and Peter Stone. Communicating with unknown teammates. In *AAMAS*, pages 1433–1434, 2014.

[Bisson *et al.*, 2011] Francis Bisson, Froduald Kabanza, Abder Rezak Benaskeur, and Hengameh Irandoust. Provoking opponents to facilitate the recognition of their intentions. In *AAAI*, 2011.

[Cakmak and Thomaz, 2012] Maya Cakmak and Andrea L Thomaz. Designing robot learners that ask good questions. In *ACM/IEEE international conference on Human-Robot Interaction*, 2012.

[Chakraborty *et al.*, 2017] Mithun Chakraborty, Kai Yee Phoebe Chua, Sanmay Das, and Brendan Juba. Coordinated vs. decentralized exploration in multi-agent multi-armed bandits. In *IJCAI*, pages 164–170, 2017.

[Devin *et al.*, 2019] Coline Devin, Daniel Geng, Pieter Abbeel, Trevor Darrell, and Sergey Levine. Plan arithmetic: Compositional plan vectors for multi-task control. *arXiv preprint arXiv:1910.14033*, 2019.

[Epshteyn *et al.*, 2008] Arkady Epshteyn, Adam Vogel, and Gerald DeJong. Active reinforcement learning. In *Proceedings of ICML*, pages 296–303. ACM, 2008.

[Fern *et al.*, 2007] Alan Fern, Sriraam Natarajan, Kshitij Judah, and Prasad Tadepalli. A decision-theoretic model of assistance. In *IJCAI*, pages 1879–1884, 2007.

[Foerster *et al.*, 2016] Jakob Foerster, Ioannis Alexandros Assael, Nando de Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. In *NIPS*, pages 2137–2145, 2016.

[Griffith *et al.*, 2013] Shane Griffith, Kaushik Subramanian, Jonathan Scholz, Charles L Isbell, and Andrea L Thomaz. Policy shaping: Integrating human feedback with reinforcement learning. In *NIPS*, pages 2625–2633, 2013.

[Hernandez-Leal *et al.*, 2019] Pablo Hernandez-Leal, Bilal Kartal, and Matthew E Taylor. A survey and critique of multiagent deep reinforcement learning. *AAMAS*, pages 1–48, 2019.

[Howard, 1966] Ronald A Howard. Information value theory. *IEEE Transactions on systems science and cybernetics*, 1966.

[Kapourchali and Banerjee, 2019] Masoumeh Heidari Kapourchali and Bonny Banerjee. State estimation via communication for monitoring. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2019.

[Kautz and Allen, 1986] Henry A Kautz and James F Allen. Generalized plan recognition. In *AAAI*, 1986.

[Keren *et al.*, 2014] Sarah Keren, Avigdor Gal, and Erez Karpas. Goal recognition design. In *ICAPS*, 2014.

[Kulkarni *et al.*, 2019] Anagha Kulkarni, Siddharth Srivastava, and Subbarao Kambhampati. A unified framework for planning in adversarial and cooperative environments. In *AAAI*, 2019.

[Lowe *et al.*, 2017] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *NeurIPS*, pages 6379–6390, 2017.

[Mirsky *et al.*, 2018] Reuth Mirsky, Roni Stern, Kobi Gal, and Meir Kalech. Sequential plan recognition: An iterative approach to disambiguating between hypotheses. *Artificial Intelligence*, 260:51–73, 2018.

[Mordatch and Abbeel, 2018] Igor Mordatch and Pieter Abbeel. Emergence of grounded compositional language in multi-agent populations. In *AAAI*, 2018.

[Ravula *et al.*, 2019] Manish Ravula, Shani Alkoby, and Peter Stone. Ad hoc teamwork with behavior switching agents. In *IJCAI*, 2019.

[Roth *et al.*, 2006] Maayan Roth, Reid Simmons, and Manuela Veloso. What to communicate? execution-time decision in multi-agent pomdps. In *Distributed Autonomous Robotic Systems 7*, pages 177–186. Springer, 2006.

[Shvo and McIlraith, 2020] Maayan Shvo and Sheila A McIlraith. Active goal recognition. In *AAAI*, 2020.

[Singh, 1998] Munindar P Singh. Agent communication languages: Rethinking the principles. *Computer*, 31(12):40–47, 1998.

[Stone *et al.*, 2010] Peter Stone, Gal A Kaminka, Sarit Kraus, and Jeffrey S Rosenschein. Ad hoc autonomous agent teams: Collaboration without pre-coordination. In *AAAI*, 2010.

[Stone *et al.*, 2013] Peter Stone, Gal A Kaminka, Sarit Kraus, Jeffrey S Rosenschein, and Noa Agmon. Teaching and leading an ad hoc teammate: Collaboration without pre-coordination. *Artificial Intelligence*, 203:35–65, 2013.

[Torrey and Taylor, 2013] Lisa Torrey and Matthew Taylor. Teaching on a budget: Agents advising agents in reinforcement learning. In *AAMAS*, pages 1053–1060, 2013.

[Wang *et al.*, 2020] Rose E Wang, Sarah A Wu, James A Evans, Joshua B Tenenbaum, David C Parkes, and Max Kleiman-Weiner. Too many cooks: Coordinating multi-agent collaboration through inverse planning. *AAMAS*, 2020.