ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ARTIFICIAL INTELLIGENCE

MASTER THESIS

in

Autonomous and Adaptive Systems

DECENTRALISED COORDINATION AND COMMUNICATION IN MULTI-AGENT REINFORCEMENT LEARNING SYSTEMS

CANDIDATE	SUPERVISOR
Giovanni Minelli	Prof. Mirco Musolesi

Academic year 2021-2022

Session 3rd

`self.dedicatedTo(p) for p in Friends+Family`

Contents

List of Figures

Abstract

Coordination of actions plays a crucial role in multi-agent systems, as it allows entities to work in a shared environment, together towards a common goal, or individually without hindering each other's progress. In order for this to occur, agents must demonstrate high levels of spatial awareness and collaborative skills that enable them to understand and acknowledge each other's intentions. Added to this challenge are all those constraints related to real-world implementation, such as decentralisation of information and efficiency requirements that cannot be easily ignored. This thesis aims to contribute to the field of research by studying coordination among agents habilitated to exchange information. Existing challenges and solutions are discussed, then an alternative approach is presented to address the problems. Specifically, the paper argues that explicitly allowing agents to choose whether to coordinate with others or to act independently provides them with adaptability to different scenarios while still ensuring an optimal understanding when needed. To support this claim, CoMix is presented as a novel method that reflects this strategy. Extensive tests with a focus on the scalability of the solution show its positive results in different scenarios, and a comparative analysis highlights the ability of agents to learn strategic behaviour.

Chapter 1

Introduction

Cooperation and competition are commonly studied behaviours in multi-agent environments, and state-of-the-art methods have shown great success in these tasks [65, 82], but, it is important to recognise that most real-world scenarios do not fit neatly into a defined category that includes one or the other. Instead, by focusing on coordination dynamics, it would be possible to develop methods enabling optimal system evolution regardless of individual objectives set by the environment [62]. This will require tackling the difficult challenge of coordinated decision-making, but developments in this direction could lead to significant advances in technology applications, especially in areas involving social skills such as swarm robotics and timely interactions such as autonomous vehicle navigation.

1.1 Multi-agent systems

A Multi-Agent System (MAS) encompasses multiple autonomous entities, referred to as agents, that interact with each other within a shared environment [79]. The goal of each agent is to accomplish a specific task which, depending on the difficulty, may also require cooperative or competitive interactions with others. Due to the complexity of such systems, instead of developing intelligent behaviours from scratch, it would be possible to inject intelligence into the agents, pre-programming responses to interactions, or adopting fixed shared rules which decrease the space of uncertainty. However, it is generally more desirable for agents to possess the ability to adapt and learn over time. One prominent framework for that learning ability is Reinforcement Learning (RL), which entails modifying behaviour through a process of trial and error. Recent advances in Artificial Intelligence and Deep Learning have led to a surge of research interest in Deep Reinforcement Learning (DRL). This approach has demonstrated success in a wide range of fields, including robotics, natural language processing, game playing, and network security. In particular, RL (and MARL, in multi-agent settings) has been utilised to develop intelligent robots that can navigate and manipulate objects in their environment [2]; strategy selections for acting in impractical spaces explorations [12] or optimisation of resources [45]; in classic game playing it has been used to develop agents that can compete at human levels, such as playing chess, Go and poker [64, 66, 7], or in the case of strategic multiplayer online games where cooperation with other agents is required to achieve a common goal, such as DOTA 2 [49] and StarCraft II [76].

Despite the impressive results that can be achieved with DRL, its application to a multi-agent setting poses unique challenges: the concurrent and heterogeneous behaviour of the agents leads to an unpredictable environment, phenomenon referred to as non-stationarity [63, 8, 21]; the exponential explosion of states leads to the curse of dimensionality [63, 8], making it difficult to assign credit to specific agents for a given outcome [80, 1]; large action spaces coupled with the need for global exploration [44], which increase the complexity of the learning process; and potential for relative overgeneralisation [17, 78, 51]. For this reason, a successful single-agent RL methodology cannot be simply applied to a multi-agent setting without re-evaluating its learning approach.

1.2 Coordination

Intelligent coordination refers to the coordinated effort of a group of agents, capable of making intelligent decisions, in acting not only on the basis of their own goals, but also taking into account other entities in the environment. It involves establishing a shared understanding of the task at hand and developing a plan that outlines the roles and responsibilities of each agent. This concept is a promising area of research in MARL as it addresses the challenges introduced by the presence of multiple entities, enabling more effective action selection and limiting inefficient behaviours.

In general, we seek coordination ability both in cooperative and competitive settings.

- *Cooperation* refers to the act of working together towards a common goal. It involves the mutual support and assistance of multiple agents, each of whom contributes with their own unique skills and abilities to the collective effort.
- *Competition* refers to the ability of agents to compete with each other to achieve their goals. It typically involves agents taking actions that maximise their own reward or utility, potentially at the expense of other agents.

Cooperative contexts are undoubtedly the most researched [9] as by modelling an environment for this purpose, it is possible to arrive at the achievement of greater goals beyond the capabilities of the single agent and the emergence of group intelligence. This is the foundation of many applications in robotics, swarm intelligence, and social studies. However, competition is also a vital aspect worth considering, as it can motivate agents to improve their performance and explore the environment more efficiently. Furthermore, competition can prevent agents from becoming too dependent on one another and encourage them to develop more sophisticated strategies. Despite this common distinction, it should be noted that usually, we do not have a clear cut between cooperative and competitive behaviour in most real-world scenarios. In some cases, cooperative agent may temporarily act selfishly while trying to achieve a common goal, and a competitive agent may temporarily form a coalition with its opponent to achieve its own goal [8, 23]. Therefore, when designing a coordination system, it is important to avoid injecting a fixed criterion for collaboration or obstruction.

1.3 Information sharing

Intelligent agents – whether humans or artificial – can greatly benefit from the ability of information exchange to coordinate, strategise, and combine reciprocally their sensory experiences to act in the environment. Indeed, it is usually assumed that agents placed in the real world have to operate in situations of partial observability, limited in their knowledge and perception of their surroundings, and it would be unrealistic to assume otherwise.

By enabling communication, we can aid the agents in gathering information about the environment and improve their decision-making process by sharing observations, action policies, future intentions, or other relevant information. Furthermore, communication enables agents to form strong relationships and work together in groups, leading to improved behaviours and increased efficiency in task completion. This is achieved through the parallelisation of activities and optimisation of resources. We can see examples of this ability in a wide range of RL applications, like multi-player gameplay in environments simulated (e.g., DoTA, StarCraft) or physical (e.g., robot soccer), and in real applications like self-driving car networks working together for safe and efficient transportation or teams of robots deployed in hostile and rapidly-evolving environments, as well as many others.

1.4 Centralised decision

When trying to achieve coordinated behaviour, a straightforward approach is to adopt a hierarchical structure, where one agent takes on the role of coordinator to establish order and take effective decisions for the system evolution. However, the question arises of who should be in charge of this role and who better understands the situation. Approaches based on centralised coordination delegate the responsibility of coordinating the agents to a single agreed-upon entity, but even if a reliable hierarchical mechanism is in place, the question remains: How can the higher-level agent acquire the necessary information and successfully coordinate all other agents? How can this method address scalability and generalisation to different situations?

An alternative and often preferred approach is a decentralised one, where there are no agents with higher roles controlling the behaviour of others. This method eliminates the difficulties associated with centralisation, such as global coordination and scalability issues. On the other hand, agents in a decentralised system have limited knowledge and must rely solely on their local information, which increases uncertainty and variability of action and makes it difficult to predict the overall behaviour of the group. As a result, this leads to suboptimal policies and ineffective interactions during testing, especially when complex coordination is required. This particular problem has been widely studied under the name of Interactive Consistency, or in a more restricted setting as Byzantine Generals, where the agents have to take a group action with shared consensus to succeed.

1.5 From simulation to reality

The process of transition from a simulated environment to the real world is often advocated in the field of RL research. However, the lack of interpretability and explainability associated with black box algorithms, such as deep neural networks, presents a significant challenge in this transition. The inability to effectively transfer the learned behaviours of an agent outside of controlled environments due to a lack of transparency and interpretability can limit the applicability of RL research to practical scenarios. To address this difficulty, there has been a trend towards using more realistic simulation environments and combining RL and robotic research efforts. Thanks also to independent organisations [74] and open source communities [72], many simulated environments are available to test physics interactions, structural properties, and complex behaviour at scale.

It is also necessary to mention the implementation of the communication process, which despite playing a key role in the deployment is often taken for granted. There is a vast research activity in the telecommunication field and under the Internet of Things (IoT) umbrella, that tries to design infrastructure and cope with interconnected entities limited in their computation and decision ability. However, this thesis will not cover the physical aspects that agents deployed in real scenarios are usually subject to, such as unreachability or disrupted communications. The analysis and reasoning will assume an agent-to-agent (A2A) communication, following the traditional device-to-device (D2D) paradigm, where physically close devices (e.g., two agents), can communicate directly over a so-called sideline. Compared to regular centralised uplink-downlink communication, D2D communications benefit from a shorter link distance and fewer hops, which is better in terms of reliability. Moreover, since communication is direct, i.e., without intermediate nodes, D2D has the potential to provide lower latency in the transmission of information.

These topics represent central points of the research field, and developments in their directions are crucial for the extensive use of AI in the wild. The central focus of this thesis work is the development of new architectures and strategies for autonomous agents operating in shared environments.

Specifically, the research aims to investigate the ability of agents to learn autonomously, to act, and purposely coordinate when necessary. The proposed architecture and the deriving analysis described in the following chapters aim to answer the following research questions:

- Can agents acting in a shared environment with the ability to communicate, learn autonomously when coordination is necessary and preferable to selfish behaviour?
- Is it possible to design an action strategy for independent agents that uses simple communication to achieve effective group coordination?

The intent, therefore, is to demonstrate, under different constraints and needs, whether a consensus in behaviour can be reached through the individual striving for a better reward without forcing coordination.

Chapter 2

Background

This chapter provides a comprehensive introduction to Reinforcement learning and Multi-Agent Reinforcement Learning. Beginning with a naturalistic explanation of the mechanisms involved, it proceeds by discussing the taxonomy of problems and tractability properties. The focus then shifts to the area of communication and coordination mechanisms, which constitutes the core of this dissertation.

2.1 Origins and definition of RL

The field of RL has its origins in the study of animals, specifically in the psychological literature [57] and animal experimentation [61], which have shown that animals can learn to perform complex tasks through trial-and-error, with the help of rewards and punishments. Later, the concept of RL was formalised in the field of artificial intelligence, defining it as a type of learning in which an agent learns to perform actions that maximise a scalar reward signal. The agent's goal is to learn a policy, which is a mapping from states of the environment to actions, that maximises the expected cumulative reward over time [57].

The standard mathematical framework for modelling sequential decision-making problems is the Markov decision process (MDP), which is defined as a tuple $M = (S, A, P, r, \gamma)$, where:

- S is the state-space, a finite set of world states, represented as $S = 1, 2, \ldots, |S|$.
- A is the action-space, a finite set of actions, represented as $A = 1, 2, \ldots, |A|$.
- $P(s_{t+1}|s_t, a)$ is the state transition probability function that expresses the probability of transitioning from state s_t to state s_{t+1} by selecting action a.
- $r = R(s_t, a, s_{t+1})$ is the reward obtained from the reward function, given the transition from state s_t to state s_{t+1} by taking action a.
- $\gamma \in [0, 1]$ is the discount factor, which is used to handle both finite and infinite-horizon problems.

The goal of an RL agent in a MDP is to find a deterministic, optimal policy $\pi^* : S \rightarrow A$, which will dictate how the agent should act in order to maximise its rewards. Mathematically, the optimal policy can be defined as,

$$\pi^* = \arg\max_{\pi \in \theta} \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(s_t), s_{t+1}) \mid s_0 = s] \tag{2.1}$$

where θ is the set of all admissible deterministic policies, and $(s_0, a_0, s_1, a_1, \ldots)$ is a state-action trajectory generated by the Markov chain under policy π. The optimal policy is the one that maximises the expected cumulative reward over an infinite horizon.

Alternatively to directly search for the optimal policy, is possible to define two utility functions that capture the concept of expected return. These are the value function and the state-action value function, also known as the Q-function. The value function for a given policy π is defined as: $V^\pi(s) = E[\sum_{t=0}^{\infty} \gamma^t r(s_t, \pi(s_t))|s_0 = s]$. It encodes the expected cumulative reward when starting in state s and following the policy π thereafter.

The state-action value function, or Q-function, is defined as:

$Q^{\pi}(s,a) = E[\sum_{t=0}^{\infty} \gamma^t r(s_t, \pi(s_t)) | s_0 = s, a_0 = a]$. It measures the expected cumulative reward when starting from state s, taking action a, and then following the policy π. In the context of DRL, the policy, the value functions, or both are typically represented by neural networks.

2.2 MARL

In Multi-Agent Reinforcement Learning, we extend the single-agent case by introducing a different formulation known as a Markov Game. This is a generalisation of Markov Decision Processes and allows for modelling more complex decision-making scenarios where agents need to make strategic decisions based on the actions of other agents. In a Markov Game, each agent acts according to its own policy, which may differ from one another, and influences the rewards and future states of others. This is formalised by the tuple $(N, S, A_i, P, R_i, \gamma)$, where:

- $N = \{1, \ldots, n\}$ denotes the set of $n > 1$ interacting agents
- S is the set of states observed by all agents
- $A = A_1 \times \cdots \times A_n$ joint action space is the collection of individual action spaces from agents $i \in N$
- $P : S \times A \rightarrow P(S)$ is the transition probability function and describes the chance of a state transition
- R_i is the reward function defined as $R_i : S \times A \times S \rightarrow \mathcal{R}$ associated to each agent $i \in N$
- $\gamma \in [0, 1]$ is the discount factor.

At stage t, each agent $i \in N$ selects and executes an action based on the individual policy $\pi_i : S \rightarrow P(A_i)$. The system evolves from state $\mathbf{s} =$

$\{s_1, \ldots, s_n\}$ under the joint action $\mathbf{a} = \{a_1, \ldots, a_n\}$ with respect to the transition probability function P to the next state $\mathbf{s}'$, while each agent receives $\mathbf{r}$ as immediate feedback to the state transition. The goal of each agent, similar to a single-agent problem, is to modify its policy in order to maximise its long-term reward [57].

2.3 Taxonomy of the problem

The study of MAS often involves categorising situations using standard taxonomies to understand the system's characteristics clearly, compare it with other multi-agent systems, and identify specific challenges and opportunities. For example, we can distinguish between settings where the rewards obtained are shared or individually assigned. The relative taxonomy classifies them as:

- **Fully cooperative** setting, in which all agents receive the same reward for state transitions, i.e. $R = R_i = \cdots = R_N$. Agents are motivated to collaborate in order to maximise the performance of the team.

- **Fully competitive** setting, where the problem is described as a zero-sum Markov Game. In this setting, the sum of rewards equals zero for any state transition, i.e. $R = \sum_{i=1}^{N} R_i(s, a, s') = 0$. Agents are motivated to maximise their own individual reward while minimising the reward of others.

- **Mixed** setting, also known as a general-sum game, the setting is neither fully cooperative nor fully competitive, and therefore does not impose any restrictions on the goals of the agents.

Another commonly used taxonomy regards the learning and execution process, where we have a distinction based on the information available to agents:

- **Decentralised** settings [6, 29], are characterised by the presence of independent learners who are unaware of the existence of other agents and are unable to observe their rewards or actions. A lack of global observability and coordination among agents usually marks this type of setting.

- **Centralised** settings [25, 37], feature joint-action learners capable of observing the actions taken by all other agents a-posteriori. Usually, it is adopted to introduce coordination among agents and some level of global observability.

The rewards and information available to agents in MAS can significantly impact the complexity of the problem. In cases where all agents receive a common reward and have complete knowledge of the environment (fully observable and fully cooperative), the problem can be reduced to a single-agent problem, allowing for the identification of exact optimal policies without coordination among agents [21]. However, these assumptions are often not met in reality, and even if they were, it would be beneficial to factorise the joint stochastic policy into $\pi(a \mid s) = \forall i \; \pi_i(a_i \mid s)$ to avoid the exponential growth of the action space A with the number of agents, which can make greedy action selection, exploration, and learning intractable. On the other hand, this inevitably leads to the problem of partial observability, where agents must act and learn with limited knowledge of the state of the world.

Another problem with independent agents is that it is equivalent to having N learning algorithms running in a shared environment that is constantly changing due to unpredictable rules, as each agent is simultaneously learning an action policy of its own. This effect of non-stationarity of the environment is referred to as the moving target problem and can be formulated as $P(s' \mid s, a, \pi_1, \ldots, \pi_N) \neq P(s' \mid s, a, \bar{\pi_1}, \ldots, \bar{\pi_N})$, which is the change in transition probability function as a result of the co-evolution of all agents' policies.

As the focus of this thesis is on decentralised control settings under the assumption of partial observability, it is important to emphasise the difficulty of solving the decision problem for Decentralised Partially Observable Markov Decision Processes (Dec-POMDPs). In fact, computing even an approximately optimal policy for Dec-POMDPs is NEXP-complete [3, 56]. Despite some recent empirical successes [33, 15], finding an exact solution to Dec-POMDPs using RL methods with theoretical guarantees is still an open research question. Nonetheless, by introducing the relaxation of free communication between agents, we can expand the knowledge of the agents and move the problem into P-SPACE [4], without introducing unrealistic abstractions that can only be achieved in simulations.

2.4 Coordination and agreement

In order to achieve coordination among decentralised, independent agents, some form of communication and agreement on actions must be established. This can be thought of as a distributed optimisation problem, where consensus in policy development (the development of an optimal policy attainable in multi-agent contexts) is achieved through local computation and communication with neighbouring agents.

In standard consensus algorithms, we have a set of agents $A = \{a_i | i \in 1, 2, \dots, N\}$, each initialised with some initial state $\in S$. To enable communication, we can imagine the agents being interconnected over a reliable communication network, ideally represented as an oriented graph. To reach consensus, every agent communicates with others by exchanging values, locally processes the information, and then proposes a single value v, drawn from the set $V = \{v_i | i \in 1, 2, \dots, M\}$. The agents are said to reach a consensus if, from a certain time step t, it holds, $lim_{t \to \infty} v_1^t = v_2^t = \cdots = v_N^t$, for every set of initial states $\in S$.

A consensus algorithm is considered to be formally correct [10] if it satisfies the following three conditions in every execution:

- **Termination**: eventually, all `correct` processes set their decision value.
- **Agreement**: the decision value of all `correct` processes is the same.
- **Integrity**: if all `correct` processes propose the same value, then any `correct` process in the `decided` state must choose that value.

By keeping these conditions in mind but relaxing some strict constraints like the convergence to a single state (value) of consensus, this research will focus on the design of a method of agreement among agents to optimise the local policy for individual needs but in accordance with the group's intentions.

2.5 Common approaches

MARL encompasses all those methods used to train multiple agents in learning to interact in a shared environment. They are therefore designed to handle the added complexity of having multiple interacting entities, capable of handling large environments and most often, designed with scalability in mind. [18, 22, 69]. One common technique able to deal with these challenges is imitation learning [26], in which agents learn to imitate the actions of an expert demonstrator by using a set of collected trajectories. For instance, [50, 43] uses this approach to transfer the driving ability of human experts to an agent that can control a physical car, reducing the need for extensive trial and error exploration of the enormous state space that characterises autonomous driving. However, this approach requires a significant amount of expert trajectories, which, being specific to the environment and task at hand, are hardly reusable for other tasks, as well as having the potential for overfitting during training.

Another relevant method is hierarchical reinforcement learning (HRL) [27].

In HRL, the learning process is divided into multiple levels or layers of abstraction, with each one being used to represent the space or the goal with a different degree of granularity. For example, an RL agent learning to play a video game might have a lower level that focuses on actions such as moving and jumping, and a higher level that rewards strategies such as exploring the environment or attacking enemies. By breaking the learning process into multiple levels, HRL can simplify the problem and make it more manageable at scale. Indeed, it is currently considered a state-of-the-art technique in the field of robotic control problems [46]. The downside is that HRL can be difficult to implement, as it requires careful design of the hierarchical structure and abstraction at each level, with the associated risk of leading to sub-optimal solutions during training or lack of generalisation.

On the other hand, we find extensions of single-agent algorithms or fully decentralised approaches that focus on learning directly from the interactions of multiple agents in the environment, allowing them to handle non-stationary and evolving environments more effectively.

2.5.1 Centralised and decentralised approaches

When dealing with simple MARL applications, we can adopt a centralised approach where the environment is viewed as a whole and the interactions between agents are observed from a global perspective. While this simplifies interactions and makes policy computation easier, it would not be suitable for the scalability of the system, a requirement that has recently attracted much attention in the development of new methods [18, 22, 69]. Additionally, the assumptions of centralisation may be difficult to attain in practice, as the presence of a central entity in the system may not be feasible.

An alternative approach is decentralised control, where each agent makes its own decisions independently, without the need for a central controller or global coordination. The independent learning framework can obtain good empirical performance in several benchmarks [52], but there are few theoretical guarantees for decentralised learning optimality, and the interpretability is often insufficient. Recent work has focused on a hybrid approach [40, 16, 71, 58], where global information is required only during the training phase, freeing the algorithm from the need to continuously know the behaviour of other agents during testing. Centralised Training with Decentralised Execution (CTDE) is one such approach, which has been expanded into two main lines of research that align with standard MARL frameworks. Multi-Agent Deep Deterministic Policy Gradient (MADDPG) [40], is an example of actor-critic model which uses a centralised per-agent critic to estimate the Q-function and decentralised actors to optimise the agents' policies. There is no explicit communication in this approach, as the other agents' actions are inferred from their respective policies. Another similar approach is Counterfactual Multi-Agent (COMA) [16], which addresses the challenges of multi-agent credit assignment by using a counterfactual baseline that marginalises out a single agent's action while keeping the other agents' actions fixed. However, these actor-critic models require on-policy learning, which can be sample-inefficient, especially when the state space is large.

An alternative CTDE approach is to learn a centralised Q-function [71, 58, 68, 81, 77], in which the optimality is reached by considering the relationship between joint action value and optimal local actions. For example, Value Decomposition Network (VDN) [71] learn the joint-action Q-values by factoring them as the sum of each agent's Q-value, and QMIX [58] extends VDN to allow the joint action Q-function to be a monotonic combination of each agent's Q-value that can vary depending on the global state. Despite achieving excellent results, QMIX has faced criticism for its limited representation capacity due to the monotonic constraint, and several alternatives have been developed to address this limitation. Between the most importants, we find QTRAN [68], which learns an unrestricted joint action-value function and aims to solve a constrained optimisation problem in order to decentralise it, and QPLEX [77], which takes advantage of the dueling network architecture to factor the joint Q-function in a manner that does not restrict the representational capacity. In this thesis, I will use the advances of the centralised Q-function approach to train agents in a non-stationary environment characterised by partial observability.

2.5.2 Communication channel

Mechanisms for information sharing and communication are introduced to reduce non-stationarity effects [20, 59]. Communication between agents can take the form of explicit communication using talk channels, or implicit communication by observing other agents' actions or their effect on the environment. In the former case, one option is to rely on standardised message formats, such as the Agent Communication Language (ACL), to enable independent agents to communicate with a precisely defined syntax, semantics and pragmatics [55, 14]. On the other hand, ad-hoc communication protocols with learnt languages are mostly adopted when complex coordination is required. Even though the first approach may result in a lack of generality and flexibility due to the imposed form, having a standardised and well-defined structure is helpful when the goal necessitates common understanding. Some research tries to find a balance between standardisation and flexible communication taking the best of both approaches [41, 31].

Efficiency, as well, is a common driver in designing effective communication, as real-world environments involve other factors such as security overhead, message brokering time and dynamism of the whole environment. There are differing opinions on the best format to adopt for messages (structured, discrete, continuous, etc.) [40, 31, 34, 70], and the optimal method for exchanging information in terms of costs and benefits. Some methods use a common memory buffer where agents can write and read to share information [54], or adopt an event-based framework where communication occurs only under certain circumstances [24], or choose to integrate implicit communication mechanisms by observing the actions of others and inferring their policies to reduce the communication needs [13].

Communication can take place in one or both directions: direct messages are shared between two agents by opening a communication channel [47, 83], or by broadcasting it to everyone [30, 36, 70, 11]. The latter approach is more expressive under partial observability assumptions, but it is also more expensive in terms of transmission traffic and can lead to situations where communications are dominated by useless transmissions. Some approaches aim to create smaller groups of agents that focus on inter- and intra-communications to limit irrelevant reasoning and improve coordination performance [38, 47, 39, 47, 28]. Others reduce the number of messages sent by learning to understand when communication is really necessary or when the information held is redundant and communication avoidable [39, 13]. Some solutions aim at targeting communication only to those who are interested in it [83], while others act on the side of the listener by adopting different mechanisms of attention to filter out irrelevant messages [34, 30, 70, 11]. While these methods of obscuration or filtering are often effective, they typically do not take into consideration the messages in the context of the whole communication channel, but filter the individual message only for its relevance to the agent, ignoring possibilities of more complex coordination.

Lastly, the adoption of cooperative assumptions makes it much easier to design solutions in this research area [16, 34, 38, 42, 13, 70], in contrast to mixed environments [73, 5] where the lack of fixed directions to succeed can make the task more challenging. This work, in particular, does not impose a fixed structure on communication, while maintaining a focus on efficiency, and does not limit its applicability only to cooperative frameworks.

2.5.3 Message content

The message content is a crucial aspect for the recipient agent. It should be designed to provide additional information about the communicator's perspective, reducing uncertainty about his behaviour and facilitating coordination. One option is to use a highly expressive message encapsulating the agent's reasoning process. For instance, some works [70, 67, 11, 53] have structured their architectures around recurrent modules, using the hidden state as a signal message for others. However, this is typically used internally to encode past and current information and reusing the individual reasoning vector for communication intent can have limiting effects in difficult tasks. Other works [70, 67] merge multiple incoming messages into a single communication vector using weighted operations, which may not result in strong coordination when many agents participate in the communication.

[11, 28] use crafted messages to transmit elaborate information and train successful agents able to understand each other. They also use weight sharing between agents [30, 11, 53, 28], which is quite common to reach a better action understanding and counteract scalability issues. Nevertheless, this implies that agents will have the same reasoning ability on the information gathered, and that policies will be learned to be effective for the average agent, but not for each individual.

More straightforward transmission approaches [39, 30] use current or time-delayed observation as a communication message, usually with additional information expressing intentions. This implementation delegates most of the interpretation and coordination to the receiver but allows for more flexible interaction dynamics. In combination with the use of a module to learn a better representation [36], this method can also be used without the burden of raw information exchange.

Finally, in some cases it is not even considered the presence of an explicit message and, assuming full mutual knowledge of each other, the effort will only be focused on learning a policy expressing coordination. Previous works [60, 48] included the use of a mechanism where the consensus is mandatory in order to proceed, while current methods implemented with agents able to ”learn”, prefer a looser agreement, aiming instead for convergence in the choice of target and coordination in actions. For example, [35] forms groups of agents with similar objectives to have tighter cooperation and variety of strategies between teams, but the proposed architecture is not end-to-end differentiable. The main point of this work is that coordination should not be seen as a global requirement, but rather, only necessary in certain situations, as extensive reasoning about the beliefs and intentions of others can slow down convergence towards a good individual policy and even be harmful.

Chapter 3

Approach

In this section, the proposed solution for addressing the coordination problem in multi-agent systems will be presented. Based on established algorithms and principles, this work attempts to make a significant contribution in terms of a method for implementing reasoning processes in RL agents. The claims are supported in the next chapter by in-depth tests in two different environments, with a focus on scalability performance, and by conducting an ablation study to assess the impact of the novelties introduced and strategies adopted.

3.1 Description

The traditional approach to address the problem of coordination in MARL systems is to prioritise group dynamics over other individual considerations. While this approach may be appropriate in scenarios where cooperation is the only way to succeed, it can prove detrimental in scenarios where agents are permitted to exhibit different behaviours to achieve their own individual goals. An intuitive example of this is a car driver who wants to go from a starting point to their destination. If the driver were to constantly consider the intentions and actions of other drivers at each ”step” of the trip, e.g. evaluating the possibility of allowing another driver to go first or taking the lead themselves, the trip would become endless. This reasoning is accentuated in large and sparsely-rewarded environments, where it is harder to extract rules and the exponential possibilities of interactions with others can lead to high uncertainty of action and slow down the convergence to a good policy or suboptimal convergence.

Another criticism of other common approaches is the reliance on complex communication channels for establishing coordination. The primary goal of communication should be to reduce uncertainty and non-stationarity effects by providing additional information about the actions and intentions of other agents. However, many methods rely on the exchange of complex vectors that encapsulate agents' history or personal thoughts, which can be cryptic for thirds. This can make it difficult for agents to fully understand the true intentions of the speaker in a general context. Additionally, complex environments are more challenging in terms of state evaluation, thus it is important to keep the size of the state space small by providing agents with only relevant information.

As a final thought, the proposed work does not claim a declarative and imperative agreement between agents since coordination can arise from continued interactions. In fact, since all agents are subject to the same environmental rules, they will eventually avoid making decisions that prove to be self-damaging and will occasionally cooperate to maximise their respective reward signals.

On top of that, I propose a policy that can adopt both egoistic and altruistic behaviours, a reasoning process that takes into account information from other agents, and a simple yet effective communication channel for exchanging information. The architecture of coordination and policy modules are implemented with recurrent neural networks to maintain consistency in decisions over time and are trained using a centralised training decentralised execution paradigm. Details of each module are discussed in the following sections, and the overall architecture is depicted in figure 3.1.

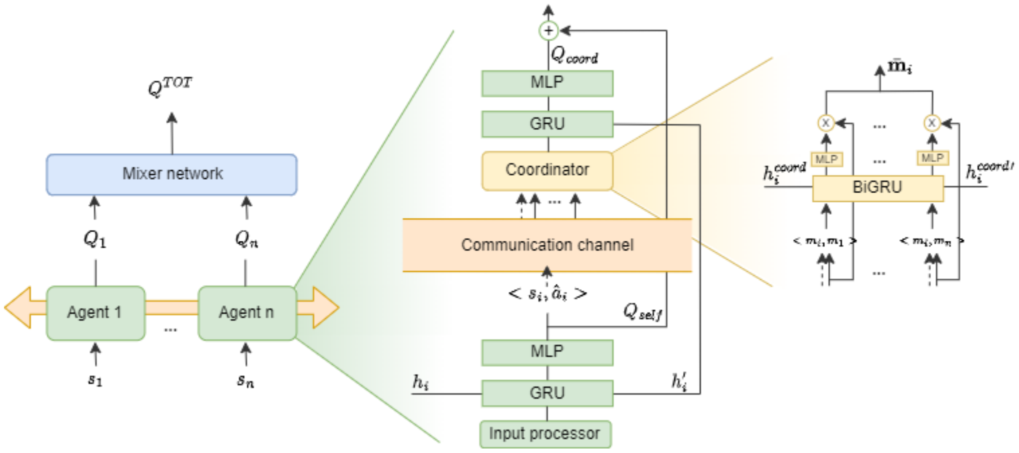


Figure 3.1: **CoMix architecture overview.** The figure depicts the CoMix architecture, with each module expanded into its components. (left) Individual agents are trained under a CTDE framework, with a mixer network used only during training that takes partial observations and state-action values to evaluate the system's performance. (middle) At each step, the agents partially observe the environment and process the information to produce an independent choice of action, Q_{self}, which is then communicated through a communication channel. Every agent then considers all received messages to decide how to modify its previous selfish decision, producing the additional term Q_{coord}, which incorporates additional information filtered by the Coordinator. (right) The Coordinator computes a boolean mask to filter the communication channel, taking into account each neighbouring agent's communicated intention in relation to personal objectives. All decision-making components are implemented with recurrent modules to maintain consistency over time and enable agents to develop more structured strategies.

3.1.1 Action policy

The Q-network is responsible for predicting the state-action value for each agent based on the information at his disposal. We can define it with the following formula: $Q_i(s_i, a_i, h_i)$ where s_i, a_i, and h_i are the observation, action and hidden state of agent i respectively. This would be considered sufficient for an implementation of an independent learner, however, in this setting, we incorporate communication messages as additional information that can aid the decision-making process when available. To achieve this, we reformulate the previous definition as the sum of two terms: $Q_i = Q_{self} + Q_{coord}$. Here, Q_{self} represent the selfish action intention, which, once computed, produces state-action values for the current state and updates the hidden state for the next state $h_i \rightarrow h_i'$. The second term Q_{coord} modifies the first on the base of the current updated hidden state and incoming messages filtered out by a coordination module. Furthermore, the introduction of a feature extractor is made in the proposed solution. When executed on raw observation data, it extracts meaningful information, enabling the policy network to reason within a more defined space. It is worth noting that the input processing module used has shared weights among all agents. As highlighted by [36], this is an important implementation detail to allow all agents to reason about data from the same distribution.

The final formula representing the policy network involved is the following (superscript to indicate the timestep is omitted for brevity):

$$Q_i(s_i, a_i, h_i, \bar{\mathbf{m}}_i) = Q_{self}(s_i, a_i, h_i) + Q_{coord}(a_i, h_i', \bar{\mathbf{m}}_i) \tag{3.1}$$

The overall network is implemented with two distinct GRU modules and related linear layers to extract the value corresponding to each action, sharing only the sequential vector of the hidden state. The recurrence of the first is necessary to process the new observation in relation to the past, while the second aims to properly mix the self-interests with the new information obtained from other agents.

3.1.2 Coordinator

The coordination module is responsible for determining the relevance of other agents' communications in relation to the agent's intentions by producing a coordination mask used to filter out incoming messages. A message is defined as the communicated intention of an agent to take a certain action in order to achieve a personal objective, $\hat{a}_i = \arg\max_a Q_{self}(s_i, a, h_i)$, and is represented by the tuple $m_i =< s_i, \hat{a}_i >$. Consequently, we define $\mathbf{m} = \{m_1, \ldots, m_n\}$, as the set of incoming messages sent by n agents at a certain timestep, and $\bar{\mathbf{m}}_i$, as the filtered set for agent i obtained with the application of the communication mask $\mathbf{c}_i$, result of the Coordinator execution:

$$\mathbf{z}_i = \{< m_i, m_1 >, \ldots, < m_i, m_n >\}^{-<m_i,m_i>} \tag{3.2}$$

$$\mathbf{c}_i = Coord(\mathbf{z}_i) \tag{3.3}$$

where $\mathbf{c}_i = \{c_{i,1}, \ldots, c_{i,n}\}^{-c_{i,i}}$ and $c_{i,j} = \begin{cases} 1 & \text{if agent } i \text{ coordinate with agent } j, \\ 0 & \text{otherwise} \end{cases}$

$$\bar{\mathbf{m}}_i = \mathbf{m} \odot \mathbf{c}_i \tag{3.4}$$

As previously proposed in [28], the module for this reasoning is implemented using a BiGRU layer, in order to take into account the intentions of all other agents under the same circumstances, but also in a reciprocal relation. The individual scores produced by this layer are then used to calculate an independent probability of communication through a two-way softmax.

3.1.3 Centralised network

Learning communication and action policy at the same time, in a setting of partial observability, may lead agents to inaccurately estimate their local Q-function. The adoption of a CTDE learning algorithm prevents this, providing current observations **s** and state-action values **q** of the agents to a centralised network to learn a joint action-value function. The specific implementation used is the one proposed by [58] (QMIX), which decomposes the joint function into factors depending only on individual agents, enabling it to cope with large joint action spaces. Therefore, when defining $Q^{TOT}(\mathbf{s}, \mathbf{q})$, we have to respect the following two properties:

- Q^{TOT} yields the same result as a set of individual argmax operations performed on each Q_i:

$$\arg\max_{\mathbf{a}} Q^{TOT}(\mathbf{s}, [\mathbf{a}, \mathbf{h}, \bar{\mathbf{m}}]) = \left\{ \begin{array}{c} \arg\max_{a_1} Q_1(s_1, a_1, h_1, \bar{\mathbf{m}}_1) \\ \vdots \\ \arg\max_{a_n} Q_n(s_n, a_n, h_n, \bar{\mathbf{m}}_n) \end{array} \right\} \tag{3.5}$$

- The relationship between Q^{TOT} and each Q_i is constrained to be monotonic:

$$\frac{\partial Q^{TOT}}{\partial Q_i} \geq 0, \forall i \in [i, n] \tag{3.6}$$

Despite its limitations in terms of representational ability due to the monotonicity constraint, which limits QMIX to suboptimal value approximations, this algorithm has long been considered a state-of-the-art approach in the field. Nonetheless, the idea at the basis of the method proposed is not bound to the use of QMIX and can be adapted to different CTDE algorithms.

3.2 Training

3.2.1 Loss functions

The agents' Q policy network is trained end-to-end with the error propagation being able to flow between the two reasoning processes by means of the hidden state shared. The rule of update for their parameters θ^Q follows the common implementation of QMIX [58], with the computation of a temporal difference error:

$$L_Q(\theta^Q) = |y^{TOT} - Q^{TOT}(\mathbf{s}, \mathbf{q}; \theta^Q)|, \tag{3.7}$$

where $y^{TOT} = \mathbf{r} + \gamma \max_{\mathbf{q}'} Q^{TOT}(\mathbf{s}', \mathbf{q}'; \theta^{Q\prime})$ and θ^Q, $\theta^{Q\prime}$ are respectively the parameters of the online policy network and target policy network, periodically copied from θ^Q as in standard DDQN [75]. Note that compared to an individual network per agent, having a centralised function leads to considerably lower variance in policy gradient estimates since it takes into account actions from all agents. At test time, this is not needed, and policy execution is fully decentralised.

In designing an efficient learning schema for the coordination module, different options were evaluated following the same general intuition. Since the probability of communication produced by the single agent for each other sees its effects in a modified set of values for the state-action pairs, we can effectively measure an improvement or decrease in action performance by considering an alternative coordination mask and assuming an optimal estimate of the state-action value from the policy. Therefore, we can define the formula of update of the Coordinator's parameters as a clipped delta between the maximum state-action value obtained with filtered messages $\bar{\mathbf{m}}_i$ from the predicted mask of coordination with respect to the alternative estimated value obtained from messages filtered with inverted probabilities of communication. That is $\tilde{\mathbf{c}}_i = (1 - Coord(\mathbf{z}_i; \theta^C))$ and $\tilde{\mathbf{m}}_i = \mathbf{m} \odot \tilde{\mathbf{c}}_i$ from Eq. 3.3 and Eq. 3.4.

$$
\begin{aligned}
L_C(\theta^C) &= \sum_{i=1}^{n} w_i \Delta Q_i \\
&= \sum_{i=1}^{n} w_i \max(0, \max_{a_i} Q_i(s_i, h_i, \tilde{\mathbf{m}}_i, a_i) - \max_{a_i} Q_i(s_i, h_i, \bar{\mathbf{m}}_i, a_i))
\end{aligned} \tag{3.8}
$$

Following this reasoning, we can further improve the method at the cost of additional computation by performing $n - 1$ inferences of the Q-function, each considering a set of different messages. Instead of computing the Q value considering communicating with the opposite set of agents – with respect to the predicted one – we can compute an averaged expectation of the state action value, taking into account the advantage of reversing the communication probability per each single agent. Although this training implementation is much more expensive, it allows for improved results, especially when many agents are present. The choice is discussed further in the section dedicated to the ablation study (See Sec. 4.4).

The adoption of the QMIX approach gives us an additional element that can be used to better estimate the gain in using a mask of coordination in place of another. Being the mixer network a mapping from states to a set of weights in the hidden space used independently in linear combination with the state-action value of the respective agent, we can reuse these parameters to weight each ΔQ_i. In these terms, we can make updates more precisely related to the gain given by the communication with a certain agent in a certain state. We find mention of this in the above formula with the term w_i.

3.2.2 Training details

Randomness was incorporated in the training of the model to capture the uncertainty of the predictions and to further explore the action space. On the Coordinator's side, the coordination mask selection process was implemented by applying the Gumbel Softmax to the two decision logits, which differs from the Softmax function by adding random noise from the Gumbel distribution to the output. For exploration in action selection, an epsilon decay technique was used, with a decreasing value from 0.9 to 0.05 in 60% of the training time. To optimise the Q-networks and Coordinators, ADAM was utilised with learning rates of $7e^{-4}$ and $8e^{-5}$, respectively. To mitigate the risk of catastrophic forgetting and overfitting, a weight decay value of $1e^{-4}$ was used as a regularisation term, and a soft update technique was adopted for the target network weights. The network sizes, learning rates and decay parameters were carefully chosen for each environment in order to optimise the algorithm's performance and achieve higher results. The learning algorithm also proved to be very sensitive to the number of recurrent steps considered during training, the value of which was chosen according to the dynamics of the task in each environment.

Chapter 4

Experiments

The aim of this chapter is to evaluate the proposed approach and its effectiveness in addressing the research questions stated earlier. The evaluation process will include a comparative analysis with other relevant methodologies in the field, based on results obtained through a comprehensive and systematic evaluation process. The results will be examined in detail, along with a study of the components of the proposed architecture and implementation details, in order to identify overall strengths and limitations. In summary, this evaluation aims to shed light on the performance of the proposed methodology and enable the reader to draw informed and insightful conclusions on its effectiveness in practical applications.

4.1 Baselines of comparison

Individualised Controlled Continuous Communication Model (IC3Net)[67] is proposed as an extension of a previous method [70], introducing a gating mechanism on each other agent's communication channel. They use the current observation encoded, both as internal thought and communication message, and a mask of communication to filter out incoming messages computed at the previous step. The reasoning process then occurs by processing the individual thoughts in an LSTM module and averaging the remaining incoming messages. The result is used as input to two output heads to obtain the actions' probabilities and the individualised probability of communication.

ATOC [28] is an attentional communication model proposed to learn effective and efficient communication at scale by adopting weight sharing between each agent's network. In their proposal, the communication message is represented by the hidden state of a recurrent module that processes the observation at each step, thus sharing a vector which resembles the agent's history. Based on the internal reasoning, the agent will decide whether or not to communicate with its neighbours, observing that this decision will create a group of maximum m agents maintained for T steps, with m and T as hyperparameters. The incoming messages from each group are mixed with a bidirectional LSTM module, where the final output is merged with the internal thoughts and processed to obtain the state-action values.

4.2 Environments

For the evaluation, two testing environments were considered. The first of these is the **Switch** environment [32] (Fig. 4.1), a grid world where four agents must navigate to reach their respective switch on the map, having as input only the current position and target distance. The agents' challenge lies in the layout of the map, which features two rooms with starting points and a narrow corridor connecting them. The corridor can only be crossed by one agent at a time, meaning that coordinated behaviour is necessary to prevent the agents from getting stuck. It is worth mentioning that the agents are unable to see each other, and must therefore use a communication channel to gather information about the positions and intentions of their neighbours. The task ends after 250 steps of execution, or when all agents have successfully reached their destination.

The second testing environment (Fig. 4.2) is an instance of the **Predator-Prey** problem [19], which implementation requires particularly high levels of

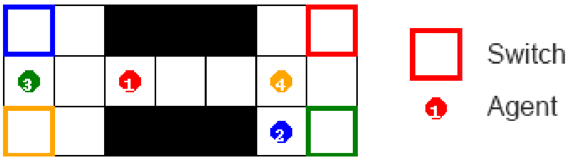


Figure 4.1: **Environment visualisation - Switch.** Four agents, with limited observation abilities, are tasked with reaching the switch of the corresponding colour. Coordination between the agents is necessary to avoid getting stuck in the corridor indefinitely.

coordination to succeed. The environment consists of a grid world in which some agents chase randomly moving entities with the aim of capturing them. The predators have limited visibility of the world, extending up to three units in each direction from their positions, and same speed as the prey. While an individual agent can only earn a small reward for ”tagging” a target, i.e. being in its same position, a group of organised agents can receive a bigger shared reward ”capturing” it, i.e. surrounding its position in the 4 axis. The task is considered complete either when all evaders have been captured or after 500 steps of execution.

4.3 Results

In evaluating the efficacy of the proposed method in comparison to the baselines, it is important to account for the disparity in the training methodologies employed. Specifically, IC3Net uses an on-policy training approach, in contrast to CoMix and ATOC. Furthermore, the latter two are trained with epsilon decay action selection, which allows for a more thorough exploration of the action space, but also leads to greater variance in the interactions with the environment.

Switch - Figure 4.3a displays the learning curve of the CoMix method and the two baseline approaches in terms of the total reward received by the agents in an episode. A higher value indicates not only the agents’ capability to reach their respective target, but also their ability to do so in a minimal

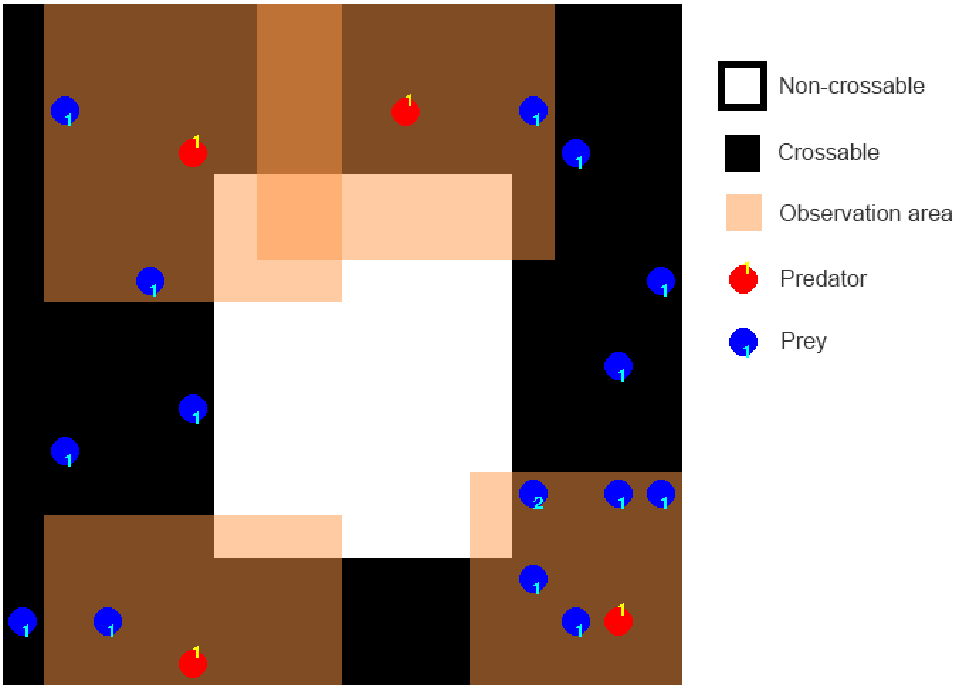
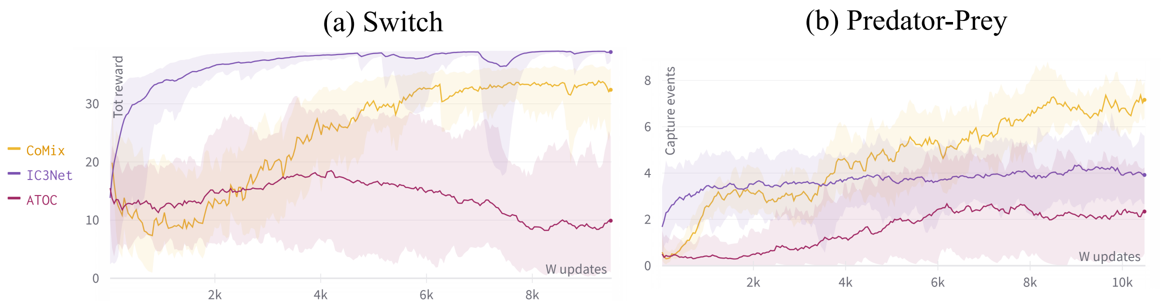


Figure 4.2: **Environment visualisation - Predator-Prey.** Predator agents navigate through space to capture prey. The agents' observation is restricted to a small area in their proximity, but they can communicate with each other to gather additional information and coordinate their actions strategically.


(a) Switch

(b) Predator-Prey

Figure 4.3: **Learning results.** Comparison of IC3Net [67], ATOC [28], and CoMix approaches in the two environments. The data presented are the average results of 10 executions with random seed initialization. The methods are compared in terms of the sum of agents' rewards in (a) and the number of captures in (b), per episode, against the number of weight updates. Note that IC3Net uses on-policy learning in contrast to ATOC and CoMix and therefore the number of steps seen by the former is two orders of magnitude higher than the others.

number of steps as the reward halves from 10 to 5 during the agents' lifespan. Given the simplicity of the task and its low potential for misunderstandings between agents, IC3Net's straightforward communication mechanism proves to be the most effective among the three methods compared. After about a thousand updates, the agents have already learnt almost deterministic behaviour and consistently achieve optimal rewards. CoMix follows, with slightly lower results, while ATOC struggles in the task. A deeper analysis of the latter reveals that in the execution, all agents tend to adopt the same action selection policy, moving in the same direction. This usually ensures that at least two agents reach their intended target, while the others rarely show any further intention of backtracking. The behaviour is probably caused by weight sharing used for all components, which was not even mitigated by the introduction of an ID in the policy computation. On the other hand, IC3Net reaches optimal performance, likely because the task resolution requires low variability and therefore does not necessitate strong communication skills. If the task is too easy, the policy may adopt a deterministic sequence of actions, and the communication channel may become of secondary utility. CoMix's method of communication and coordination maintains variability in the choices, which often leads to a slightly slower resolution but still enables all agents to reach their target with high scores.

Predator-Prey - The agents were required to decide whether to act individually or coordinate in groups to capture prey and receive higher rewards. Figure 4.3b show the evolution of learning for the basic configuration which involves four agents in a 16x16 map with 16 prey.

The analysis revealed that IC3Net performed quite well even in this case, with the agents often strategizing to converge in a fixed corner of the map then surrounding prey when they move within the agents' observable area. However,

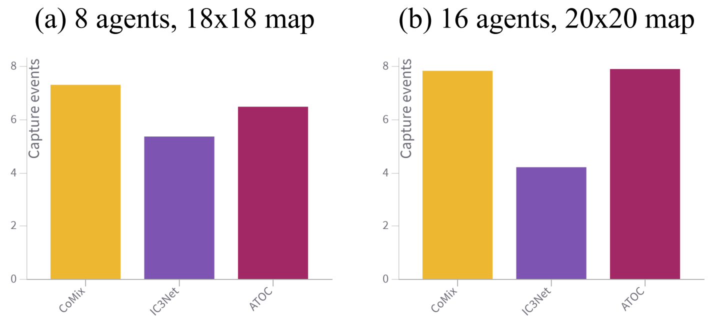


Figure 4.4: **Predator-Prey results at scale.** Comparison of IC3Net [67], ATOC [28], and CoMix approaches in two Predator-Prey scenarios at scale. In scenario (a), 8 agents were trained on an 18x18 map, while in scenario (b), 16 agents were trained on a 20x20 map. Both scenarios included 16 prey, and a time limit of 500 steps was used.

this strategy relies on the randomicity of prey movements and does not provide exploration to the agents, therefore could not be beneficial in general. In contrast, ATOC uses a more elaborate method of mixing the thoughts of others and shows agents very dedicated to catching prey. However, they often get ’distracted’ and drift away from the group. This explains the variance in the reported results. CoMix, on the other hand, challenges both methods by introducing the right amount of randomness during learning. It combines exploration with the preservation of cohesive relationships between agents leading to effective coordination for the capture of nearby prey.

The results obtained in the basic configuration of the environment demonstrated the need for further testing in larger spaces and with more agents in action. Consequently, two additional configurations were considered:

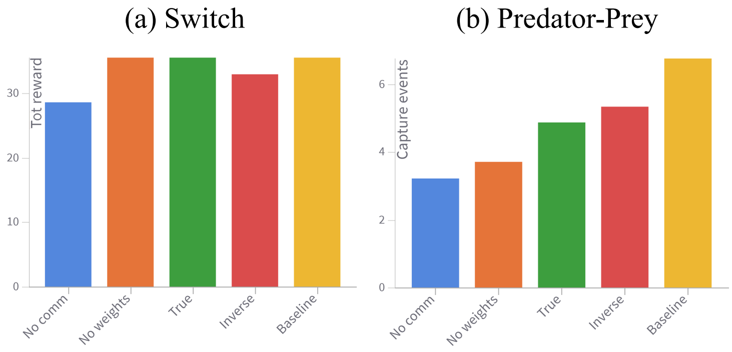
- 8 agents in a 18x18 map
- 16 agents in a 20x20 map

Despite scaling the space and the number of predators, we kept the number of prey and episode length constant at 16 and 500, respectively, to ensure comparability between results and agents’ abilities. Figure 4.4 shows the best results obtained after an equivalent training time for the three methods. As hypothesized, IC3Net's performance degrades due to the sparsity of the map and its inability to cover larger portions of it effectively. In contrast, the coordinated planning ability demonstrated by CoMix and ATOC allows them to perform better as the scale of the scenario increases. Interestingly, despite their comparable performance, they seem to adopt different methodologies, with CoMix leading to partial map coverage and preferring compactness, while ATOC leads to agents more distributed over the entire map surface but also to situations with too small groups trying to catch prey or lone agents wandering around the map. Overall, the results demonstrate the effectiveness of CoMix in balancing exploration and coordination, even in larger environments.

4.4 Ablation study

Building upon the quantitative and qualitative analysis presented in the comparison against other methods, this section delves deeper into the factors that contribute to the success of the proposed method. Tests were repeated in both environments – only in the smaller version in the case of Predator-Prey, to limit the use of computational resources – considering the final proposal as a baseline against variants that differ in the architecture or training methodologies.

- **'*no comm*'** (Q w/o Q_{coord}), uses the base structure of learning, but with agents lacking communication abilities. Agents are required to understand the environment dynamics thoroughly to achieve their objective, as centralised training is the only mechanism for information sharing.
- **'*no weights*'** (L_C w/o $\mathbf{w}$), does not use the weights provided by the QMIX framework for the computation of the Coordinator loss. This variant shows the performance of the base method if extrapolated from the current CTDE framework.



(a) Switch

(b) Predator-Prey

Figure 4.5: **Ablation results.** Comparison between the ablated variants of the proposed method and the basic implementation of CoMix. The data presented are the average results of 10 (a) and 5 (b) episodes, after an equivalent amount of training, averaged over 3 executions each.

- **’*true*’** ($\tilde{\mathbf{c}}^{true}$) and **’*inverse*’** ($\tilde{\mathbf{c}}^{inverse}$) adopt different flavours of training for the Coordinator. The first use an all-true mask of coordination, and the second use a single-inference full inversion of probabilities, as explained in 3.2.1.

4.4.1 Performance analysis

Figure 4.5 puts in perspective the average results obtained after an equivalent time of training for each variant discussed. As in the method evaluation, the total reward is considered as a metric of success for the Switch environment and the number of captures in Predator-Prey. The comparison also provides very interesting insights into how different choices influence agent training and consequently the strategies developed.

In the Switch environment, we can see *inverse* and *no weights* demonstrating the same coordination performances as the baseline. However, they require more training steps to reach the same achievements as they obtain lower rewards over time. In both variants we can identify situations in which the agents struggle to reach their position, but this is mainly due to individual misbehaviour rather than coordination impediments. On the other hand, we observe *true* obtaining very good results even if with a slower convergence and a weaker coordination (See Fig. 4.6). Interestingly, while the baseline implementation allows agents to learn incrementally and navigate the environment, *true* does not initially report successes. This is due to the fact that this method of supervision incentivises considering everyone's intentions instead of limiting the space for collaboration to local coordination.

In the Predator-Prey environment, the baseline outperformed the other variants, demonstrating superior performance and strategies not observed in others. For instance, the agents employed a group exploration strategy of moving along the map edges, as capturing prey in these positions requires fewer predators. The *inverse* and *true* flavours ranked behind the baseline with similar results in terms of performance. However, *true* exhibited coordination issues and achieved few captures even when opportunities were present, while *inverse* as well as *no weights* encountered significant exploration difficulties.

The absence of communication capabilities results in an apparent lack of coordination between agents in both environments. In the Predator-Prey, this was evidenced by the tendency of the *no comm* agents to disperse and not remain cohesive, leading to poor performance compared to the other variants. In the Switch environment, *no comm* agents seem disturbed by each other's actions, moving back and forth several times when facing one another, unable to anticipate or understand their movements.

To further analyse the performance of the alternatives, we examine the Coordinator module's training loss. *no weights* exhibited high spikes in training loss, while the *true* and *inverse* variants showed more stable learning. However, it's possible that the stability is due to the ineffectiveness of the methods, as both *true* and *inverse* performed lower than the baseline and *no weights*.

Looking at the results as a whole, it can be deduced that enabling communication between agents with the inclusion of the additional term in the state-action computation is crucial for improved performance. Regarding the tested learning modalities, it can be observed that, except for *true*, which is highly situation and environment-dependent, the others can be considered simplified variants of the baseline leading to slower results.

4.4.2 Communication analysis

To conclude the analysis, the communication mechanism of CoMix has been investigated by analysing the evolution of the predicted communication masks. Coordination success was determined by comparing the predicted state-action values against the values obtained with the alternative coordination mask when computing the Coordinator's loss. The "Good/bad coordination ratio", shown in Figure 4.6, indicates the percentage of agents making correct predictions per each step. The increasing metric for all ablated strategies shows promising results, with the base method achieving the maximum value. However, although the proposed strategy is generally applicable, adopting a learning process for the Coordinator tailored to the specific environment dynamics could potentially yield better results (e.g., in a fully cooperative environment, could be more proficient training against the maximum amount of information and then learn to filter out what is perceived as irrelevant for the current step). Another important finding is given by the number of times agents coordinate with others during training. In the Switch environment, we observe a decrease in this value, whereas in the Predator-Prey environment, the value remains almost constant and higher overall. This finding aligns with the intuition, as the latter environment requires all four agents to coordinate in order to capture prey.

Furthermore, it should be noted that the CoMix implementation inherently offers interpretability for the agents' decisions on actions. We can trace an action back to single interactions with other agents or to the agents' self-imposed objectives. For example, in the Switch environment, when a single agent remains, Q_{coord} term has a null value since its actions are not influenced by others. In the case of the Predator-Prey environment, we can observe the norm of Q_{self} and Q_{coord} to determine whether an agent acts primarily by following its own will or by adopting a strategy aimed at coordination.

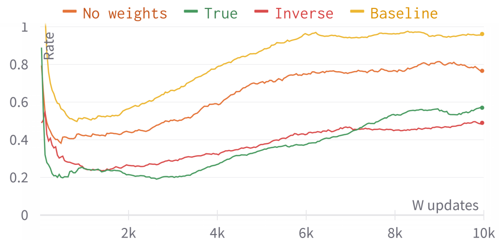


Figure 4.6: **Good/bad coordination ratio - Switch.** Ratio relative to the number of agents predicting action-state values above or below their loss target. As this value is directly influenced by the communication mask, it can be read as the ratio of agents communicating effectively.

4.5 Conclusions

This master thesis addresses the problem of coordination in multi-agent reinforcement learning systems, proposing a new approach in which agents make decisions based on an explicit combination of self-interest and willingness to coordinate. As an implementation of this approach, CoMix is presented, demonstrating performance comparable to, if not superior to, important works in the field. The evaluation analysis provides insights into the underlying mechanisms of the method and its effectiveness in training agents to cope with complex environmental dynamics, such as partial observability and hidden reward structures. The ablation study provides useful information on the role of the different components of the architecture and underlines the importance of design and training choices. Overall, CoMix appears to be a promising approach to coordination in multi-agent systems, with potential for implementation in various contexts beyond virtual environments.

4.5.1 Future work

Future research may focus on the further development of the method with alternative training frameworks to QMIX, thus testing its effectiveness in different environments not subject to its limitations. In addition, future studies could explore the interpretability of CoMix and its ability to learn coordination strategies, providing valuable insights into the dynamics that may occur in simulated situations and better understanding their evolution. Overall, I see these directions as an opportunity to advance the field of research on smarter and more flexible solutions for multi-agent systems.

4.5.2 Limitations

While this approach has demonstrated promising results in various scenarios, it is important to acknowledge certain limitations of it to ensure its successful implementation and to suggest further improvements in research. One of the main limitations is the observed slowness in the training process, attributed to the inherent complexity of the approach. The absence of mechanisms limiting computation, such as restrictions in communication range or attention modules, together with the presence of multiple action selection mechanisms, are the main causes. However, it is worth noting that this limitation can be addressed partly by incorporating additional mechanisms on top of the CoMix approach, drawing on the extensive literature on multi-agent systems.

It should be noted that, despite the accuracy of the results presented, the testing and evaluation phase was limited to the specific environments presented, thus not adequately evaluated in purely competitive/cooperative scenarios. Furthermore, since the final choice of CTDE framework fell on QMIX, it is necessary to emphasise how its limitations and representation constraints influenced the choice and set-up of the test environments. For instance, QMIX is not suitable for competitive implementations, and the environments were necessarily modelled without negative rewards due to the monotonic constraint. Although these constraints do not automatically represent simplifications in solving tasks or shortcuts in learning for agents, they do limit the general applicability of the method.

Bibliography

[1] A. K. Agogino and K. Tumer. Unifying temporal and structural credit assignment problems. *3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*:980–987, 2004.

[2] S. V. Albrecht and P. Stone. Autonomous agents modelling other agents: a comprehensive survey and open problems. *ArXiv*, abs/1709.08071, 2017.

[3] D. S. Bernstein, S. Zilberstein, and N. Immerman. The complexity of decentralized control of markov decision processes. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2000.

[4] V. D. Blondel and J. N. Tsitsiklis. A survey of computational complexity results in systems and control. *Automatica*, 36(9):1249–1274, September 2000. ISSN: 0005-1098. DOI: `10.1016/S0005-1098(00)00050-9`. URL: `https://doi.org/10.1016/S0005-1098(00)00050-9`.

[5] J. Blumenkamp and A. Prorok. The emergence of adversarial communication in multi-agent reinforcement learning. *ArXiv*, abs/2008.02616, 2020.

[6] M. Bowling and M. M. Veloso. Multiagent learning using a variable learning rate. *Artificial Intelligence (AIJ)*, 136:215–250, 2002.

[7] N. Brown and T. Sandholm. Superhuman ai for heads-up no-limit poker: libratus beats top professionals. *Science*, 359:418–424, 2018.

[8] L. Buşoniu, R. Babuvska, and B. D. Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 38:156–172, 2008.

[9] Y. Cao, W. Yu, W. Ren, and G. Chen. An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial Informatics (TIE)*, 9:427–438, 2012.

[10] G. Coulouris. Distributed systems : concepts and design / george coulouris ... [et al.] In 2012.

[11] A. Das, T. Gervet, J. Romoff, D. Batra, D. Parikh, M. G. Rabbat, and J. Pineau. Tarmac: targeted multi-agent communication. *ArXiv*, abs/1810.11187, 2018.

[12] F. L. da Silva and A. H. R. Costa. A survey on transfer learning for multiagent reinforcement learning systems. *Journal of Artificial Intelligence Research (JAIR)*, 64:645–703, 2019.

[13] Z. Ding, T. Huang, and Z. Lu. Learning individually inferred communication for multi-agent cooperation. *ArXiv*, abs/2006.06455, 2020.

[14] T. W. Finin, R. Fritzson, D. P. McKay, and R. McEntire. Kqml as an agent communication language. In *International Conference on Information and Knowledge Management*, 1994.

[15] J. N. Foerster, Y. Assael, N. de Freitas, and S. Whiteson. Learning to communicate to solve riddles with deep distributed recurrent q-networks. *ArXiv*, abs/1602.02672, 2016.

[16] J. N. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson. Counterfactual multi-agent policy gradients. *ArXiv*, abs/1705.08926, 2017.

[17] N. Fulda and D. Ventura. Predicting and preventing coordination problems in cooperative q-learning systems. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2007.

[18] S. Gronauer and K. Diepold. Multi-agent deep reinforcement learning: a survey. *Artificial Intelligence Review*, 55:895–943, 2021.

[19] J. K. Gupta, M. Egorov, and M. Kochenderfer. Cooperative multi-agent control using deep reinforcement learning. In *The 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 66–83. Springer, 2017.

[20] F. Heider and M. L. Simmel. An experimental study of apparent behavior. *American Journal of Psychology (AJP)*, 57:243–259, 1944.

[21] P. Hernandez-Leal, M. Kaisers, T. Baarslag, and E. M. de Cote. A survey of learning in multiagent environments: dealing with non-stationarity. *ArXiv*, abs/1707.09183, 2017.

[22] P. Hernandez-Leal, B. Kartal, and M. E. Taylor. A survey and critique of multiagent deep reinforcement learning. *Autonomous Agents and Multi-Agent Systems*, 33:750–797, 2019.

[23] P. J. Hoen, K. Tuyls, L. Panait, S. Luke, and H. L. Poutre. An overview of cooperative and competitive multiagent learning. In *Learning and Adaption in Multi-Agent Systems*, 2005.

[24] G. Hu, Y. Zhu, D. Zhao, M. Zhao, and J. Hao. Event-triggered communication network with limited-bandwidth constraint for multi-agent reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, PP, 2021.

[25] J. Hu and M. P. Wellman. Nash q-learning for general-sum stochastic games. *Journal of Machine Learning Research (JMLR)*, 4(Nov):1039–1069, 2003.

[26] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne. Imitation learning: a survey of learning methods. *ACM Computing Surveys*, 50(2), April 2017. ISSN: 0360-0300. DOI: `10.1145/3054912`. URL: `https://doi.org/10.1145/3054912`.

[27] M. Hutsebaut-Buysse, K. Mets, and S. Latré. Hierarchical reinforcement learning: a survey and open research challenges. *Machine Learning and Knowledge Extraction*, 4(1):172–221, 2022. ISSN: 2504-4990. DOI: `10.3390/make4010009`. URL: `https://www.mdpi.com/2504-4990/4/1/9`.

[28] J. Jiang and Z. Lu. Learning attentional communication for multi-agent cooperation. In *Neural Information Processing Systems*, 2018.

[29] S. Kapetanakis and D. Kudenko. Reinforcement learning of coordination in cooperative multi-agent systems. *AAAI Innovative Applications of Artificial Intelligence Conference (IAAI)*, 2002:326–331, 2002.

[30] W. Kim, J. Park, and Y. Sung. Communication in multi-agent reinforcement learning: intention sharing. In 2021.

[31] M. Koes, I. R. Nourbakhsh, and K. P. Sycara. Communication efficiency in multi-agent systems. *IEEE International Conference on Robotics and Automation (ICRA)*, 3:2129–2134 Vol.3, 2004.

[32] A. Koul. Ma-gym: collection of multi-agent environments based on openai gym. `https://github.com/koulanurag/ma-gym`, 2019.

[33] J. Z. Leibo, V. F. Zambaldi, M. Lanctot, J. Marecki, and T. Graepel. Multi-agent reinforcement learning in sequential social dilemmas. *ArXiv*, abs/1702.03037, 2017.

[34] S. Li, Y. Zhou, R. Allen, and M. J. Kochenderfer. Learning emergent discrete message communication for cooperative reinforcement learning. *International Conference on Robotics and Automation (ICRA)*:5511–5517, 2022.

[35] W. Li, X. Wang, B. Jin, J. Sheng, Y. Hua, and H. Zha. Structured diversification emergence via reinforced organization control and hierarchical consensus learning. In *Adaptive Agents and Multi-Agent Systems*, 2021.

[36] T. Lin, M. Huh, C. Stauffer, S. N. Lim, and P. Isola. Learning to ground multi-agent communication with autoencoders. *ArXiv*, abs/2110.15349, 2021.

[37] M. L. Littman et al. Friend-or-foe q-learning in general-sum games. In volume 1, pages 322–328, 2001.

[38] W. Liu, S. Liu, J. Cao, Q. Wang, X. Lang, and Y. Liu. Learning communication for cooperation in dynamic agent-number environment. *IEEE/ASME Transactions on Mechatronics (TMECH)*, 26:1846–1857, 2021.

[39] Y.-C. Liu, J. Tian, N. Glaser, and Z. Kira. When2com: multi-agent perception via communication graph grouping. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*:4105–4114, 2020.

[40] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *ArXiv*, abs/1706.02275, 2017.

[41] L. Luncean and A. Becheru. Communication and interaction in a multi-agent system devised for transport brokering. In 2015.

[42] H. Mao, Z. Gong, Y. Ni, X. Liu, Q. Wang, W. Ke, C. Ma, Y. Song, and Z. Xiao. Accnet: actor-coordinator-critic net for ”learning-to-communicate” with deep multi-agent reinforcement learning. *ArXiv*, abs/1706.03235, 2017.

[43] P. Maramotti, A. P. Capasso, G. Bacchiani, and A. Broggi. Tackling real-world autonomous driving using deep reinforcement learning. *2022 IEEE Intelligent Vehicles Symposium (IV)*:1274–1281.

[44] L. Matignon, G. J. Laurent, and N. L. Fort-Piat. Independent reinforcement learners in cooperative markov games: a survey regarding coordination problems. *The Knowledge Engineering Review (KER)*, 27:1–31, 2012.

[45] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. A. Riedmiller, A. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.

[46] O. Nachum, S. S. Gu, H. Lee, and S. Levine. Data-efficient hierarchical reinforcement learning. In *Neural Information Processing Systems*, 2018.

[47] Y. Niu, R. R. Paleja, and M. C. Gombolay. Multi-agent graph-attention communication and teaming. In *Adaptive Agents and Multi-Agent Systems*, 2021.

[48] R. Olfati-Saber, J. A. Fax, and R. M. Murray. Consensus and cooperation in networked multi-agent systems. *IEEE*, 95:215–233, 2007.

[49] OpenAI. Openai five. `https://blog.openai.com/openai-five/`, 2018.

[50] B. Osinski, A. Jakubowski, P. Milos, P. Ziecina, C. Galias, S. Homoceanu, and H. Michalewski. Simulation-based reinforcement learning for real-world autonomous driving. *IEEE International Conference on Robotics and Automation (ICRA)*:6411–6418, 2020.

[51] G. Palmer, K. Tuyls, D. Bloembergen, and R. Savani. Lenient multi-agent deep reinforcement learning. *ArXiv*, abs/1707.04402, 2017.

[52] G. Papoudakis, F. Christianos, L. Schafer, and S. V. Albrecht. Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks. In *NeurIPS Datasets and Benchmarks*, 2020.

[53] P. Peng, Y. Wen, Y. Yang, Q. Yuan, Z. Tang, H. Long, and J. Wang. Multiagent bidirectionally-coordinated nets: emergence of human-level coordination in learning to play starcraft combat games. *ArXiv*, 2017.

[54] E. Pesce and G. Montana. Improving coordination in small-scale multi-agent deep reinforcement learning through memory-driven communication. *Machine Learning*:1–21, 2019.

[55] S. Poslad. Specifying protocols for multi-agent systems interaction. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 2:15, 2007.

[56] Z. Rabinovich, C. V. Goldman, and J. S. Rosenschein. The complexity of multiagent systems: the price of silence. In *Adaptive Agents and Multi-Agent Systems*, 2003.

[57] R. P. N. Rao. Reinforcement learning: an introduction; r.s. sutton, a.g. barto (eds.); mit press, cambridge, ma, 1998, 380 pages, isbn 0-262-19398-1, $42.00. *Neural Networks*, 13:133–135, 2000.

[58] T. Rashid, M. Samvelyan, C. S. D. Witt, G. Farquhar, J. N. Foerster, and S. Whiteson. Qmix: monotonic value function factorisation for deep multi-agent reinforcement learning. *ArXiv*, abs/1803.11485, 2018.

[59] A. Rasouli, I. Kotseruba, and J. K. Tsotsos. Agreeing to cross: how drivers and pedestrians communicate. *IEEE Intelligent Vehicles Symposium (IV)*:264–269, 2017.

[60] W. Ren and R. W. Beard. Consensus seeking in multiagent systems under dynamically changing interaction topologies. *IEEE Transactions on Automatic Control (TAC)*, 50:655–661, 2005.

[61] R. A. Rescorla. Behavioral studies of pavlovian conditioning. *Annual Review of Neuroscience*, 11(1):329–352, 1988.

[62] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, T. P. Lillicrap, and D. Silver. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588:604–609, 2019.

[63] Y. Shoham, R. Powers, and T. Grenager. If multi-agent learning is the answer, what is the question? *Artificial Intelligence*, 171:365–377, 2007.

[64] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. P. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529:484–489, 2016.

[65] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018. DOI: `10.1126/science.aar6404`. URL: `https://www.science.org/doi/abs/10.1126/science.aar6404`.

[66] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. baker, M. Lai, A. Bolton, Y. Chen, T. P. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis. Mastering the game of go without human knowledge. *Nature*, 550:354–359, 2017.

[67] A. Singh, T. Jain, and S. Sukhbaatar. Learning when to communicate at scale in multiagent cooperative and competitive tasks. *ArXiv*, abs/1812.09755, 2018.

[68] K. Son, D. Kim, W. J. Kang, D. E. Hostallero, and Y. Yi. Qtran: learning to factorize with transformation for cooperative multi-agent reinforcement learning. *ArXiv*, abs/1905.05408, 2019.

[69] G. B. Stone, D. A. Talbert, and W. Eberle. A survey of scalable reinforcement learning. *International Journal of Intelligent Computing Research (IJICR)*, 2022.

[70] S. Sukhbaatar, A. D. Szlam, and R. Fergus. Learning multiagent communication with backpropagation. In *NIPS*, 2016.

[71] P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls, and T. Graepel. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *The 17th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, AAMAS '18, pages 2085–2087, Stockholm, Sweden. International Foundation for Autonomous Agents and Multiagent Systems, 2018.

[72] TheFaramaFoundation. Farama group for open source reinforcement learning tools. `https://farama.org/`, 2022.

[73] J. Tu, T.-H. Wang, J. Wang, S. Manivasagam, M. Ren, and R. Urtasun. Adversarial attacks on multi-agent communication. *IEEE/CVF International Conference on Computer Vision (ICCV)*:7748–7757, 2021.

[74] Unity. Unity ml agents. `https://unity.com/products/machine-learning-agents/`, 2017.

[75] H. Van Hasselt, A. Guez, and D. Silver. Deep reinforcement learning with double q-learning. In *AAAI Conference on Artificial Intelligence*, volume 30 of number 1, 2016.

[76] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, J. Oh, D. Horgan, M. Kroiss, I. Danihelka, A. Huang, L. Sifre, T. Cai, J. P. Agapiou, M. Jaderberg, A. S. Vezhnevets, R. Leblond, T. Pohlen, V. Dalibard, D. Budden, Y. Sulsky, J. Molloy, T. L. Paine, C. Gulcehre, Z. Wang, T. Pfaff, Y. Wu, R. Ring, D. Yogatama, D. Wunsch, K. McKinney, O. Smith, T. Schaul, T. P. Lillicrap, K. Kavukcuoglu, D. Hassabis, C. Apps, and D. Silver. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*:1–5, 2019.

[77] J. Wang, Z. Ren, T. Liu, Y. Yu, and C. Zhang. Qplex: duplex dueling multi-agent q-learning. *ArXiv*, abs/2008.01062, 2020.

[78] E. Wei and S. Luke. Lenient learning in independent-learner stochastic cooperative games. *Journal of Machine Learning Research (JMLR)*, 17:84:1–84:42, 2016.

[79] G. Weiss. *Multiagent systems: a modern approach to distributed artificial intelligence*. MIT press, 1999.

[80] D. H. Wolpert and K. Tumer. Optimal payoff functions for members of collectives. *Adv. Complex Syst.*, 4:265–280, 2001.

[81] Y. Yang, J. Hao, B. Liao, K. Shao, G. Chen, W. Liu, and H. Tang. Qatten: a general framework for cooperative multiagent reinforcement learning. *ArXiv*, abs/2002.03939, 2020.

[82] T. Zhang, H. Xu, X. Wang, Y. Wu, K. Keutzer, J. Gonzalez, and Y. Tian. Multi-agent collaboration via reward attribution decomposition. *ArXiv*, abs/2010.08531, 2020.

[83] L.-y. Zhao, T. Chang, L. Zhang, J. Zhang, K. Chu, and D.-p. Kong. Targeted multi-agent communication algorithm based on state control. *Defence Technology*, 2022.

Acknowledgements

I am deeply grateful for the support and guidance of my supervisor, Mirco Musolesi, who provided me with valuable insights and feedback during the development of this work. I would also like to extend my appreciation to the ISSLab group at UCL for their warm welcome and the research environment that facilitated my work.

To my dear friends, I am grateful for bearing with me, for distracting me and cheering me up when I needed it. Your encouragement was invaluable and helped me to stay motivated throughout the process.

Finally, I would like to express my sincere gratitude to my family and those close to me for their constant support and love, which has been unwavering throughout. Thanks.