Learning to cooperate using deep reinforcement learning in a multi-agent system

A THESIS
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Nabil Khan

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

Maria Gini

December, 2020

Acknowledgements

There are many people that have earned my gratitude for their contribution to my time in graduate school.

Dedication

To those who held me up over the years.

Abstract

In this thesis we address the problem of emergence of cooperation between agents that operate in a simulated environment, where they need to accomplish a complex task that is decomposed into sub-tasks and that can be completed only with cooperation. A deep reinforcement learning approach using a multi-layered neural network is used by the agents within a reinforcement learning algorithm to learn how to accomplish the task of moving heavy loads that require cooperation between agents. The goal of this work is to empirically show that cooperation can emerge without explicit instructions, whereby agents learn to cooperate to perform complex tasks, and to analyze the correlation between task complexity and training time. The series of experiments we conducted helps establish that cooperation can emerge but becomes unlikely in partially observable environments as the environment size scales up. Another series of experiments shows that communication makes the cooperative behavior more likely, even as environments scale up, when the environment is only partially observable. However, communication is not a necessary condition for cooperation to emerge, since agents with knowledge of the environment can also learn to cooperate, as demonstrated in the fully observable environment.

Contents

List of Figures

Chapter 1

Introduction

Multi-agent systems are increasingly popular and are often tasked with accomplishing complex tasks that require cooperation. People experience the reality of their physical limitations as individuals when trying to move a bulky or irregular sized object which requires them to coordinate with others to accomplish the task. The need to work cooperatively arises from trying to interact in the environment with artifacts that are larger and more complex than what a single person can reasonably manipulate. Given the complexity and multitude of tasks that have to be performed all around us, cooperative team work is inevitable for successful functioning in organizations, sports, warehouses and almost every facets of society. In an ever increasing pace of automation we see that autonomous agents need to work cooperatively, where they operate in an environment with objects that are bigger and more complex than themselves and hence cannot be manipulated by a single agent.

Robotics and artificial intelligence are bound to evolve into the area of multi-agent cooperation as more and more autonomous entities are deployed to accomplish tasks that cannot be adequately performed by a single agent. A relevant example of an environment that needs multiple agents coordinating tasks is that of a large factory warehouse, where robots are tasked with automatic storage and retrieval of products. Products that are bulky might require multiple robots to cooperate in order to move the object without having to instrument the environment just as two human workers would cooperatively work to move a bulky furniture instead of attempting to move it alone. Another area where a multi-agent cooperative model could be extremely beneficial is the manufacturing process. The way car manufacturing is automated today is to have an assembly line for the different parts to come together, where an engine is placed into a chassis and the vehicle progresses down the line where human and robotic elements take turns placing finer components. However, this requires instrumentation of the environment to enable an adequate level of automation and even then a large workforce is required to attend to specific tasks. Instrumentation is expensive and hence there are significant opportunities to further automate manufacturing while managing costs. If a team of robots could learn to accomplish cooperative tasks, such as inventory control in a warehouse, or in the assembly line of the manufacturing process, there would be a significant overall economic benefit.

Task specific teamwork, as evidenced in sports, is a learned effort through training and practice. We can also assume that some of the programming for enabling teamwork already exists within the human cognitive psyche, particularly taking into account that we are social beings and need to depend on others for survival. However, cooperation with people also depends on experience to a large extent. For example, a person with no prior experience in playing soccer as a team may not know upfront the fundamental strategy necessary for winning. A simple winning strategy that exhibits team cooperation might be to pass the ball to the person closest to the goal. Such a person can either learn through observing other team members or the person could be instructed by others via an explanation of how the co-ordination effort works. Similarly, when we are tasked to move a large table that is too bulky to haul by one person we can instruct another person to coordinate the move. We may give specific step-by-step directional guidance to move the object to a target location. Of course, human intelligence is sophisticated enough to decipher a simple instruction such as “help me move the table”, and the brain automatically charts a navigation path all the while perceiving the weight of the object and the necessary dynamics within us to balance the table while gripping the part of the object for maximum leverage. In case of a robot, however, with rudimentary coordination skills we may pose the question on how to generate cooperative behavior without directly programming instructions to do so. We may broadly inquire if when multiple agents are placed in an environment where complex tasks can only be accomplished through teamwork, would we witness emergent cooperative behavior? The goal of this thesis is to prove that such behavior can emerge with reasonable training and that communication can enhance this learning process even further.

Reinforcement learning has been quite successful with training agents in a single agent environment. However, it also has limitations in the continuous space of the real environment because both the state space and the action space can be arbitrarily large. Moreover, the solution space needed for task accomplishment can grow exponentially large as an agent learns a sequence of tasks that eventually leads to the goal state. This can be adequately understood using the constraints of a chess board. On average, a game will last about 40 moves and each state will have about 30 possible moves. So even in a toy environment reinforcement learning alone may not explore the state space adequately to learn a winning strategy. Moreover, there is the limitation of observation in a real environment where perceived information is either noisy or incomplete. In a partially observable environment, an agent has to rely on relevant information to make an optimal decision. However, the relevance of specific information can be challenging to identify. For example, an autonomous vehicle might adequately decide to move into another lane as long as the lane is free of traffic. Under usual circumstances, the action of the vehicle can be decided adequately with sensors that detect that the lanes are clear. However, if there is a speeding car also attempting to pass the autonomous vehicle from behind and it is also imminently expected to appear in the next lane, this vehicle's location and speed will also become relevant.

In a different scenario of autonomous robots, let us consider a team of warehouse robots that are tasked with cooperatively moving objects to a specific location. The robots have to learn the optimal path to the object itself and then synchronously move the object to the goal location. Robots can be easily programmed to do this but if a robot were to be made truly adaptive it has to have the ability to learn the optimal set of actions in any environment. The robot will explore the space and, given the correct set of rewards, will learn to navigate to the location of the object and then possibly learn to cooperatively move the object to the target location. The purpose of this thesis is to investigate how robots learn to cooperate in a simulated environment that has been broadly inspired by a factory warehouse. The series of experiments have been completed in three phases with different conditions of the environment: (1) fully observable, (2) partially observable, and (3) partially observable with communication.

The focus and main contribution of this thesis is to comprehensively study how properties of the environment affect learning in a new problem scenario and how communication will affect the overall learning in a partially observable environment when compared to a fully observable environment. To the best of our knowledge, the use of neural networks with reinforcement learning for the purpose of studying the emergence of cooperation in the three categories listed above, based on observability and communication, is a first. This work also shows that the availability of complete information in a fully observable environment does not guarantee cooperation but can be rather detrimental to the learning effort in a large environment whereas communication provides a more efficient way to make the relevant information available to agents.

The simulation framework for this study has been built from the ground up to enable further research in multi-agent cooperation, with a focus on inter-agent communication. The software built for this thesis includes objects that represent the environment, agents, and loads, along with an implementation of the Q-Learning algorithm that uses a neural network. Plotting and other rendering functionality for the objects are all part of the software and will be made publicly available to further contribute to this area of research. The following chapters detail out the work on this thesis.

- Chapter 1 introduces the topic of this thesis along with the motivations for the research work.
- Chapter 2 provides an overview of related work in the area of multi-agent cooperation.
- Chapter 3 defines the specific problem in multi-agent cooperation addressed in this thesis.
- Chapter 4 presents in detail the experiments we conducted in simulation.
- Chapter 5 provides an analysis of the results obtained in Chapter 4.
- Chapter 6 provides conclusions and directions for future work.

Chapter 2

Related Work

Multi-agent cooperation has seen significant progress in the past two decades. Recent application of deep learning in fields of computer vision and natural language processing has contributed to its application in multi-agent task environments. One of the earlier papers by Mahadevan et al. [12] explores behavior-based robots using reinforcement learning from a single agent perspective. A real robot called OBELIX combined with a simulator for gathering data is used for tasks that involve pushing boxes. Q-learning is used as the reinforcement learning algorithm. The paper makes a case for subsumption architecture where an agent is organized into layers where each layer is responsible for achieving specific aspects of the overall task at hand. An emphasis on the learning of actions via reinforcement learning is made in the paper because it removes the burden of programming the behavior into the agents. Reinforcement learning poses a convergence problem in a large search space. This can be overcome by decomposing a large problem into a set of smaller problems and thereby learning the smaller problems. The paper also claims to address the perceptual aliasing problem using this technique which means that the relationship between the state and the percept is rarely one-to-one. The issue of temporal credit assignment problem is addressed with the use of subsumption architecture since reward functions specific to modules limits the sequence of actions needed before a reward can be awarded. The Q-learning algorithm is used which involves learning the utility values of each state for a given action denoted by $Q(s, a)$ where s represents a state and a is an action. A weighted hamming distance computation is used to determine the similarity of states so the same action can be applied. It's shown that the algorithm successfully learns to find and push boxes. The subsumption approach is also compared to a monolithic approach where the robot learns to complete the entire task without decomposing it and as expected the monolithic approach was considerably worse. Although this paper did not consider tasks that required cooperation it formed the bedrock for using reinforcement learning in robotic learning agents.

In further development, Tan [22] studies cooperative agents with independent agents as a benchmark. Agents can be cooperative by communicating different types of information such as perception, rewards, action, episodes. The paper presents three case studies each with three different types of information, instantaneous information, sequential information, learned decision policies. In the first case study the information shared is the perception of another agent. The second case study shares learned policies and episodes. The third case study involves joint tasks where more than one agent is needed to accomplish the task. Tan's paper makes the claim that cooperative agents outperform independent agent although learning might take longer. The experiments are setup to measure success in terms of the discounted reward per action. The extension of reinforcement learning from single agents in prior work to multiple agents is defined by Tan as the comparison between n independent agents and n cooperative agents. The main thesis of the paper is that agents can benefit from the information shared between them. The experiments involve the capture of prey by hunter agents in a fixed 10×10 grid environment. Each agent has four possible actions involving the directional movements: up, down, left, right. Successful capture of prey yields positive reward whereas movements have a negative reward associated with it. Agents also have a limited visual field. The initial experiment established that a learning hunter outperformed a random one which is expected. Tan, also goes on to showing that cooperative agents outperform independent agents. The cooperative endeavor used in the experiments allows a scout to share percept information with the hunter for the location of the prey and conclusively demonstrates performance improvement which improves even further with increasing visual field depth.

Kasai et al. [7] further extends the work in [22] by enabling the learning of communication codes. The pursuit problem where agents are tasked with cooperatively capturing a prey is used for the purpose of learning communication codes. Learning of communication code is called Signal Learning in the paper and it is accomplished by encoding key features of the state space such as location of prey as a sequence of bits of fixed length. The paper examines the effect on performance when the signal length is varied between experiments. Kasai et al. claims that learning relevant information to share between agents is crucial because irrelevant information can have a detrimental effect on the overall learning process. Some of the issues identified with multi-agent reinforcement learning are concurrent learning problem, perceptual aliasing problem and credit assignment problem. The work in this paper essentially automates the information that is shared between agents whereas communication code was predefined in [22]. For the pursuit problem, the hunter sends an appropriate signal to a helper agent and also selects an appropriate direction to move based on the signal received from a helper agent and its own percept. Prey is captured when two of the adjacent four cells surrounding a prey are occupied. The hunter can perceive positions of the prey and the helping agent, and also the signal received. Encodings are formed using predefined rules for the positions and also the moving direction. Learning curves showed a trade-off between training time and accuracy. In the series of experiments performed, the length of the signal is varied from 0 to 4 and it is established that the performance improves with an increasing signal length. That is, the number of steps to catch prey decreases with an increasing signal length.

In [3] agents learn communication protocols to solve switch riddle and MNIST games in a fully cooperative, partially observable environment. Two approaches are used Reinforced Inter-Agent Learning (RIAL) and Differentiable Inter-Agent Learning (DIAL). Since agent behavior has to be coordinated, agents have to discover communication protocol. During learning communication bandwidth is not restricted, that is learning is centralized and execution is decentralized. RIAL uses Q-learning with a recurrent network and two different approaches are used within this method. The first approach is independent Q-learning where agents learn their own network parameters. The second approach of independent Q-learning parameters are shared. In DIAL gradients are passed between agents as real valued messages. During execution, since only limited bandwidth channels are available, the real valued messages are discretized. Agents don't observe the underlying Markov state but receive a private observation. Agents select an environment action and communication action at each time step. Protocols are defined as mappings from action-observation histories to sequence of messages. Since the space of protocols is high dimensional and because agents need to coordinate both sending and interpreting messages, discovering an effective protocol is challenging. Positive reward for the coordination of correct interpretation of message sent by another agent is necessary for discovering an effective protocol. Experimental results show that DIAL outperform both RIAL methods. Parameter sharing is also found to be crucial for learning the optimal protocol. The baseline also shows that no learning takes place without any communication. The paper establishes that differential communication is essential to utilize the full potential of centralized learning. The paper claims to be a first attempt at learning communication and language with deep learning approaches.

Gupta et al. [5] consider cooperation in partially observable environments without communication. The problem of cooperation can be formulated as a decentralized partially observable Markov decision process as seen before. Gupta et al. claim that policy gradient methods outperform both temporal difference and actor-critic methods. However, since exact solutions are intractable, approximates have to be applied. It is worth considering that since approximate methods are limited to discrete action spaces applying them to real-world problems require careful design considerations. Further, the combination of deep learning and approximate methods has given rise to deep reinforcement learning and has seen considerable success in complex tasks such as robotic locomotion. The paper focuses on temporal-difference learning using Deep Q Networks, policy gradient using Trust Region Policy Optimization and A3C. Three different training schemes are also considered and that includes centralized training/execution, concurrent training with decentralized execution and parameter sharing during training with decentralized execution. Four complex multi-agent tasks that required cooperative behavior were chosen to benchmark the performance of the approaches. A3C was found to outperform in the discrete space whereas TRPO outperformed other algorithms in the continuous space. The paper goes on to show that deep reinforcement learning allows scalability in large continuous space.

Machine learning is capable of capturing statistical patterns of natural language which are insufficient to make interaction with human agents possible. Emergence of compositional language is investigated in a multi-agent environment in [15]. Compositional language is the combination of multiple words to create meaning. The motivation for this work is to understand the development of human language and cognition. A pragmatic view is adopted which states that use of language implies understanding of language. The compositional language in focus is a stream of symbols with a coherent structure uttered by agents while also performing actions to cooperatively accomplish tasks. Agents form concepts and arbitrarily assign symbols. Symbols refer to landmarks, action verbs and agents. Language that emerge are specialized for the environment. Words for concepts do not form if disambiguation is unnecessary in a given context. The work also provides insights for the emergence of compositional structure. Compositionality emerges only when the number of concepts exceed vocabulary size. Moreover, training on a diverse set of configurations also has an effect on compositionality.

The setting is an extension of Markov Decision Process (MDP) since multiple agents are involved. All agents have identical actions and observations spaces. The policy is used by all the agents to execute an action and the reward agents receive is shared. The problem is formulated to finding a policy that maximizes the expected shared reward and uses a reinforcement learning approach. Agents and landmarks possess physical location and characteristics such as color and shape. Agent actions that are allowed are motion in the environment and direction of their gaze. Agents also utter verbal communication symbols at every time step which are observed by all the other agents. The symbols have no predefined meaning and it is during the training that agents assign meaning to symbols. Agent goals are kept private and include moving to a specific location or gazing at a location. Agents also have a private memory bank that they learn to use effectively. Since goals are private and may require action from a different agent such as one agent telling another to go to a specific location, communication is necessary for achieving goals. Communication is allowed using only discrete symbols from a fixed vocabulary. Goal positions cannot be directly communicated since agents only possess positions from their own reference frames. Moreover, the association between the position vector and the symbols have to learned. When trained on environment with two agents, symbols begin to appear for landmarks and actions. When more agents are introduced, symbols begin to appear for agents to refer to them.

Lan et al. [9] show that discrete language appears even when using a continuous space. Their work uses signaling games where an agents perceives some information and produces a message for a receiver that takes an action. Discreteness, which is clustering of words in acoustic space and displacement, which occurs with communication beyond the immediate context are the focus here. Besides the emergence of discrete language, using a continuous communication protocol means that standard back-propagation can be used for end-to-end training. Function Games are introduced which include a set of contexts and actions, as well as a family of functions from contexts to actions. The sender perceives context c and function f from the environment and sends a message m. The receiver receives the message along with a different context c' and chooses action a'. Both sender and receiver gets a reward if $a' = f(c')$. The function f determines the appropriate action. As an example, the context could be a set of objects. Lan et al. use the Extremity Function Game and success in communication is measured in terms of the accuracy of the receiver retrieving the correct object.

Multi-agent cooperative problems can also be formulated using a game theoretic approach. The work of Bhat et al. [1] is a study of the distributed task execution problem where the goal is to maximize the total value of completed tasks in a discrete environment within a time window. The problem is formulated as a potential problem and it was shown that a payoff-based learning algorithm can maximize agent performance. Marden et al. review the game theoretic concept of potential in [13]. A related problem is a distributed graph coverage problem and [24] presents a study of this formulation which is described by a group of agents which are arbitrarily deployed on nodes in a graph. Agents have to maximize coverage of the nodes by exploring the graph and covering nodes according to the regions they are assigned responsibility. The paper shows that with sufficient communication range agents can maximize the covered nodes using the proposed approach of a potential game.

Several more papers have been published regarding multi-agent cooperation with variations such as those focusing on task allocation or task execution. So far the related works cited have been those that are deliberate in their approach and the agents themselves have a level of sophistication. Yet another research area focuses on numerous yet simple agents to demonstrate swarm intelligence. Inspired by the natural foraging behavior of ants, [18] discusses optimization algorithm for combinatorial optimization problems which are generally NP-hard. The paper approaches the problem of finding optimal combinations of problem components by partitioning the problem. For ants, swarm intelligence is exhibited by individuals cooperating to accomplish a task while indirect communication takes place through chemical signals. The experiment is set up so that given two paths of differing lengths individual agents will show a higher preference for the shorter path because of their ability to cover those trails faster and hence being covered by more agents leaving a stronger pheromone.

Gerkey [4] presents a formal study of multi-agent task allocation which essentially determines which robot should execute which task. Task independence is assumed and order constraints are not permitted. Two type of cooperative behavior are identified, intentional and emergent. Intentional cooperation occurs through communication and negotiation. In an emergent approach agents cooperate through interaction and with no explicit communication or negotiation. The paper only considers intentional cooperation and does not propagate to the level of task execution.

Korsah [8] deals with intentional cooperation where tasks are explicitly assigned to agents as a follow up to the taxonomy provided by [4] and provides a more complete taxonomy involving interrelated utilities. The new taxonomy, called iTax, addresses different levels of dependencies between tasks for different problems and breaks it down into four categories – no dependency, in-schedule dependency, cross schedule dependency, and complex dependency. In cross-schedule dependency the utility of a robot depends not only on its own schedule but also on the schedule of other robots and is relevant to the simulation experiments performed for this paper although task execution is the main focus. The more recent paper by Nunes at al. [16] focuses on task allocation with an emphasis on temporal and ordering constraints, and extends the taxonomy to include different types of temporal and ordering constraints.

Stone [20] discusses coordination without preconditions where an ad-hoc team of multiple agents with different capabilities coordinate on a set of tasks with perfectly aligned goals. Robot soccer with the variation of a pickup game is an example of such an ad-hoc cooperative strategy that can be studied within the framework of game theory. The paper elaborates on the collaborative multi-arm bandit with a teacher and a learner and challenges the community to develop other approaches. Multiple papers have been written since that initial paper on ad-hoc teams.

Other forms of learning such as social learning have been investigated in [19]. Social learning as opposed to individualistic learning involves the learner interacting with a mediator who sets constraints, provides feedback and demonstrates actions so the learner can associate meaning to words. Individualistic learning takes a more passive role to learning where feedback from the speaker is seldom provided. Social learning is essentially an imitation of the learning process enacted by an infant acquiring language. The experiments performed in [19] focus on naming objects in the form of imitation games.

Chapter 3

Problem Definition

Given the different approaches outlined in the related work Chapter, the problem of cooperation can be broadly split into three categories of the environment which affects how much relevant information is available to the agents.

The first category of fully observable environment has all state information available to the agents, however, some of this information may not be relevant for learning the cooperative task.

The second category of environment is a partially observable one with no communication between agents. Agents may not have all the relevant information available at their disposal for learning the cooperative task at hand.

The third category is a partially observable environment with communication where state information may not be available but agents have the ability to share, send and receive additional information that might be relevant to the task at hand. We want to investigate if agents under each of these conditions tasked with a novel problem will learn to cooperate.

The specific problem created to investigate these conditions is the "warehousing problem" which is a simulated environment motivated by a real-world warehouse where robots might be tasked to find an object and then cooperatively move it using the help of another agent. Since fully observable environments by definition make all the state information available to the agents this is expected to hold true, however, it is not expected to be scalable since in a large environment the additional information that is not relevant to the learning task would potentially have a negative impact on learning because the state space would grow exponentially. Hence, given the large observation space we also want to investigate the scalability of the learning schemes in a fully observable model. We use neural networks with reinforcement learning algorithm for scalability. However, we also want to investigate the feasibility of training a neural network since learning with a large state space requires additional parameters.

The second problem we wanted to explore is whether cooperation will emerge in a partially observable environment using the same warehouse problem. In this case the visibility of the agent is limited only to its current location, which ensures a very small observation space regardless of the size of the environment. The agent is also given information about the location of the load and that of the goal. No information about the location of the other agents is available in the partially observable version of the warehouse problem. We want to investigate if a partially observable environment with no communication is conducive to learning. Moreover, the same question of scalability concerning the environment and the neural network still hold in this scenario.

The third problem we wanted to address also involves a partially observable environment but where communication between the agents is enabled. In a partially observable environment, agents may not have all the information needed to learn the optimal action for each given state. Communication give the agents the ability to share information that the other agents may not possess. Besides the question of whether communication will emerge, we also investigated if communication will allow the agents to learn more or less efficiently, specifically using a neural network.

The problem can be seen as an instance of the piano mover problem [17], simplified, because we assume there is space to move the load from the initial place to the destination. We also need the robots to be in touch with the object, but we do not use force sensors on the robots to sense the forces they exert on the load.

According to Gerkey's taxonomy [4] our problem is ST-MR-IA. The taxonomy is based on three main characteristics of the robots, tasks, and time, as follows:

- Single-task robots (ST) vs. multi-task robots (MT): ST robots can do at most one task at a time, while MT robots can work on multiple tasks simultaneously.

- Single-robot tasks (SR) vs. multi-robot tasks (MR): SR tasks require a single robot to be completed, while multiple robots are needed to complete an MR task.
- Instantaneous (IA) vs. time-extended (TA) assignments: in IA, tasks are allocated as they arrive, while in TA, tasks are scheduled over a planning horizon.

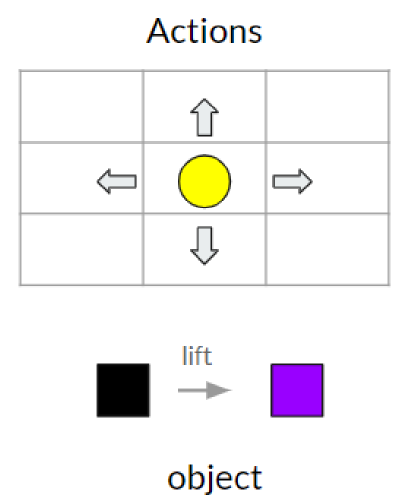

Figure 3.1: Action space; the color change of the load from black to purple indicates it has been successfully lifted

More formally, we define our environment as a $n \times m$ grid with discrete space identifiable by an x and y coordinate. Agents $r_i, r_{i+1}, ..r_{i+j} \in R$ are randomly placed in this discrete space. Loads l_i, l_{i+1}, l_{i+k} are also available in the discrete space in a fixed location, along with a fixed goal location g. Loads are allocated to one or more agents. The task to be completed by agent r_i is to first learn the sequence of actions, $a_i \in A$ to move to the location of the load, l_i that the agent is assigned to. Loads can have a minimum threshold number of agents to be moved, typically 2, so assigned agents have to cooperatively move the load to target g. The action space A seen in 3.1 includes the four directional movements and lift. At each time step t, agents take an action. Moving the load requires a synchronous series of actions by different robots, including the lift action. If agents move in different directions after lifting an object, they lose hold of the object and the object has to be lifted again for moving. The robots will likely learn an optimal, i.e., shortest, path from the load location to the goal, but optimality of the path is not a requirement, unlike what happens often in robot motion planning [10].

The approach we take is to use the Q-learning algorithm [23] with a neural network to learn the Q-values. The use of a neural network in conjunction with Q-learning was proposed in [2], but instead of using backpropagation we use a dense network and deep learning.

Chapter 4

Experiments

In the simulation experiments, a varying number of agents are trained in the grid world to perform the two tasks of (1) finding the loads assigned to them and (2) haul them to their destination with the help of one or more other agents. The agents use an emergent approach to learn the behavior, since there is no direct communication or negotiation to induce cooperation. Reinforcement learning, and more specifically the Q-learning algorithm was chosen as the learning algorithm to see if cooperative behavior would emerge. A multi-layered neural network was used for learning the Q-values.

The agents can execute an action $a \in up, down, left, right, lift$. The loads that are already on the grid have to be picked up using the lift action and synchronously moved to the target location. The loads are not allowed to move without the aid of agents and may require multiple agents for the lift to produce the intended effect on the object. Learning to cooperate between agents involves performing the lift operation and also learning to synchronize the move actions after picking up the load to cause the object to move in a specific direction.

At each step, the agents are either penalized or rewarded following the completion of an action. A collision with the wall does not change the state of the agent but penalizes more heavily compared (penalty is -5 for collision) to a non-rewarding step (penalty is -1). Successfully finding a load rewards the agents significantly ($+50$ reward) and moving the load to the target location results in the maximum possible reward ($+100$ reward). All motion are deterministic and the environment is fully observable in the first set of experiments. The series of experiments are then repeated in a partially observable environment with and without communication.

An example of the environment with a load in grid (0,0), target in (2,1) and two agents, in grids (2,0) and (0,1) respectively are shown in Figure 4.1. The agents are shown as yellow circles, the load as a black square, and the target location as a green cross.

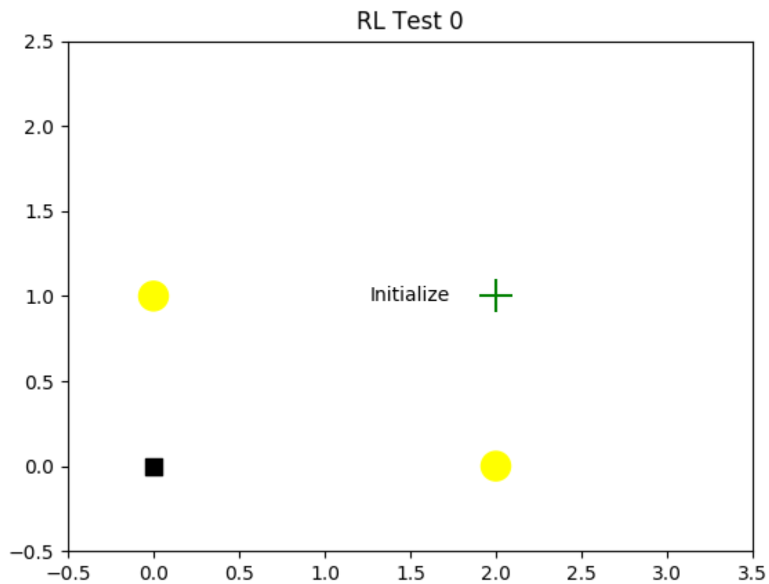

Figure 4.1: 4×3 grid world with 2 agents and one load

The neural network used for the experiments involving the fully observable environment is a densely connected network of 4 layers, including the input and output layer. The input layer takes in the state, which is the grid with numeric values assigned according to the occupant of each cell within the grid. The state also includes a bit that indicates whether an agent has successfully picked up a load. The ReLU activation function was necessary to allow the agents to learn to pick up the object and also to find the target location. For the training itself minimizing the mean squared error was sufficient to achieve a successful completion of the task. The mean squared error formulates the loss function for training the neural network.

In the case of the partially observable environment the neural network is a densely connected network of 3 layers including input/output layer.

Both the fully observable and partially observable case use separate neural networks for each phase of the task: (1) finding the object, and (2) moving the object to the target location.

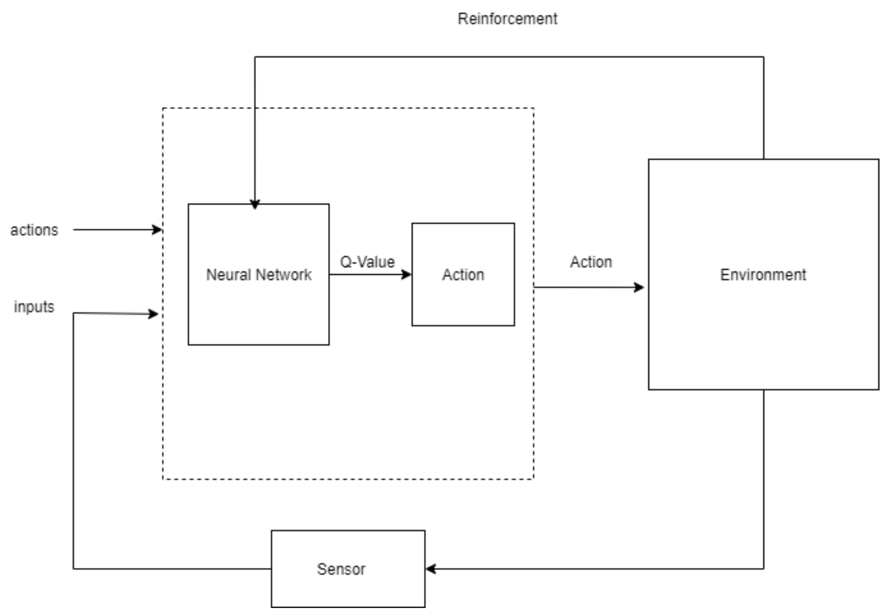

Figure 4.2: Neural Network Based Reinforcement Learning

The Q-learning algorithm used for the series of experiments uses a neural network backend, as shown in Figure 4.2. At each step, agents do an action in the environment and receive a sensor percept, which includes the reward for the outcome state. The percepts form the input to the neural network which is trained over a fixed number of times steps. The output of the neural network are the Q-values. The action corresponding to the highest Q-Value for a given state determines the best action.

Agents can use a fixed communication protocol that involves placing messages in the environment that are essentially the x and y coordinates of their current location. Agents place their location information in a shared buffer that is available for all the agents in the environment. The shared buffer becomes part of the percept sequence of all the agents, making the location information of the other agents available at each time step. This communication is enabled in a series of experiments in the partially observable model and is compared to the series of experiments with no communication.

4.1 Experiments and Results for Fully Observable Environments

4.1.1 4 × 3 grid with a single agent and a single load

The first experiment with a single agent and a single load is a relaxed version of the problem, where the load is allowed to be lifted by a single agent. All subsequent experiments involve more than one agent and require more than one agent to move the load. Figure 4.3 shows the initial and final state.

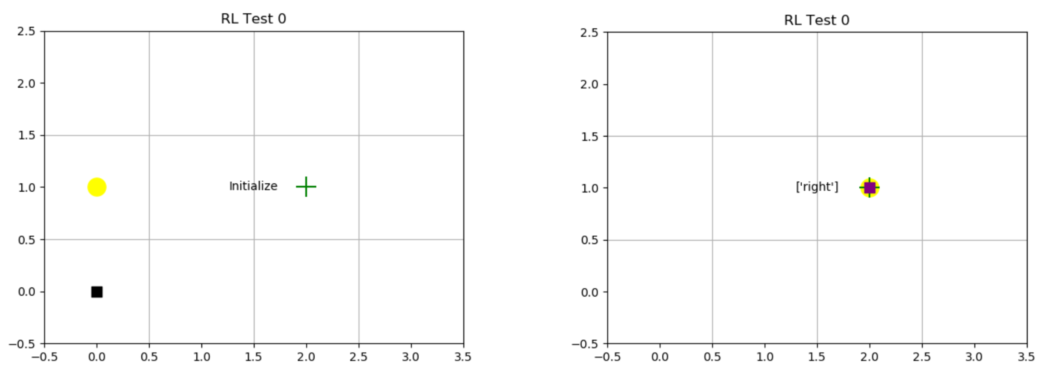

Figure 4.3: Initial state and final state for the experiment with a 4 × 3 grid, a single agent, and a single load

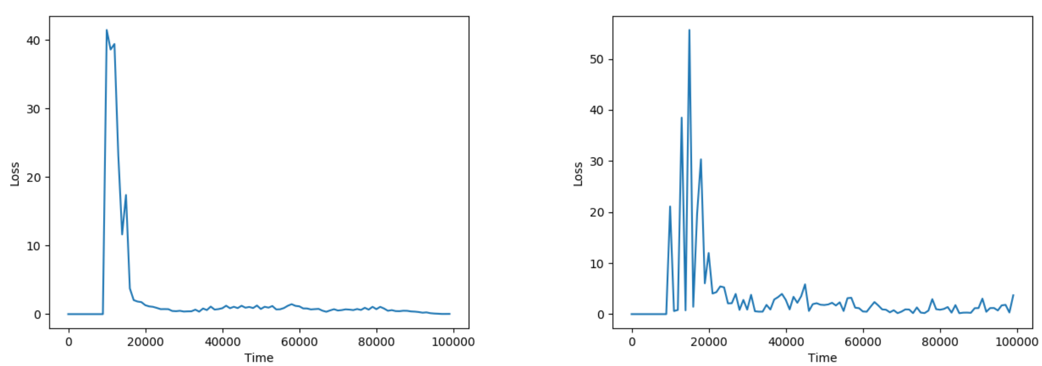

Figure 4.4: Training for the experiment with a 4 × 3 grid with a single agent and a single load, showing the loss over time steps. The left figure is for the search part and the right figure is for the haul part

Figure 4.4 shows convergence within 200,000 epochs for the agent to learn the task. The agent successfully completes the task following the training and achieves the maximum possible reward, as seen in Figure 4.3, where the plus sign indicates the target location.

We ran the experiments 10 times, each with different initial locations for the agents. This prevents the neural network to memorize the path of the agent from the initial location to the load.

The results were used to establish baseline numbers for the training time and to prove that a single agent can be successfully trained to find the load, pick it up, and move it to the target location. The constraint for the load to require multiple agents was relaxed only for this experiment. Even for this relatively straightforward task involving a single agent, two separate neural networks were necessary, the first for the search task, where the agents find the load, and the second for the haul task, where the load is moved to the target location.

4.1.2 4×3 grid with two agents that move a single load

Multi-agent cooperation was established in this phase where two agents were trained simultaneously, following an exploration phase of the grid. Similar to the single agent training, the task of search and haul were separately handled using two distinct neural networks. The constraint for requiring at least two agents to handle the load was applied in this phase. The convergence was expected to be slower, because the probability of hitting the target is greatly reduced with two agents, so a larger training set was used. Training between a single agent and multi-agents differs essentially in the way data points are generated during the training session. Every step which occurs simultaneously generates two intermediary states that are fed into the agent percepts, and appears as a training data point within the neural network. An alternate solution might have been to simply train a single agent longer. However, the synchronization required for cooperation may not occur if agents are trained separately. The agents successfully completed the task following the training and achieved the maximum possible cumulative rewards.

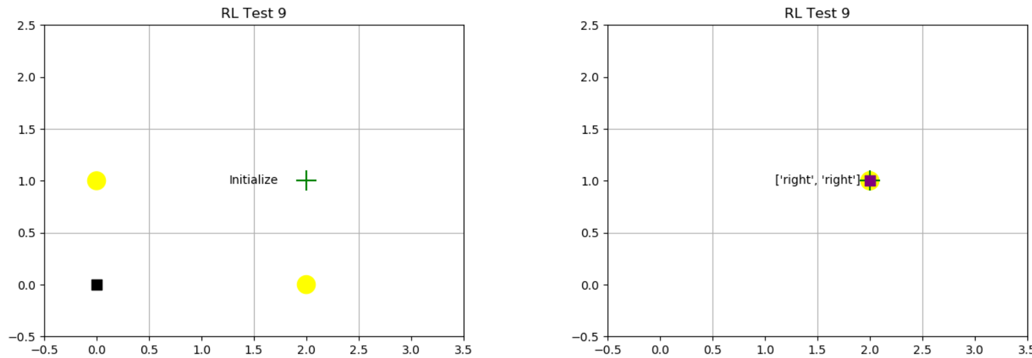

Figure 4.5: Initial state and final state for a 4 × 3 grid with two agents that have to move a single load

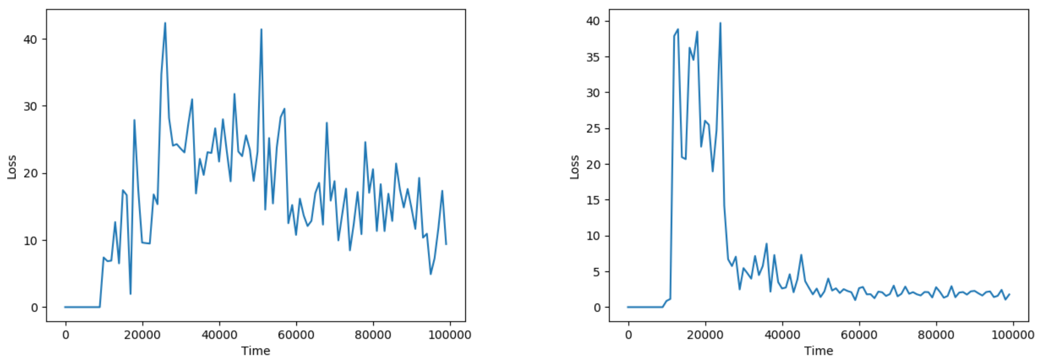

Figure 4.6: Training for the experiment with a 4 × 3 grid, two agents and a single load showing loss over time steps. The left figure is for the search part and the right figure is for the haul part

4.1.3 8 × 6 grid with 8 agents that move multiple loads in pairs

Training for the 8 × 6 grid with 8 agents required the pre-allocation of loads to agents. This allocation remained fixed through the learning process. The agents find the loads they are assigned and move them to the target location indicated in Figure 4.7 by a green plus symbol.

With the expanded search space and the increase in the number of agents training had to be performed on a substantial amount of data to achieve convergence. However, in one case one of the agents failed to reach its target location in all the test runs. All the other agents successfully completed the task after the training and achieved the maximum possible cumulative rewards.

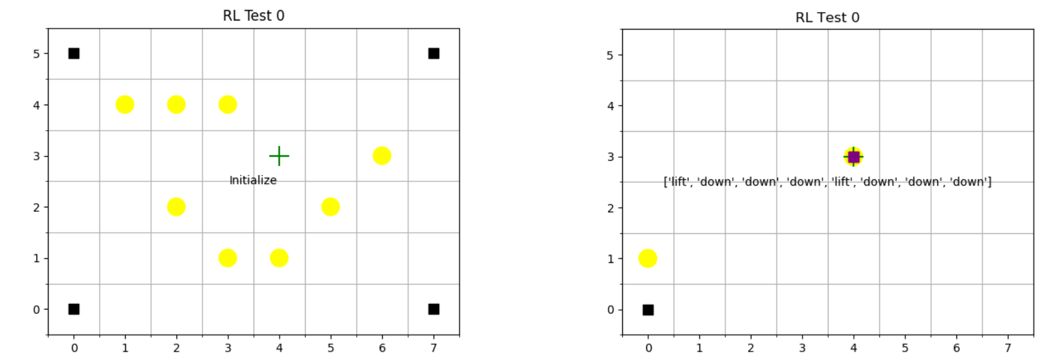

Figure 4.7: Initial state and final state for a 8 $\times$ 6 grid world with 8 agents that have to move 4 loads

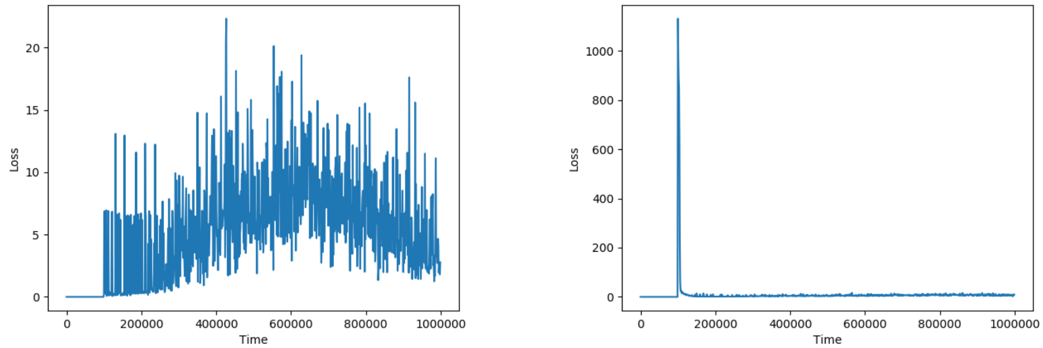

Figure 4.8: Training for the experiment with a 8 $\times$ 6 grid, with 8 agents and 4 loads. The left figure is for the search part and the right figure is for the haul part

4.2 Experiments and Results for Partially Observable Environments

4.2.1 4×3 partially observable grid with 2 agents and a single load

The same experiment was repeated in a 4×3 grid in a partially observable environment, where agent perception was limited to its own location, the goal location and the load location. No information about other agents was made available in the percept sequence. The neural network architecture used was similar to the one for the previous experiments, with one network for the search and a second one for the haul task. However, fewer layers were needed to achieve learning. Since, the number of input parameters were also reduced, the neural network had fewer overall parameters and hence could be trained more efficiently compared to the fully observable model. After $100,000$ iterations the agents learned to accomplish the task of finding the object and cooperatively moving it to the goal location. In Figure 4.9, the initial state and final state are shown in the test run following training. Figure 4.10 shows that training was achieved within just 100,000 iterations.

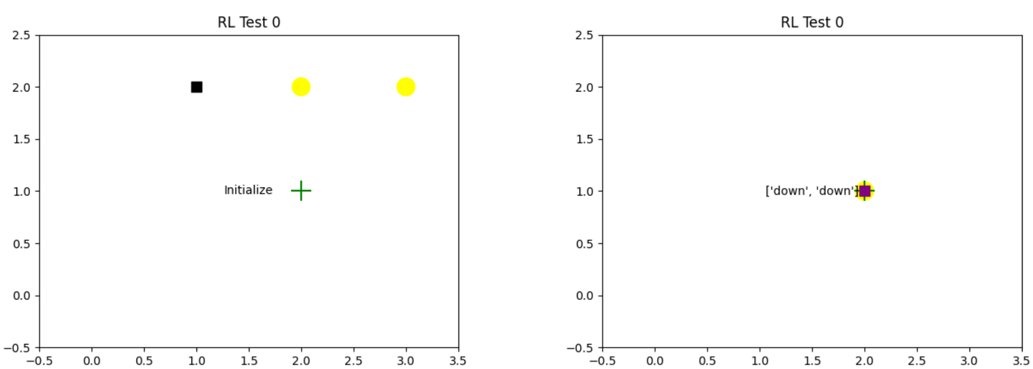

Figure 4.9: Initial state and final state for a 4×3 grid with two agents that have to move a single load in a partially observable environment with no communication

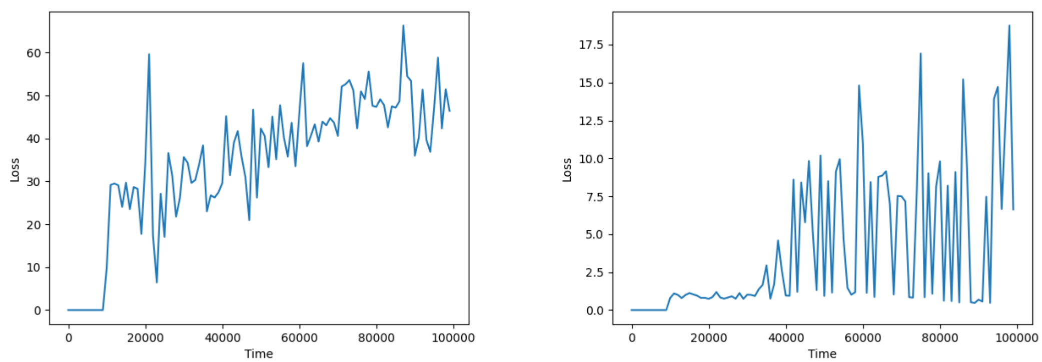

Figure 4.10: Training for the experiment with a 4×3 grid, two agents, and a single load showing loss over time steps. The left figure is for the search part and the right figure is for the haul part

4.2.2 5×5 partially observable grid with 2 agents and a single load

In a 5×5 grid with a partially observable environment, the agents also learned to find the object and cooperatively move it to the goal location. However, the training time doubled compared to the 4×3 case for the learning to take place. This is expected because the environment size is about twice as large in terms of the number of discrete locations compared to the 4×3 case. Figure 4.11 shows the initial and final states from the test run following the training of $200,000$ epochs shown in Figure 4.12.

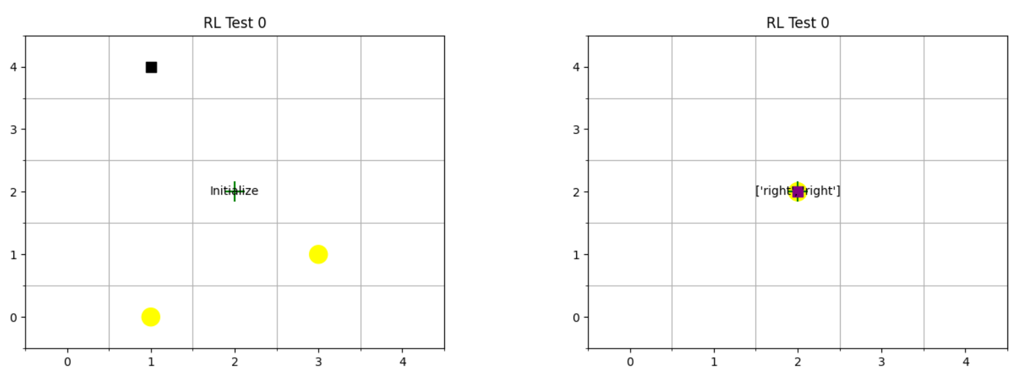

Figure 4.11: Initial state and final state for a 5×5 grid with two agents that have to move a single load in a partially observable environment with no communication

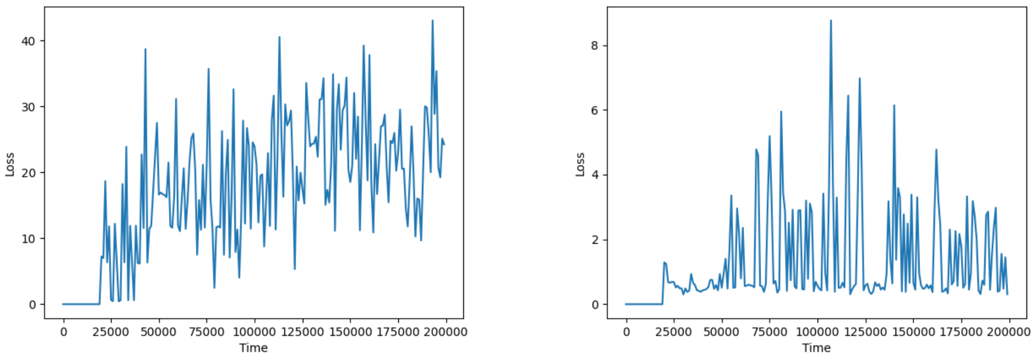

Figure 4.12: Training for the experiment with a 5 × 5 grid, two agents, and a single load showing loss over time steps. The left figure is for the search part and the right figure is for the haul part

4.2.3 8×6 partially observable with 2 agents and a single load

Cooperation was not achieved in the partially observable case of the 8×6 grid as seen in Figure 4.13. The training time was increased to $1,000,000$ epochs which is shown in Figure 4.14. This was a 5-fold increase compared to the 4×3 case but the agents failed to complete the tasks in the test runs. The rewards achieved were also much less compared to the optimal case but still better than a random test run.

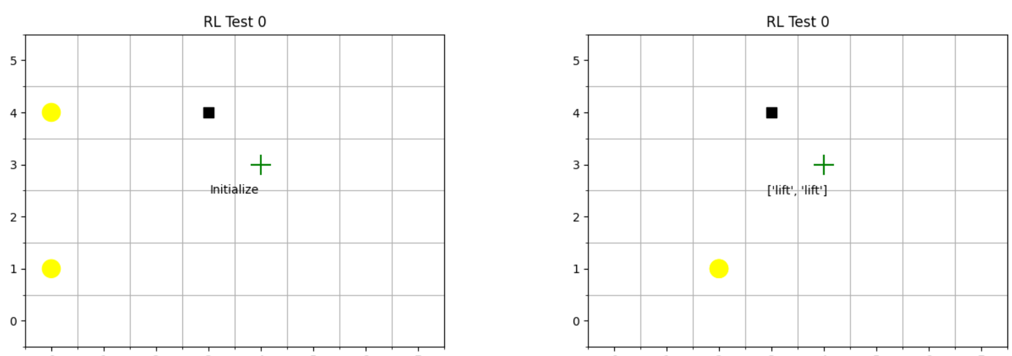

Figure 4.13: Initial state and final state for a 8 × 6 grid with two agents that have to move a single load in a partially observable environment with no communication

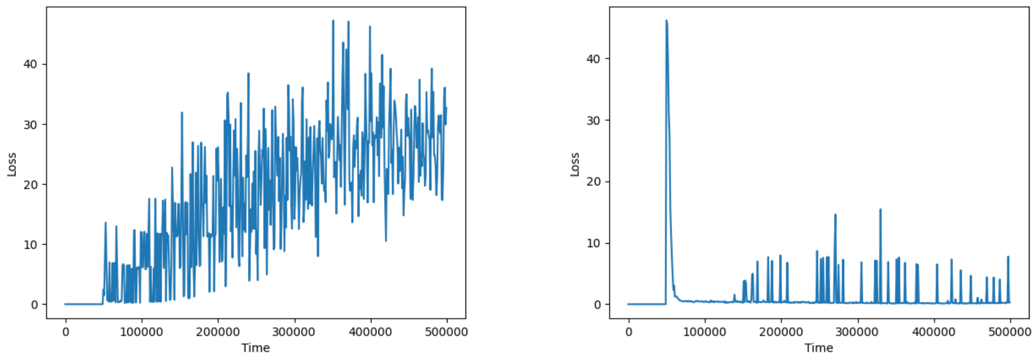

Figure 4.14: Training for the experiment with a 8×6 grid, two agents, and a single load showing loss over time steps. The left figure is for the search part and the right figure is for the haul part

4.3 Experiments and Results for Partially Observable Environments with communication

4.3.1 4×3 partially observable grid with 2 agents and a single load and with communication

The series of experiments from the partially observable model were repeated using communication. In the first experiment with the 4×3 grid the agent perception was just limited to the goal location and obstacle location. However, agents were allowed to send and receive messages from other agents. The neural network architecture used was the same as the partially observable case with two separate networks for the search and for the haul phase. The number of input parameters was increased since the communication messages were also included in the training. Despite the increase in the overall training parameters, after $200,000$ epoch, as seen in Figure 4.16, the agents learned to accomplish the task of finding the object and cooperatively moving it to the goal location. The initial and final location are seen in Figure 4.15.

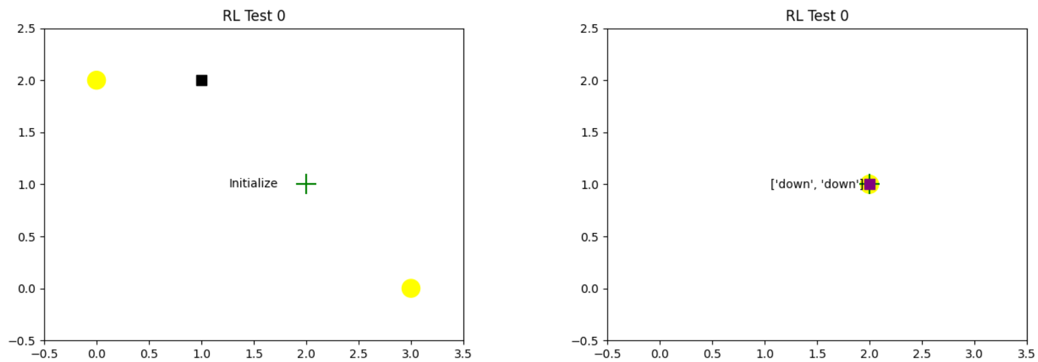

Figure 4.15: Initial state and final state for a 4 × 3 grid with two agents that have to move a single load in a partially observable environment with communication

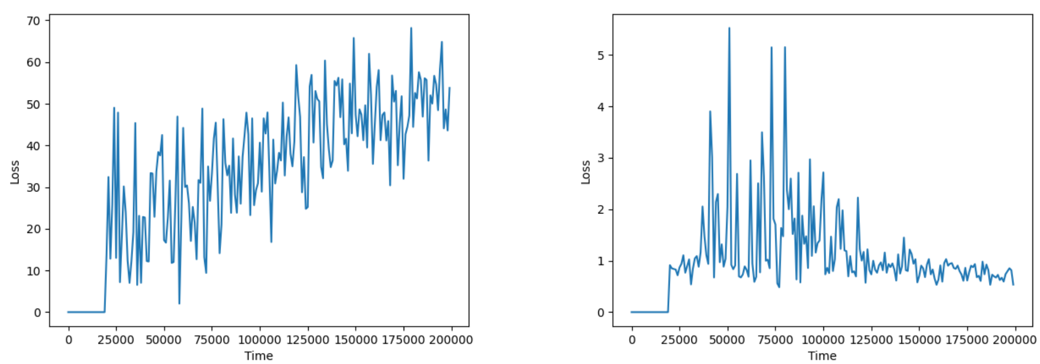

Figure 4.16: Training for the experiment with a 4 × 3 grid, two agents, and a single load showing loss over time steps. The left figure is for the search part and the right figure is for the haul part

4.3.2 5 × 5 partially observable grid with 2 agents and a single load and with communication

In a 5 × 5 grid with a partially observable environment and with communication, the agents also learned to find the object and cooperatively move it to the goal location, as seen in Figure 4.17. However, the training time shown in Figure 4.18 was significantly greater than for the partially observable case with no communication. The was likley due to the increase in the number of input parameters to the neural network.

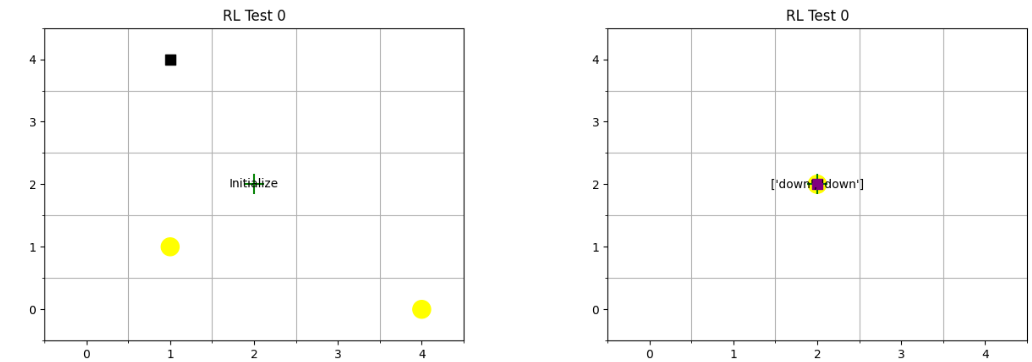

Figure 4.17: Initial state and final state for a 5 × 5 grid with two agents that have to move a single load for the partial observable environment with communication

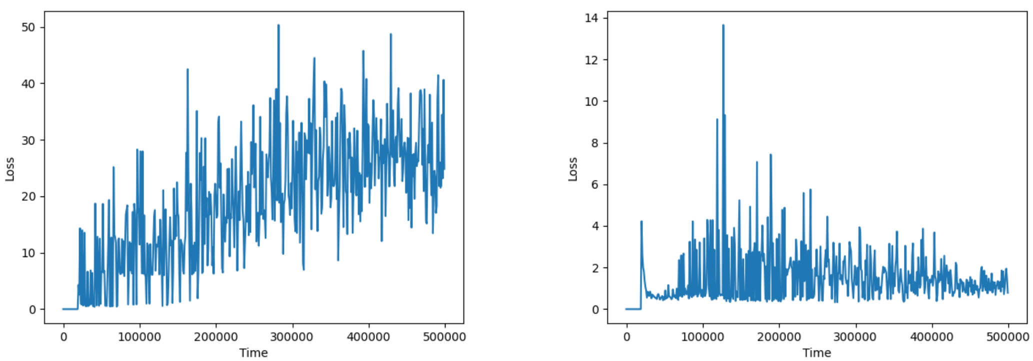

Figure 4.18: Training for the experiment with a 5 × 5 grid, two agents, and a single load showing loss over time steps. The left figure is for the search part and the right figure is for the haul part

4.3.3 8×6 partially observable with 2 agents and a single load and with communication

Learning was successfully achieved in the partially observable case with communication with the larger grid size of 8×6. However, the training time increased to about $1,000,000$ epochs. The rewards achieved were smaller compared to the previous cases but still close to the expected maximum reward achievable. This is because since even the optimal policy would take more steps in a larger grid space. Figure 4.19 shows the initial and final state of the test runs for the 8×6 case. The increased training time as seen in

Figure 4.20 is comparable to the training time seen for the partially observable case with no communication.

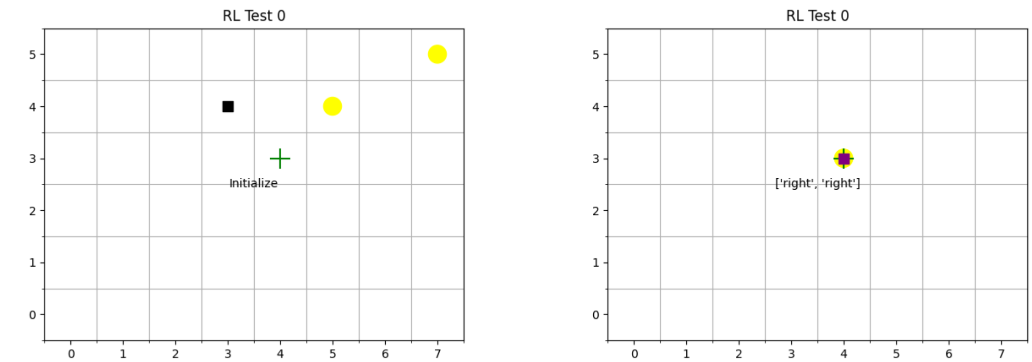

Figure 4.19: Initial state and final state for a 8×6 grid with two agents that have to move a single load for the partially observable environment with communication

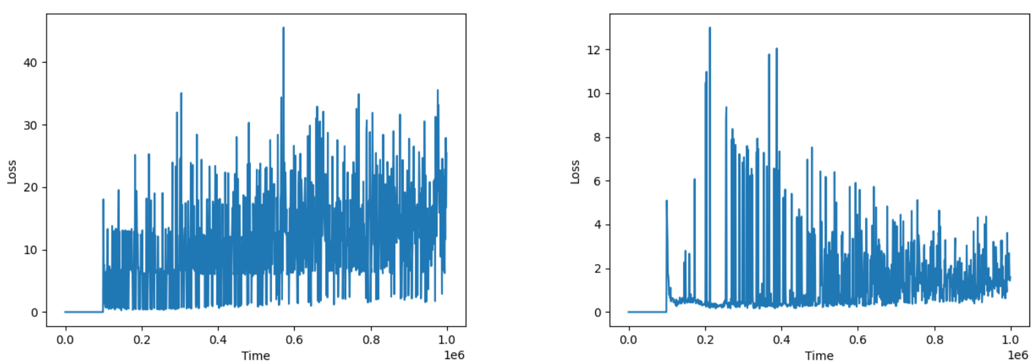

Figure 4.20: Training for the experiment with a 8×6 grid, two agents, and a single load. The left figure is for the search part and the right figure is for the haul part

4.4 Experiments and Results for a Single Neural Network

A series of experiments using a single neural network was performed where agents had to learn a single task, which included the two subtasks search and haul. The task would require agents to learn to find the object and haul the object to the goal location using a single neural network. However, agents failed to learn these tasks even when a single agent was used in the relaxed version of the problem where loads could be moved by a single agent. This happened regardless of the size of the grid space and the number of training steps. This probably failed because the two subtasks required learning two different sets of parameters, since the destination location is different in each case.

4.5 Fixed versus Random Initial Locations

Another variation that was tried in the series of experiments is to randomize the location of the obstacles and the goal. Since a fixed initial position for each agent was found to be limiting in its ability to learn, a random series of positions were chosen for the experiments. However, the goal and obstacle location remained fixed since learning within a reasonable amount of training time could not be achieved.

Chapter 5

Analysis

The series of experiments in the three different cases of fully observable, partially observable with no communication, and partially observable with communication, show that cooperation is achieved in the case of the 8×6 partially observable environment. In all the other cases the agents learned the optimal set of actions in both the search phase and the haul phase. Of particular importance is that the agents learned to synchronize their actions to haul the object at each step exhibiting cooperation. In the specific case of the 8×6 fully observable environment, one set of agents consistently failed to move the load to the goal location whereas the other set of agents succeeded in all the test runs. Cooperation did not emerge in the two cases with no communication between agents but with communication we see that it is possible for cooperation to emerge even in a large environment. This seems to indicate that communication is necessary for cooperation in large partially observable environments. The training time across all the experiments were consistently kept below $1,000,000$ time steps. It appears that the slightly better performance with the larger grid in the partially observable environment with communication can be attributed to the agent's ability to share information. However, more experiments would be needed to establish this conclusively, in particular with even larger environments. Although the series of experiments we performed does not provide conclusive evidence, it motivates further work in this area, particularly on the observation space and information that is shared between agents.

A more detailed analysis of the rewards achieved during the series of experiments for the fully observable environment (see Figure 5.1) shows that a reinforcement learning agent definitively performs better than an agent randomly executing an action. Rewards achieved through reinforcement learning are significantly higher compared to just random actions even in the case of 8×6 grid where one set of agents was unable to learn cooperation. Furthermore, agents executing random actions rarely see any rewards, particularly in a large environment, even when the environment is fully observable because agents do not use the information that is available to them. This is further evidence that the agents are indeed learning to perform the tasks and since the tasks can only be accomplished through cooperation, they are exhibiting cooperative behavior without explicit instructions to do so.

A further analysis of the training time shows that the larger grids need an increase in the number of time steps needed to train. Moreover, the training time with communication was greater compared to the training time with no communication. This is because the larger state space correlates with a larger input size of the neural network.

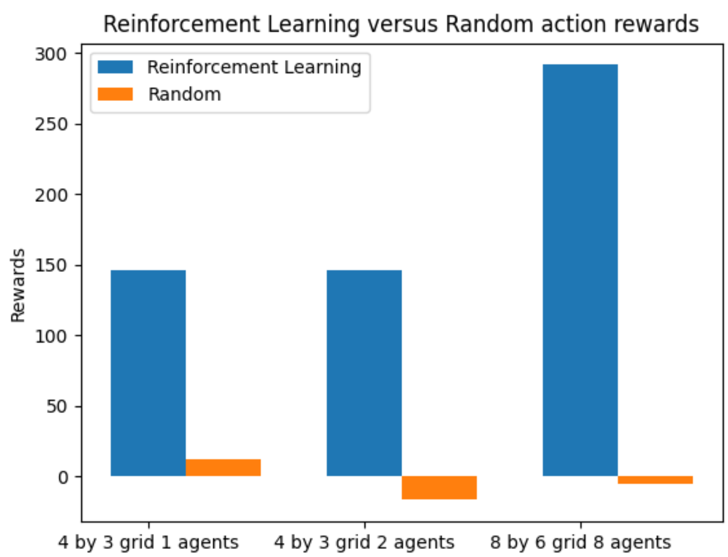

Figure 5.1: Reward comparison for the different experiments: 4×3 grid with a single agent and a single load, 4×3 grid with two agents and a single load, and 8×6 grid with 8 agents moving multiple loads in pair

Figure 5.2 shows that agents successfully learn to cooperate in the smaller grid spaces but in the larger 8×6 grid, agents succeed in finding the objects but fail to move the objects to the target location. They still perform considerably better than random agents but do not achieve the maximum possible rewards.

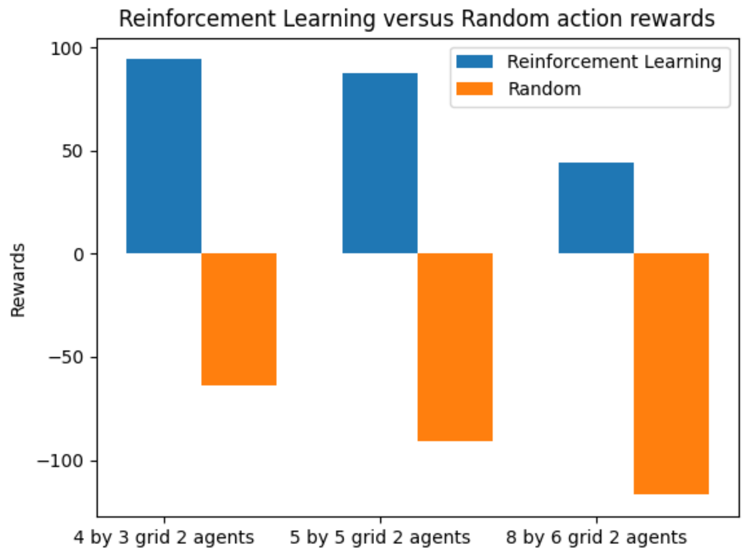

Figure 5.2: Reward comparison for Partially Observable Environment: 4×3 grid with two agents and a single load, 5×5 grid with two agents and a single load, and 8×6 grid with 2 agents and a single load

Besides the successful experiments where the expected behavior was observed with the agents, the experiment with a single neural network was particularly unexpected when the grid was at its minimum size at 2×2 and only one agent was placed in the grid only one step away from the load. Even with just one agent and one load, a single neural network failed to learn the task completely. Two separate neural networks had to be used for learning the sequence of actions for the two phases of search and haul, since these could not be learned using a single network. A variation in the sequence of steps for accomplishing the task was also tried, which involved automatically lifting the object when the agent was at the location of the object so the lift action did not need to be learned. This led to successful training for the single agent and completion of the task. However, this particular approach was not used because the premise of the problem was that agents learn each action including the lift operation that are crucial for the warehousing problem.

Chapter 6

Conclusions and Future Work

We have shown that the agents successfully achieved the desired cooperative behavior by working as a team to accomplish a complex task which included two subtasks. Although two separate neural networks were needed to learn each phase of the task, we have shown that complex tasks that can be broken down into simpler subtasks and there is potential for emergent cooperation, that is, without explicitly teaching the agents to cooperate, agents can learn to cooperate.

The Q-learning algorithm that was used proved to be effective and maximized the utility values in all the experiments performed in the fully observable environment. In all cases, the learning agents outperformed a random agent, including the case where a set of agents failed to learn the overall task. Similarly, in the partially observable cases all agents succeeded in learning the task, except in the case of the larger grid. However, this was remedied by allowing the agents to communicate which made the necessary information available for learning to occur. Given the limitation of learning in a fully observable environment and a partially observable environment we can conclude that agents with the ability to communicate will generally outperform those with no means of communication.

In our work, learning is centralized. The information that comes from the agents and the environment is fed into a neural network, which returns Q-values and the best action. Execution is decentralized. Each agent executes the policy, communicating with the other agents only when communication is allowed. Centralized learning and decentralized execution are quite common, but an avenue for future research is to explore distributed learning. Social influence, which has been shown to be a motivation for deep learning [6], could be used to support distributed learning.

In the natural world emergent cooperation is seen in animals and insects that do not necessarily exhibit higher intelligence in the form of language. However, cooperation is also evident in social animals that exhibit some forms of communication. Communication can also emerge in competitive situations, where competition acts as an additional pressure to improve performance (e.g., [11]). Further work investigating intentional cooperation to contrast the results with emergent cooperation will provide interesting insights into the relation between language and cooperation.

Another avenue to investigate is the use of recurrent neural networks to improve the learning process using a single neural network. This would further simplify the overall learning process for problems that can be decomposed in simpler subtasks and could be applied to even more complex tasks that require a decomposition in several more subtasks. The taxonomy for task allocation presented in [8] distinguishes "simple tasks" which can be decomposed into atomic non-decomposable tasks allocated to a single agent, from "compound tasks" which can be decomposed in only one way into a set of simple or compound sub-tasks and allocated to different agents, from "complex tasks", which can be decomposed in multiple ways and allocated in different ways to multiple agents. In our experiments we used a simple task decomposable in only one way in two atomic tasks, but we added complexity by requiring cooperation between agents to execute the haul subtask.

The work presented in this thesis provides several avenues to investigate the role of communication in a partially observable environment. Exploring even larger grid sizes could conclusively determine the limitation of discrete environments and provide insight on the feasibility of training in such an environment using neural networks. Additional experiments with more variations on the information available to the agents can further clarify the role of relevant information when it comes to learning. As an example, during our simulation runs, agents never learned when critical pieces of information were suppressed, such as the location of the goal. We considered to make the goal information, which is critical to learning, available via the communication channel but decided to focus on the location of agents themselves for use in the communication protocol. However, investigating further into other pieces of information that are critical for task accomplishment would help determine how learning is affected by how the information appears to the agent.

The system presented here allows us to set the minimum number of agents for cooperating to move a load and our experiments were limited to two agents. Conducting another series of experiments with a variable number of agents that have to cooperate as a group can further contribute to our understanding of how information sharing between agents affects their ability to learn.

Furthermore, the task allocation was predetermined for all our experiments, that is, loads were assigned to the agents upfront. There is an opportunity to learn task allocation, which would require another level of cooperation, since agents would have to learn to form a team. This would follow along the work of [21] and [14] which suggest the formation of ad-hoc teams with particular attention to the role of communication.

References

[1] Raghavendra Bhat, Yasin Yazıcıoğlu, and Derya Aksaray. Distributed path planning for executing cooperative tasks with time windows. *IFAC-PapersOnLine*, 52(20):187–192, 2019.

[2] R. Crites and A. Barto. Improving elevator performance using reinforcement learning. In *Neural Information Processing Systems (NIPS)*, 1996.

[3] Jakob Foerster, Ioannis Alexandros Assael, Nando de Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 2137–2145, 2016.

[4] Brian P. Gerkey and Maja J. Matarić. A formal analysis and taxonomy of task allocation in multi-robot systems. *International Journal of Robotics Research*, 23(9):939–954, 2004.

[5] Jayesh K Gupta, Maxim Egorov, and Mykel Kochenderfer. Cooperative multi-agent control using deep reinforcement learning. In *Proceedings International Conference on Autonomous Agents and Multiagent Systems*, pages 66–83, 2017.

[6] Natasha Jaques, Angeliki Lazaridou, Edward Hughes, Çaglar Gülçehre, Pedro A. Ortega, DJ Strouse, Joel Z. Leibo, and Nando de Freitas. Intrinsic social motivation via causal influence in multi-agent RL. arXiv:1810.08647v4 [cs.LG], 2018.

[7] Tatsuya Kasai, Hiroshi Tenmoto, and Akimoto Kamiya. Learning of communication codes in multi-agent reinforcement learning problem. In *Proceedings IEEE Conference on Soft Computing in Industrial Applications*, pages 1–6. IEEE, 2008.

[8] G. Ayorkor Korsah, Anthony Stentz, and M. Bernardine Dias. A comprehensive taxonomy for multi-robot task allocation. *International Journal of Robotics Research*, 32(12):1495–1512, 2013.

[9] Nur Geffen Lan, Emmanuel Chemla, and Shane Steinert-Threlkeld. On the spontaneous emergence of discrete and compositional signals. arXiv:2005.00110v1 [cs.CL], 2020.

[10] Steven M. LaValle. *Planning Algorithms*. http://planning.cs.uiuc.edu/, 2006.

[11] Paul Pu Liang, Jeffrey Chen, Ruslan Salakhutdinov, Louis-Philippe Morency, and Satwik Kottur. On emergent communication in competitive multi-agent teams. In *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems*, pages 735–743, 2020.

[12] Sridhar Mahadevan and Jonathan Connell. Automatic programming of behavior-based robots using reinforcement learning. *Artificial Intelligence*, 55(2-3):311–365, 1992.

[13] Jason R Marden, Gürdal Arslan, and Jeff S Shamma. Cooperative control and potential games. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(6):1393–1407, 2009.

[14] Reuth Mirsky, William Macke, Andy Wang, Harel Yedidsion, and Peter Stone. A penny for your thoughts: The value of communication in ad hoc teamwork. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence*, pages 254–260, 2020.

[15] Igor Mordatch and Pieter Abbeel. Emergence of grounded compositional language in multi-agent populations. In *Proceeedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[16] Ernesto Nunes, Marie Manner, Hakim Mitiche, and Maria Gini. A taxonomy for task allocation problems with temporal and ordering constraints. *Robotics and Autonomous Systems*, 90:55–70, April 2017.

[17] Jacob T. Schwartz and Micha Sharir. On the "piano movers" problem I. the case of a two-dimensional rigid polygonal body moving amidst polygonal barriers. *Communications on Pure and Applied Mathematics*, 36(3):345–398, 1983.

[18] Krzysztof Socha and Marco Dorigo. Ant colony optimization for continuous domains. *European Journal of Operational Research*, 185(3):1155–1173, 2008.

[19] Luc Steels and Frederic Kaplan. Aibo's first words: The social learning of language and meaning. *Evolution of Communication*, 4(1):3–32, 2000.

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

[21] Peter Stone, Gal A Kaminka, and Jeffrey S Rosenschein. Leading a best-response teammate in an ad hoc team. In *Agent-mediated Electronic Commerce. Designing Trading Strategies and Mechanisms for Electronic Markets*, pages 132–146. Springer, 2009.

[22] Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 330–337, 1993.

[23] C. J. C. H. Watkins. *Learning from Delayed Rewards*. PhD thesis, Cambridge University, 1989.

[24] A Yasin Yazıcıoğlu, Magnus Egerstedt, and Jeff S Shamma. A game theoretic approach to distributed coverage of graphs by heterogeneous mobile agents. *IFAC Proceedings Volumes*, 46(27):309–315, 2013.