Coevolution of Heterogeneous Multi-Robot Teams

Matt Knudson
Oregon State University
Corvallis, OR, 97331
knudsonm@engr.orst.edu

Kagan Tumer
Oregon State University
Corvallis, OR, 97331
kagan.tumer@oregonstate.edu

ABSTRACT

Evolving multiple robots so that each robot acting independently can contribute to the maximization of a system level objective presents significant scientific challenges. For example, evolving multiple robots to maximize aggregate information in exploration domains (e.g., planetary exploration, search and rescue) requires coordination, which in turn requires the careful design of the evaluation functions. Additionally, where communication among robots is expensive (e.g., limited power or computation), the coordination must be achieved passively, without robots explicitly informing others of their states/intended actions. Coevolving robots in these situations is a potential solution to producing coordinated behavior, where the robots are coupled through their evaluation functions. In this work, we investigate coevolution in three types of domains: (i) where precisely n homogeneous robots need to perform a task; (ii) where n is the optimal number of homogeneous robots for the task; and (iii) where n is the optimal number of *heterogeneous* robots for the task. Our results show that coevolving robots with evaluation functions that are locally aligned with the system evaluation significantly improve performance over robots evolving using the system evaluation function directly, particularly in dynamic environments.

Categories and Subject Descriptors

I.2.6 [**AI**]: Learning

General Terms

Algorithms, Experimentation

Keywords

Robot coordination; Coevolution; Team Formation

1. INTRODUCTION

Coordinating multiple robots to achieve a system-wide objective in an unknown and dynamic environment is critical to many of today's relevant applications, including the autonomous exploration of planetary surfaces and search and rescue in disaster response. In such cases, the environment may be dangerous, uninhabitable to humans all together, or sufficiently distant from central control that response times require autonomous, coordinated behavior. Evolutionary algorithms are particularly relevant to these applications, as solutions to robotic behavior in such complex environments are difficult or impossible to model.

In general, most multi-robot tasks can be broadly categorized into [8]: (i) tasks where a single robot can accomplish the task, but where having a multi-robot system improves the process (for example, terrain mapping or trash collection); and (ii) tasks where multiple robots are necessary to achieve a task (for example to carry an object). In both cases, coordination requires addressing many challenges (low level navigation, high level decision making, inter-robot coordination) each of which requires some degree of information gathering [17]. However, in the first case, a failure of coordination leads to inefficient use of resources, whereas in the second, it leads to a complete system breakdown. Therefore, a delicate balance must be established within a robots' behavior such that coordination is achieved without an overly strict adherence to a specific coordination protocol. Through coevolution, robots are given the freedom to develop their own protocols to benefit the system objective.

In this work, we focus on problems of the second type, and investigate the robot evaluation functions that need to be derived for the overall system to achieve high levels of performance. To that end, we investigate the use of difference evaluation functions to promote team formation [3]. Such evaluation functions have previously been applied to multiagent coordination problems of the first type [1, 18]. The key contribution of this work is to extend those results to coordination problems of the second type where unless tight coordination among the agents is established and maintained, the tasks cannot be accomplished. We develop teams within the multi-robot system using passive means (e.g., no explicit coordination directives) through the coupling of the robots' evaluation functions.

The application domain we selected is a distributed information gathering problem. First we explore the case where unless a particular point of interest is observed by n robots, the point of interest is not considered as observed. Second we explore the case where there is an optimal number of robots (n) that need to observe a point of interest, but where the system receives some value for observations by teams with other than n members. Finally, we construct a system where

GECCO'10, July 7–11, 2010, Portland, Oregon, USA.

the individuals are of differing capabilities, and one of each type is needed to provide optimal behavior.

In Section 2 we discuss the robot exploration problem. In Section 3, we present the problem requiring team formation. In Section 4 we present the problem of encouraging rather than requiring team formation, and in Section 5 we present heterogeneous teams with robots of two types. Finally in Section 6 we discuss the implication of these results and highlight future research directions.

1.1 Related Work

Extending single robot approaches to multi-robot systems presents difficulties in ensuring that the robots learn a particular task beneficial to the overall system. New approaches that are particularly well suited to multi-robot systems include using Markov Decision Processes for online mechanism design [15], developing new reinforcement learning based algorithms [4, 6, 9, 10], devising agent-specific evaluation functions [3], and domain based evolution [5]. In addition, forming coalitions for purposes of reducing search costs [11], employing multilevel learning architectures for the formation of coalitionsl [16], and market based approaches [21] have been examined.

The use of evolutionary algorithms in a multiagent domain is attractive due to the complex, non-Markovian nature of most systems. Coevolution furthers the advantages by evaluating the performance of individuals based on the interactions with others within the system. Coevolution algorithms tend to favor stability over optimality however [19], finding stable equilibria in agent behavior. One method used to alleviate this tendency is biasing the evaluation functions such that the fitness is evaluated on the most beneficial collaborative agents [13, 14]. The work in this paper is similar, where the most beneficial collaborators are those robots that most closely observe a Point of Interest, evaluated through a difference function. In addition, cooperative coevolution was further classified by defining a robustness criterion, demonstrated on a set of standard multiagent problems [20]. An interesting further extension to coevolution encodes individual agents with a base skill-set [7], preventing coevolved agents from having to learn the same thing independently.

2. ROBOT COORDINATION

The multi-robot information gathering problem we investigate in this work consists of a set of robots that must observe a set of points of interest (POIs) within a given time window [3]. The POIs have different importance to the system, and each observation of a POI yields a value inversely related to the distance the robot is from the POI. In addition, and particular to the work presented in this paper, multiple observations of a POI are either required (Section 3) or highly beneficial (Section 4) to the system objective.

2.1 Robot Capabilities

Each robot uses an evolutionary algorithm to map its sensor inputs to an x, y translation relative to the current position of the robot. Each robot utilizes a two layer sigmoid activated artificial neural network to perform this mapping.

The inputs to this neural network are four POI sensors (Equation 1) and four robot sensors (Equation 2), where x_q^{POI} and x_q^{ROBOT} provide the POI and robot "richness" of each quadrant q, respectively, V_j and L_j are the value and location of POI j respectively, L_i is the location of the current robot i and $\theta_{j,q}$ is the separation in radians between the POI and the center of the sensor quadrant.

$$x_{i,q}^{POI} = \sum_j \frac{V_j}{\delta(L_j, L_i)} \left(1 - \frac{|\theta_{j,q}|}{(\pi/4)}\right) \qquad (1)$$

$$x_{i,q}^{ROBOT} = \sum_{k,k \neq i} \frac{1}{\delta(L_k, L_i)} \left(1 - \frac{|\theta_{k,q}|}{(\pi/4)}\right) \qquad (2)$$

The two outputs indicate the velocity of the robot (in the two axes parallel and perpendicular to the current robot heading). The weights of the neural network are adjusted through an evolutionary search algorithm [3, 2] for ranking and subsequently locating successful networks within a population [12, 3]. The algorithm maintains a population of ten networks, utilizes mutation to modify individuals, and ranks them based on a performance metric specific to the domain. The search algorithm used is shown in Figure 1 which displays the ranking and mutation steps.

```
Initialize $N$ networks at $T = 0$
For $T < T_{max}$ Loop:
    1. Pick a random network $N_i$ from population
       With probability $\epsilon$: $N_{current} \leftarrow N_i$
       With probability $1 - \epsilon$: $N_{current} \leftarrow N_{best}$
    2. Mutate $N_{current}$ to produce $N'$
    3. Control robot with $N'$ for next episode
    4. Rank $N'$ based on performance
       (evaluation function)
    5. Replace $N_{worst}$ with $N'$
```

Figure 1: Evolutionary Algorithm: An ϵ-greedy evolutionary algorithm to determine the weights of the neural networks. See text body for definitions. T indexes episodes, N indexes networks with appropriate subscripts, and N' is the modified network for use in control of the current episode.

In this domain, mutation (Step 2) involves adding a randomly generated number to every weight within the network. This can be done in a large variety of ways, however it is done here by sampling from a random Cauchy distribution where the samples are limited to the continuous range [-10.0,10.0] [3]. Ranking of the network performance (Step 4) is done using a domain specific evaluation function, and is discussed in the following section.

2.2 Robot Objectives

In these experiments, we used three different evaluation functions [3] to determine the performance of the robot: the system evaluation function which rates the performance of the full system; a local evaluation function that rates the performance of a "selfish" robot; and a difference evaluation function that aims to capture the impact of a robot in the multi-robot system [3]. These three evaluation functions are:

- The system evaluation reflects the performance of the full system. Though robots optimizing this evaluation function guarantees that the robots all work toward the same purpose, robots have a difficult time discerning their impact on this function, particularly as the number of robots in the system increases.

- The local evaluation reflects the performance of the robot operating alone in the environment. Each robot is rewarded for the sum of the POIs it alone observed. If the robots operate independently, optimizing this evaluation function would lead to good system behavior. However, if the robots interact frequently, then each robot aiming to optimize its own local function may lead to competitive rather than cooperative behavior.

- The difference evaluation reflects the impact a robot has on the full system [3, 2]. By removing the value of the system evaluation where robot i is inactive, the difference evaluation computes the value added by the observations of robot i alone. Because only POIs to which robot i were closest need this difference computed, this evaluation function is "locally" computable in most instances.

Though conceptually the same, the specifics of these evaluations are different for each of the problems described in the following sections. We derive those specific evaluation structures and present the experimental results below.

3. REQUIRING TEAM FORMATION

In the first problem we examine, the robots need to form teams to perform a task and contribute to the system objective. In this problem, a POI is considered observed only if n robots visit that POI from within a certain observation distance. Neither the robot, nor the system receive any value unless multiple observations of a POI occur. This problem formulation ensures that the problem is one that cannot be solved by a single robot and that the team formation is essential to the completion of each task.

3.1 Problem Definition

To formalize this problem, let us first focus on a problem where the observations of the two robots closest to a POI are tallied. If more than two robots visit a POI, only the observations of the closest two are considered and their visit distances are averaged in the computation of the system evaluation (G), which is given by:

$$G(z) \quad = \quad \sum_i \sum_j \sum_k \frac{V_i \; N^1_{i,j} \; N^2_{i,k}}{\frac{1}{2}(\delta_{i,j} + \delta_{i,k})} \qquad (3)$$

where V_i is the value of the ith POI, $\delta_{i,j}$ is the closest distance between jth robot and the ith POI, and $N^1_{i,j}$ and $N^2_{i,k}$ determine whether a robot was within the observation distance δ_o and the closest or second closest robot, respectively, to the ith POI:

$$N^1_{i,j} = \begin{cases} 1 & \text{if } \delta_{i,j} < \delta_o \text{ and } \delta_{i,j} < \delta_{i,l} \; \forall l \neq j \\ 0 & \text{otherwise} \end{cases} \qquad (4)$$

and

$$N^2_{i,k} = \begin{cases} 1 & \text{if } \delta_{i,k} < \delta_o \text{ and } \delta_{i,k} < \delta_{i,l} \; \forall l \neq j,k \\ 0 & \text{otherwise} \end{cases} \qquad (5)$$

The single robot evaluation function used by each robot only focuses on the value a robot receives for observing a particular POI, and results in:

$$P_j\left(z\right) = \sum_i \frac{V_i}{\delta_{i,j}} \qquad \text{if } \delta_{i,j} < \delta_o \qquad (6)$$

This evaluation promotes selfish behavior only, providing a clear, easy-to-learn signal, but one not aligned with the system objective as a whole.

Finally, the difference evaluation for a robot aims to provide system-wide beneficial behavior, while remaining sensitive to the actions of a robot [3]. This difference evaluation function is given by:

$$D_j\left(z\right) = \begin{cases} \sum_i \left(\frac{V_i}{\frac{1}{2}\left(\delta_{i,j}+\delta_{i,k}\right)} - \frac{V_i}{\frac{1}{2}\left(\delta_{i,j}+\delta_{i,l}\right)} \right) & \text{if } \delta_{i,j}, \delta_{i,k} < \delta_{i,l} < \delta_o \\ \sum_i \frac{V_i}{\frac{1}{2}(\delta_{i,j}+\delta_{i,k})} & \text{if } \delta_{i,j}, \delta_{i,k} < \delta_o \\ 0 & \text{otherwise} \end{cases} \qquad (7)$$

where l is the third closest robot to POI i (meaning that robots j and k are the closest two for the first two conditionals). All three of these evaluations were applied for learning in many different situations, though for brevity, only an environment with 50 POIs and 40 robots (which was representative of the general performance of the evaluations) is presented.

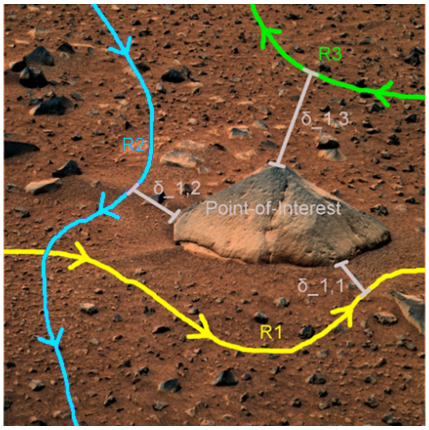

Figure 2: Sample robot paths in an exploration scenario. Multiple observations are made of a particular point of interest. In the team formation domain, multiple observations must be made for the POI to have any value to the system. Background courtesy of JPL.

Figure 2 shows a schematic of how these evaluation functions are computed, given that all three robots are within the observation radius. Only robots 1 and 2 ($R1$ and $R2$) are taken into consideration when calculating $G(z)$ because their observation distance ($\delta_{1,1}$ and $\delta_{1,2}$) are closer than $R3$ ($\delta_{1,3}$). For $G(z)$, robot 3's observation is discarded. For the difference evaluation for robots 1 or 2, robot 3 is taken into consideration. For example, in calculating Equation 7 for $R2$, the first term considers $R1$ and $R2$, where the second term considers $R1$ and $R3$. That is, $R2$ receives the difference between the observation values of $R1$ and $R2$ and the observation values of R1 and R3.

3.2 Results

The environment used for presentation in this paper contained 40 robots and 50 POIs, providing a great deal of

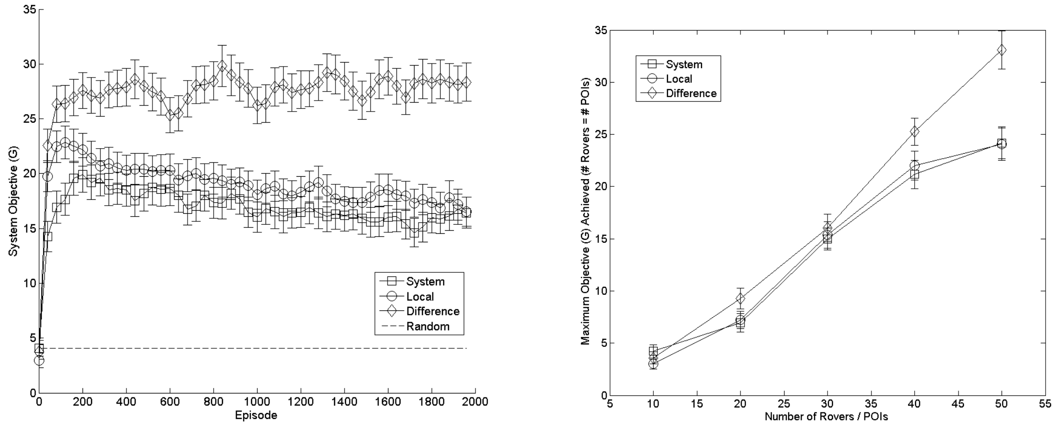

Figure 3: *Team Formation Required* **Left: System evaluation is plotted versus episode for learning in an environment containing 40 robots and 50 POIs. Right: Maximum evaluation achieved is plotted for equal numbers of robots and POIs. Learning is done with system, local, and difference evaluations requiring the formation of teams of two robots.**

information to be gathered, while simultaneously creating a congested situation. In addition, the environment was highly dynamic, where 10% of the POIs (selected randomly) changed location and value at each episode. This was done to encourage specific coordination behavior based on sensor inputs rather than specific x-y coordinates. The results are based on 2000 episodes of 30 time-steps each, and are averaged for significance.

Figure 3 (*left*) shows that robots using all three evaluations perform significantly better than random behavior. It also shows that the difference evaluation provides a signal that allows the robots to learn to coordinate their actions, whereas using the system and local evaluations do not. Additionally, Figure 3 (*right*) shows that the difference evaluation does not provide benefits until the system reaches the point of high complexity.

4. ENCOURAGING TEAM FORMATION

In the second problem we examine, multiple robots are encouraged (rather than required) to form teams to perform a task and contribute to the system objective. In this problem, a POIs value is optimized for n robots observing it, but the system receives lesser value for other numbers of robots observing the POI. Figure 4 shows the functional form of the two system evaluations used in Section 3 and Section 4.

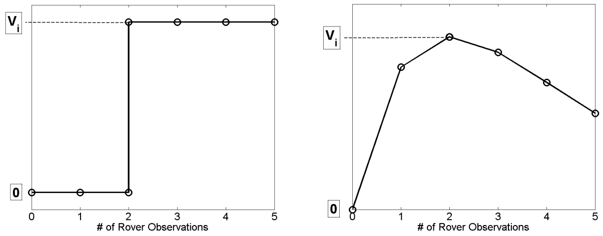

Figure 4: POI value structure is compared between the required (left) and encouraged (right) team formation systems.

4.1 Problem Definition

For these evaluations, δ_o remains the same, however the distance of observation is no longer explicitly included in the evaluation function, relying on inherent inclusion in the observation radius of the POI. As before, three evaluation functions are defined, beginning with the system evaluation given by:

$$G(z) = \sum_i \alpha V_i x e^{\frac{-x}{\beta}} \tag{8}$$

where i indexes POIs, x is the number of robots within δ_o, β is the observation capacity, and α is a constant chosen to be 1.37 such that the maximum of the exponential curve approximates the POI value V_i.

For this new system evaluation, the selfish robot evaluation is defined as:

$$P_j(z) = \sum_{i_j} \alpha V_{i,j} x e^{\frac{-x}{\beta}} \tag{9}$$

where indexing and constant selection is the same as above. This evaluation includes no information regarding contribution to the system as a whole, rather indicating only what robot j can directly observe. This robot evaluation is the component of the system objective for which robot j was within the observation distance δ_o of each POI.

Finally, the difference evaluation function for this system results in:

$$D_j(z) = \sum_{i_j} \alpha V_{i,j} \left[x e^{\frac{-x}{\beta}} - (x-1) e^{\frac{-(x-1)}{\beta}} \right] \tag{10}$$

where indexing and constant selection is the same as above. This evaluation aims to provide the contribution of robot j to the system. The performance of all three evaluation functions are presented in the next section.

4.2 Results

All training parameters were maintained from those used in Section 3.2, including the number of POIs and robots.

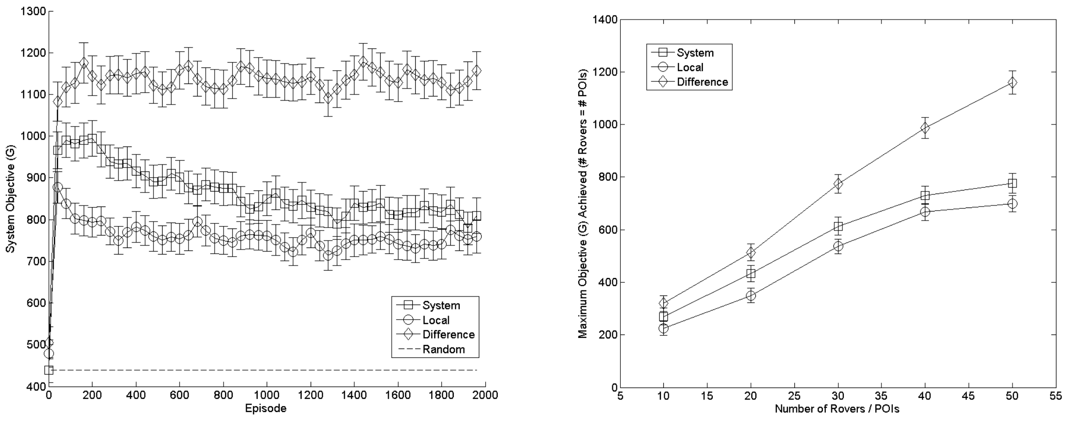

Figure 5: ***Team Formation Encouraged*** **Left: System evaluation is plotted versus episode for learning in an environment containing 40 robots and 50 POIs. Right: Maximum evaluation achieved is plotted for equal numbers of robots and POIs. Learning is done with system, local, and difference evaluation functions requiring the formation of teams of two robots.**

The results presented in Figure 5 are qualitatively similar to those seen in Figure 3. This is a good result, demonstrating that the team requirement in general is applicable and successful for multiple formulations of the problem (does not depend on the exact form of G). As before, the difference evaluation provides consistent behavior throughout, where the system evaluation function (aligned with system, but not sensitive to a given robot's actions) and local evaluation (sensitive to a robot's action, but not necessarily aligned with the system evaluation) break down.

Here again Figure 5 (*right*) shows that as the system increases in complexity, the difference evaluation, through providing a better learning signal, provides consistent behavior through the increased complexity of the system. The system and local learning evaluation function performance tapers off, where using the difference evaluation maintains its' performance slope, clearly indicating that when the number of robots within the system becomes large, the difference evaluation is able to maintain successful dynamic team formation. In addition, through encouraging team formation, rather than requiring it, we have presented a simpler problem to learn.

4.3 Higher Coordination Requirements

The previous two sections investigated coordination for $n = 2$, for both required and encouraged team formation scenarios. The behavior of the three evaluation functions was similar for both cases. In this section we investigate the behavior for $n = 3$, a change that has significant impact on the computation of G, particularly when the observation distance is not increased.

Figure 6 (*left*) shows the learning results for requiring three robots to observe a POI. The all-or-nothing learning structure in this evaluation function makes it very difficult for a robot using passive team formation to extract the relevant signal. This brings the difference evaluation closer to the system objective by reducing its sensitivity to a particular robot's actions (that is, in most cases, removing a robot from the system has no impact on the system performance). As a consequence, the difference evaluation fails to promote good system-level behavior.

By contrast, Figure 6 (*right*) shows the behavior of the system where team formation is encouraged by a decaying value assignment to POI observations. In this case, moving from $n = 2$ to $n = 3$ does not affect the difference evaluation. This is because in this problem, removing a robot has a computable impact on the system objective. This creates a "gradient" for evaluating the impact of a robot on the system as a whole. As a consequence, the difference evaluation performs better than system or local evaluation functions.

We combine the conclusions that a) encouraging dynamic teams, rather than requiring them, is more robust to changes in system definition and, b) difference evaluations are more successful in systems changing in the number of robots and POIs from the above sections to formulate a problem for heterogeneous team formation in the following section.

5. HETEROGENOUS TEAM FORMATION

The success in team formation shown in the above sections points to an investigation of teams constructed of heterogenous robots. When the entire team is made of robots of identical construction, the tasks are limited to general redundant observations of an environment to provide robustness, or mechanical tasks that require multiple individuals to provide enough effort. In contrast, if the individuals can learn to dynamically partner with one-another, the question arises whether or not, given additional sensing, individuals of differing construction can partner to provide a more specific suite of tasks.

5.1 Problem Definition

In the final problem we investigate, we define two robot types; *blue* and *green*. These can represent any number of possible construction differences, including sensing and articulation, depending on the system in which they are installed. The individuals must have the ability to determine the difference between the two, for example a blue robot must be able to determine that there are green robots elsewhere in the environment. In addition, the evaluation function must again be modified to represent the need for robots of differing capabilities to visit a POI.

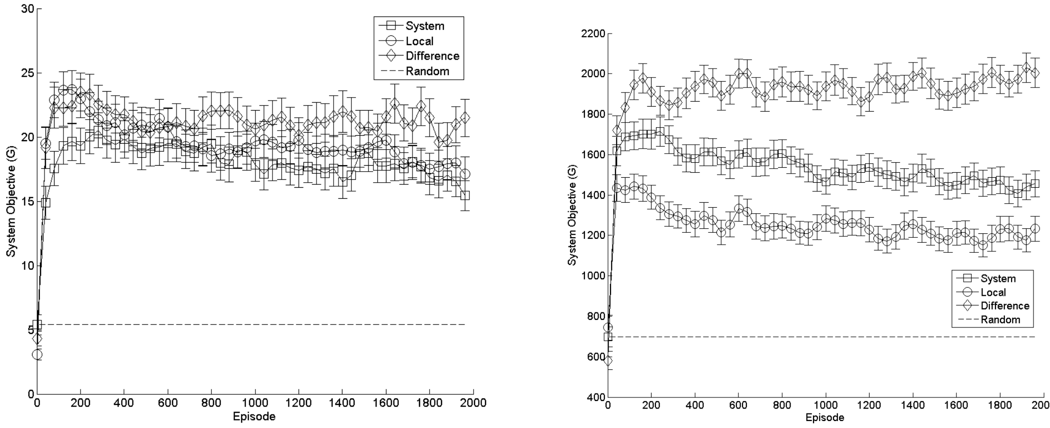

Figure 6: ***Higher Coordination Requirements*** **(*n* = 3) Left: Required Team Formation. Right: Encouraged Team Formation. System evaluation is plotted versus episode for learning in an environment containing 40 robots and 50 POIs. Learning is done with system, local, and difference evaluation functions for three robots to observe a POI.**

The sensing capabilities are similar to those shown in Section 2.1. For each quadrant q however, the robot sensor is split into two, one indicating the density of "blue" robots and the other indicating "green" robots. This increases the number of inputs to the neural network from 8 to 12, and the number of hidden units was increased accordingly. This configuration maintains comparability to homogeneous applications while providing the differentiation between robot types needed by the new problem.

We showed that encouraging team formation is more beneficial to the learning process over requiring team formation, and therefore the modified evaluation function reflects the exponential form as much as possible. Again, δ_o remains the same, and the functional form includes the number of robots in the observation radius of a given POI. The number of observations however is separated into the number of blue robots and green robots that made observations. Therefore, the optimal solution is not only that two robots visit, but that one of each type visits each POI.

As with previous work, three evaluation functions were defined for comparison, reflecting the styles discussed in Section 2.2. Beginning with the system-level evaluation:

$$G\left(z\right) = \sum_{i} \alpha V_i x_b x_g e^{\frac{-x_b x_g}{\beta_b \beta_g}} \tag{11}$$

where x_{type} is the number of observations of a POI i of each type of robot, α is a scaling constant to ensure the maximum of the function approximates the POI value V_i (set to 2.72 for these experiments), and β_x are the constants to produce functional peaks at the desired number of observations of each type of robot. For example, to have one of each type observe a POI, $\beta_b = \beta_g = 1$, which is the configuration for subsequent experiments.

The local evaluation is similar to the above, however it reflects only the POIs that robot j has visited. Therefore it is locally computable and easy to learn, but does not indicate the robot's impact on the system as a whole:

$$P_j\left(z\right) = \sum_{i_j} \alpha V_{i,j} x_b x_g e^{\frac{-x_b x_g}{\beta_b \beta_g}} \tag{12}$$

where indexing and constant selection is the same as the above.

Finally, the difference evaluation includes information contained in the system-level evaluation, but is easier to learn as it directly indicates how robot j contributed to the system as a whole. It is contingent on the type of robot j:

$$D_j\left(z\right) = \sum_{i_j} \alpha V_{i,j} \left(x_b x_g e^{\frac{-x_b x_g}{\beta_b \beta_g}} - \left(x_b - 1\right) x_g e^{\frac{-\left(x_b - 1\right) x_g}{\beta_b \beta_g}} \right) \tag{13}$$

where indexing and constant selection is the same as above. The equation shown is for robot j of type *blue*, where if the type is *green*, 1 is subtracted from the *green* robot observations rather than the *blue*. The experimental results for the use of all three evaluation functions follows in the next section.

5.2 Results

The domain for the experiments involving heterogeneous teams is the same as that used in the above work. Each robot is randomly assigned a type at the beginning of each experiment based on a given team ratio. Learning time is adjusted from 2000 episodes to 3000 as the network has increased in size, and the problem has increased in difficulty, slightly decreasing convergence speed. The environment maintains its dynamic nature, where 10% of the POIs change location and value at every episode, though the robots maintain their type throughout the learning process.

Figure 7 (*left*) shows the results of training in an environment where 40 robots and 50 POIs are present. The ratio of *blue* to *green* robots is 50%, meaning there are 20 of each type present. With the increased problem complexity we observe that the local evaluation is entirely incapable of learning a good solution, in fact learning the wrong thing, performing worse than random parameter selection (network weights) after convergence.

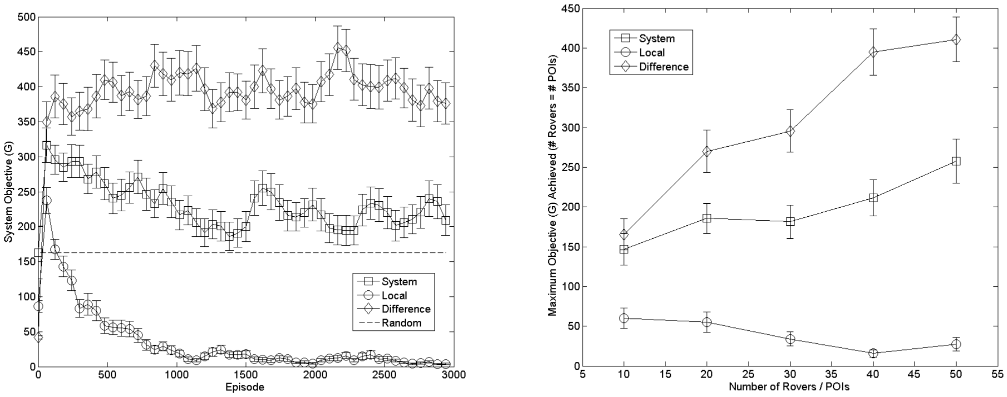

Figure 7: ***Heterogeneous Team Formation*** **Left: System performance for an environment containing 40 robots and 50 POIs. Learning is done with system, local, and difference evaluation functions requiring the formation of teams of two robots, one of each type. Right: Maximum performance achieved for equal numbers of robots and POIs. Learning is done with system, local, and difference evaluation functions encouraging the heterogeneous formation of teams of two robots.**

As with the results in Section 4.2, learning with the system-level evaluation function proves difficult, as there is a great deal of information contained in the signal; too much regarding other robots for each individual to ascertain what actions are best in contributing to the system as a whole. The difference evaluation however, as expected, learns quickly and maintains performance through the learning process. This confirms the applicability of the difference evaluation in general, and specifically indicates that dynamically requiring heterogeneous team formation in a congested and dynamically changing environment is achievable, indeed successful.

We next examine the impact of increasing both the number of robots and the number of POIs within the system simultaneously. Figure 7 (*right*) shows the maximum system-level evaluation function achieved for varying numbers of robots and POIs (where the number of robots and POIs is the same). The local evaluation begins poorly and decreases further as the system complexity increases, as shown in previous figures. Using the system-level evaluation for learning, while increasing slightly as complexity increases, is strongly outperformed by the difference evaluation. As with all previous dynamic team formation work in this paper, utilization of the evaluation function significantly improves performance over the others, and provides an excellent learning signal for dynamic team formation, particularly in domains absent of communication and heterogeneous in construction.

In varying the ratio between robot types present in the system, we can determine if the robots are able to modify their behavior to suit changes in system consistently. For example, if a large set of robots of a specific type fail, the system must have the ability to adjust coordination behavior to maintain success in accomplishing the tasks requested. Figure 8 shows the maximum system performance achieved when the ratio between *blue* and *green* robots is varied. The variance is symmetrical, therefore 10% *blue* and 90% *green* is the same as 10% *green* and 90% *blue*. The number of robots and POIs present in the system is held constant.

The local evaluation always performs poorly, and the ratio of types within the system has little impact on the performance of the system evaluation. This points to a lack of attention paid to the heterogeneous nature of the team in the behavior of the robots that learn with the system evaluation. The difference evaluation however varies significantly when the teams are strongly unbalanced, particularly when the ratio is set to 20%. This is due to the variance in sensing information during the learning process. For example, when there are much fewer robots of one type within the system, the sensors detecting the two types return significantly different levels of information, and therefore the algorithm can learn to focus on the sensors showing where robots of a different type are located. This provides additional information to the algorithm regarding the actions that will lead directly to an increase in the learning evaluation performance.

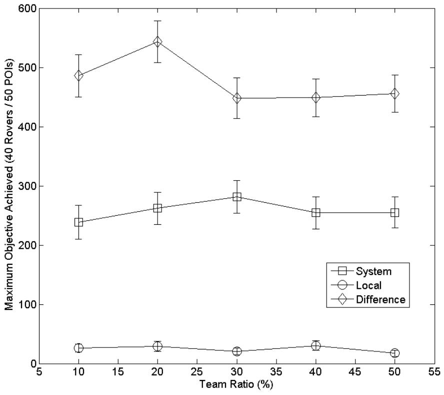

Figure 8: **Heterogeneous Team Ratios: System evaluation is plotted versus episode for learning in an environment containing 40 robots and 50 POIs. Learning is done with system, local, and difference evaluation functions requiring the formation of teams of two robots, one of each type. The ratio between** ***blue*** **and** ***green*** **robots varies in the system.**

6. DISCUSSION AND FUTURE WORK

Exploration of planetary surfaces or in disaster response requires that robotic solutions operate in unknown and dynamic environments. Coordinating multiple robots in such domains presents additional challenges. In this work, we explore multi-robot coordination domains where multiple robots are necessary to achieve a task (for example to carry an object). We focus on passive coordination that is accomplished through the robots' evaluation functions.

The work presented is this paper explores three types of problems where robot coordination is beneficial. First, we explore a problem where n robots must coordinate to receive a reward. Then, we explore a problem where the system reward is optimized for n robots, but other number of robots observing a POI also contribute to the system objective. Finally we develop a heterogeneous system where two types of robots are present, and an observation by one of each produces optimal behavior.

In all three cases, coordination and team formation is established and maintained through passive means encoded in the robots evaluation functions. The difference evaluation yielded the best results because it provided an evaluation that was aligned with the overall system evaluation, while maintaining sensitivity to a robot's actions, even when many robots were active within the coordinated system. That approach also extended to three or more robots encouraged to complete a task. This is an interesting result showing that the difference evaluation is best suited to domains where the impact of a robot on a system can be ascertained.

We are currently implementing the work discussed in this paper in robot hardware. This involves investigating non-episodic learning such that coordination and ad-hoc team formation can be learned while the robot is in current operation. In addition, extensions to the learning algorithm used in this paper will be investigated to facilitate the restrictions of physical hardware.

Acknowledgments

This work was partially supported by AFOSR grant FA9550-08-1-0187 and NSF grant IIS-0910358.

7. REFERENCES

[1] A. Agogino and K. Tumer. Distributed evaluation functions for fault tolerant multi rover systems. In *Proceedings of the Genetic and Evolutionary Computation Conference*, Seattle, WA, July 2006.

[2] A. K. Agogino and K. Tumer. Analyzing and visualizing multiagent rewards in dynamic and stochastic environments. *Journal of Autonomous Agents and Multi Agent Systems*, 17(2):320–338, 2008.

[3] A. K. Agogino and K. Tumer. Efficient evaluation functions for evolving coordination. *Evolutionary Computation*, 16(2):257–288, 2008.

[4] M. Ahmadi and P. Stone. A multi-robot system for continuous area sweeping tasks. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 1724–1729, May 2006.

[5] M. Alden, A.-J. van Kesteren, and R. Miikkulainen. Eugenic evolution utilizing a domain model. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2002)*, San Francisco, CA, 2002.

[6] C. Claus and C. Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. In *Proceedings of the Artificial Intelligence Conference*, pages 746–752, Madison, WI, July 1998.

[7] D. B. D'Ambrosio and K. O. Stanley. Generative encoding for multiagent learning. In *Genetic and Evolutionary Computation Conference*, 2008.

[8] B. P. Gerkey and M. J. Mataric. Multi-robot task allocation: Analyzing the complexity and optimality of key architectures. In *Proceedings of the IEEE Int. Conference on Robotics and Automation*, pages 3862–3868, 2003.

[9] C. Guestrin, M. Lagoudakis, and R. Parr. Coordinated reinforcement learning. In *Proceedings of the 19th International Conference on Machine Learning*, page 41Ű48, 2002.

[10] J. Hu and M. P. Wellman. Multiagent reinforcement learning: Theoretical framework and an algorithm. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 242–250, 1998.

[11] E. Manisterski, D. Sarne, and S. Kraus. Enhancing mas cooperative search through coalition partitioning. In *Proc. Int'l Joint Conference on Artificial Intelligence*, pages 1415–1421, 2007.

[12] S. Nolfi, D. Floreano, O. Miglino, and F. Mondada. How to evolve autonomous robots: Different approaches in evolutionary robotics. In *Proc. of Artificial Life IV*, pages 190–197, 1994.

[13] L. Panait. Improving coevolutionary search for optimal multiagent behaviors. In *International Joint Conference on Artificial Intelligence*, pages 653–658. Morgan Kaufmann, 2003.

[14] L. Panait, S. Luke, and R. P. Wiegand. Biasing coevolutionary search for optimal multiagent behaviors. *IEEE Transactions on Evolutionary Computation*, 10(6):629–645, 2006.

[15] D. Parkes and S. Singh. An MDP-based approach to online mechanism design. In *NIPS 16*, pages 791–798, 2004.

[16] L. Soh and X. Li. An integrated multilevel learning approach to multiagent coalition formation. In *Proc. Int'l Joint Conference on Artificial Intelligence*, pages 619–625, 2003.

[17] S. Thrun and G. Sukhatme. *Robotics: Science and Systems I*. MIT Press, 2005.

[18] K. Tumer and A. Agogino. Coordinating multi-rover systems: Evaluation functions for dynamic and noisy environments. In *The Genetic and Evolutionary Computation Conference*, 2005.

[19] R. P. Wiegand, W. Liles, and K. D. Jong. *Modeling variation in cooperative coevolution using evolutionary game theory*, pages 203–220. Morgan Kaufmann, 2002.

[20] R. P. Wiegand and M. A. Potter. Robustness in cooperative coevolution. In *Genetic and Evolutionary Computation Conference*, pages 369–376. ACM Press, 2006.

[21] Y. Ye and Y. Tu. Dynamics of coalition formation in combinatorial trading. In *Proc. Int'l Joint Conference on Artificial Intelligence*, pages 625–632, 2003.