Crucial Factors Affecting Cooperative Multirobot Learning

Poj Tangamchit [1] John M. Dolan [2] Pradeep K. Khosla [1,2]
E-mail: poj@andrew.cmu.edu jmd@cs.cmu.edu pkk@cs.cmu.edu
Dept. of Electrical and Computer Engineering [1], The Robotics Institute [2]
Carnegie Mellon University, 5000 Forbes Ave. Pittsburgh, PA 15213, USA

Abstract

The effectiveness of multirobot learning in achieving optimal, cooperative solutions is potentially affected by various factors having to do with the nature and configuration of the robots and the nature of the learning entities. Varying one factor wrongly may lead to undesirable results. There is no reported work on how systematically to set up these factors. In this paper, we methodically test the effect of varying four common factors (reward scope, global information delay, diversity of robots, and number of robots) in a decentralized multirobot system, first in simulation and then on real robots. The results show that two of these factors, reward scope and global information delay, if set up incorrectly, can prevent optimal, cooperative solutions.

1. Introduction

Learning allows multirobot systems to adjust and fine-tune themselves to fit their tasks and achieve optimal performance. It alleviates the tedious work of manual tuning: first when we newly install a multirobot system, second when the task changes during operation. However, multirobot learning is still in the developmental stage. One of the main problems is the multiplicity of factors affecting the learning performance and a lack of systematic knowledge about their effects. Another problem is the size of the discretized sensory learning space due to the analog nature of the real world. We bypass this problem by giving our robot systems a fixed set of low-level behaviors, and locating the learning at the level of how to use these behaviors. In other words, the learning is done at a high level (called the task level), where robots learn how to choose appropriate actions according to their internal states.

This paper tackles the first problem, the multiplicity of factors affecting learning. One of these factors is the value function of the learning algorithm. In [10], we demonstrated that for multi-robot systems, learning based on average awards, such as the Monte Carlo algorithm, is superior to that based on discounted rewards, such as Q learning, which generates sub-optimal, non-cooperative solutions. In this paper, we investigate four other factors that might be expected a priori to have an impact on the learning performance. The four factors investigated can be grouped under the nature of the learning entities (reward scope and delay of global information) and the nature of the robots (number and diversity).

The rest of this paper is organized as follows. Section 2 describes previous work. Section 3 gives our approach by first explaining the taxonomy of multirobot learning and the learning algorithm that we used. It then describes the multirobot patrolling task and the four factors that we investigated. Section 4 gives experimental results from simulation and on real robots. Section 5 presents a discussion of how the factors under consideration affect learning, and section 6 gives conclusions.

2. Previous Work

Learning in decentralized multirobot systems has not been systematically studied. Nevertheless, some aspects of single-robot learning can be applied to multirobot systems. For example, reinforcement learning [9] is still suitable for use in multirobot systems, as is Sutton's Dyna architecture [4]. Dyna reuses training data and creates a hypothetical world in order to use the training data to the full extent. It can speed up learning, and we use it to make experiments with real robots feasible. An example of a single-robot learning method being applied to multirobot systems is Mataric's work [1]. With the use of progress estimators, she successfully implemented multirobot learning with a single-robot learning method.

Despite the partial applicability of single-robot learning, multirobot systems have unique features that introduce additional considerations. One of these is the potential for cooperation when using different learning value functions, and in [10]

we showed that discounted-reward-based Q-learning [5], although effective for single robots, cannot produce cooperation, for which an average-rewards-based scheme such as Monte Carlo learning [4] is necessary. Another consideration is the diversity of multirobot teams, which was first investigated by Balch [6]. He showed that diversity can have an impact on the performance of robot teams in some types of tasks. Inspired by his work, we included diversity among the factors whose impact on learning performance was tested, using a hybrid, rather than Balch's purely reactive architecture. Because our structure of tasks and actions is different from Balch's, we cannot rely on his results being applicable to our architecture.

3. Approach

Multirobot learning is still in an early stage. Implementing multirobot learning is more complex than single-robot learning because there are many more factors to be considered. There are papers [1,6,10] that indicate the insufficiency of using single-robot learning theory to obtain good performance in multirobot systems. This paper is an attempt to begin to build a fundamental theory of multirobot learning. To systematically study the factors introduced in multirobot learning, we made a taxonomy of these factors, referring to earlier work by Dudek et al. [7] and Balch [8]. This paper explores a portion of this taxonomy. We selected four factors that seemed capable of affecting learning in multirobot systems, then varied each factor individually and observed how the learning result changed.

Our work and the previous work in multirobot learning [1,2,6] use simple tasks as testbeds in experiments. Among those tasks are foraging [1,6] and box pushing [2]. The main reason is that we are initially more interested in validating learning theory than in implementing learning on a complex task. Simple tasks are easy to observe and analyze. We chose the multirobot patrolling problem to be our testbed for that reason. Moreover, it is a surveillance problem that is a prototype of general exploration tasks. Its details are described in section 3.4.

3.1 Taxonomy of environmental factors

Environmental factors are various characteristics that have to be chosen when researchers implement learning in multirobot systems. Due to the absence of guidelines for systematically specifying these factors, researchers currently require trial-and-error in choosing them until the desired results are achieved. A multirobot learning system has three main components: robots, learning algorithms and tasks. Our taxonomy is constructed based on the first two components. The nature and configuration of the robots entails the overall structure of the robot team. The nature of the learning entities involves the structure of learning algorithms and rewards. We do not make a taxonomy of tasks because tasks are user-specific. Tasks can vary indefinitely with different aspects according to users' requirements. We instead test the two components on different types of common multirobot tasks that we consider to be prototypes for general multirobot applications.

The nature of the robots involves robot architecture (reactive / deliberative / hybrid), group architecture (centralized / decentralized), number of robots (small-size / large-size), and diversity in capabilities (heterogeneous / homogeneous).

The nature of the learning involves learning entities (centralized / decentralized), model of the value function (discounted / average), delay of global information (delay/non-delay) and reward scope (local / global).

This paper presents multirobot learning with fixed parameters as follows: hybrid robot architecture, decentralized group architecture, decentralized learning entities, and average-reward value function. The effect of the value function of learning has already been tested in our previous work [10]. This paper tests the effect of varying the following parameters: number of robots, diversity in capabilities, delay of global information, and reward scope. These factors were tested in the multirobot patrolling problem.

3.2 Reinforcement Learning Algorithm

We use distributed learning entities that run asynchronously on each robot. In this paper, we also fix the value function of the learning algorithm. We choose the Monte Carlo algorithm (MC), which has an average-reward value function as the learning algorithm. MC's slowness is a major weakness, so we use Sutton's Dyna architecture [4] to speed up the learning. Its basic idea is that most learning algorithms do not use training samples to their full extent. Therefore, Dyna keeps a record of training samples and reuses them. In other words, it uses past

training samples to create a hypothetical world and learn from this world. There are therefore two sources of learning: one from new training samples and one from the hypothetical world.

3.3 The Multirobot Patrolling Problem

We chose the multirobot patrolling problem for our learning experiment because it is easy to implement and does not require object manipulation. The problem involves the general mobile robot task of locomotion, and it typically has multiple solutions that differ from one another in efficiency. The task is to guard a large area with mobile robots equipped with cameras. The robots have to move around and search for intruders. We use the "security guard rule" by defining checkpoints. The robots have to travel and visit all checkpoints in repetitive cycles. The objective of learning is to share the responsibility for those checkpoints in order to complete a cycle as fast as possible. This will cause each checkpoint to be visited as often as possible. The multirobot patrolling problem can be a prototype of tasks that involve moving through paths or areas such as exploration.

Our testbed problem consists of a rectangular area with four checkpoints and two robots. In each cycle, the robots have to visit each checkpoint at least once. After all checkpoints are visited, the new cycle will begin. Due to the horizontal distance between points being greater than the vertical distance, the optimal solution is where one robot takes on two checkpoints on the left-hand side and the other robot takes on two checkpoints on the right-hand side.

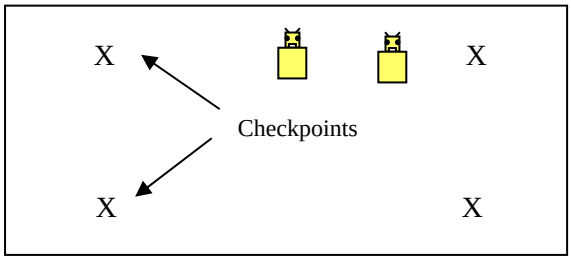


Figure 1: The Multirobot Patrolling Problem

3.4 Details of Tested Environmental Factors

In this paper, we investigate two factors from the nature of the learning entities (reward scope and delay of global information) and two factors from the nature and configuration of the robots (number and diversity). The factors in the nature and configuration of the robots that we fix are the hybrid robot architecture and the decentralized group architecture. The factors in the nature of the learning entities that we fix are the decentralized learning entities and the average-reward value function. Each factor varied is detailed below.

3.4.1 Reward Scope

Rewards are an important component of reinforcement learning. A reward is given to a robot when it does something good, e.g., reaching the goal. There are two reward schemes in multirobot learning: local and global. A local reward scheme keeps rewards within each robot individually, whereas a global reward scheme broadcasts rewards generated within each robot to all other teammates. Therefore, with a global reward scheme, robots receive reward or punishment together, as a team. The reward (cost) used in the patrolling problem is the time needed to travel from one checkpoint to another checkpoint.

3.4.2 Delay of Global Information

In decentralized systems, communication among robots is necessary in order to make the system efficient and consistent. The information about the world that robots exchange among themselves is called global information. In multirobot learning, the state of each robot is dependent not only on its own state information, but also on information from its teammates. Communication is required in order to synchronize the world state among all robots. Generally, a robot will tell its teammates when there is a change in the world. In the patrolling problem, a robot has to tell it teammates about which checkpoint it chooses to visit next. A delay in this information is created by broadcasting the message after the robot already reaches that checkpoint.

3.4.3 Diversity of Robots' Capabilities

In a robot team, robots can have the same or different capabilities. This property is referred to as the diversity of a robot team. Teams consisting of robots with the same capabilities are termed homogeneous. Teams consisting of robots with different capabilities are termed heterogeneous. The impact of diversity on robot teams was first investigated by Balch [6]. Using a reactive robot architecture, he showed that diversity is beneficial in

some types of tasks, but unsuited to others. However, with our hybrid robot architecture, the same result is unlikely. In a reactive robot architecture, the robots are ignorant of the global environment. They will only read sensors and take local actions without caring about their teammates. Therefore, the robots perform the tasks independently and cooperation cannot be ensured. In the patrolling problem, the robots main action is moving to checkpoints. Making difference in robots' capabilities can be done by changing this. Generally, all robots have full capability in going to all checkpoints. This is a homogeneous team. We can create a heterogeneous team by artificially preventing one of the robots from going to some checkpoints (cripple it). In the experiment we chose one checkpoint at upper right-hand corner.

3.4.4 Number of Robots

We classify this factor into two types: small-size (2-3 robots) and large-size (>20 robots). Although we do not test the large-size case, we vary the number of robots within small-size range in order to test the scalability of the learning algorithm. We tested this factor by adding both robots and checkpoints and then restarting the learning to see if the division of labor has changed. In our experiment, one robot and two more checkpoints were added (see Figure 2). We did not change any other parameters. The expected optimal division of labor is shown, in which each robot takes two points in the same column.

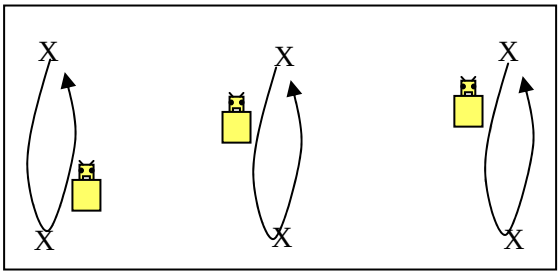


Figure 2: Patrolling problem with 3 robots/6 points

4. Experiments and Results

We conducted simulated and real experiments on the patrolling problem using the Monte Carlo algorithm with Dyna. The simulation results were verified on real robots.

4.1 Simulation and Results

Our simulation was written in Visual C++. The robots had predefined low-level behaviors: move to a point and avoid obstacles. The robots were assumed to be equipped with GPS, sonar sensors and a communication channel. Learning entities were implemented on each robot independently. The communication channel was used for sending global rewards and the world state update. The state of each robot was computed from the following parameters:

State = {last point visited, point already visited}

The set of possible actions for each robot was as follows:

Actions = {move to point n, wait and do nothing}

The reward given was the cost of each movement from one point to another. There was therefore no positive reward in the patrolling problem. The movement cost was calculated as the time (in seconds) needed to travel between points, including avoidance of other robots. When the robot chose to wait and do nothing, it received a small cost due to overhead. For patrolling with 4 checkpoints, there are 64 (2^4x4) states and 5 (4+1) actions. The state is dependent on which checkpoints have already been visited (2^4 possible combinations) and which checkpoint the robot visited last (4). The possible actions are going to any of the points (4) plus the "wait" command (1).

After randomly varying factors and running the experiments, we found two types of results: the optimal case where each robot takes two points in the same column, and a sub-optimal case where one robot or more got stuck and always chose to do nothing. The second case happens because the robots learn that choosing the "wait" action repetitively will give it the highest total reward. We designate the first case "converge" and the second case "deadlock". Moreover, the effect of factors that create deadlock seems to be dominant regardless of the setting of other factors. For example, using a local reward scheme always creates deadlock regardless of the setting of other factors. There was also no change in results when the factors that did not create deadlock were simultaneously adjusted. This suggests that the effects of the four factors are independent of one another. The results presented here are based on changes in each factor compared to the "standard case" of two robots, four checkpoints, global reward, a homogeneous team, and no delay in global information. Because the

learning performs random exploration, we ran the experiment multiple (10) times for each case to ensure the consistency of the results. Each run ended when the learning reached a stable state defined by the total reward staying within 5% of its value for 5 cycles. The tables below show the results.

Factors		No. times converged	Median time to converge (epochs)
Reward Scope	Global	10	145
	Local	0	N/A
Delay of Global Info.	No Delay	10	145
	Delay	5	324
Diversity of robots' capability	Homogeneous	10	145
	Heterogeneous	10	91
Number of robots	2 robots 4 points	10	145
	3 robots 6 points	10	830

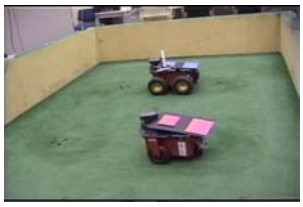
The results were consistent across runs except for the delay of global information, which had 50% convergence and 50% deadlock. Increasing the learning time did not change this result. The reward scope had a great effect on the learning result, whereas robot diversity and the number of robots had no effect.

4.2 Real Robot Experiment

As a general rule, robot simulations need to be confirmed on real robots due to unexpected or unmodeled incidents or behaviors, such as noise in sensors. We verified our simulation by porting the same learning algorithm onto Pioneer robots from Activmedia Corp. equipped with an onboard PC104, sonar sensors, and a wireless LAN card. We used an overhead camera to detect robot position (act as a GPS). Although we got different values of reward, the results were qualitatively the same as those in simulation. The optimal solution was found as each robot took two points of the same column. Additionally, deadlock occurred when we used a local reward scheme and when we added the delay of global information.


The Robots


The field


Converge


Deadlock

Figure 3: Real Robot Experiment

5. Discussion

The results indicate that only the reward scope and the delay of global information affect the final results of learning. In this section, we analyze the effect of each of the four factors considered.

With a local reward scheme, each robot will be unwilling to do any task. An explanation of this result is as follows. Monte Carlo learning is episodic. It keeps counting the total reward accumulated until the end of each cycle and learns the whole series of actions needed to complete the job. In other words, the learning algorithm learns the series of actions, not just each individual action. With a local reward scheme, there is a possibility for a robot to avoid costs by doing nothing and have the job done by its teammates. When all robots learn this strategy, the final result will be that each robot is unwilling to do any task. This will make the robots greedy and finally lead to a deadlock. With a global reward scheme, a robot cannot avoid costs by doing nothing and have the job done. This is because the work done by its teammates will also generate a cost for itself. The best strategy for minimizing costs is therefore to cooperate and do the part of the job most suited to oneself. Therefore, global reward generates a fair division of labor and utilizes the capability of the team to the full extent. Global reward can therefore induce cooperation among robots.

When robots have different capabilities, the learning algorithm on each robot will be able to find the optimal division of labor. In our experiment, the optimal solution is the same for both homogeneous and heterogeneous teams. For a heterogeneous team, the robot that cannot go to the upper right corner learns to patrol the two

checkpoints on the left. The other robot compensates for the inability of the first robot by patrolling the checkpoints on the right.

The number of robots also has no effect on the final solution, but it slows the learning significantly. When there are more robots, there are more states and situations for the robots to learn. The learning time will grow exponentially. With 6 checkpoints, the number of states will be 2^6x6 = 384 states. Thus, a 50% increase in checkpoints will result in about a 600% increase in the number of states to explore. The result shows that an increase in number of robots does not prevent the learning from achieving the optimal result. However, the multirobot patrolling problem does not scale well because it is NP–complete.

The delay in global information also affects the learning. The result was sometimes convergence and sometimes deadlock. Delayed information has an effect equivalent to that of false information. It tells the robot that the robot is in one state when the robot is actually in another state. Therefore, the learning matches the states with the wrong actions and a wrong solution is created. Consider Figure 4: when there is no delay, the correct state-action pairs are S1-A1, S2-A2, and S3-A3. When there is a delay in global information, the state-action pairs will be S1-A2 and S2-A3, which are incorrect.

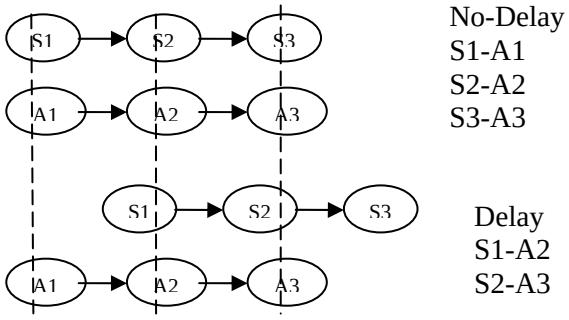


Figure 4: Delay of Global Information

6. Conclusions

The effectiveness of multirobot learning in achieving optimal, cooperative solutions is potentially affected by various factors. In earlier work [10], we showed the importance of average-rewards-based learning for cooperation. In the current work, using a hybrid, decentralized group architecture and the average-rewards-based Monte Carlo learning algorithm, we demonstrated the sensitivity of learning performance to reward scope and global information delay, and its insensitivity to the diversity and number of robots. In future work we will investigate the effects of these factors on different types of tasks, such as those requiring object manipulations.

References

[1] Mataric M.J., "Interaction and Intelligent Behavior", Ph.D. thesis, MIT EECS, 1994.

[2] Parker L.E., "Heterogeneous Multi-Robot Cooperation", Ph.D. thesis, MIT EECS, 1994.

[3] Tangamchit P., Dolan J.M. and Khosla P.K., "Dynamic Task Selection: A Simple Structure for Multirobot Systems", DARS 2000, pp.483-484.

[4] Sutton R.S. and Barto A.G., "Reinforcement Learning: An Introduction", MIT Press, Cambridge, MA, 1998.

[5] Watkins C.J.C.H., "Learning from Delayed Rewards", Ph.D. thesis, King's College, Cambridge, UK, 1989.

[6] Balch T., "Behavioral Diversity in Learning Robot Teams", Ph.D. thesis, Dept. of Computer Science, Georgia Tech., 1998.

[7] Dudek G., Jenkin M.R., Milios E. and Wilkes D., "A Taxonomy for Multi-Agent Robotics", Autonomous Robots 3 (4):375-397, December 1996, Kluwer Academic Publishers.

[8] Balch T., "Taxonomies of Multirobot Task and Reward", Technical Report Robotic Institute, CMU, 1998.

[9] Kaelbling L., Littman M. and Moore A., "Reinforcement Learning: A Survey", Journal of AI Research 4, pp.237-285, 1996.

[10] Tangamchit P., Dolan J.M. and Khosla P.K., "The Necessity of Average Reward in Cooperative Multirobot Learning", ICRA 2002.

[11] Fukuda T.,Kawauchi Y., "Cellular Robotics", pp. 745-782, Springer-Verlag 1993.