In *Plan, Activity, and Intent Recognition*,
Elsevier, Philadelphia, PA, USA, 2013.

Role-Based Ad Hoc Teamwork

Katie Genter[a], Noa Agmon[b], Peter Stone[a]

[a] *Department of Computer Science*
The University of Texas at Austin
Austin, TX, USA
{katie,pstone}@cs.utexas.edu
[b] *Department of Computer Science*
Bar Ilan University[1]
Ramat Gan, Israel
agmon@cs.biu.ac.il

Abstract

An ad hoc team setting is one in which teammates work together to obtain a common goal, but without any prior agreement regarding how to work together. We introduce a role-based approach for ad hoc teamwork, in which each teammate is inferred to be following a specialized role. In such cases, the role an ad hoc agent should select depends on its own capabilities and on the roles selected by its teammates. In this chapter we formally define methods for evaluating the influence of an ad hoc agent's role selection on the team's utility and show that use of these methods facilitates efficient calculation of the role yielding maximal team utility. We examine empirically how to choose the best suited method for role assignment and show that once an appropriate assignment method is determined for a domain, it can be used successfully in new tasks that the team has not encountered before. Unlike much of the rest of the book, this chapter does not focus on *methods* for recognizing the roles of the other agents. Rather, it examines the question of how to *use* successful role recognition towards successful multiagent decision-making.

Keywords: ad hoc teamwork, role-based ad hoc teamwork, multiagent teamwork, multiagent collaboration, multiagent planning

[1] This work was conducted while Noa Agmon was at The University of Texas at Austin.

1. Introduction

As software and robotic agents become increasingly common, there becomes a need for agents to collaborate with unfamiliar teammates. Consider a disaster situation where rescue robots from all around the world are brought in to assist with search and rescue. Ideally, these robots would be able to collaborate immediately — with little to no human assistance — to divide and conquer the necessary tasks according to their relative abilities. Some agents would locate victims, while other agents would lift fallen debris away from trapped victims. However, most existing agents are designed to only collaborate with known teammates that they were specifically pre-programmed to work with. As such, collaborating on the fly with unknown teammates is impossible for most current agents.

Ad hoc teamwork is a relatively new research area that examines this exact problem — how an agent ought to act when placed on a team with other agents such that there was no prior opportunity to coordinate behaviors. In ad hoc teamwork situations, several agents find themselves in a situation where they all have perfectly aligned goals, yet they have had no previous opportunity to coordinate their teamwork [1]. This problem arises quite often for humans, who tend to solve the problem quite naturally. However, autonomous agents—such as robots and software agents—do not currently handle this problem as gracefully.

Consider briefly that you recently arrived in a foreign country where you do not speak the local language. Now assume that you come across a pickup soccer game. After watching the game for a few minutes, some of the players on one team motion you to join. Despite not being able to verbally communicate with any of your teammates, you can still contribute to the team and work as a cohesive unit with your teammates. Through observing your teammates, you can quickly make a rough analysis of their strengths and weaknesses, and determine how you should play to best help the team.

Throughout this chapter we refer to the agents that make up a team as either *ad hoc agents* or *teammates*. Ad hoc agents are agents whose behavior we can control. Teammates, on the other hand, are agents that we have no control over, potentially because they were programmed by other groups or at different times such that future collaboration with our agents was unforeseeable.

In some team domains, such as search and rescue missions and many team sports, the team behavior can easily be broken down into *roles*. Under a *role-based approach* to ad hoc teamwork, each teammate is inferred to be following a specialized role that accomplishes a specific task or exhibits a particular behavior. Using this information, an ad hoc agent's main task is to decide which role to assume such that the team's performance is maximized. This decision is situation-specific: it depends on the task the team performs, the environment in which it acts, the roles currently performed by the team members, and the capabilities of the team members. One trivial approach is for an ad hoc agent to assume the role at which it is most *individually* capable. However, the choice of optimal role—one that results in highest *team* utility—rarely depends only on the ad hoc agent, but also on the ability and behavior of the other team members. Hence, an ad hoc agent will sometimes not adopt the role that it performs best if adopting a different role is optimal for the team. We examine the contribution of an ad hoc agent to the team by the measure of *marginal utility*, which is the increase in a team's utility when an ad hoc agent is added to the team and assumes a particular role. An *optimal mapping* of an ad hoc agent to a role is, therefore, one that maximizes the marginal utility, hence maximizing the contribution of the ad hoc agent to the team's utility.

In this chapter, we present a *role-based approach* for ad hoc teamwork. We begin by noting related work in Section 2. In Section 3, we formally define the role-based ad hoc teamwork problem. In Section 4, we emphasize the importance of accurate role recognition. In Section 5, we define several methods for modeling the marginal utility of an ad hoc agent's role selection as a function of the roles performed by the other teammates. Then we empirically show in a foraging domain that each method is appropriate for a different class of role-based tasks. In Section 6, we demonstrate that use of these methods can lead to efficient calculation of the role that yields maximal team utility. We then include an empirical examination in a more complex Pacman Capture-the-Flag domain of how to choose the best suited method for role assignment in a complex environment where it is not trivial to determine the optimal role assignment. Finally, we show that the methods we describe have a predictive nature, meaning that once an appropriate assignment method is determined for a domain, it can be used successfully in new tasks that the team has not encountered before and for which only limited experience is available. In Section 7 we conclude and present some avenues for future work.

Unlike much of the rest of the book, this chapter does not focus on methods of role recognition. Instead, it examines the question of how to use successful role recognition to assist in multiagent decision-making. Indeed, this work contributes towards answering the ad hoc teamwork challenge presented by Stone, Kaminka, Kraus, and Rosenschein [1]. Specifically, this ad hoc teamwork challenge is 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, Kaminka, Kraus, and Rosenschein laid out three abstract technical challenges. This chapter presents an approach towards solving one of these challenges — finding theoretically optimal and/or empirically effective algorithms for behavior — in role-based ad hoc teamwork settings.

2. Related Work

Although there has been much work in the field of multiagent teamwork, there has been little work towards getting agents to collaborate towards a common goal without pre-coordination. In this section, we review some selected work that is related to the role-based ad hoc teamwork approach presented in this chapter. Specifically, we consider work related to multiagent teamwork, multiagent plan recognition, and our two experimental domains.

2.1. Multiagent Teamwork

Most prior multiagent teamwork research requires explicit coordination protocols or communication protocols. Three popular protocols for communication and coordination — SharedPlans [2], Shell for TEAMwork (STEAM) [3], and Generalized Partial Global Planning (GPGP) [4] — all provide collaborative planning or teamwork models to each team member. Each of these protocols work well when all agents know and follow the protocol. However, in ad hoc teamwork settings we do not assume that any protocol is known by all agents, so protocols such as these cannot be successfully used.

Some multiagent teams are even designed to work specifically with their teammates in pre-defined ways, such as via "locker-room agreements" [5]. Specifically, a "locker-room agreement" is formed when there is a team synchronization period during which a team can coordinate their teamwork structure and communication protocols. This work divides the task space via the use of roles like we do, but our work differs in that we do not assume the availability of a team synchronization period.

Liemhetcharat and Veloso formally defined a weighted synergy graph that models the capabilities of robots in different roles and how different role assignments affect the overall team value [6]. They presented a team formation algorithm that can approximate the optimal role assignment policy given a set of teammates to choose from and a task. They applied this algorithm to simulated robots in the RoboCup Rescue domain and to real robots in a foraging task, and found that the resulting role assignments outperformed other existing algorithms. This work determines how to best form a team when given many agents to choose from, while our work determines how a particular agent should behave to best assist a preexisting team.

Wu, Zilberstein, and Chen present an online planning algorithm for ad hoc team settings [7]. Their algorithm constructs and solves a series of stage games, and then uses biased adaptive play to choose actions. They test their algorithm in three domains: cooperative box pushing, meeting in a 3×3 grid, and multi-channel broadcast. In these tests, they show they are able to perform well when paired with suboptimal teammates. Their work is different than ours in that we choose the best suited role assignment method and then assign the ad hoc agent to perform a role using the chosen role assignment method, whereas in their work they optimize each individual action taken by the ad hoc agent.

Bowling and McCracken examined the concept of "pick-up" teams in simulated robot soccer [8]. Similarly to us, they propose coordination techniques for a single agent that wants to join a previously unknown team of existing agents. However, they take a different approach to the problem in that they provide the single agent with a play book from which it selects the play most similar to the current behaviors of its teammates. The agent then selects a role to perform in the presumed current play.

Jones, Browning, Dias, Argall, Veloso and Stentz perform an empirical study of dynamically formed teams of heterogeneous robots in a multirobot treasure hunt domain [9]. They assume that all of the robots know they are working as a team and that all of the robots can communicate with one another, whereas in our work we do not assume that the ad hoc agent and the teammates share a communication protocol.

Han, Li and Guo study how one agent can influence the direction in which an entire flock of agents is moving [10]. They use soft control in a flocking model where each agent follows a simple control rule based on its neighbors. They present a simple model that works well in cases where the agents reflexively determine their behaviors in response to a larger team. However, it is not clear how well this work would apply to more complex role-based tasks such as those studied and discussed in our work.

2.2. Multiagent Plan Recognition

An ad hoc team player must observe the actions of its teammates and determine what plan or policy its teammates are using before determining what behavior it should adopt. In this work, we assume that the ad hoc agent is given the policy of each teammate so as to focus on the role selection problem. However, in many situations this is not a valid assumption, so recognizing the plan or policy of each teammate is an important part of solving the ad hoc teamwork challenge.

Barrett and Stone present an empirical evaluation of various ad hoc teamwork strategies [11]. In their work they show that efficient planning is possible using Monte Carlo Tree Search. Additionally, they show that an ad hoc agent can differentiate between its possible teammates on the fly when given a set of known starting models, even if these models are imperfect or incomplete. Finally, they show that models can be learned for previously unknown teammates. Unlike our work, this work does not take a role-based approach to solving the ad hoc teamwork problem. Instead, they evaluate the ability of various algorithms at generating ad hoc agent behaviors in an on-line fashion.

Sukthankar and Sycara present two approaches for recognizing team policies from observation logs, where a team policy is a collection of individual agent policies along with an assignment of individual agents to policies [12]. Each of their approaches — one model-based and one data-driven — seem generally well suited for application to the ad hoc teamwork challenge. Specifically, their approaches would be best suited for ad hoc teamwork settings that (1) are turn-based tasks, (2) do not require analysis of observation logs in real-time, and (3) do not require excessive amounts of training data to avoid over-fitting.

Zhuo and Li provide a new approach for recognizing multiagent team plans from partial team traces and team plans [13]. Specifically, given a team trace and a library of team plans, their approach is to first create a set of constraints and then solve these constraints using a weighted MAX-SAT solver. The required library of team plans might be difficult to obtain in some ad hoc teamwork settings though, so this approach is not well-suited for all ad hoc teamwork settings.

2.3. Experimental Domains

In this chapter we use two experimental domains: a Capture-the-Flag domain and a foraging domain. There has been previous research on multiagent teamwork in both the Capture-the-Flag domain and the foraging domain, and we discuss a few examples below. However, most of this work focuses on coordination between all teammates instead of coordination of one or more ad hoc agents with existing teammates, and hence does not address the ad hoc teamwork problem.

Blake, Sorensen, Archibald, and Beard present their Capture-the-Flag domain, which they implemented both physically and in simulation [14]. In their work, they focus on effective path planning for their robots as they navigate through a maze-like environment. Although similar to our Capture-the-Flag domain, their domain is different than ours because they assume that target coordinates will be communicated to the ground robots from an overhead flying robot. This communication might not be possible in an ad hoc teamwork setting because we can not assume that the teammates and the ad hoc agent will be able to communicate and exchange data.

Sadilek and Kautz used a real-world game of Capture-the-Flag to validate their ability to understand human interactions, attempted interactions, and intentions from noisy sensor data in a well defined multiagent setting [15]. Our work currently assumes that the actual behaviors of the teammates are provided to us, such that we can focus on determining the best behavior for the ad hoc agent. However, work such as Sadilek and Kautz's that focuses on recognizing interactions and inferring their intentions will be necessary to solve the complete ad hoc teamwork problem.

Mataric introduced the multiagent foraging domain, and focused on teaching agents social behaviors in this foraging domain [16]. Specifically, the foraging agents learned yielding and information sharing behaviors. In Mataric's work she assumed that all of the foraging robots are capable of communicating with each other. However, in our work we do not assume that the foraging robots share a communication protocol.

Lerman, Jones, Galstyan, and Mataric considered the problem of dynamic task allocation in a multiagent foraging environment [17]. They designed a mathematical model of a general dynamic task allocation mechanism. As long as all the agents use this mechanism, a desirable task division can be obtained in the absence of explicit communication and global knowledge. However, such an approach does not work in ad hoc teamwork settings because we can not assume that the ad hoc agent will be able to use the same mechanism as its teammates.

3. Problem Definition

In this chapter we introduce the *role-based* ad hoc teamwork problem, which is one that requires or benefits from dividing the task at hand into roles. Throughout this chapter we refer to the agents that make up a team as either *ad hoc agents* or *teammates*. Ad hoc agents are agents whose behavior we can control. Teammates, on the other hand, are agents that we have no control over, potentially because they were programmed by other groups or at different times such that future collaboration with our agents was unforeseeable.

Under a role-based ad hoc teamwork approach, the ad hoc agent first must infer the role of each teammate and then decide what role to assume such that the team's performance is maximized. The teammates do not need to believe or understand that they are performing a role. Indeed, the classification of teammates into roles is merely done so that the ad hoc agent can determine the best role for itself. Using the pickup soccer example presented in Section 1, under a role-based ad hoc teamwork approach you might determine what positions your teammates are playing and then adopt a position accordingly. You might adopt the most important position that is unfilled or the position that you are best at that is unfilled — or more likely you would adopt a position based on a function of these two factors.

Each teammate's role will be readily apparent to the ad hoc agent in many domains. For example, the goalie in a pickup soccer game is immediately apparent due to her proximity to her team's goal. However, in some domains it may take more extended observations for the ad hoc agent to determine the actual role of each teammate. In such domains, the role of each teammate may be determined with increasing certainty as observations are made regarding the agent's behavior and the areas of the environment it explores.

We assume that different roles have different inherent values to the team, and that each agent has some ability to perform each role. As such, an ad hoc agent must take into account both the needs of the team and its own abilities when determining what role to adopt. A team receives a score when it performs a task. Therefore, the goal of an ad hoc agent is to choose a role that maximizes the team score and hence maximizes the marginal utility of adding itself to the team. Hence, an ad hoc agent will sometimes not adopt the role that it performs best if adopting a different role is optimal for the team. Using the pickup soccer example, if your team is in dire need of a goalie, it may be best for the team if you play goalie even if you are better at other positions.

3.1. Formal Problem Definition

In this section, we define our problem more formally and introduce the notation that we use throughout the chapter.

Let a task d be drawn from domain D, where task d has m roles $R(d) = \{r_0, ..., r_{m-1}\}$. Each role r_i has an associated relative importance value v_i, where r_x is more critical to team utility than r_y if $v_x > v_y$. Each v_i is constant for a particular task d and set of potential roles $R(d)$, but might change if d or $R(d)$ were different. Let $\mathbf{A} = \{a_0, ..., a_{n-1}\}$ be the set of ad hoc agents and $\mathbf{B} = \{b_0, ..., b_{k-1}\}$ be the set of teammates such that $T = A \cup B$ is the team that is to perform task d. Each agent $t_j \in T$ has a utility $u(t_j, r_i) \geq 0$ for performing each role $r_i \in R(d)$. This utility $u(t_j, r_i)$ represents player t_j's ability at role r_i in task d.

Using the pickup soccer example, let domain D = {soccer}, task d = {game against a local boys high school team}, and $R(d)$ = {goalie, sweeper, stopper, outside back, center midfield, outside midfield, striker}. If A = {Katie} and B = {Jake, Noa, Peter, Sam, Todd}, then $T = A \cup B$ = {Jake, Katie, Noa, Peter, Sam, Todd}.

Let mapping $\mathbf{P} : B \rightarrow R(d)$ be the mapping of the teammates in B to roles $\{r_0, ..., r_{m-1}\}$ such that the teammates associated with role r_i are B_i^P, where $|B_i^P| = m_i^P$ and $B_0^P \oplus B_1^P \oplus ... \oplus B_{m-1}^P = B$. Without loss of generality, the agents in each B_i^P are ordered such that $u(b_j, r_i) \geq u(b_{j+1}, r_i)$. In the pickup soccer example, assume B_0^P = {Todd}, B_4^P = {Sam, Jake}, and B_6^P = {Peter, Noa}. This assumption means that Todd is playing goalie under mapping P, Sam and Jake are playing center midfield under mapping P (where Sam's utility for playing center midfield is greater than Jake's utility for playing center midfield), and Noa and Peter are playing striker under mapping P (where Peter's utility for playing striker is greater than Noa's utility for playing striker). Mapping P may be given fully or probabilistically, or it may need to be inferred via observation. However, it is important to note that ad hoc agents can not alter P by commanding the teammates to perform particular roles. For simplicity, we assume in this work that the ad hoc agents have perfect knowledge of mapping P.

Let mapping $\mathbf{S} : A \rightarrow R(d)$ be the mapping of the ad hoc agents in A to roles $\{r_0, ..., r_{m-1}\}$ such that the ad hoc agents performing role r_i are A_i^S, where $|A_i^S| = m_i^S$ and $A_0^S \oplus A_1^S \oplus ... \oplus A_{m-1}^S = A$. In the pickup soccer example, if $A_6^S = \{\text{Katie}\}$, then Katie is playing striker under mapping S. Additionally, let mapping $\mathbf{SP} : T \rightarrow R(d)$ be the combination of mappings S and P. As such, agents $T_i^{SP} = B_i^P \cup A_i^S$ are performing role r_i and $T_0^{SP} \oplus T_1^{SP} \oplus ... \oplus T_{m-1}^{SP} = T$. In other words, mapping SP is the association of *all* team members to the roles they are performing. Without loss of generality, the agents in each T_i^{SP} are ordered such that $u(t_j, r_i) \geq u(t_{j+1}, r_i)$. In the pickup soccer example, if T_6^{SP} ={Peter, Katie, Noa}, then Peter, Katie and Noa are all playing striker under mapping SP and Peter's utility for playing striker is greater than Katie's utility for playing striker, which is greater than Noa's utility for playing striker. A team score $U(W, d, T)$ results when the set of agents T perform a task d, with each $t_j \in T$ fulfilling some role $r_i \in R(d)$ under mapping W. Team score U is a function of individual agent utilities, but its precise definition is tied to the particular domain D and the specific task $d \in D$. For example, in the pickup soccer example, the team score might be the goal differential after ninety minutes of play. The marginal utility $MU(S, P)$ obtained by mapping S, assuming P is the mapping of the teammates in B to roles, is the score improvement obtained when each ad hoc agent $a_j \in A$ chooses role $r_S(a_j)$ under mapping S. Assuming that either B can perform the task or that $U(P, d, B) = 0$ when B cannot complete the task, marginal utility $MU(S, P) = U(SP, d, T) - U(P, d, B)$[2]. Going back to the pickup soccer example, the marginal utility obtained by mapping S is the difference in the expected score differential when B mapped by P plays a local boys high school team and the expected score differential when T mapped by SP plays a local boys high school team.

Given that mapping P is fixed, the role-based ad hoc team problem is to find a mapping S that maximizes marginal utility. The problem definition and notation provided above are valid for any number of ad hoc team agents. Hence, although for the remainder of this chapter we focus our attention on the case where there is only one ad hoc agent such that $A = \{a_0\}$, our general theoretical contributions can be applied in teams to which multiple ad hoc agents are added. For example, multiple ad hoc agents could coordinate and work together as a single "agent" and the theoretical results presented below would still hold. Note that multiple ad hoc agents could not each individually determine a mapping to a role that maximizes marginal utility

[2] MU is actually a function of d, B, T, P, and S, however throughout this chapter we use this more compact notation.

using the approach presented below, since each ad hoc agent would merely choose the role that would be optimal were it the only ad hoc agent to join the team. These mappings would not be guaranteed to collectively maximize marginal utility.

4. Importance of Role Recognition

The work presented in this chapter concerns how an ad hoc agent should select a role in order to maximize the team's marginal utility. However, in order to do that using the role-based ad hoc teamwork we present, the roles of the teammates must be correctly identified. This need for accurate role recognition is what ties this chapter to the other chapters in this book.

We do not focus on how to do role recognition in this chapter. Indeed, we assume that the ad hoc agents have complete knowledge concerning the roles of the teammates. As such, accurate role recognition is an important prerequisite for our work. Role recognition might be easy in some situations, but extremely difficult in others. However, role recognition always becomes easier as time passes and more experience is gained. For example, in the pick-up soccer example presented in Sections 1 and 3, it might be easy to recognize that a player positioned in the goal is playing the goalie role. However, a player positioned around midfield could be playing a variety of roles; one might be able to better determine his role as time passes and play continues.

If the ad hoc agents have imperfect or noisy knowledge of the mapping of the teammates to roles — in other words, if the role recognition is imperfect — the general processes presented in this chapter can still be useful. However, the resulting mapping of the ad hoc agents to roles may not maximize marginal utility when role recognition is imperfect. Such uncertainty can be dealt with formally if the agents have probabilistic knowledge of the mapping of the teammates to roles. Given a prior probability distribution over roles, a Bayesian approach could be used to modify the ad hoc agent's beliefs over time and to enable it to act optimally given its uncertain knowledge — including acting specifically so as to reduce uncertainty using a value of information approach.

In this work we assume that the roles of the agents remain fixed throughout the task. If the roles were to change, the ad hoc agent would first need a new process to identify such changes, perhaps by noticing that the teammates' observable behavior has changed. After detecting the change, the processes described in this chapter could be used to find a new mapping of the ad hoc agents to roles that maximizes marginal utility.

5. Models for Choosing a Role

The gold standard way for an ad hoc agent to determine the marginal utility from selecting a particular role, and hence determine its optimal role, is to determine $U(SP, d, T)$ for each possible role it could adopt. However, in practice, the ad hoc agent may only select *one* role to adopt. Hence, the ad hoc agent must *predict* its marginal utility for all possible roles and then select just one role to adopt. In this section we lay out three possible models with which the ad hoc agent could do this prediction based on the roles its teammates are currently filling. We also empirically verify that each model is appropriate for a different class of role-based tasks in the multiagent foraging domain described below.

In this foraging domain, a team of agents is required to travel inside a given area, detecting targets and returning them to a preset station [16]. We use a 50×50 cell map in our experiments, but an example 10×10 cell map can be seen in Figure 1. Each target can be acquired by a single agent that enters the target's cell as long as the agent is not already carrying a target and has the capability to collect targets of that type. To avoid random wandering when no collectable targets are visible, we assume that each agent has perfect information regarding target locations. Additionally, we allow multiple agents and targets to occupy the same cell simultaneously.

The goal of each agent is for the team to collect as many targets as possible, as quickly as possible. We consider a special case of the foraging problem in which targets are divided into two groups: red targets to the North and blue targets to the South, where the blue targets are worth twice as much to the team as the red targets. As such, the blue targets and agents that collect blue targets are randomly initialized on the lower half of the map, while the red targets and agents that collect red targets are randomly initialized on the upper half of the map.

As each model is presented, we also describe a foraging task in which the model has been empirically shown to be most appropriate using the methods described later in this chapter. Results of each model on each task are presented in Table 1 after all three models have been introduced.

For all of the models, except the Unlimited Role Mapping model, we assume that the ad hoc agent a_0 knows the utilities $u(b_j, r_i), \forall b_j \in B, r_i \in R(d)$

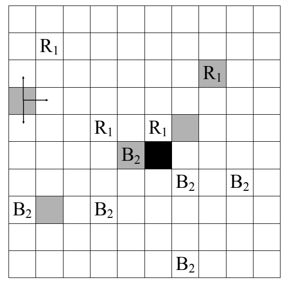


Figure 1: A sample 10×10 cell map of our foraging environment. Each target occupies one cell — blue targets are represented by B_2 and red targets are represented by R_1, where the subscript denotes the target's point value. Agents are represented by gray boxes, occupy only one cell at a time, and can move up, down, left, or right one cell each time step, as long as the move does not carry them off the board. The station to which targets should be delivered is shown as a black box.

and the mapping $P : B \rightarrow R(d)$. Additionally, when considering the following three models, note that the marginal utility of agent a_j choosing to fulfill role r_i under mapping S is often given by an algorithm $\mathsf{MU\text{-}X}$ (Algorithms 1, 2). In these cases, $\mathsf{MU\text{-}X}(a_j, r_i, P) = U(SP, d, T_i^{SP}) - U(P, d, B_i^P)$.

5.1. Unlimited Role Mapping Model:

Consider the multiagent foraging example presented above. Assume there are 100 red targets and 5000 blue targets in a constrained area with limited time (50 time steps) for the agents to collect targets. This example is Task A in Table 1. Due to the ample amount of valuable blue targets and the limited time, it would never be optimal for an agent to collect red targets instead of blue targets: the blue targets will be surrounding the base after the agent delivers its first target, and hence easy to collect and deliver quickly, whereas the red targets will be less numerous and likely farther from the base and slower to collect and deliver. Hence, in tasks such as this one, the benefit the team receives for an agent performing a role does *not* depend on the roles fulfilled by its teammates.

In such a case, the contribution to the team of an agent t_j performing role r_i is simply the agent's utility $u(t_j, r_i)$ at role r_i multiplied by the value of the role v_i. As such, the team utility can be modeled as

$$U(SP, d, T) = \sum_{i=0}^{m-1} rs_i * v_i \tag{1}$$

where

$$rs_i = \sum_{t_j \in T_i^{SP}} u(t_j, r_i) \tag{2}$$

Note that in this model, agent utility $u(t_j, r_i)$ for performing each role r_i and the importance v_i of each role r_i are parameters that can be tuned to match the characteristics of a particular task.

Theorem 1 describes the optimal role mapping under this model.

Theorem 1. *In Unlimited Role Mapping tasks, mapping S, under which a_0 chooses the role r_i that obtains $\underset{0 \leq i \leq m-1}{\operatorname{argmax}}(u(a_0, r_i) * v_i)$, maximizes marginal utility such that $\forall S' \neq S \ MU(S', P) \leq MU(S, P)$.*

5.2. Limited Role Mapping Model:

Returning to the multiagent foraging example, assume that there are 1000 red targets and 5000 blue targets in a constrained environment with limited time (50 time steps) for the agents to collect the targets. Also assume that there are rules prohibiting an agent from collecting the valuable blue targets if there is not at least one other agent also collecting blue targets or if more than three other agents are already collecting blue targets. This task is Task B in Table 1. Due to the ample amount of valuable blue targets, the ad hoc agent should collect blue targets if the conditions for being able to collect blue targets are satisfied. However, if the conditions are not satisfied, the ad hoc agent should default to collecting the less valuable red targets.

In tasks such as this, each role r_i has an associated r_i^{min} value and r_i^{max} value that represent the minimum and maximum number of agents that should perform role r_i. For all i, let $0 \leq r_i^{min} \leq r_i^{max} \leq n$. If the number of agents performing role r_i is less than r_i^{min}, then the team gains no score from their actions. On the other hand, if the number of agents performing role r_i is greater than r_i^{max}, then only the r_i^{max} agents with highest utility, $T_i^{SP}[r_i^{max}]$, will be taken into account when calculating the team score. As such, the team utility for Limited Role Mapping tasks can be modeled as

$$U(SP, d, T) = \sum_{i=0}^{m-1} rs_i * v_i \tag{3}$$

where

$$rs_i = \begin{cases} \sum_{t_j \in T_i^{SP}} u(t_j, r_i) & \text{if } r_i^{min} \leq m_i^{SP} \leq r_i^{max} \\ \sum_{t_k \in T_i^{SP}[r_i^{max}]} u(t_k, r_i) & \text{if } m_i^{SP} > r_i^{max} \\ 0 & \text{if } m_i^{SP} < r_i^{min} \end{cases}$$

The function MU-1(a_j, r_i, P) displayed in Algorithm 1 gives the marginal utility obtained from the ad hoc agent a_j choosing to perform role r_i, where the current mapping of teammates to roles is described by P. In this model, agent utility $u(t_j, r_i)$ for performing each role r_i, the importance v_i of each role r_i, and the minimum and maximum number of agents that should perform each role r_i are all tunable model parameters.

Function MU-1(a_j, r_i, P) uses some special notation. Specifically, let $posB(a_j, r_i)$ denote the 0-indexed position in T_i^{SP} that the ad hoc agent a_j occupies. Additionally, let $T_i^W(num)$ denote the agent that is performing role r_i under mapping W with the num highest utility on role r_i. For example, if agents A, B, C, and D are performing role R under mapping Y with the following utilities for role R: $A = 1$, $B = 2$, $C = 3$, and $D = 4$, then $T_R^T(0) = D$, $T_R^T(1) = C$, and $T_R^T(2) = B$.

Algorithm 1 MU-1(a_j, r_i, P)

if $m_i^P + 1 < r_i^{min}$ **then**
  return 0
else
  if $m_i^P + 1 = r_i^{min}$ **then**
    return $\sum_{t_j \in T_i^{SP}} u(t_j, r_i) * v_i$
  else
    if $r_i^{max} < m_i^P + 1$ **then**
      if $posB(a_j, r_i) \leq r_i^{max}$ **then**
        return $u(a_j, r_i) * v_i - u(T_i^P(r_i^{max}), r_i) * v_i$
      else
        return 0
    else
      return $u(a_j, r_i) * v_i$

Theorem 2 describes the optimal role mapping for the ad hoc agent under this model.

Theorem 2. *In Limited Role Mapping tasks, mapping S, under which a_0 chooses the role r_i that obtains $\underset{0 \le i \le m-1}{\text{argmax}} \mathsf{MU\text{-}1}(a_0, r_i, P)$, maximizes marginal utility such that $\forall S' \neq S \; MU(S', P) \leq MU(S, P)$.*

5.3. Incremental Value Model:

Continuing the multiagent foraging example, consider a task in which there are 500 blue targets and 5000 red targets in a constrained environment with limited time (50 time steps) for the agents to collect the targets. This example is Task C in Table 1. Since the time to collect targets is limited and there are more red targets than blue targets, the optimal role will sometimes be one that collects less plentiful but more valuable blue targets and sometimes be one that collects less valuable but more plentiful red targets. Collecting the less valuable red targets may be optimal if there are many other agents collecting blue targets and few other agents collecting red targets. This is because if there are many other agents collecting blue targets, the blue targets close to the base will be quickly collected, which forces all of the agents collecting blue targets to venture farther away from the base in order to collect blue targets. In such a case, collecting less valuable red targets may prove to be optimal since competition for them is less fierce and hence they can be collected and returned to the station more frequently.

In tasks like this, the value added by agents performing a role may not be linearly correlated with the number of agents performing that role. As such, the team utility in incremental value tasks can be modeled as

$$U(SP, d, T) = \sum_{i=0}^{m-1} rs_i * v_i \tag{4}$$

where

$$rs_i = \sum_{t_j \in T_i^{SP}} u(t_j, r_i) * F(i, j) \tag{5}$$

In particular, we consider the following three functions F—each with two parameters that can be tuned to match the characteristics of a particular task—that describe how the value added to the team by each subsequent agent performing a role incrementally increases or decreases as more agents perform that role.

Logarithmic Function $F(i, j) = \log_{j+1}(x_i) + k_i$, where k_i represents the amount added to the role score rs_i for each agent performing role r_i

and x_i sets the pace at which the function decays for agents performing r_i.

Exponential Function $F(i,j) = g_i^{(j/t_i)}$, where g_i is the growth factor and t_i is the time required for the value to decrease by a factor of g_i—both for each agent performing role r_i.

Sigmoidal Function $F(i,j) = \frac{1}{1+e^{s_i*(j+b_i)}}$ where s_i determines the sharpness of the curve and b_i dictates the x-offset of the sigmoid from the origin for each agent performing role r_i.

The function MU-2(a_j, r_i, P) displayed in Algorithm 2 gives the marginal utility obtained when an ad hoc agent a_j chooses to perform role r_i, where the current mapping of teammates to roles is described by P. Remember from function MU-1(a_j, r_i, P) that $posB(a_j, r_i)$ denotes the 0-indexed position in T_i^{SP} that the ad hoc agent a_j occupies. In this model, agent utility $u(t_j, r_i)$ for performing each role r_i, the importance v_i of each role r_i, and the parameters used in function F are all tunable parameters. Note that although we generally assume benefit is obtained as additional agents join a team, our models can also handle the case where additional agents add penalty as they join the team.

Algorithm 2 MU-2(a_j, r_i, P)

if $m_i^P = 0$ **then**
  return $v_i * (u(a_j, r_i) * F(j, 1))$
else
  if $posB(a_j, r_i) = m_i^P$ **then**
    return $v_i * u(a_j, r_i) * F(j, m_i^P + 1)$
  else
    return $v_i * u(a_j, r_i) * F(j, posB(a_j, r_i) + 1) - \sum_{y=b_{posB(a_j,r_i)}}^{m_i^P - 1} (u(b_y, r_i) * v_i * F(j, posB(b_y, r_i) + 1) - u(b_y, r_i) * v_i * F(j, posB(b_y, r_i) + 2))$

Theorem 3 describes the optimal role mapping for the ad hoc agent under this model.

Theorem 3. *In Incremental Value tasks, mapping S, under which a_0 chooses the role r_i that obtains $\underset{0 \le i \le m-1}{\operatorname{argmax}} \text{MU-2}(a_0, r_i, P)$, maximizes marginal utility such that $\forall S' \neq S$ $MU(S', P) \le MU(S, P)$.*

5.4. Empirical Validation of Models

As each model was presented, a foraging task in which the model has been empirically shown to be most appropriate was also described. Results of each model on each task are presented in Table 1. Task D — a task with 5000 blue targets, 5000 red targets, limited time (50 time steps), and crowding penalties that only let one agent move from each cell in a time step — is also included to represent a case where the limited model and two of the incremental models do poorly.

Model	**Task A**	**Task B**	**Task C**	**Task D**
Unlimited	**0/36**	1/18	14/22	0/36
Limited	3/33	**1/26**	6/30	9/27
Incremental w/ Logarithmic	0/36	1/19	**3/33**	8/27
Incremental w/ Exponential	0/36	1/20	**2/34**	8/27
Incremental w/ Sigmoidal	0/36	1/18	14/22	0/36

Table 1: The number of statistically significant incorrect/correct decisions made by an ad hoc agent agent under each model in four different tasks presented throughout Section 5.

Throughout this chapter, we evaluate models based on how often they lead an ad hoc agent to make the "correct" decision about which role to assume. An ad hoc agent's decision to perform role r_1 instead of r_2 is "correct" if empirical data shows that performing r_1 yields a team score that is better by a statistically significant margin than the team score obtained by performing r_2. Likewise, an ad hoc agent's decision to perform role r_1 instead of r_2 is "incorrect" if empirical data shows that performing r_1 yields a team score that is worse by a statistically significant margin than the team score obtained by performing r_2. If the margin is not statistically significant, then the decision is not counted as correct or incorrect. We determine statistical significance by running a two-tailed Student's t-Test assuming two-sample unequal variance.

As seen in Table 1, each model presented earlier in this section is indeed appropriate for some task. Interestingly enough, no model is best — or worst — for all tasks. In fact, each model performs poorly in at least one task. Hence, this experiment serves to show that each of the models presented is worth considering. Tasks C and D of this experiment also serve to highlight the differences between the incremental model with sigmoidal function and the incremental model with logarithmic and exponential functions.

6. Model Evaluation

The role-based ad hoc teamwork problem lies in determining what role an ad hoc agent should select when faced with a *novel* teamwork situation. Hence, the main questions in terms of role selection are: given a task in a particular environment, how should the correct model be selected? Additionally, once a model is selected, how should we determine reasonable parameters for the model given limited gold standard data? Answering these questions makes substantial progress towards solving the role-based ad hoc teamwork problem. Hence, in this section we examine both of these questions in the Pacman Capture-the-Flag environment.

6.1. Pacman Capture-the-Flag Environment

We empirically examine each of the three models described above in a Capture-the-Flag style variant of Pacman designed by John DeNero and Dan Klein[18]. The foraging domain used earlier in the paper was a simple and easily configurable domain. However, we move to the Pacman domain now both as an example of a more complex domain and to validate that our approach works in multiple domains.

The Pacman map is divided into two halves and two teams compete by attempting to eat the food on the opponent's side of the map while defending the food on their side. A team wins by eating all but two of the food pellets on the opponent's side or by eating more pellets than the opponent before three thousand moves have been made. When a player is captured, it restarts at its starting point.

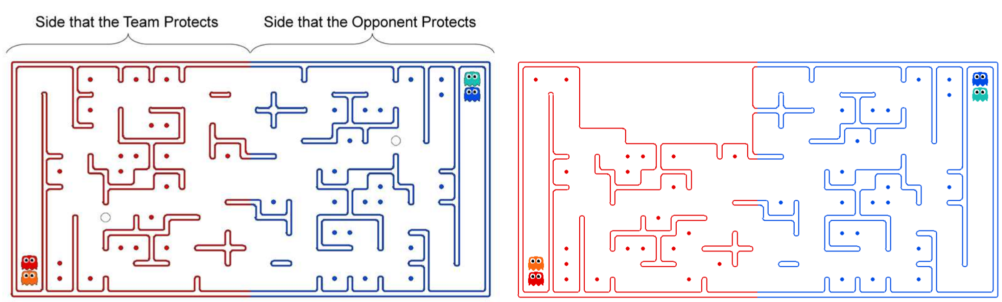
The result of each game is the difference between the number of pellets protected by the team and the number of pellets protected by the opponent—we refer to this as the *score differential*. Wins result in positive score differentials, ties result in zero score differentials, and losses result in negative score differentials. High positive score differentials indicate that the team dominated the opponent, while score differentials closer to zero indicate that the two teams were well matched. We mainly care whether we win or lose, so we transform each score differential using a sigmoid function in order to emphasize differences in score differentials close to zero. We input the score differential from each game into the sigmoid function $\frac{1}{1+e^{-0.13*\text{scoreDifferential}}}$ to obtain *gold standard data*. We examined different values for the multiplicand and found that 0.13 yielded the most representative score differential spreads in the three tasks presented below.

In each experiment we consider two roles that could be performed: $R = \{\text{offense, defense}\}$. Offensive players move toward the closest food on the opponent's side, making no effort to avoid being captured. Defensive players wander randomly on their own side and chase any invaders they see. These offensive and defensive behaviors are deliberately suboptimal, as we focus solely on role decisions given whatever behaviors the agents execute when performing their roles.

We consider the opponents and map to be fixed and part of the environment for each experiment. Half of the opponents perform defensive behaviors and half perform offensive behaviors. Additionally, all of the agents run either the offensive or defensive behavior just described. As such, all agents performing a particular role have the same ability for performing that role. In other words, for agents T_i^{SP} performing role r_i, $u(t_0, r_i) = \ldots = u(t_{m_i^P - 1}, r_i)$.

6.2. Determining the Best-Suited Model

We use three tasks to determine which of the models best represents the marginal utility of a role selection for the Pacman Capture-the-Flag environment. In particular, a *task* is defined by the number of opponents and the map. The first task "vs-2" is against two opponents on the "Basic" map shown in Figure 2(a), the second task "vs-6" is against six opponents on the "Basic" map, and the third task "vs-2-SmallDefense" is against two opponents on the "SmallDefense" map shown in Figure 2(b).



(a) "Basic" Map (b) "SmallDefense" Map

Figure 2: Maps used to determine which of the models best represents the marginal utility of a role selection for the Pacman Capture-the-Flag environment.

In order to decide which of the models is most representative of the marginal utility of a role selection in the Pacman Capture-the-Flag environment, we first gather full sets of *gold standard data*. In particular, in each task we gather scores over one thousand games for each team of zero to six offensive agents and zero to six defensive agents (i.e., 49 teams). Then we calculate the gold standard data for each team by putting the score differential from each of the one thousand games through the sigmoidal function given above and then averaging the results. The gold standard data from the "vs-2" environment is shown in Table 2. Note that 0.09 is the worst possible gold standard performance, and corresponds to obtaining 0 pellets and losing all 18 pellets to the opponent. Likewise, 0.88 is the best possible gold standard performance, and corresponds to obtaining 18 pellets and losing no pellets to the opponent.

We then use the gold standard data to determine the *gold standard decision* of whether an ad hoc agent should perform an offensive role or a defensive role on any team composed of zero to five offensive agents and zero to five defensive agents in each of the three tasks. To determine the gold standard decision of whether it is better for the ad hoc agent to perform an offensive or defensive role when added to a particular team we look at whether the gold standard data is higher for the team with one extra defensive player or the team with one extra offensive player. If the former is true, then the gold standard decision is for the ad hoc agent to play defense. Likewise, if the latter is true, then the gold standard decision is for the ad hoc agent to play defense offense. We determine whether a gold standard decision is statistically significant by running a two-tailed Student's t-Test assuming two-sample unequal variance.

As an example, consider the gold standard data from the "vs-2" environment that is shown in Table 2. The gold standard decision of whether an ad hoc agent should perform an offensive role or a defensive role on a team composed of two offensive agents and one defensive agent can be determined by considering whether the gold standard data for a team with two offensive agents and two defensive agents is greater than or less than the gold standard data for a team with three offensive agents and one defensive agent. By looking at the gold standard data from the "vs-2" environment shown in Table 2, we can see that the gold standard data for a team with two offensive agents and two defensive agents is 0.75, while the gold standard data for a team with three offensive agents and one defensive agent is 0.71. Since the gold standard data for a team with two offensive agents and two defensive agents is greater than the gold standard data for a team with three offensive agents and one defensive agent, the gold standard decision regarding a team composed of two offensive agents and one defensive agent is to perform a defensive role.

	0d	**1d**	**2d**	**3d**	**4d**	**5d**	**6d**
0o	0.09 (+o)	0.09 (+o)	0.09 (+o)	0.13 (+o)	0.23 (+o)	0.31 (+o)	0.36
1o	0.29 (+d)	0.49 (X)	0.64 (+o)	0.74 (+o)	0.79 (+o)	0.81 (+o)	0.82
2o	0.42 (+d)	0.63 (+d)	0.75 (+d)	0.81 (+d)	0.83 (X)	0.85 (X)	0.86
3o	0.54 (+d)	0.71 (+d)	0.80 (+d)	0.83 (+d)	0.85 (X)	0.85 (X)	0.86
4o	0.56 (+d)	0.74 (+d)	0.81 (+d)	0.84 (+d)	0.85 (+d)	0.87 (X)	0.87
5o	0.61 (+d)	0.75 (+d)	0.83 (+d)	0.84 (+d)	0.86 (X)	0.87 (+d)	0.88
6o	0.64	0.79	0.83	0.86	0.87	0.88	0.88

Table 2: Rounded gold standard data and decisions from the "vs-2" environment. The rows represent the 0...6 agents performing an offensive role, while the columns represent the 0...6 agents performing a defensive role. A '+o' ('+d') decision means that the ad hoc agent should adopt an offensive (defensive) role if added to a team with teammates performing the roles indicated by the row and column. An 'X' decision means that the decision of which role to perform was not statistically significant at $p = 0.05$.

Once we calculate the gold standard decisions for the ad hoc agent in each of the three tasks, we can determine which of the three models best captures the actual marginal utility of role selection in each task. First, we input the gold standard data and the model function into Matlab's lsqcurvefit algorithm (which uses the trust region reflexive least squares curve fitting algorithm) and obtain *fitted parameters* for the model function. The fitted parameters vary in type and number for each of the three models, but always include the role importance value v_i, the agent's utility $u(a_j, r_i)$ at performing role v_i, and parameters of the model function—all for each role $r_i \in R(d)$. The obtained fitted parameters for each of the models in the "vs-2", "vs-6", and "vs-2-SmallDefense" tasks can be found in Tables 3, 4, 5, 6, and 7. We use the fitted parameters to calculate *fitted results* for teams of zero to six offensive agents and zero to six defensive agents. Lastly, we translate these fitted results into *fitted decisions* using the same methodology used to translate the gold standard score differentials into gold standard decisions.

Now that we have gold standard decisions for each of the three tasks and fitted decisions for all three models in the three tasks, we compare the number of times the gold standard decision (for example, '+o') is statistically significant but does not match the fitted decision for a particular team arrangement (for example, '+d')—in other words, the number of times the ad hoc agent made an *incorrect decision*.

Parameter	Initial	vs-2	vs-6	vs-2-SmallDefense
$u(*, offense)$	1	0.3456	0.2277	0.3508
$u(*, defense)$	1	0.3010	0.2918	0.2827
$v_{offense}$	1	0.3456	0.2277	0.3508
$v_{defense}$	1	0.3010	0.2918	0.2827

Table 3: Obtained fitted parameters for the Unlimited Role Mapping Model.

Parameter	Initial	vs-2	vs-6	vs-2-SmallDefense
$u(*, offense)$	1	0.6764	0.2937	0.5437
$u(*, defense)$	1	0.2924	0.2926	0.2849
$v_{offense}$	1	0.6764	0.2937	0.5437
$v_{defense}$	1	0.2924	0.2926	0.2849
$r^{min}_{offense}$	1	1	1	1
$r^{min}_{defense}$	1	1	1	1
$r^{max}_{offense}$	3	1.2876	3.1476	2.0388
$r^{max}_{defense}$	3	3.3742	4.7504	3.2672

Table 4: Obtained fitted parameters for the Limited Role Mapping Model.

As is apparent from Table 8, all three incremental model functions perform rather well. Unfortunately we have yet to discover any clear insight as to when each of the incremental model functions are most appropriate. Hence, for now, it seems to be something that must be determined empirically for each new domain using gold standard data.

In this Pacman domain, as can be seen in Table 8, the exponential and sigmoidal functions of the incremental model make the fewest incorrect decisions across the three tasks. Hence, we conclude that in the Pacman Capture-the-

Parameter	Initial	vs-2	vs-6	vs-2-SmallDefense
$u(*, offense)$	1	0.5916	1.5926	0.4655
$u(*, defense)$	1	0.5650	0.5630	0.5164
$v_{offense}$	1	0.8081	0.1355	1.7277
$v_{defense}$	1	0.3675	0.1322	0.6257
$k_{offense}$	1	-0.5163	-0.3232	-0.1447
$k_{defense}$	1	-0.2510	0.4069	-0.0622
$x_{offense}$	1	2.4560	1.6866	1.7287
$x_{defense}$	1	1.9180	1.4599	1.3864

Table 5: Obtained fitted parameters for the Logarithmic Incremental Value Model.

Parameter	Initial	vs-2	vs-6	vs-2-SmallDefense
$u(*, offense)$	1	1.1761	0.3837	1.0742
$u(*, defense)$	1	0.3398	0.8327	0.1804
$v_{offense}$	1	1.3387	0.4658	1.2690
$v_{defense}$	1	0.5712	0.1280	0.8254
$g_{offense}$	1	0.9123	0.6285	0.2809
$g_{defense}$	1	0.5510	0.8831	0.7534
$t_{offense}$	1	0.0706	0.9366	1.0924
$t_{defense}$	1	1.3116	1.1912	0.6606

Table 6: Obtained fitted parameters for the Exponential Incremental Value Model.

Parameter	Initial	vs-2	vs-6	vs-2-SmallDefense
$u(*, offense)$	1	1.8120	0.7068	1.9380
$u(*, defense)$	1	1.7941	0.5136	0.6841
$v_{offense}$	1	1.7544	0.6969	1.6301
$v_{defense}$	1	0.2533	0.4558	0.5369
$s_{offense}$	1	1.3934	0.5801	1.2311
$s_{defense}$	1	0.5263	0.1577	0.4917
$b_{offense}$	1	0.3318	1.1634	0.5131
$b_{defense}$	1	0.9327	1.3573	1.1455

Table 7: Obtained fitted parameters for the Sigmoidal Incremental Value Model.

Flag domain, at least on the maps and opponents we studied, the incremental model using either an exponential function or a sigmoidal function most accurately models team utility. However, to conclude this we generated a full set of gold standard data for each of the three tasks, amounting to 49,000 games per task and used this data to fit the parameters of the model. Next we consider how to use the chosen model for predictive modeling when substantially less gold standard data is available.

6.3. Predictive Modeling

The main research problem in role-based ad hoc teamwork is how to choose a role for the ad hoc agent that maximizes marginal utility. This problem is particularly important when the ad hoc agent is placed in a situation it has never encountered before.

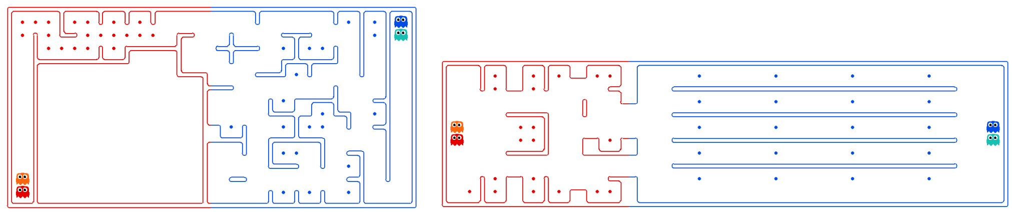
Once a model type has been selected for a domain, the ad hoc agent can use this model to *predict* the marginal utility of role selection on new tasks in this domain for which we have limited gold standard data. Essentially

Model	vs-2	vs-6	vs-2-Small Defense
Unlimited Role Mapping	19/10	8/21	14/16
Limited Role Mapping	3/26	10/19	3/27
Logarithmic Incremental Value	3/26	2/27	1/29
Exponential Incremental Value	**1/28**	**1/28**	**1/29**
Sigmoidal Incremental Value	**0/29**	**1/28**	**2/28**

Table 8: The number of statistically significant incorrect/correct decisions made by the ad hoc agent in each of the three tasks. Fewer incorrect and greater correct decisions is desirable.

we want to be able to determine how the ad hoc agent should behave in a new task—including never seen before situations—without the expense of gathering substantial amounts of gold standard data for every scenario. We do this by choosing fitted parameters for the new task based on the data that is available. Remember that fitted parameters can be obtained by inputting the gold standard data and the chosen model function into Matlab's lsqcurvefit algorithm, as this will fit the chosen model to the limited gold standard data using a least squares curve fitting algorithm. Then these fitted parameters can be used to calculate fitted results and fitted decisions, which represent the decisions chosen by the agent given each possible set of teammates.

Below we evaluate the accuracy of the incremental value model in our Pacman domain on multiple tasks when various amounts of randomly selected data is available for choosing fitted parameters. We use two new tasks in this section, both against two opponents. One task "vs-2-alley" is on the "AlleyDefense" map shown in Figure 3(a) and the other task "vs-2-33%" is on the "33%Defense" map shown in Figure 3(b). Both the "AlleyDefense" and "33%Defense" maps include a smaller defensive area than offensive area for the team that the ad hoc agent is added to, but the alley in "AlleyDefense" calls for the ad hoc agent to behave very differently than in the "33%Defense" map where the opponent's food pellets are relatively easy for the ad hoc agent's team to capture. Specifically, in the "33%Defense" map it is desirable—up to a certain threshold—to add an offensive agent as long as there is at least one defensive agent, whereas in the "AlleyDefense" map it is desirable to have substantially more defensive agents than offensive agents as long as there is at least one offensive agent.



(a) "AlleyDefense" Map (b) "33%Defense" Map

Figure 3: The maps used for the predictive modeling tasks.

Consider the case in which the ad hoc team agent is given, either through experience or observation, randomly selected data points that represent some sparse experience in a task, where a data point consists of the number of agents fulfilling each role and the average gold standard data calculated over 25 games. We choose 25 games because this proved to be an appropriate trade-off between the time required to collect data and the value of minimizing incorrect predictions. However, in practice the agent will usually not get to determine how many games it receives information about, and instead must do the best it can with whatever experience it is given. Note that if only one data point is used to fit the model, then score differentials from 25 games are required. Likewise, use of ten data points requires 250 (10*25) games. Even if all forty-nine data points are used, only 1,225 (49*25) games are required. To put these game numbers in perspective, we can usually run 500 games on our high-throughput computing cluster in under five minutes.

We evaluate the prediction accuracy of each of the three function variations of the incremental value model on two tasks for which we have limited data ranging from one to forty-nine randomly selected data points. In this experiment we endeavor to determine how many data points are needed to obtain reasonable prediction accuracy — in other words, we want to find the point at which it might not be worth obtaining additional data points. Note that since a data point can not be selected more than once, all data points are being used when forty-nine data points are selected. Prediction accuracy is reported as the number of statistically significant incorrect predictions made by the model. Figure 4 shows the accuracy of each variation of the chosen model on the "vs-2-alley" task, while Figure 5 shows accuracy on the "vs-2-33%" task, both when given varying amounts of randomly selected data points calculated from 25 games.

As would be expected, the accuracy of each variation of the chosen model

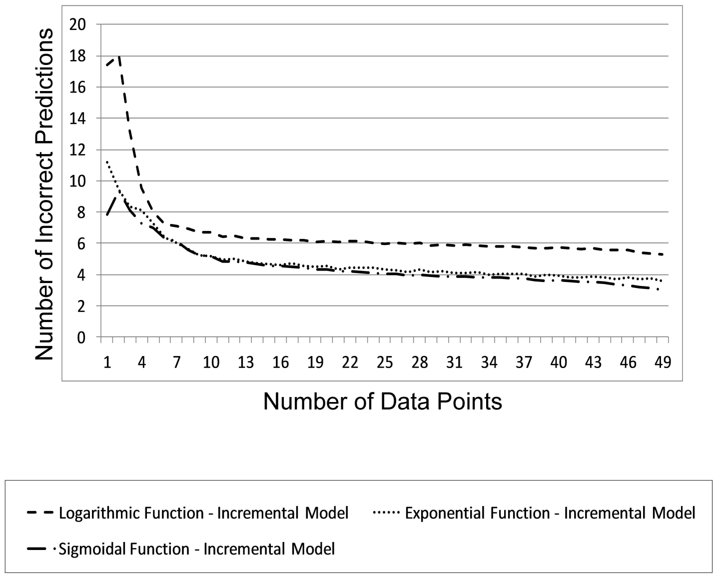


Figure 4: Accuracy of each variation of the incremental value model (averaged over 1000 trials) using various amounts of randomly selected data points from 25 games in the "vs-2-alley" task.

improves steadily in both tasks as additional data points are used fit the model. Note that in both tasks, using as few as ten data points yields about as many incorrect predictions on average as using all forty-nine data points. One interesting result to note is that in the "vs-2-alley" task, the incremental model with the logarithmic function does worst of the incremental models, whereas in the "33%Defense" task it does best. This performance variability is acceptable — although the incremental model with either an exponential function or a sigmoidal function was shown in the previous section to be best for the Pacman Capture-the-Flag domain, it will not always be the absolute best. What is notable is that the chosen model — the incremental model with either an exponential function or a sigmoidal function in our domain — does best or close to best in both tasks.

6.3.1. Importance of Determining the Fitted Parameters

In the previous section we presented the idea that once an ad hoc agent has chosen a model for a particular domain, it can use this chosen model to predict the marginal utility of role selection on new tasks in the same domain

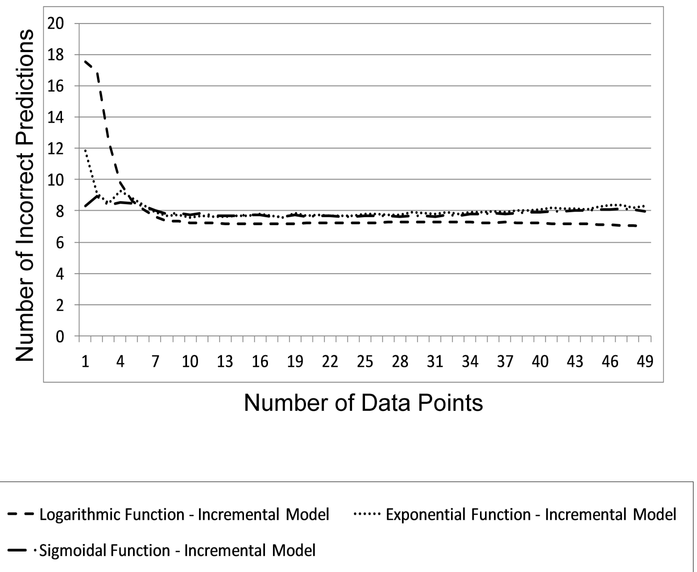


Figure 5: Accuracy of each variation of the incremental value model (averaged over 1000 trials) using various amounts of randomly selected data points from 25 games in the "33%Defense" task.

by using limited gold standard data to determine new fitted parameters for the model.

However, how important is it to use limited gold standard data to determine appropriate fitted parameters for the chosen model function in a new task? We found experimentally that if parameters fit on one task are used on another task, the results can be quite poor. For example, using parameters fit for the "vs-6" task (which yield one incorrect decision on the "vs-6" task) yields 14 incorrect decisions on the "vs-2-33%" task. As such, it is almost always important and worthwhile to find new fitted parameters when a new task is encountered and there is opportunity to obtain *any* gold standard data.

7. Conclusions and Future Work

This chapter presented a formalization of role-based ad hoc teamwork settings, introduced several methods for modeling the marginal utility of an ad hoc agent's role selection, and empirically showed that each of these methods is appropriate for a different class of role-based tasks. We assume in this work that the roles of the teammates are known and that we know how well some team configurations do in a particular task. However, we do not know how much an ad hoc agent could help the team if added to each role. As such, we show that it is possible to use a particular functional form to model the marginal utility of a role selection in a variety of tasks. Additionally, we show that only a limited amount of data is needed on a new task in order to be able to fit the function such that it can be used as a predictive model to determine how an ad hoc agent should behave in situations of a task that it has not previously encountered.

This research is among the first to study role-based ad hoc teams. As such, there are many potential directions for future work. One such direction would be to expand this work into more complicated environments with more than two potential roles to fulfill and more than one ad hoc agent. Another direction would be to consider the case in which the ad hoc agents encounter teammates that are running unfamiliar behaviors, as this would force the ad hoc agents to model their teammates.

8. Acknowledgements

This work has taken place in the Learning Agents Research Group (LARG) at UT Austin. LARG research is supported in part by NSF (IIS-0917122), ONR (N00014-09-1-0658), and the FHWA (DTFH61-07-H-00030).

References

[1] P. Stone, G. A. Kaminka, S. Kraus, J. S. Rosenschein, Ad hoc autonomous agent teams: Collaboration without pre-coordination, in: Proceedings of the Twenty-Fourth Conference on Artificial Intelligence (AAAI'10), 2010.

[2] B. J. Grosz, S. Kraus, Collaborative plans for complex group action, Artificial Intelligence Journal 86 (2) (1996) 269 – 357.

[3] M. Tambe, Towards flexible teamwork, Journal of Artificial Intelligence Research 7 (1997) 83–124.

[4] K. S. Decker, V. R. Lesser, Designing a family of coordination algorithms, in: Proceedings of the First International Conference on Multi-Agent Systems (ICMAS'95), 1995.

[5] P. Stone, M. Veloso, Task decomposition, dynamic role assignment, and low-bandwidth communication for real-time strategic teamwork, Artificial Intelligence Journal 110 (2) (1999) 241–273.

[6] S. Liemhetcharat, M. Veloso, Weighted synergy graphs for role assignment in ad hoc heterogeneous robot teams, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'12), 2012.

[7] F. Wu, S. Zilberstein, X. Chen, Online planning for ad hoc autonomous agent teams, in: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI'11), 2011.

[8] M. Bowling, P. McCracken, Coordination and adaptation in impromptu teams, in: Proceedings of the Twentieth Conference on Artificial Intelligence (AAAI'05), 2005.

[9] E. Jones, B. Browning, M. B. Dias, B. Argall, M. Veloso, A. T. Stentz, Dynamically formed heterogeneous robot teams performing tightly-coordinated tasks, in: Proceedings of the International Conference on Robotics and Automation (ICRA), 2006.

[10] J. Han, M. Li, L. Guo, Soft control on collective behavior of a group of autonomous agents by a shill agent, Journal of Systems Science and Complexity 19 (2006) 54–62.

[11] S. Barrett, P. Stone, S. Kraus, Empirical evaluation of ad hoc teamwork in the pursuit domain, in: Proceedings of Eleventh International Conference on Autonomous Agents and Multiagent Systems (AAMAS'11), 2011.

[12] G. Sukthankar, K. Sycara, Policy recognition for multi-player tactical scenarios, in: Proceedings of the Sixth International Conference on Autonomous Agents and Multiagent Systems (AAMAS'07), 2007.

[13] H. H. Zhuo, L. Li, Multi-agent plan recognition with partial team traces and plan libraries, in: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI'11), 2011.

[14] M. Blake, G. Sorensen, J. Archibald, R. Beard, Human assisted capture-the-flag in an urban environment, in: Proceedings of the International Conference on Robotics and Automation (ICRA), 2004.

[15] A. Sadilek, H. Kautz, Modeling and reasoning about success, failure, and intent of multi-agent activities, in: Proceedings of the Twelfth International Conference on Ubiquitous Computing (UbiComp'10) - Workshop on Mobile Context-Awareness, 2010.

[16] M. Mataric, Learning to behave socially, in: Proceedings of From Animals to Animats 3: The Third International Conference on Simulation of Adaptive Behavior, 1994.

[17] K. Lerman, C. Jones, A. Galstyan, M. Mataric, Analysis of dynamic task allocation in multi-robot systems, International Journal of Robotics Research 25 (2006) 225–242.

[18] J. DeNero, D. Klein, Teaching introductory artificial intelligence with pac-man, in: Proceedings of the First Symposium on Educational Advances in Artificial Intelligence (EAAI'10), 2010.