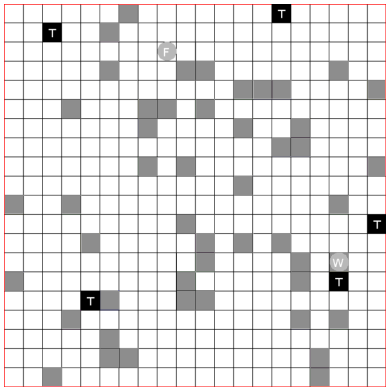

Figure 2: Example instance of the tool fetching domain. Workstations are shown by the grey boxes, toolboxes are shown by the black boxes with a T, the fetcher is shown as the gray circle labeled with an F, while the worker is shown as a gray circle labeled with a W.

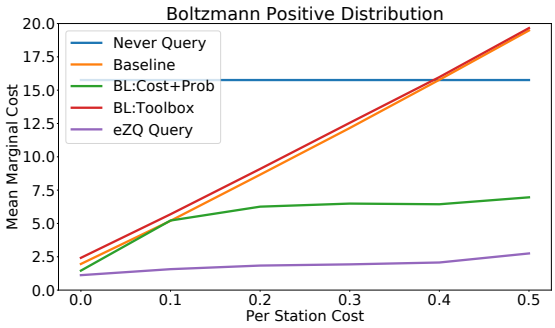

Figure 3: Per-station cost vs marginal cost of domain with Boltzmann distribution of the worker's distances to goals.

Experimental Setup

Previous work in SOMALI CAT introduced an experimental domain known as the tool fetching domain (**?**). This domain consists of an ego agent, the *fetcher*, attempting to meet a teammate, the *worker*, at some workstation with a tool. The worker needs a specific tool depending on which station is its goal, and the worker's goal and policy are unknown to the fetcher. It is the job of the fetcher to deduce the goal of the worker based on its actions, to pick up the correct tool from a toolbox and to bring it to the correct workstation. At each timestep, the agents can execute one ontic action each. In this work, the fetcher infers the worker's true goal by setting a goal's probability to zero and normalizing the belief distribution when observing a non-optimal action for that goal. Alternatively, the fetcher can query the worker with questions of the form "Is your goal one of the stations $g_1, g_2 ... g_N$?", where $g_1, \ldots, g_N \subseteq G$ is a subset of all workstations, and the worker replies truthfully. All queries are assumed to have a cost identical to moving one step, regardless of the content of the query. To show the benefits of our algorithm, we introduce 3 generalizations to the tool fetching domain that make the planning problem for the ego agent more complex to solve.

Multiple Z_Bs We allow multiple toolboxes in the domain. Each toolbox contains a unique set of tools, and only one tool for each station is present in a domain. Including this generalization means that each pair of goals may have different Z_B values, which makes determining query timing and content more challenging.

Non-uniform Probability We allow non-uniform probability distributions for assigning the worker a goal. For instance, goals may be assigned with probability corresponding to the Boltzmann distribution over the negative of their distances. This modification means that the worker is more likely to have goals that are closer to it. Including this generalization means that querying about certain goals may be more valuable than others, and the fetcher will have to consider this distribution when deciding what to query about.

Cost Model We allow for a more general cost model, where different queries have different costs. In particular, we consider a cost model where each query has some *base cost* and an additional cost is added for each station asked, a *per-station cost*. So for instance if queries have a base cost of 0.5 and a per-station cost of 0.1, then the query "Is your goal station 1, 3 or 5?" would have a cost of $0.5 + 3 * 0.1 = 0.8$. Including this generalization means that larger queries will cost more, and it may be more beneficial to ask smaller but less informative queries. Results used a cost of 0.5 when initiating a query and varied the per station cost of each query.

We compare the performance of Algorithm 1 (eZ_Q *Query*) against two baseline ego agents: one agent that never queries but always waits when it is uncertain about which action to take (*Never Query*). The other baseline is the algorithm introduced by Mirsky et al. (**?**), where the ego agent chooses a random query once inside a Z_Q (*Baseline*). In addition, we extended *Baseline* in two ways to make it choose queries in a more informed way. First by accounting for the changing cost of different queries, as well as for the probability distribution over the teammate's goals (*BL:Cost+Prob*) (**?**). The second method involves creating a set of stations that each ontic action would be optimal for, and then querying about the set with the median size of these sets. Intuitively, this method first attempts to disambiguate which toolbox to reach and then attempts to disambiguate the worker's station (*BL:Toolbox*). All methods take a NOOP action if they are uncertain of the optimal action and do not query. Additional details about the setup and the strategies can be found in the Appendix.

Results

We ran experiments in a 20×20 grid, with 50 workstations and 5 toolboxes. Locations of the stations, toolboxes, and agents in each domain instance are chosen randomly. All results are averaged over 100 domain instances.

Figure 2 shows an example of such a tool-fetching domain. We now compare the new algorithm with previous

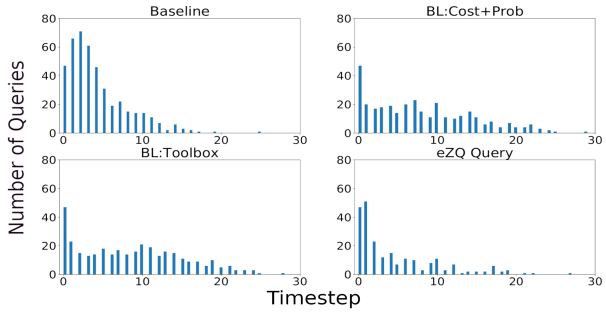

Figure 4: Histogram of number of queries (y-axis) executed per timestep (x-axis) by each method with the Boltzmann distribution of the worker's distances to goals and a per-station cost of 0.

work and the heuristics described above. We demonstrate that eZ_Q *Query* is able to effectively leverage the additional information from our generalizations to obtain a better performance over previous work and the suggested heuristics, in terms of marginal cost (Definition 7).

Figure 3 shows the marginal cost averaged over 100 domain instances with different per-station costs (x-axis) when the probability distribution of the goals of the worker is the Boltzmann distribution over the worker's distances to each goal. This distribution means that the worker is more likely to choose goals that are farther from it. eZ_Q *Query* performs similarly to the heuristics when the per-station cost is 0, but dramatically outperforms all other methods as this value increases. Additional results showing eZ_Q *Query* under additional goal distributions can be seen in the Appendix.

We also provide an analysis of how the eZ_Q *Query* method is achieving better performance than other methods. Figure 4 shows a histogram of the number of queries executed per timestep by each method. As shown in the histogram, eZ_Q *Query* tends to ask more queries in the earlier timesteps compared to *BL:Toolbox* and *BL:Cost+Prob*. This increase is because eZ_Q *Query* is focused on learning the worker's true goal with minimal cost and is better able to leverage information about the probability distribution over goals to make informed queries, and therefore finds the correct goal more quickly than other methods, but also takes longer before it knows an optimal action. In addition, we found that as the per-station cost increases from 0 to 0.5, the total number of queries executed by eZ_Q query over all simulations decreases by 23%, showing that eZ_Q *Query* executes fewer queries when the cost is higher.

Finally, there is an increased computational cost of using eZ_Q *Query* over other approaches and the proposed heuristics. While calculating EDp is expensive and takes several orders of magnitude longer than the rest of the querying algorithm (several hours per domain on average), these values can be computed a priori regardless of the teammate's actions. As such, the following results are under the assumption that the EDP computation is performed in advance, and the following time measurements do not include these offline computations. On average, all heuristic methods took < 0.23 seconds to complete each simulation, while eZ_Q *Query* took on average 8.9 seconds on an Ubuntu 16.04 LTS Intel core i7 2.5 GHz, with the genetic algorithm taking on average 6.1 seconds to run. In practice, the increased time should not be a major detriment. If a robot is communicating with either a human or another robot, the major bottleneck is likely to be the communication channel (e.g. speech, network speed, decision making of the other agent) rather than this time. In addition, when using a genetic algorithm for optimization as we do in this paper, the eZ_Q *Query* computation should only grow in terms of $O(|G|^2 log(|G|))$ with the number of goals (assuming that EDP is precomputed ahead of time and that the number of members and generations do not grow with the number of goals).

Discussion and Future Work

In this paper, we investigated a new metric to quantify ambiguity of teammate policies in ad hoc teamwork settings, by estimating the expected divergence point between different policies a teammate might posses. We then utilized this metric to construct a new ad hoc agent that reasons both about *when* it is beneficial to query, but also about *what* is beneficial to query about in order to reduce the ambiguity about its teammate's goal. Our empirical results show that regardless of the goal-choosing policy of the worker and a varying query cost model, eZ_Q *Query* remains more effective than any of the other methods tested, and even when querying is almost never beneficial, it is still able to adapt and obtain performance that is consistently better than *Never Query*.

The scope of this work is limited to SOMALI CAT problems. In addition, our current methods are designed to work in relatively simple environments with finite state spaces and a limited number of goals. However, the EDP formalization opens up new avenues for investigating other complicated SOMALI CAT scenarios and other CAT scenarios, such as those in which an agent can advise or share its beliefs with its teammates. We conjecture that the eZ_Q algorithm can be modified relatively easily to address such challenges, as long as the ego agent remains the initiator of the communication. For instance, EDP may be able to be calculated in domains with larger and continuous state spaces by leveraging more sophisticated RL techniques than the policy evaluation algorithm. It might be more challenging to extend this work to domains in which the teammate is the one to initiate the communication, as other works have investigated in the context of reinforcement learning agents (**??**). Nonetheless, this work provides the means to investigate collaborations in ad hoc settings in new contexts, while presenting concrete solutions for planning in SOMALI CAT settings.

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), ARL, 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