Path Exploration with Random Network Distillation on Multi-Agent Reinforcement Learning

Mr. Korawat Charoenpitaks

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science in Computer Science
Department of Computer Engineering
FACULTY OF ENGINEERING
Chulalongkorn University
Academic Year 2019

การสำรวจเส้นทางด้วยการกลั่นตัวโครงข่ายแบบสุ่มบนการเรียนรู้เสริมกำลังหลายตัวแทน

นายกรวัฒน์ เจริญพิทักษ์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2562

Thesis Title	Path Exploration with Random Network Distillation on Multi-Agent Reinforcement Learning
By	Mr. Korawat Charoenpitaks
Field of Study	Computer Science
Thesis Advisor	Associate Professor Dr. Yachai Limpiyakorn

Accepted by the FACULTY OF ENGINEERING, Chulalongkorn University in Partial Fulfillment of the Requirement for the Master of Science

..... Dean of the FACULTY OF ENGINEERING

(Professor Dr. SUPOT TEACHAVORASINSKUN)

THESIS COMMITTEE

..... Chairman

(Assistant Professor Dr. SUKREE SINTHUPINYO)

..... Thesis Advisor

(Associate Professor Dr. Yachai Limpiyakorn)

..... External Examiner

(Dr. Paskorn Apirukvorapinit)

กรวัฒน์ เจริญพิทักษ์ : การสำรวจเส้นทางด้วยการกลั่นตัวโครงข่ายแบบสุ่มบนการเรียนรู้เสริมกำลังหลายตัวแทน. (Path Exploration with Random Network Distillation on Multi-Agent Reinforcement Learning) อ.ที่ปรึกษาหลัก : รศ. ดร.ญาใจ ลิ่มปิยะกรณ์

แรงจูงใจภายในเป็นทางเลือกหนึ่งที่มีศักยภาพช่วยเพิ่มขีดความสามารถของอัลกอรึทึมการเรียนรู้เสริมกำลังในสภาพแวดล้อมที่ซับซ้อน วิธีการดังกล่าวขยายความสามารถในการสำรวจได้ โดยไม่ต้องอาศัยค่าที่ชัดแจ้งจากผู้สร้าง อีกทั้งยังสามารถใช้ได้ทั่วไปกับสภาพแวดล้อมใดๆ ทำให้วิธีการนี้มีความเหมาะสมกับการนำมาใช้ในกรณีของการเรียนรู้แบบเสริมกำลังหลายตัวแทน ซึ่งมีสภาพแวดล้อมซับซ้อนมากกว่าปกติ งานวิจัยนี้ได้เสนอโมเดลการสำรวจโดยใช้แรงจูงใจภายในจากอัลกอริทึมการกลั่นตัวโครงข่ายแบบสุ่มเพื่อเพิ่มสมรรถนะของการเรียนรู้เสริมกำลังหลายตัวแทน และเปรียบเทียบผลลัพธ์กับผลการทดลองจากผลเกณฑ์มาตรฐานในหลายๆ สภาพแวดล้อม ทั้งนี้ ผู้วิจัยได้นำเสนอแนวคิดค่าอัตราส่วนสำหรับตัดออกเพื่อบังคับจำกัดขนาดค่าความเหมาะ โดยอ้างอิงจากอัตราส่วนที่มาจากค่าแรงจูงใจภายนอก การใช้ค่าอัตราส่วนสำหรับตัดออกจะช่วยตัดขนาดค่าส่วนเกินที่อาจทำให้การหาค่าเหมาะสมไม่มีความเสถียร การทดลองได้ดำเนินการบนสถาปัตยกรรมหลายตัวแทนสองแบบที่แตกต่าง ประกอบด้วย สถาปัตยกรรมแรงจูงใจภายในแบบเดี่ยว และสถาปัตยกรรมแรงจูงใจภายในแบบรวมศูนย์ ผลการทดลองแสดงให้เห็นว่า ในกรณีที่สภาพแวดล้อมมีความซับซ้อนมาก สถาปัตยกรรมแรงจูงใจภายในแบบรวมศูนย์ร่วมกับอัตราส่วนสำหรับตัดออกที่มีค่าน้อย จะช่วยเพิ่มสมรรถนะได้มากกว่าปกติ โดยสามารถทำอัตราการชนะได้จนถึง 70% ในทั้งสองสถาปัตยกรรมซึ่งสูงกว่าอัตราที่ดีที่สุด 43% ของเกณฑ์เปรียบเทียบมาตรฐานในงานวิจัยอื่นที่ทดลองบนสภาพแวดล้อม 2s3z

สาขาวิชา	วิทยาศาสตร์คอมพิวเตอร์	ลายมือชื่อนิสิต .....
ปีการศึกษา	2562	ลายมือชื่อ อ.ที่ปรึกษาหลัก .....

6170903021 : MAJOR COMPUTER SCIENCE

KEYWORD: Reinforcement Learning, Multi-Agent Learning, Curiosity Exploration, Intrinsic Reward., Random Network Distillation

Korawat Charoenpitaks : Path Exploration with Random Network Distillation on Multi-Agent Reinforcement Learning. Advisor: Assoc. Prof. Dr. Yachai Limpiyakorn

Intrinsic motivation is one of the potential candidates to help improve performance of reinforcement learning algorithm in complex environments. The method enhances exploration capability without explicitly told by the creator and works on any environment. This is suitable in the case of multi-agent reinforcement learning where the environment complexity is more than usual. The research presents an exploration model using intrinsic motivation built from the random network distillation algorithm to improve the performance of multi-agent reinforcement learning and compare with the benchmark in different scenarios. The concept of clipping ratio is introduced to enforces the limit on optimization magnitude. Based on the extrinsic reward, the limit in the form of clipping ratio helps truncate the excessive magnitude that may cause instability to the optimization. The experiments were carried out on two different multi-agent architectures: 1) Individual Intrinsic Motivation Architecture, and 2) Centralized Intrinsic Motivation Architecture. The experimental results showed that in case of very complex environments, Centralized Intrinsic Motivation Architecture accompanied with a small clipping ratio could gain an increase in performance. The result reported the achievement of up to 70% win-rate in both architectures which is higher than those of the benchmark at the best of 43% in 2s3z environment.

Field of Study:	Computer Science	Student's Signature .....
Academic Year:	2019	Advisor's Signature .....

ACKNOWLEDGEMENTS

The completion of this thesis could not have been possible without the assistance and participation of so many people whose names may not all be enumerated. Their contributions are sincerely appreciated and gratefully acknowledged. However, the group would like to express deep appreciation and indebtedness particularly to the following.

Assoc. Prof. Dr. Yachai Limpiyakorn, Asst. Prof. Dr. Sukree Sinthupinyo and Dr. Paskorn Apirukvorapinit for their time, endless effort and generous advice during the presentation.

Last but not least, to all relatives, friends, others who in one way or another shared their support, either morally, financially and physically, thank you.

Korawat Charoenpitaks

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

Chapter 1

Introduction

1.1 Statement of the Problems

Reinforcement learning (RL) is a subfield of machine learning where the learner entities are in the form of software agents in which learn from reward signals sampled through interactions with simulation or environment. RL is becoming more and more popular and presents high potentials for many applications in different domains as the RL can be used to learn and solve a sequential decision-making problem [1]. Combined with Deep Learning to learn state representations, deep reinforcement learning (DRL) is a general framework to learn on any complex challenging problem. This allows the DRL framework to be one of the best candidates to solve any general problems. However, there are many benefits and challenges regarding the issue.

RL in general requires a large amount of interaction through simulations at the benefit of requiring no data labels. In addition, RL could introduce creativity and uncertainty based on the design, the algorithm has potential to achieve alternative solutions that achieve the goal but also introduce misbehavior that could also achieve the same goal. The potential for a wide-open sequential solution comes at the cost of optimization complexity in terms of difference objective function. On top of that, the data used from training is sampled through simulation interaction. Therefore, it is required a good sequential simulated interaction to be able to achieve optimal solutions.

One of the areas of research in RL is in exploration which is about how to seek for better data for optimization. In detail, the sampled data come from interacting with simulation where it is represented using elements in the Markov Decision Process framework which are state action and immediate reward. The exploration algorithm must enhance agent capability to obtain more quality sampled

data in different states. As the main objective of RL is to maximize the expected sum of reward [2] of the episode, the quality of sampled reward is essential for the learning process of RL. In this case, there are two problems regarding quality of the reward, sparse [3] and too complex. In case of complex rewards, it is probable that agents would get stuck in the loop of suboptimal rewards as they thought that the current path is optimal which is not the case.

There is a lot of literature researching the topics of exploration. Intrinsic motivation is one of the alternative solutions that enhance exploration capability on any tasks without manually customizing it by the creator [2]. Therefore, it can scale better and show a better potential for use in the case of more complex challenges of Multi-Agent Reinforcement Learning (MARL). The idea of self-generated curiosity of humans can be applied to create intrinsic reward to help improve optimization. It encourages a software agent to explore new states by adding intrinsic signals when agents explore the unknown state and have no effect on the known states.

In many real-world applications, the problems are multi-agent in nature. The business challenges are very complex, and in order to utilize RL effectively, the effective exploration algorithm for MARL needs to be explored. In the research, we explored different methods and architecture of Intrinsic Motivation in both Single Agent Reinforcement Learning and Multi-Agent Reinforcement Learning. In addition, we also explore the methodology of applying intrinsic motivation on MARL for robustness of applying intrinsic motivation. The preliminary experiments were conducted on many simulations such as OpenAI for single agent RL and The StarCraft Multi Agent Challenge (SMAC) environment for MARL.

1.2 Objective

To explore and improve efficiency on applying random network distillation on multi agent reinforcement learning to achieve the best outcome in terms of sample efficiency and the maximum long-term score.

1.3 Scope of Study

1. This work will be tested on The StarCraft Multi-Agent Challenge (SMAC) environment
2. This work will be using Counterfactual Multi-Agent (COMA) as a baseline algorithm
3. The experiment results are measured in term of win rate over episode used in different architectures and parameters and in different aspects of environment such as symmetry, asymmetry and micro-trick

1.4 Research Methodology

1. Research on methodology/theory/study on related topics
2. Design Develop Experiment and Analyze on the results of using random network distillation on single agent reinforcement learning
3. Design Develop Experiment and Analyze on the results of using random network distillation on multi agent reinforcement learning
4. Evaluate and Conclude on contribution of the research
5. Summarize and compile the thesis

1.5 Outcomes

1. Empirical results contributed to Random network distillation as a method to improve path exploration on RL in both single and multi-agent setups
2. More alternative choices of exploration models that could improve performance for MARL

1.6 Thesis Publication

Parts of the thesis had been published in one academic conference and one international journal as following:

[1] K. Charoenpitaks and Y. Limpiyakorn, "Curiosity-Driven Exploration Effectiveness on Various Environments," in Proceedings of the 3rd International Conference on Vision, Image and Signal Processing, 2019, pp. 1-6.

[2] Chaoenpitaks, K., and Limpiyakorn, Y, “Multi-Agent Reinforcement Learning with Clipping Intrinsic Motivation,” International Journal of Machine Learning and Computing (IJMLC), *to appear*.

Chapter 2

Literature Review

2.1 Markov Decision Process

The RL algorithm is usually modeled following a Markov Decision Process (MDP) framework which is a mathematical framework for decision making in discrete and stochastic control process [4].

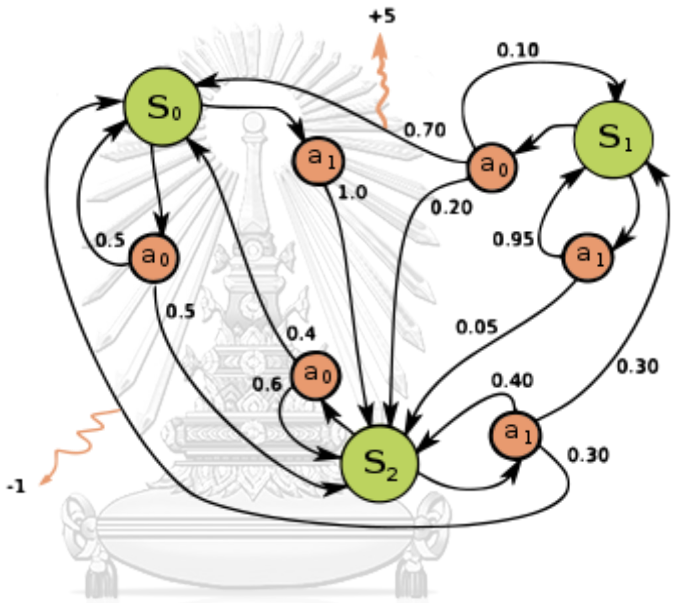

Figure 1: Markov Decision Process Example with three states (green circles), two actions (orange circles) and rewards (orange arrows)

The MDP framework is an abstraction of the goal-oriented learning problem that learn from interaction with the environment. Any kind of problems can be represented to only three signals passing back and forth between an agent and its environment: one signal to represent the action, one signal to represent the basis on which the choices are made (the states), and one signal to define the goal which is the rewards [5]. This framework may not be able to represent all decision-learning problems, but it has proved to be widely useful and applicable.

Figure 1 depicts the MDP elements which consists of four elements, state, action, state transition probability given state and action and immediate reward after state transitions. The State is represented in large green circles while a red circle is

the representation of an action. Transition probability is displayed next to the black transition line which represent the probability of taking that path. Nevertheless, curvy pink arrow represents immediate rewards.

When state transition probability and reward is known, the MDP can be solved by many algorithms such as dynamic programming. To find the optimal policy of finite state and action MDP, the algorithm requires two steps, value update and policy update. Equations (1) and (2) are the iterative equations for value update and policy update, respectively.

$$V(s) := \sum_{s'} P_{\pi(s)}(s, s')(R_{\pi(s)}(s, s') + \gamma V(s')) \quad (1)$$

$$\pi(s) := argmax_a\{\sum_{s'} P(s'|s, a)\,(R(s'|s, a) + \gamma V(s'))\} \quad (2)$$

There are many variants in optimizing the MDP using the two steps, value iteration and policy interaction. The former approximate the value of the state and predict the policy using the state value while the latter perform step one only once for many iterations of step two.

However, when probability or reward are unknown, the MDP can be optimized using reinforcement learning which can solve MDP without knowing transition probability [6]. To be specific, the transition probability is needed in optimization through value and policy iterations, however, the state transition information is accessed through the interactions with simulation. The RL can be combined with general function approximators such as Deep Neural Network (DNN) to approximate more complex functions [7, 8].

There are many other models to represent dynamic system such as predictive state representation (PSRs) where it is introduced class of models for discrete-time dynamical systems by represent the state as a set of predictions of observable outcomes of experiments one can do in the system [9]. Evolution Strategies (ES) is also capability to be an alternative to MDP-based RL techniques

[10]. Nevertheless, we only focus on solving problem based on simple MDP framework in this case.

2.2 Reinforcement Learning

The RL algorithms are also known as approximate dynamic programming, or neuro-dynamic programming [2] because most of it utilizes the dynamic programming technique as it assumes no knowledge of simulation MDP. Therefore, most basic RL are modeled in the MDP framework.

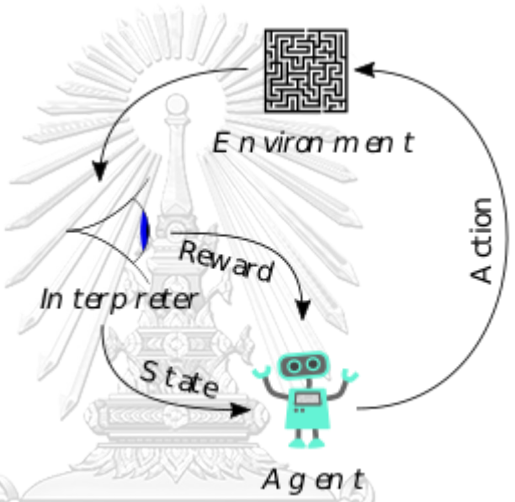

Figure 2: Basic framework of reinforcement learning: an agent takes an action based on observed state and value approximated from reward at a given discrete time

Figure 2 depicts the standard Reinforcement Learning interaction diagram where the agent interacts with simulation in the discrete time steps. In each step the agent observes the simulation and decides about the observation by choosing an action on the action space. The simulation then moves to the next state to complete the on step of sampled experience. The basic sample experiences are records with states, action, next states and immediate reward which is used in the learning process and the main optimization objective of RL is always to maximize the expected sum of rewards.

There are many concepts related to how to use past experiences to find out the optimal reward or to select the best actions such as value function and direct

policy search. But first we need to introduce the criterion of optimality and basic terminology in the optimization process.

The policy refers to the function that maps state to action of the agent which can be probabilistic or non-probabilistic selection. While state value function refers to the expected return of the state which is the expected sum of future discount of reward from the current state to the end as shown in Equation (3).

$$V_{\pi}(s) = E[R] = E[\sum_{t=0}^{\infty} \gamma^t r_t \,|s_0 = s] \quad (3)$$

2.2.1 Valued-based method

There are two main approaches to find optimal policy in MDP, namely, value-based, and policy-based reinforcement learning. The value-based or value function approach attempts to achieve the maximum expected sum of reward by estimating a set of expected return or value function given the situation. In this case, the action value function or Q-value is used to represent the estimated value of all given state action pairs. Hence, Q-value is a function of the expected sum of reward given state, action, and policy as shown in Equation (4).

$$Q^{\pi}(s, a) = E[R \,|s, a, \pi] \quad (4)$$

To find an optimal policy in the value-based approach, it can simply choose the action with the highest state value function given the state and action coordination.

Monte Carlo method is one of the methods to approximate action value function. The method approximates the value of the given state and action pair by average the sampled return from the state to the end. On the other hand, as shown in the right side of Figure 3, there is another method that is called “temporal difference learning” which learns the action value function by bootstrap from the current estimate of action value. In general, bootstrapping methods are faster to learn but it is not instances of true gradient decent, since the target depends on the weights to be estimated [6].

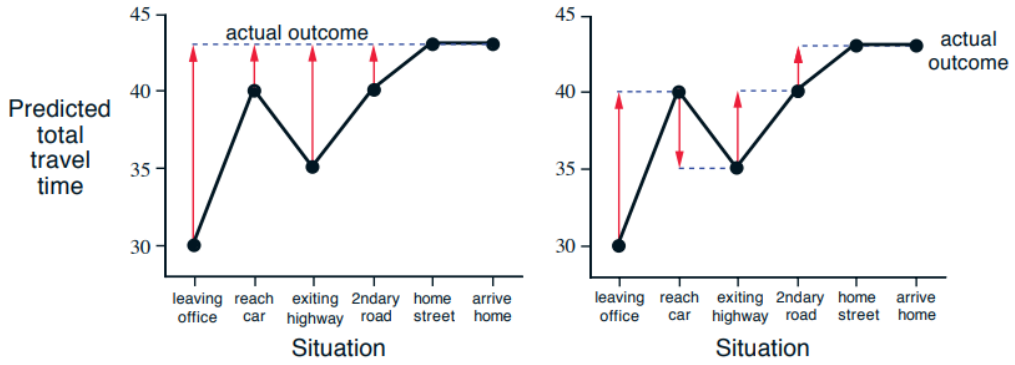

Figure 3: Comparison between Monte Carlo method (left) and temporal difference learning (right) [11]

The Monte Carlo method updates each prediction to the actual outcome which must wait to see the actual outcome first. While the temporal difference learning update at each state transition by using the sum of recorded value of the next state value and the discounted immediate reward as an update target. The Temporal Different Update is calculated as Equation (5).

$$V(s) \leftarrow V(s) + \alpha(r + \gamma V(s') - V(s)) \tag{5}$$

2.2.1 Policy-based method

Policy-based learning is another alternative method to optimize the MDP framework which is a subset of Direct policy search method. In direct policy search methods, there are two approaches which are gradient-based and gradient-free methods, the former utilize gradient to optimization while the latter search parameters by other methods such as random search.

The basic algorithm for this category is REINFORCE as shown in Equation (6) which uses the value of the state as a direction to optimization. There are many more algorithms built on top of this foundation. For example, Actor-Critic, as Equation (7), uses the standard maximum log likelihood multiplied with the estimated action-value of the state or Q-function to optimize the policy of neural networks. In addition, there are other variants such as Advantage Actor-Critic (A2C)

which is very popular and universally used as shown in Equation (8). More is TD Actor-Critic as Equation (9).

$$\nabla_\theta J(\theta) = E_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a) V_t\right] \tag{6}$$

$$\nabla_\theta J(\theta) = E_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a) Q^w(s,a)\right] \tag{7}$$

$$\nabla_\theta J(\theta) = E_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a) A^w(s,a)\right] \tag{8}$$

$$\nabla_\theta J(\theta) = E_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a) \delta\right] \tag{9}$$

2.2 Exploration Problem

In sample-based learning or reinforcement learning, the trade-off between exploration and exploitation of algorithms is well-known and always a challenge to any practitioners, especially in a very sparse reward.

The main objective of reinforcement learning algorithm is to find the best solution as soon as possible. However, it is not clear whether to find or optimize for best solution first. Although there are many modern reinforcement learning algorithms that can-do good exploitation efficiently, exploration issues are still an open topic.

Some have overcome this with other techniques such as demonstration method [12], hindsight experience [13] and Information seeking [14-16]. Back to the most basic case, the topic had been observed since the very basic multi-arm bandit problem [17] in which there is a limited resource which must be allocated to maximize the expected gain. The basic analogy can be displayed as in Figure 4, where the there are many slot machines and the octopus need to spend limit energy to roll an arm.

The problem outlines the decision of software agents to choose either explore new things to obtain new information or choose the best action based on known knowledge which is also known as exploitation. However, there is still a lack

of universally accepted exploration models that scale well with the large number of states.

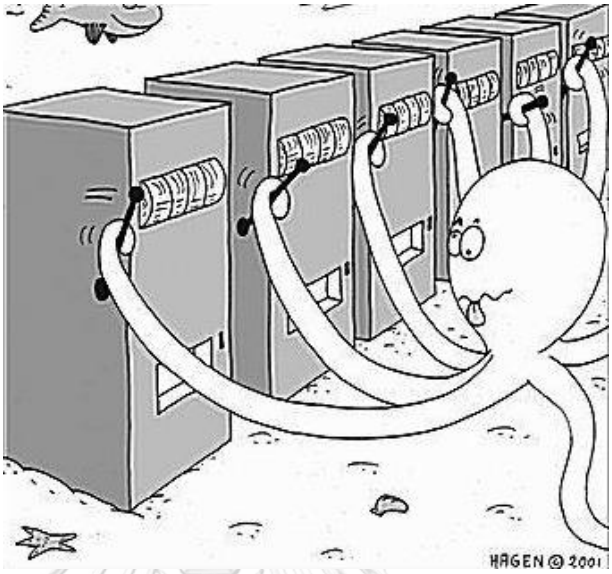

Figure 4: Intuitive picture of multi-arm bandit problem [6]

2.3 Environment

The environment implemented in the research consists of both single agent and multi agent environment or simulation. The former is used for quickly implementing and testing new ideas to make sure that it would work in standard setup. In this case, it is used for testing single agent RL. For the multi-agent environment, it is the main stage for conducting experiments and optimizing for performance improvement based on different architectures and parameters.

2.3.1 OpenAI Gym

OpenAI Gym is open source environments developed for RL research purposes [18]. It is well-known as widely used as a benchmark for testing and developing RL algorithms. The gym focuses on simple and standard game environments such as atari and often breaks down into episodic tasks. The

performance of the software agent is displayed in the in-game score where this is used to calculate objective function for optimization.

The gym is basically a single agent environment where the state transition happens when a single agent interacts with the environment. In some environments, the agent only perceives some part of it. Hence, this makes it a partially observable Markov decision process (POMDP). This is a perfect testing ground to try out the standard exploration tactic before moving on to the more complex multi-agent environment.

2.3.2 The StarCraft Multi Agent Challenge (SMAC)

The SMAC is a customized environment made of StarCraft 2 game engine for multi-agent research purposes which is built based on the SC2LE (StarCraft II Learning Environment), a reinforcement learning environment based on the game StarCraft II [19]. It provides an open source Python-based interface for communicating with the game engine. In the case of the SMAC, the environment is very complex, and it supports multi control to different individual units at the same time. Therefore, it supports many aspects of experiments and evaluation such as cooperative and competitive experiments as shown in Figure 5 where there are multiply units fighting together to win another group of units. Nevertheless, the goal of the SMAC is to achieve the highest team rewards which require agents to cooperate in complex strategy.

In SMAC, multi-agent algorithms control the red units while the in-game rule-based AI controls blue units as the opponent. The actions in the game is about skirmishing between two sides of units but there are some tactics the agent must master to be able to consistently win the fight.

Figure 5: Preview of the SMAC environments

2.4 Intrinsic Motivation

There are many researches contributed to the areas of exploration especially in complex and sparse reward environments. As mentioned earlier that there are no well-known standard solutions for exploration that would work at the scale of large state space real-world applications especially in the multi-agent setup yet. However, there are interesting fields of research on Intrinsic Motivation that might be the answer for the problems.

Intrinsic Motivation creates information-seeking behavior by using curiosity-driven characteristics which are different from task-dependent goal-directed behaviors [20]. This can be done by introducing internal reward functions that are based on maximizing other novel information such as curiosity. By combining with standard goal-directed behavior, the algorithm could potentially enhance exploration capability to encourage software agents to explore the unexplored states in the environment, and thus, achieving better results [21].

The basic concept of the curiosity-based intrinsic motivation algorithm simply creates the high intrinsic reward for unknown states of the environment while generating low intrinsic reward for the known state. This mimic the characteristic of curiosity in humans.

In general, there are two main methods: count-based and prediction-based intrinsic motivation. The count-based method refers to the algorithm that counts how many times that the agent visited the state and thus creates the intrinsic reward from the visited amount. However, for the prediction-based method such as in the case of the Random Network Distillation (RND), the algorithm creates intrinsic reward by implementing the scalable low overhead intrinsic motivation by using the prediction error of the two neural networks. This demonstrates an astonishing performance on Montezuma Revenge beating the average human performance [22]. The prediction-based Intrinsic Motivation is also applied in the work of Burda et al. [23] for large-scale study of curiosity-driven learning. The research experimented on OpenAI Atari games with no extrinsic rewards or extrinsic motivation at all.

Chapter 3

Methodology

In this chapter, we will drive deep into the technical details of each component that contributed to the final outcomes of the research. We will also show the baseline algorithms that are built in the previous published research papers that are strategically developed to build foundation toward the main research paper and the thesis.

3.1 Previous Research Methodology

The research aims to explore on the different types of Intrinsic reward of single agent reinforcement learning which is used as foundation to adapt to MARL in the main paper and this thesis. The title is named “Curiosity-Driven Exploration Effectiveness on Various Environments” because it’s evaluated the performance of different intrinsic motivation on a many single agent environment from OpenAI. The following are foundation concepts building toward the main experiments.

3.1.1 Random Network Distillation

As we briefly mentioned, the random network distillation (RND) is categorized under prediction-based curiosity driven method because it uses the error value between the two neural networks to represent the Intrinsic reward. Both networks are randomly initialized but one of them has fixed parameters [24, 25]. To find the intrinsic value, the state is fed forward to both networks to find the error between their outputs [24]. The objective function is the error between the two networks, and it can only minimize the error by adjusting parameters on the adjustable network. Figure 6 summarizes all the steps into a basic diagram of RND with single-agent environment.

The error is then used as an intrinsic motivation following the curiosity concept where the unknown state gives out more curiosity than the known one.

Therefore, the agent prefers to try actions in an unexplored state until they are familiar with the state. The benefit of applying this technique is that it incurs low overhead cost which is easy to scale on multi-agent setup.

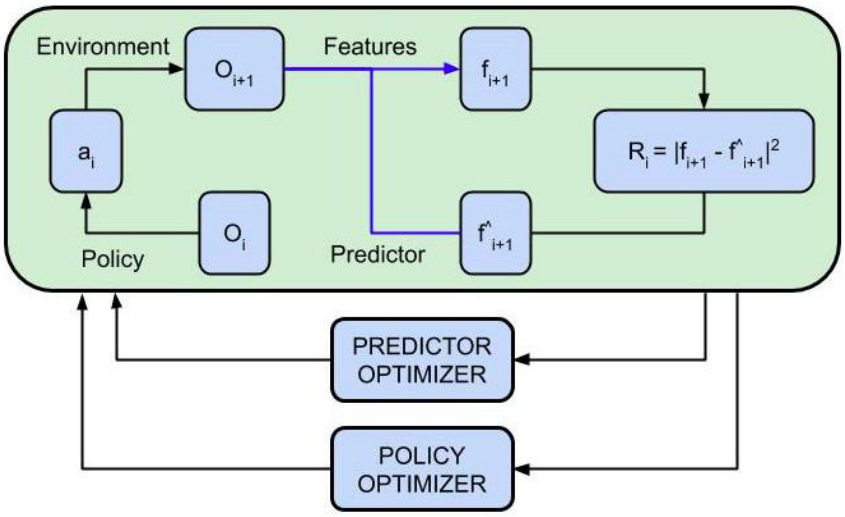

Figure 6: The first architecture of RND for single agent environment [24]

For the first architecture as shown in Figure 6, the observation states are fed through both neural network, feature and predictor networks. The predictive error between the two networks are quantified using Sum Square Error. The error is then back propagated to the Predictor Network parameters. In this case, the predictive error is used as an intrinsic motivation.

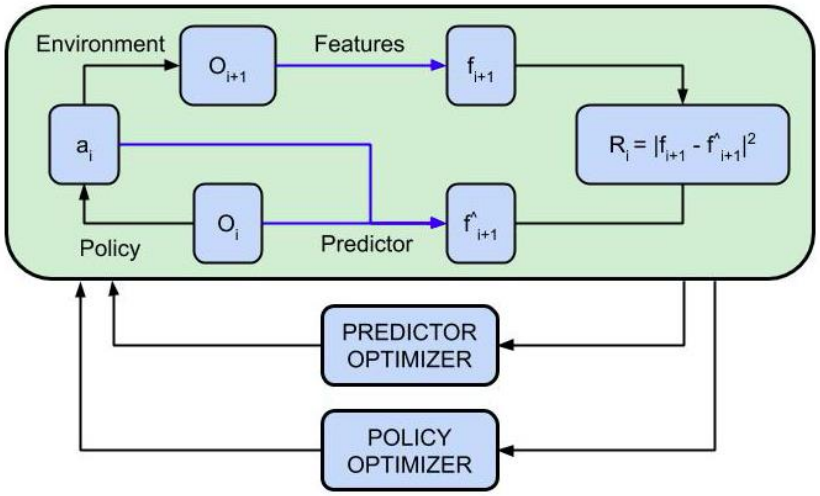

Figure 7: alternative architecture of RND for single agent environment [24]

For the other architecture as shown in Figure 7, the difference is that the predictor state uses the combination of previous observation and action instead of recent observation.

3.1.2 Reward Prediction Network

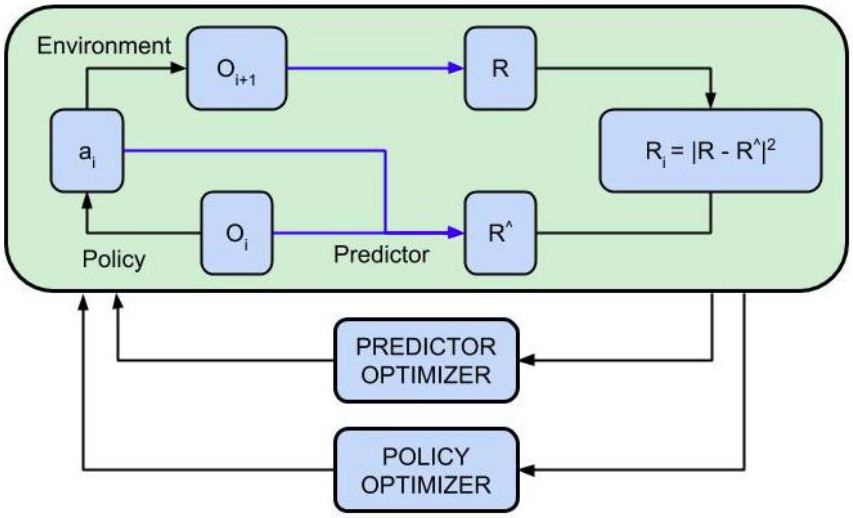

Figure 8: Reward Prediction Network

Figure 8 illustrates the model modified from the Reward Prediction Network shown in the work of Jaderberg et al. [26]. The concept is similar to the previous model but using previous state and observation to predict the immediate reward instead. The error of the prediction is used as intrinsic motivation. For the objective function, the prediction difference is used as an objective function to be minimized which can be done by adjusting the predictor network

3.1.3 Handcrafted Intrinsic Reward Noise

The Idea of the model is to improve the characteristics of exploration by introducing small noise which attenuated over time. The method is constructed as shown in Equation (10). For the Equation (11), it is the hard limit on the reward to ensure that it would never fall below zero which makes the logarithm term to be a negative number. Therefore, the intrinsic reward always equals or larger than zero.

$$Intrinsic\ Reward\ = N(0,0.1) * discount^{timesteps} * log(reward) \quad (10)$$

where

$$reward\ = max(episode\ reward, 1) \quad (11)$$

The method is a kind of random walk exploration as It did not use related environment information to guide the exploration. This is referred to as undirected exploration [27].

3.1.4 Count-based Exploration

This is another curiosity driven exploration which can generate intrinsic motivation based on the count of unique states which is called “Count-based exploration algorithms”. This is known to perform near-optimally when used in conjunction with tabular reinforcement learning (RL) such as Q-table. However, There is a surprising finding that a simple generalization of the count-based method could achieve near state-of-the-art performance on various high-dimensional RL benchmarks [22].

In this case, the experiments apply the algorithm presented in [28]. The states are divided into 16 bins for each dimension of the state space and are assigned the unique string as a label name to count. The equation is shown below as Equation (12) where N(S) denotes the count of the unique states.

$$Intrinsic\ Reward\ = \frac{0.1}{\sqrt{N(S)}} \quad (12)$$

3.1.5 OpenAI Environment

This section briefly describes all the environments related to the experiments necessary for the interpretation of the results of the paper.

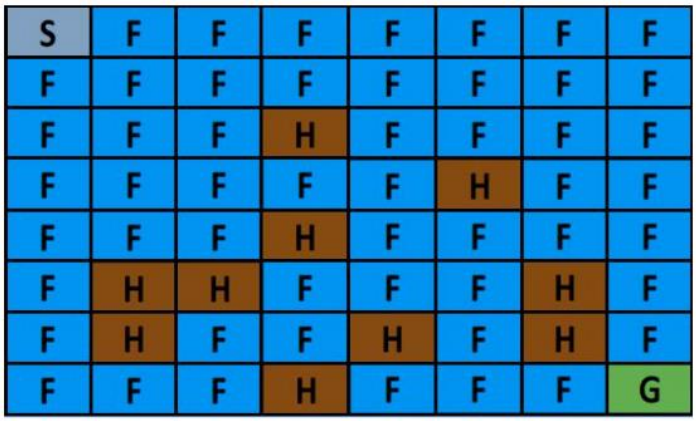

Figure 9: 8x8 FrozenLake Environment [29]

The first environment in Figure 9 is a 8x8 grid world which is called "FrozenLake8x8-v0". The main objective is to move from the starting point "S" to the finishing point "G" without falling to the Hole "H". A successful episode will be rewarded with +1 reward while the failure episode receives no reward. There are 64 possible state spaces on the 8x8 grid as it is 8x8 and only 1 combination of agent-based on the state. For the frozen lake, there are only 4 action spaces, move up, down, left, and right.

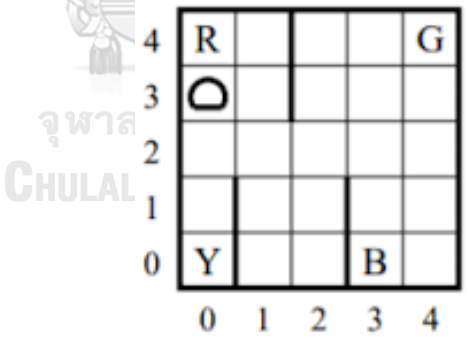

Figure 10: Taxi Game Environment [30]

The second environment is called "taxi" which is 5x5 grid world as shown in Figure 10. Still, there are many possible combination of elements such as position of the taxi (5x5), position of passenger (5 R, G,Y, B and on Taxi) and position of destination (R,G,Y,B). Hence, there are a total of 500 state spaces. For the action space, there are six in totals, left, right, up, down, pickup, drop off. The four specific

locations: R, G, Y, and B can be spawning points for passengers or a drop off destination. The objective is to pick up a passenger at a spawning location then drop them off at the destination to earn the reward which is +20 for a successful delivery.

For the third environment which is shown in Figure 11, it is called “MsPacman”. This is an arcade game in OpenAI environments and the gameplay is about controlling Pac-Man to eat all dots while escaping 4 colored ghosts. The large dot at the corner is a power bullet which turns ghosts into blue weak ghosts which allows Pac-Man to eat them for extra points. The array of 128-array integers which range from 0-255 is a state space and there are 4 action spaces which are up, down, left and right.

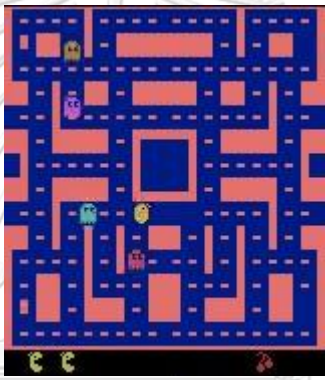

Figure 11: Taxi Game Environment [29]

3.1.6 Finding of Methodology

In literature, the Intrinsic Motivation is implemented with the single agent RL as shown in Figure 12.

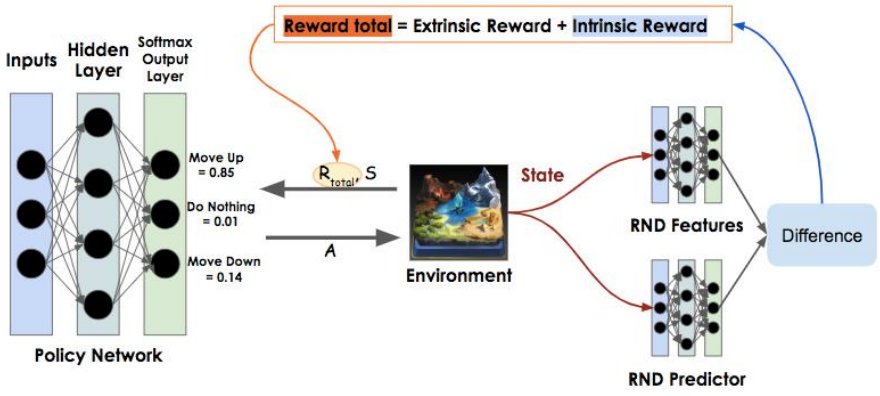

Figure 12: Single Agent Reinforcement Learning with Intrinsic Motivation

The performance of the experiments executed in different settings and environments is assessed with two dimensions measures: 1) sample efficiency (reward per episode run), and 2) average maximum reward. The sample efficiency is how many samples required to train the algorithm to reach optimal solution, while the average maximum reward shows the long-term potential which is how good of the algorithm to find optimal policy in the long term.

The visualization shows the reward or performance on the Y-axis and number of episodes on the X-axis. The sample efficiency can be observed by comparing average rewards on Y-axis with the same number of episodes on X-axis. In case of the average maximum reward, it is shown by comparing which score consistently has the highest score in the long run. There is a total of five exploration algorithms in the research. The following are descriptions of the graph colors: “BLUE” for the baseline without exploration, “GREEN” for RND using the recent state as an input, “RED” for RND using the previous state and action as inputs, “CYAN” for RPN Network, “MAGENTA” for handcrafted Intrinsic reward noise, and “YELLOW” for count-based exploration.

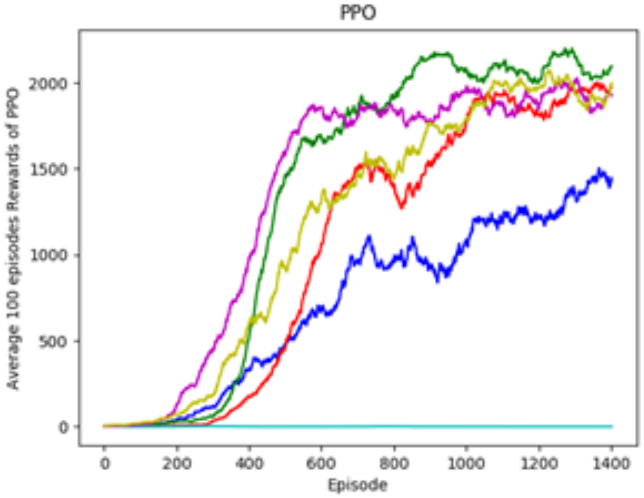

Figure 13: Experimental results on Taxi environment [31]

Figure 13 shows the baseline (blueline) is underperformed compared to most algorithms except the RPN. Both RND and Count-based algorithms display better performance in terms of both sample efficiency and average maximum reward. The

added intrinsic noise shows a better exploration value in terms of sample efficiency compared to the baseline and outperformed all other exploration techniques in the short run, but its effect does not reach the later steps of the game or it is still in suboptimal policy as the reward does not reach the top. On the other hand, the RND has potential to reach higher reward in the long run than the baselined method.

As we already seen, both count-based and prediction-based Intrinsic Motivation methods demonstrate solid improvement over the long period of time and allow the agent to seek for a more optimal policy. In detail, the RND model performs the best (Green line) as shown in the results.

3.1.6 Random Network Distillation from another research

We further investigate other scholar research that popularized the RND method to study the feasibility of applying RND in the multi agent model. The research is called Exploration by Random Network Distillation in which it aims to create an exploration algorithm on the very hard sparse reward environment like Montezuma's Revenge. The algorithm shows an astonishing performance with a proximal policy optimization algorithm which achieves the best reward in that category which shows a huge potential in the field of information-seeking type of agent [2]. The authors concluded that the RND introduce variants in the playstyles and decision making which improve the performance but not in the complex problem in the long run. Table 1 reported that in the complex environment such as Montezuma's Revenge or Venture, the RND exploration algorithm performs well and can achieve higher score than the others.

Table 1: The experiments of RND compared with State-of-the Art (SOTA) [24]

	Gravitar	Montezuma's Revenge	Pitfall!	PrivateEye	Solaris	Venture
RND	**3,906**	**8,152**	-3	8,666	3,282	**1,859**
PPO	3,426	2,497	0	105	3,387	0
Dynamics	3,371	400	0	33	3,246	1,712
SOTA	2,209[1]	3,700[2]	**0**	**15,806**[2]	**12,380**[1]	**1,813**[3]
Avg. Human	3,351	4,753	6,464	69,571	12,327	1,188

3.2 Proposed Methodology

In this section, we demonstrate the methodology used in the main research which is directly related to the topics of this thesis which is called "Multi-Agent Reinforcement Learning with Clipping Intrinsic Motivation". The paper illustrates many concepts and experiments using Intrinsic Motivation in Multi-Agent Setup. The following are the details of these methodologies.

3.2.1 The Starcraft Multi-Agent Challenge (SMAC)

This is the main environment used for multi agent reinforcement experiments due to the perfect property of this environment which is rich and complex for exploration. As mentioned earlier, the environment focuses on micromanaging part of the game and it is customized to allow individual unit control which is the main requirement for the experiment [32]. The SMAC offers a diverse set of challenge maps and recommendations for best practices in benchmarking and evaluations [33].

In detail, there are three different categories of skirmishes reported in the SMAC academic paper, symmetry, asymmetry and tactics. Symmetry refers to the fight that has an equal and identical unit. In order to win the fight, the algorithm must choose action slightly better than standard rule-based in-game AI engine. The asymmetry refers to the skirmishes that are more challenging by reducing our ally units to make handicaps for the enemy. Therefore, this requires better control to win the skirmish. Lastly, the tactic refers to the fight that requires game understanding in terms of clue or specific pattern and developing intelligent strategy in order to win the fight.

Table 2: List of SMAC environments

Name	Ally Units	Enemy Units
2s3z	2 Stalkers & 3 Zealots	2 Stalkers & 3 Zealots
3s5z	3 Stalkers & 5 Zealots	3 Stalkers & 5 Zealots
1c3s5z	1 Colossus, 3 Stalkers & 5 Zealots	1 Colossus, 3 Stalkers & 5 Zealots
5m_vs_6m	5 Marines	6 Marines
10m_vs_11m	10 Marines	11 Marines
27m_vs_30m	27 Marines	30 Marines
3s5z_vs_3s6z	3 Stalkers & 5 Zealots	3 Stalkers & 6 Zealots
MMM2	1 Medivac, 2 Marauders & 7 Marines	1 Medivac, 3 Marauders & 8 Marines
2s_vs_1sc	2 Stalkers	1 Spine Crawler
3s_vs_5z	3 Stalkers	5 Zealots
6h_vs_8z	6 Hydralisks	8 Zealots
bane_vs_bane	20 Zerglings & 4 Banelings	20 Zerglings & 4 Banelings
2c_vs_64zg	2 Colossi	64 Zerglings
corridor	6 Zealots	24 Zerglings

In our research, based on the experimental results, we selected some of the environments that would cover three categories which are 2s3z, 3s5z, 1c3s5z, 10m_vs_11m and 2s_vs_1sc as shown in Table 2. The first three are symmetry type, the fourth is asymmetry type and the last one is tactic type. The SMAC has 6 action spaces which are move up, move down, move left, move right, attack and idle. It is necessary for the active unit to choose one of the actions to transition to the next state which makes the environment following the Markov Decision Process.

To win the game, it is required to attack enemy units to reduce enemy hit point to zero before the opposite happens. The name of the map explicitly tells us about the unit in the game. The following are the brief descriptions for the unit in experiments. “s” stands for stalker which is a range unit, “sc” stands for spine crawler which is a power tower, “z” stands for zealot which is a powerful melee units, “c” stands for colossus which is a power area attack range unit, and “m” stands for marine which is weak range unit. These unit combinations and match up require a special technique of collaboration to win. For example, the 2s_vs_1sc requires both stalkers to take turns attacking the spine crawler that will result in its take turn attack stalkers. This causes the stalker to stay in game longer thus higher damage overall.

3.2.1 Actor-Critic

The Actor-Critic Built on the foundation of both policy and Value-based method as it used both elements from both methods. The method is basically a temporal different method that has a separate neural network where one represents the policy estimation and other approximates the value function of the state. The former is referred to as an actor and the latter is called a critic. The general Idea in creating actor-critic is that the current log-likelihood training are limited by the discrepancy between their training and testing modes, as the models generate tokens conditioned on their previous guesses rather than the ground-truth state. The method addresses this problem by using a critic network that is trained to predict the value of an output state [34]. Figure 14 shows the architecture of actor-critic algorithm which the actor part is called Policy and the critic part is called Value Functions.

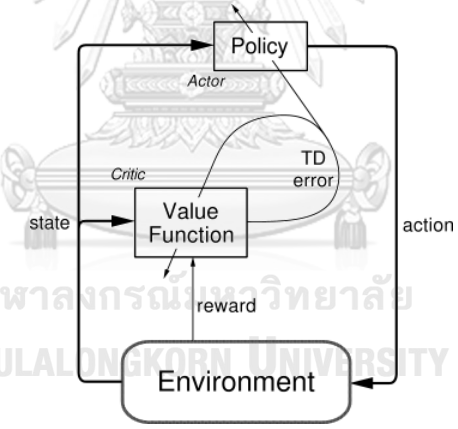

Figure 14: The actor-critic architecture

There are two significant advantages of combination between the two policy and value approaches. First, it scales very well with continuous-value action [35]. This is an advantage of using a policy method as the policy explicitly stores the action, so it does not require the search through all the action space to find the most valuable action. In addition to this, it can learn stochastic policy very well as it can learn the optimal policy of selecting various actions. Therefore, it is a good methodology to apply for a non-Nash equilibrium environment such as in the case of a competitive or cooperative environment as they are developing in the game

strategy. This is one of the advantages that may benefit applying on the multi-agent setup.

3.2.2 Counterfactual Multi-Agent Policy Gradient (COMA)

Multi-Agent Reinforcement Learning has many dimensions such as learning architecture, centralized or decentralized, or type of interaction, cooperative or competitive. One of the most popular paradigms for the Multi-Agent RL is in the centralized learning and decentralized execution where the agent shares their experiences in learning but acts independently to the best of their knowledge.

This brings us to the concept of the baseline algorithms built on top of actor-critic models and it is also implemented on the baseline environment research paper [32]. In the case of COMA, it is based on the actor-critic but with a slight modification on the critic assignment for value function. Instead of giving equal value to all the agents in the team, the critic assigns the value based on their contribution towards the team [36]. In detail, the critic neural network calculates agents' weighting by comparing the recent team value with using counterfactual baseline which is the team value when the agent is idle. As shown in Figure 15, there are multiply agents and the critic has to assign Advantage values to each particular actor based on the contribution.

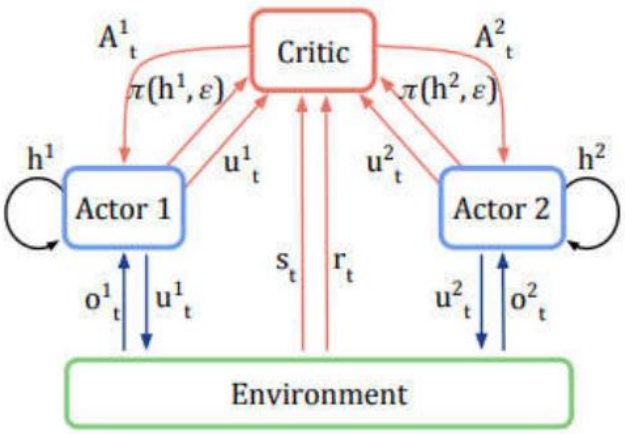

Figure 15: Counterfactual Multi-Agent Policy-Gradient Architecture

3.2.3 Intrinsic Advantage with Clipping Ratio

In the previous section, we mentioned the baseline environment and baseline architectures of the experiment. In this section, we would like to introduce the multi-agent architecture of intrinsic motivation and how we should apply it. In this case, we use the standard RND architecture from the single agent experiments and apply it to the multi-agent environment.

In this case, the notion of clipping intrinsic motivation is based on the intrinsic motivation which is generated from the RND network and the advantages value which is the terminology in Advantage Actor Critic (A2C) model. To begin with, the advantage value is the quantity of how better the action is compared to the others at a given state. The value is then used to optimize the policy network. In detail, the advantage is derived from the state action value, which is, in turn, derived from the reward. In this case, we would call it extrinsic advantage. To combine the intrinsic term, we should make it in the form of intrinsic advantage first which comes to the idea of clipping intrinsic motivation.

The idea is that we should scale the intrinsic advantage to the same scale or similar scale as the extrinsic advantage. Using Clipping method helps to clip the excessive amount to suit the needed level after having been normalized by other methods. Therefore, it allows intrinsic advantage components to never have too much impact on extrinsic advantage, and thus, appropriate for optimization. The steps of computing intrinsic advantage are as the following explanation.

The algorithm started with sampling a batch of experiences from the rollout gameplay with the environment. In batch, sampled extrinsic rewards are equivalent to episode length multiplied by the number of agents in parallel which is set to eight in this case. The sampled batch can only be used for a single iteration of optimization. After that, the intrinsic reward generated by RND is recorded in the batch in the same length. However, the intrinsic reward is scaled down by standard deviation of the batch as shown in the RND research paper [12].

As shown in Figure 16, the intrinsic advantage is obtained from the difference between the Intrinsic reward and the estimated baseline which is a prediction from another neural network. In detail, the predictor attempts to predict the scale down intrinsic reward by using observation as an input. Consequently, the intrinsic advantage is clipped by the product of extrinsic advantage of each individual agent and clipping ratio of that element in the rollout. In the last step, the total advantage is computed as the sum of intrinsic advantage and extrinsic advantage, where extrinsic advantages are the sum of all agent extrinsic values in case of Centralized Intrinsic Motivation, while separate for each agent in case of Individual Intrinsic Motivation.

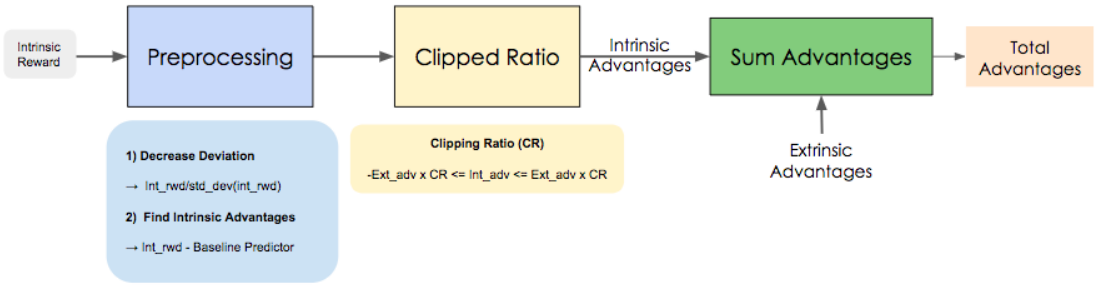

Figure 16: Intrinsic Advantage Generation Process

3.2.4 Individual Intrinsic Motivation Architecture (IIMA)

The IIMA is an extension version of the standard COMA with the built-in variable advantages. It is defined as a combination of both components, extrinsic and intrinsic reward. Previously, the single agent RND networks contained the Predictor and Target networks used to predict the intrinsic reward from the output deviation.

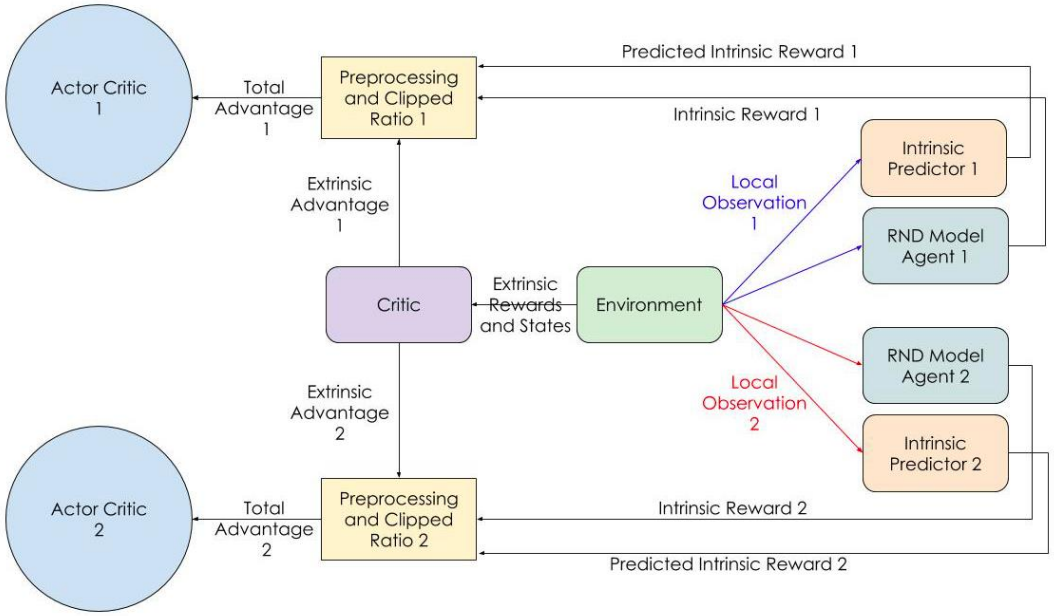

Figure 17: Individual Intrinsic Motivation Architecture

Based on IIMA Architecture (Figure 17), the individual set of RND networks is built for each individual agent, resulting in different intrinsic motivation advantages for each agent. In each IIMA, there are three neural networks, 2 for standard RND and 1 more to use as intrinsic predictor which is used to create a baseline to create intrinsic advantage by minus with Intrinsic Reward as shown in the previous pre-processing section. The IIMA is one version used to explore the multi-agent setup.

3.2.5 Centralized Intrinsic Motivation Architecture (CIMA)

The CIMA is built following the idea of centralized learning and decentralized execution paradigm which centralized the learning process off-policy in one place. In order to create the similar architecture with the intrinsic motivation, we combine all their observations to obtain an integrated observation and use it as an input to create the intrinsic reward as shown in Figure 18. The observations are concatenated together to form one large string of observation. It is expected that the architecture should be more suitable for multi-agent in the long run as it is built in alignment with the idea of the paradigm.

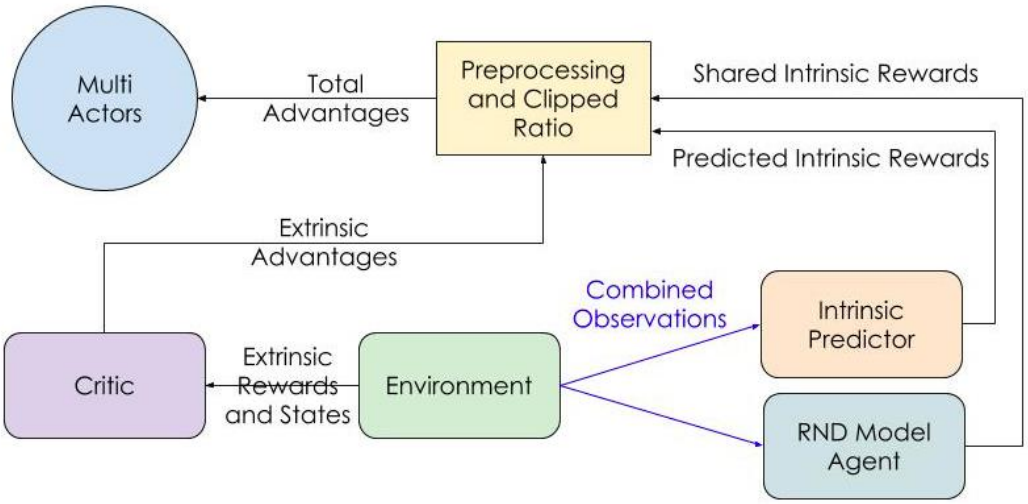

Figure 18: Centralized Intrinsic Motivation Architecture (CIMA)

Chapter 4

Experiments

The experiments demonstrated in the section are based on the previous setup aimed to answer the objectives of the thesis which is to develop a better algorithm using random network distillation on multi-agent reinforcement learning. In order to achieve the mentioned goal, we had been setting up the research paper that lay the foundation toward the thesis objective. In this case, we implement the following setups in our main research paper, "Multi-Agent Reinforcement Learning with Clipping Intrinsic Motivation" so that the results are aligned and can be used to clear our objectives.

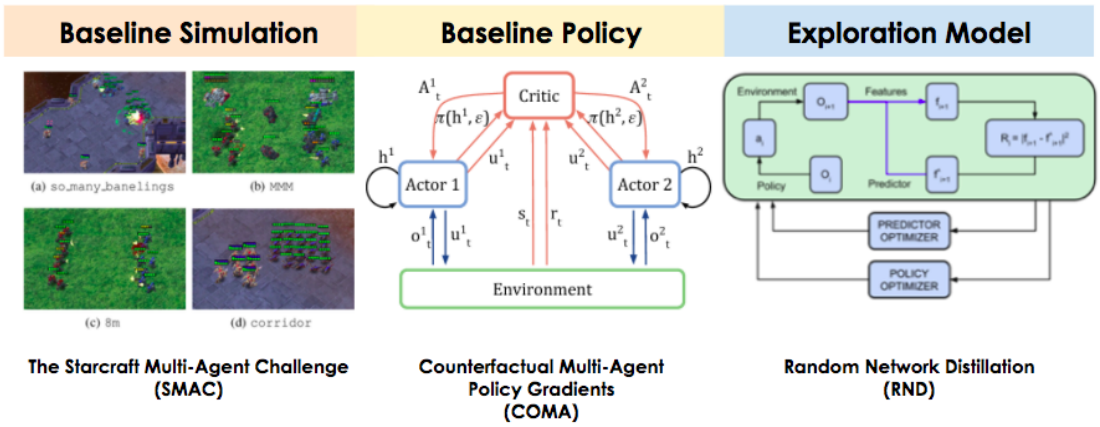

Figure 19: main components of thesis experiments

Figure 19 shows the three major components that are used to construct different testing scenarios to compare the effectiveness and performance in terms of both sample efficiency and long-term maximum score

To begin with, the experiments are implemented with different scenarios to compare the effectiveness of different architectures and clipping ratios. Based on the assumptions mentioned in the environments section, we aim to investigate 5 different simulations: 2s_vs_1sc, 2s3z, 3s5z, 1c3s5z, and 10m_vs_11m was carried out. There are 3 different scenarios: no intrinsic Motivation (green), Individual Intrinsic Motivation or IIM (blue), and Centralized Intrinsic Motivation or CIM (red). These

scenarios aim to compare the performances of each different intrinsic motivation architectures in multi-agent setup in combination with various clipping ratios: 0, 0.2, 0.5, 0.8, 1, 2 and 3 to find the optimal architecture on applying intrinsic motivation. The following are the experimental results from different categories.

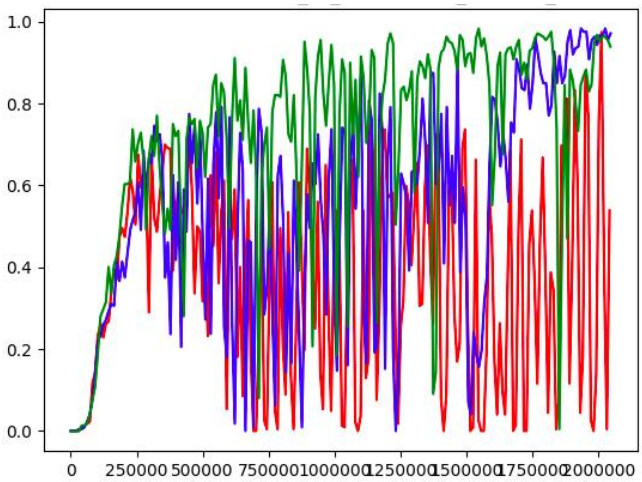

Figure 20: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 2s_vs_1sc with Clipping Ratio = 0

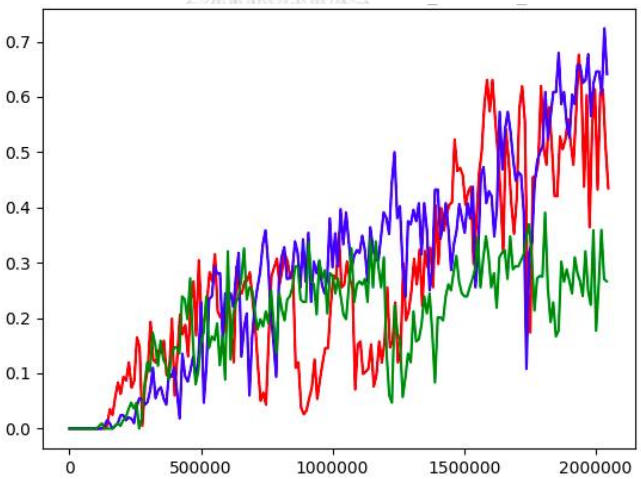

Figure 21: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 2s3z with Clipping Ratio = 0

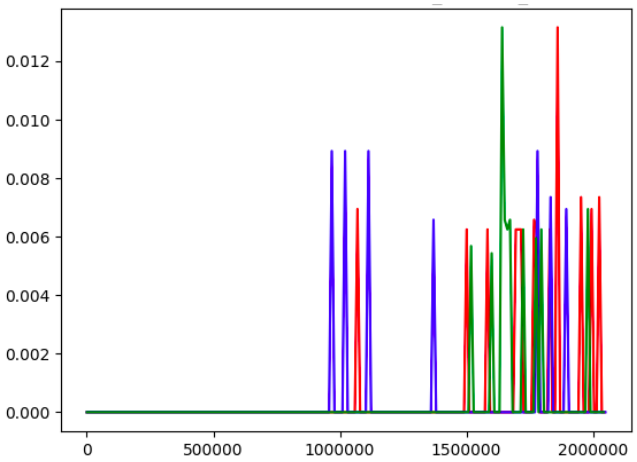

Figure 22: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 3s5z with Clipping Ratio = 0

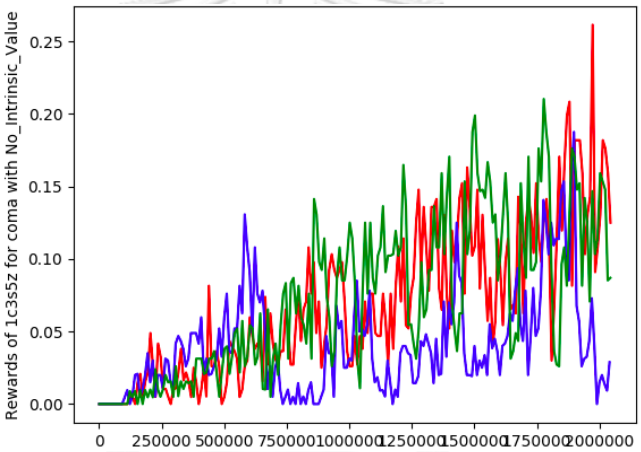

Figure 23: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 1c3s5z with Clipping Ratio = 0

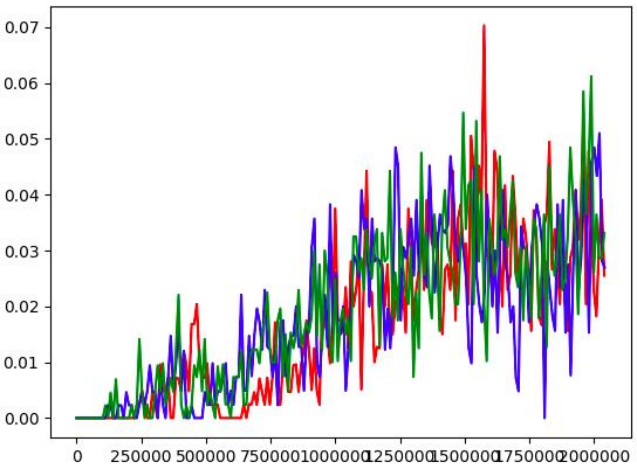

Figure 24: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 10m_vs_11m with Clipping Ratio = 0

For the clipping ratio equal to zero (Figure 20-24), the interpretation is that the intrinsic advantages are clipped to that zero times extrinsic advantages of that element in that roll out. Hence it means that the CIMA and IIMA in this case is basically equal to No Intrinsic Motivation. As you can see from the results, the results of all three lines are identical but due to the uncertainty of rollout, it may look a little bit different from each other. The difference will be clearer when the clipping ratio is not equal to zero.

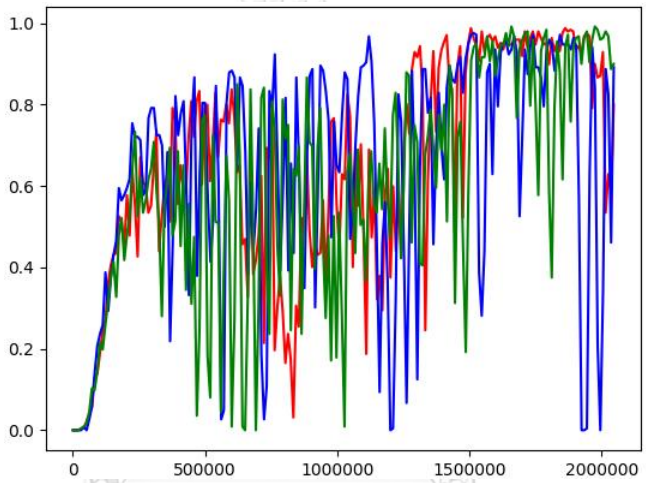

Figure 25: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 2s_vs_1sc with Clipping Ratio = 0.1

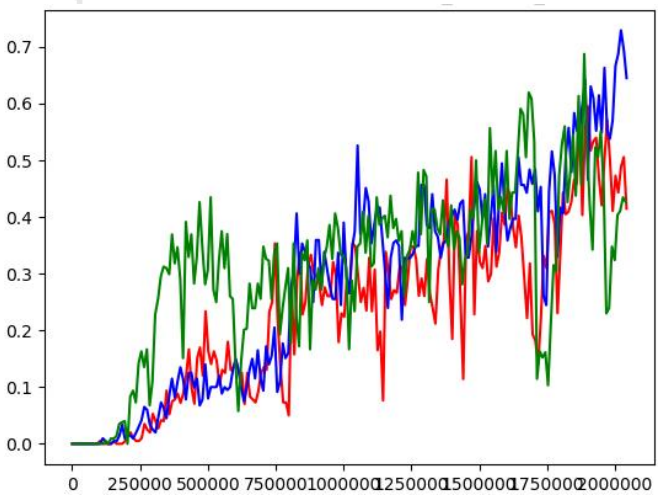

Figure 26: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 2s3z with Clipping Ratio = 0.1

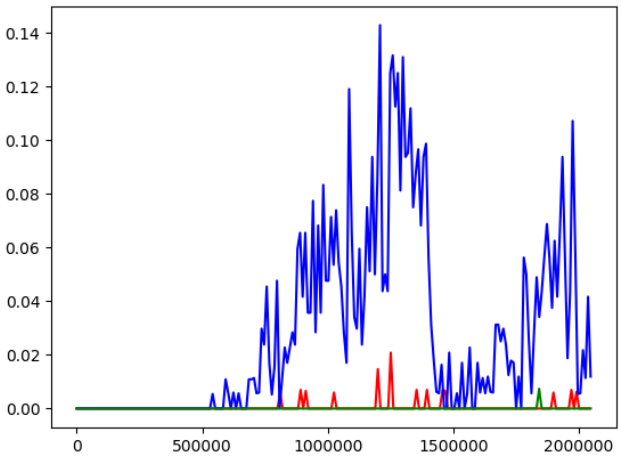

Figure 27: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 3s5z with Clipping Ratio = 0.1

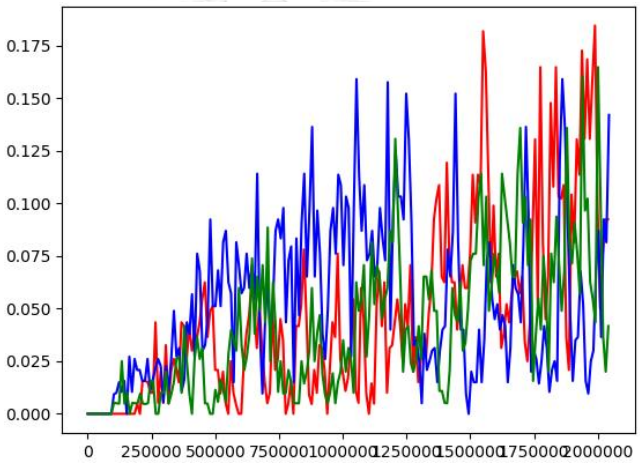

Figure 28: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 1c3s5z with Clipping Ratio = 0.1

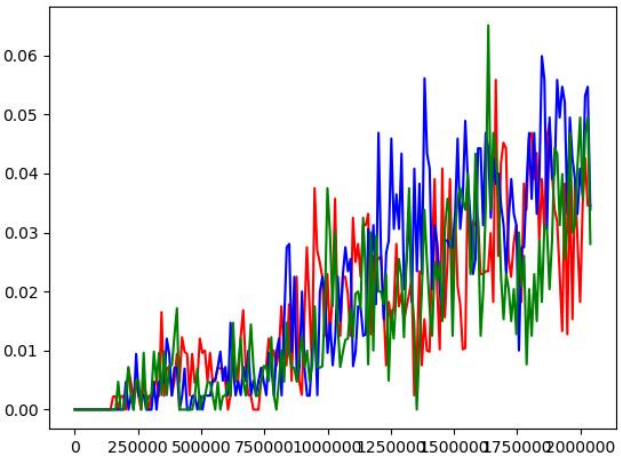

Figure 29: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 10m_vs_11m with Clipping Ratio = 0.1

The results with 0.1 clipping ratio are shown in Figures 25-29. Observing that IIMA and CIMA show an overall better performance than no intrinsic motivation in 2s3z and 3s5z. At the very low clipping ratio, the IIMA shines in exploration power reaching 70% peak performance in 2s3z and reaching up to a 14%-win rate in very hard environment 3s5z which outperform any other algorithms.

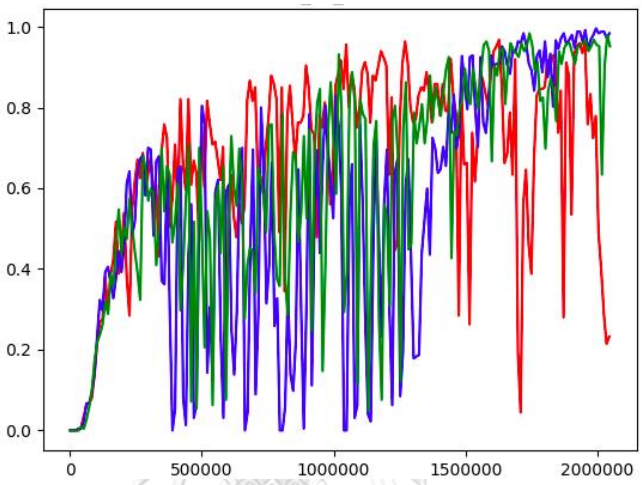

Figure 30: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 2s_vs_1sc with Clipping Ratio = 0.2

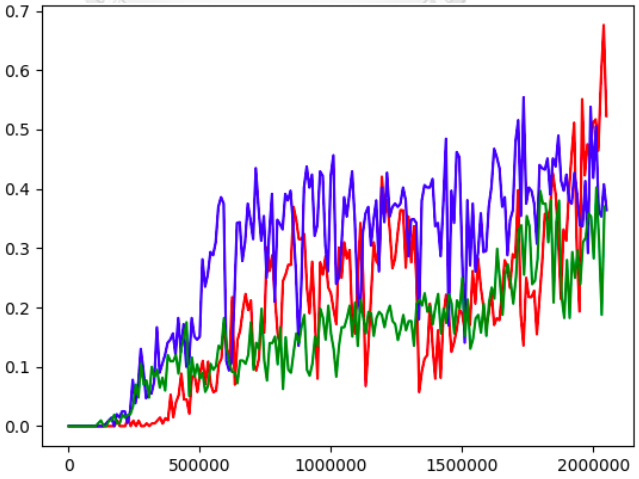

Figure 31: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 2s3z with Clipping Ratio = 0.2

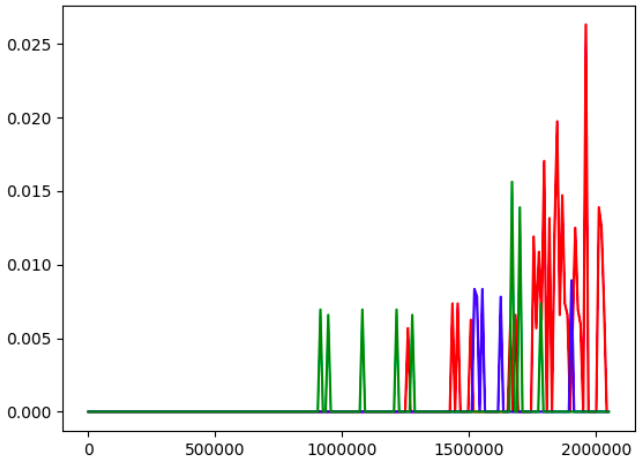

Figure 32: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 3s5z with Clipping Ratio = 0.2

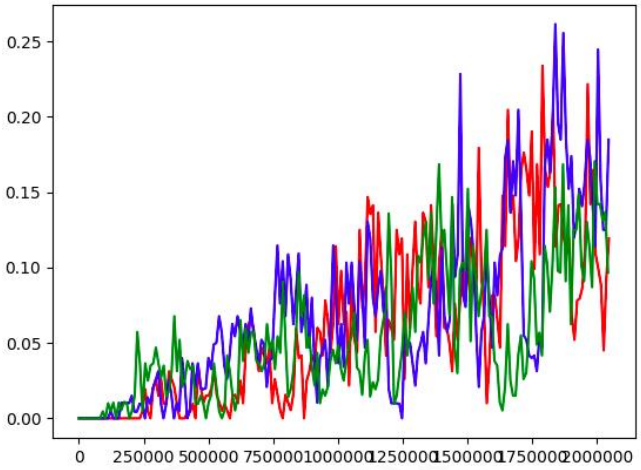

Figure 33: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 1c3s5z with Clipping Ratio = 0.2

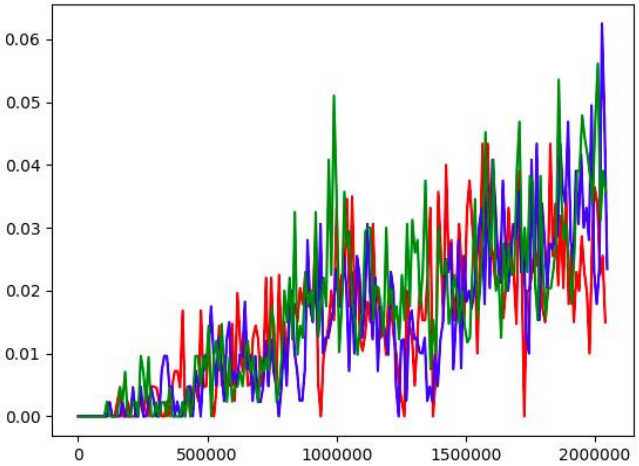

Figure 34: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 10m_vs_11m with Clipping Ratio = 0.2

Figures 30-34 show the results with 0.2 clipping ratio. Observing that the performance of IIMA and CIMA show an overall better performance than no intrinsic motivation. The CIMA shows a solid trend to outperform IIMA and no intrinsic motivation in the case of 2s3z and 3s5z which are a very difficult environment. Nevertheless, the CIMA shows a higher overhead cost as the performance ramps up very rapidly. In this case, the intrinsic motivation also shows a significant higher result than the benchmark especially in the map of 2s3z where the benchmark results only show 43% [19].

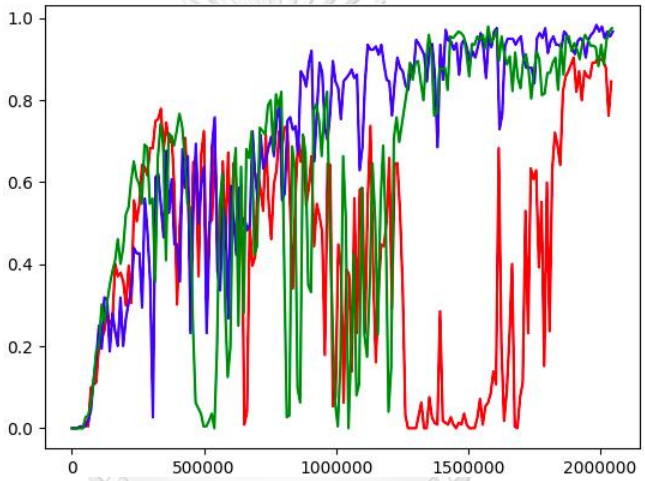

Figure 35: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 2s_vs_1sc with Clipping Ratio = 0.5

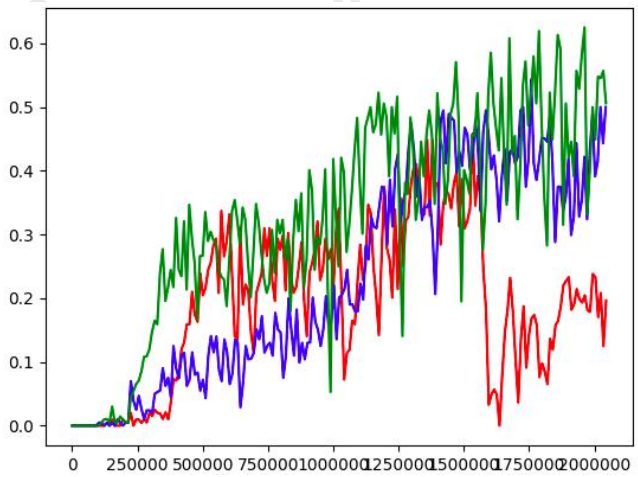

Figure 36: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 2s3z with Clipping Ratio = 0.5

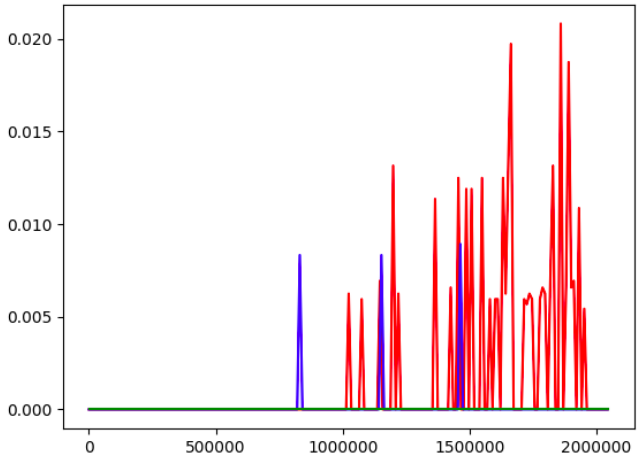

Figure 37: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 3s5z with Clipping Ratio = 0.5

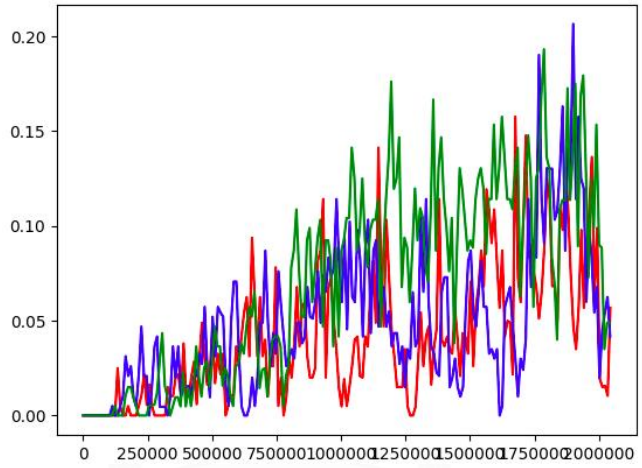

Figure 38: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 1c3s5z with Clipping Ratio = 0.5

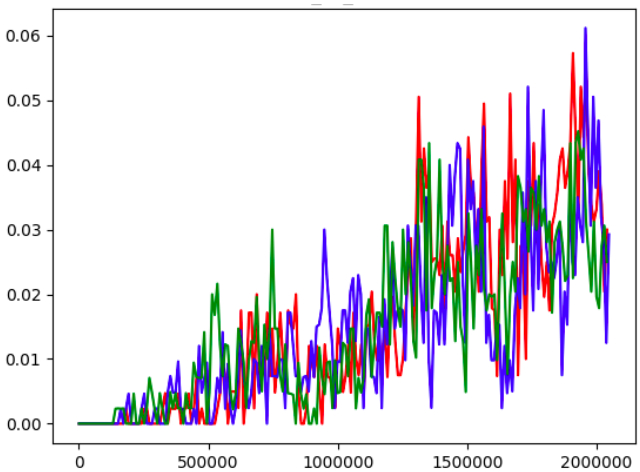

Figure 39: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 10m_vs_11m with Clipping Ratio = 0.5

In case of clipping ratio=0.5, the graphs (Figure 35-39) show that IIMA and CIMA may not outperform the standard benchmark unless the simulation is very difficult to explore in which the CIMA shows the win rate up to 2% instead of 0%. It is imperative that the higher clipping ratio or higher magnitude of exploration deteriorate the overall performance of the MARL as shown in the case of 2s3z. The CIMA shows a more powerful exploration as expected in the case of a difficulty simulation as shown in the 3s5z environment. Nevertheless, the CIMA confirm a higher overhead cost issues as it usually requires larger training episodes to achieve the same performance compared to the IIM and No intrinsic motivation in the easier environment.

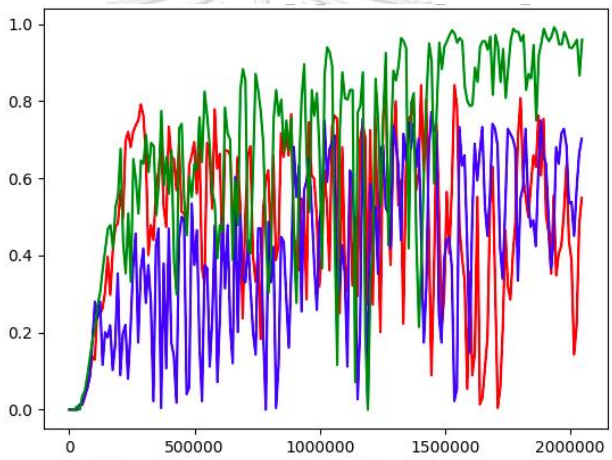

Figure 40: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 2s_vs_1sc with Clipping Ratio = 0.8

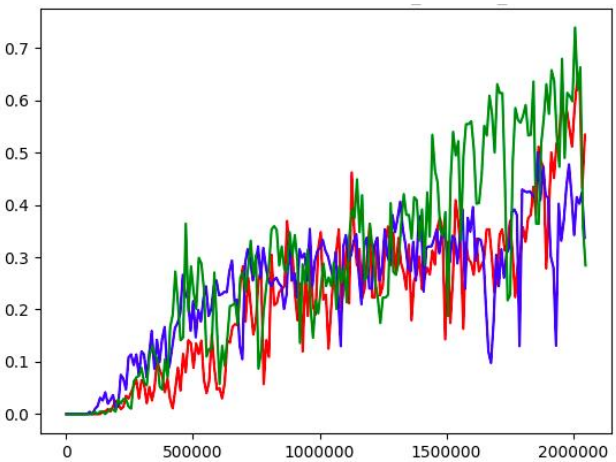

Figure 41: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 2s3z with Clipping Ratio = 0.8

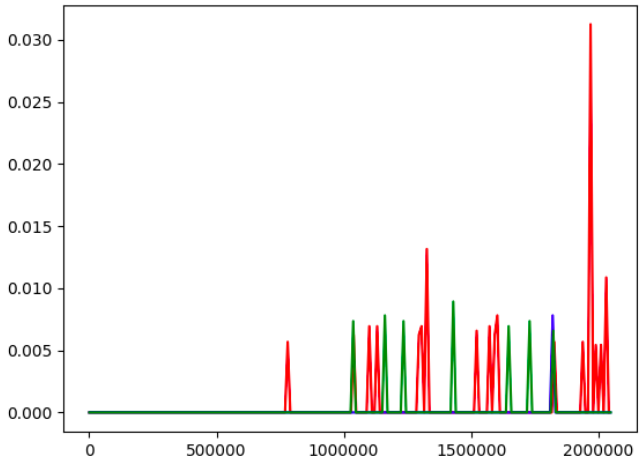

Figure 42: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 3s5z with Clipping Ratio = 0.8

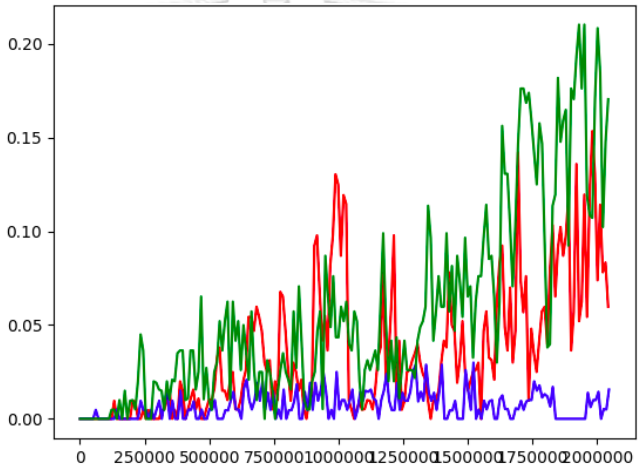

Figure 43: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 1c3s5z with Clipping Ratio = 0.8

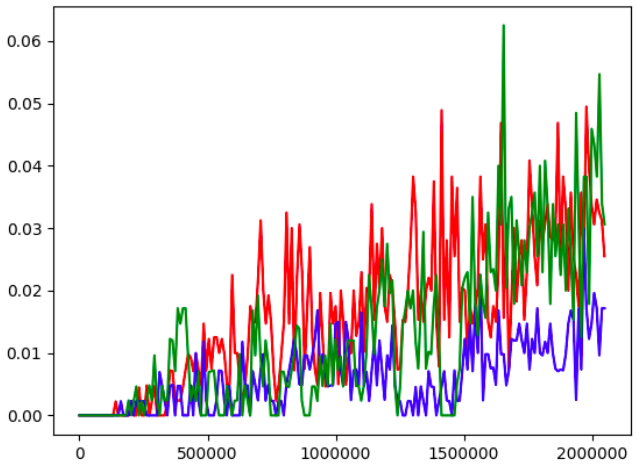

Figure 44: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of of 10m_vs_11m with Clipping Ratio = 0.8

In case of clipping ratio=0.8, the graphs (Figure 40-44) show that IIMA and CIMA show a significant drop in performance as the clipping ratio is increased. The trend is observable since the clipping ratio of 0.8. In addition, we observe that the IIMA is more sensitive to clipping ratio as the performance is dropped faster than those of CIMA.

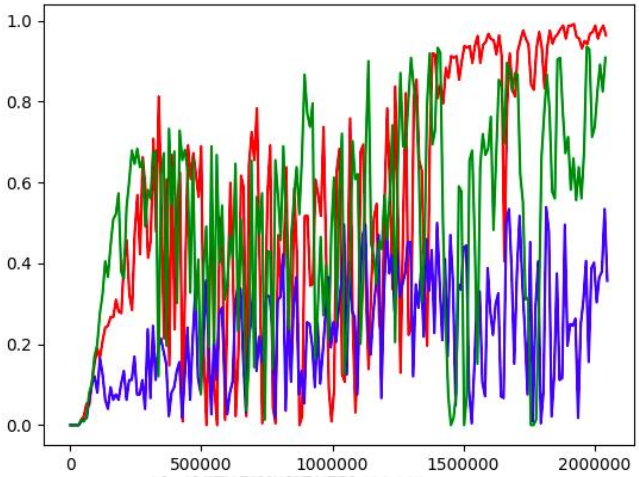

Figure 45: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 2s_vs_1sc with Clipping Ratio = 1.0

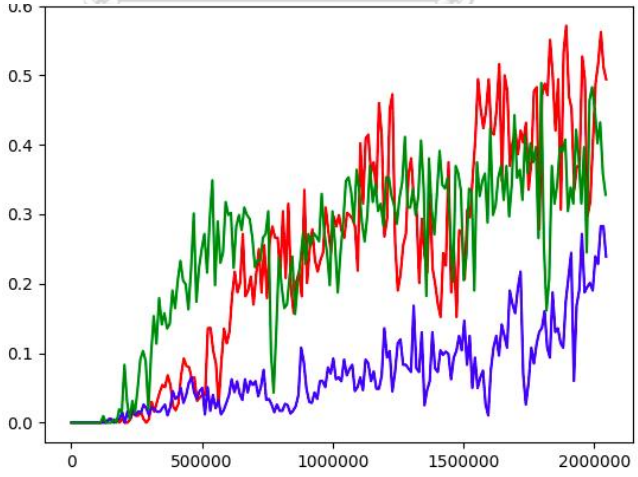

Figure 46: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 2s3z with Clipping Ratio = 1.0

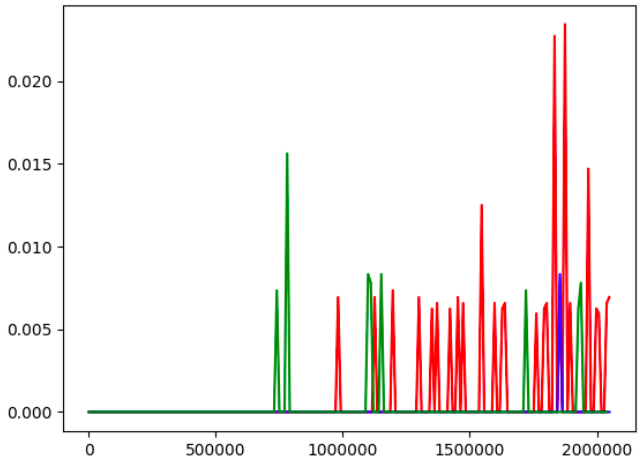

Figure 47: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 3s5z with Clipping Ratio = 1.0

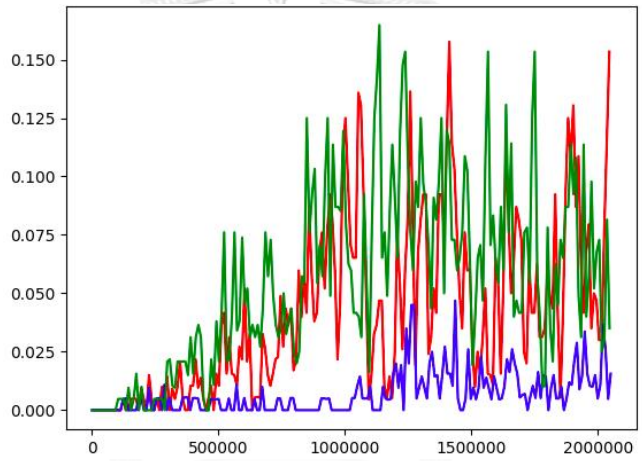

Figure 48: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 1c3s5z with Clipping Ratio = 1.0

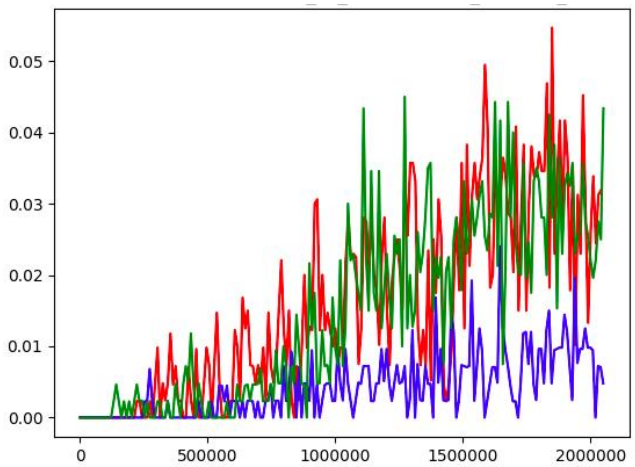

Figure 49: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 10m_vs_11m with Clipping Ratio = 1.0

In case of clipping ratio=1.0, the graphs (Figure 45-49) confirmed the deterioration of IIMA and CIMA performances especially in the case of IIMA which dropped at an astonishing degree.

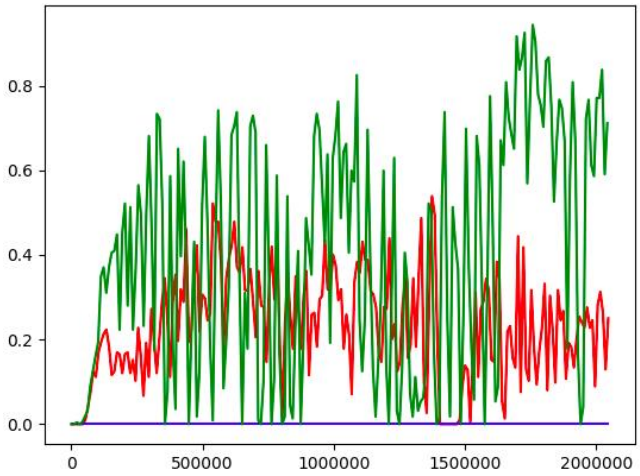

Figure 50: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 2s_vs_1sc with Clipping Ratio = 2.0

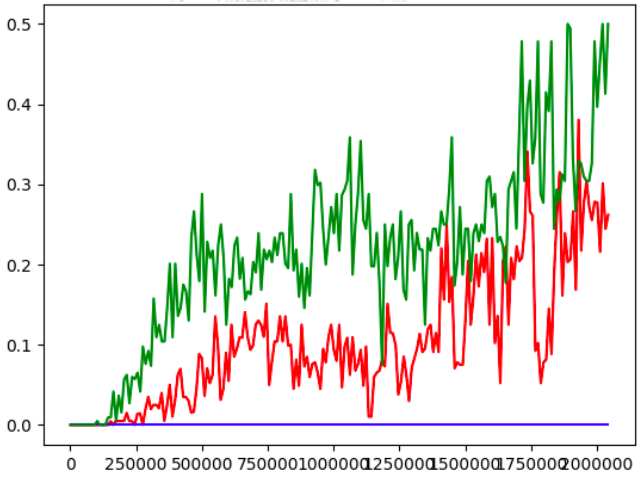

Figure 51: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 2s3z with Clipping Ratio = 2.0

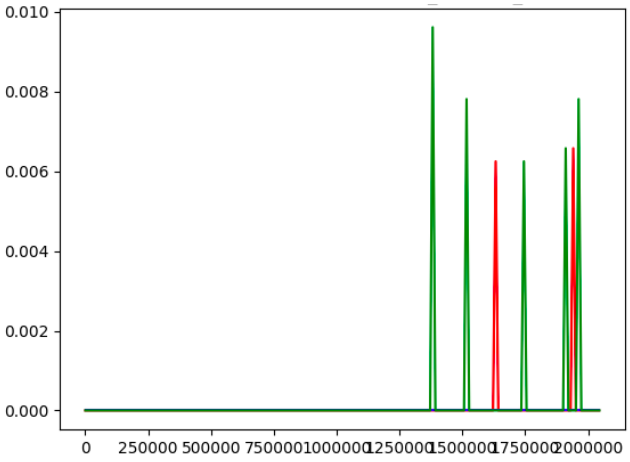

Figure 52: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 3s5z with Clipping Ratio = 2.0

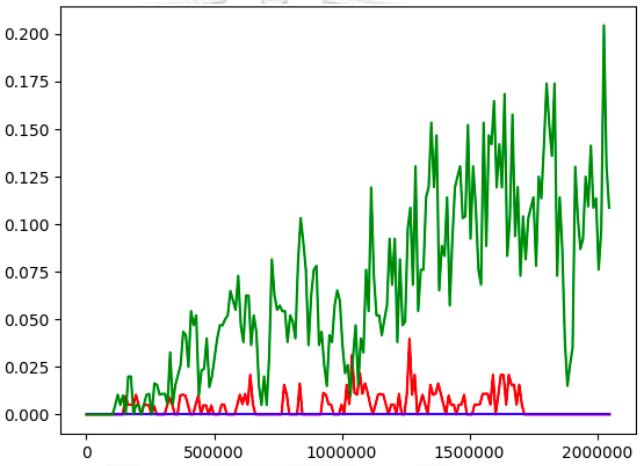

Figure 53: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 1c3s5z with Clipping Ratio = 2.0

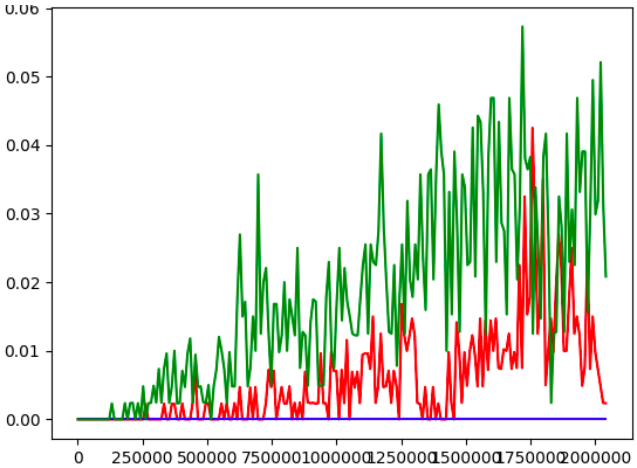

Figure 54: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 10m_vs_11m with Clipping Ratio = 2.0

In case of clipping ratio=2.0, the graphs (Figure 50-54) show that the IIMA is no longer performed in the high clipping ratio while CIMA performs significantly worse but still durable at a very high clipping ratio even though it does not add any exploration value anymore.

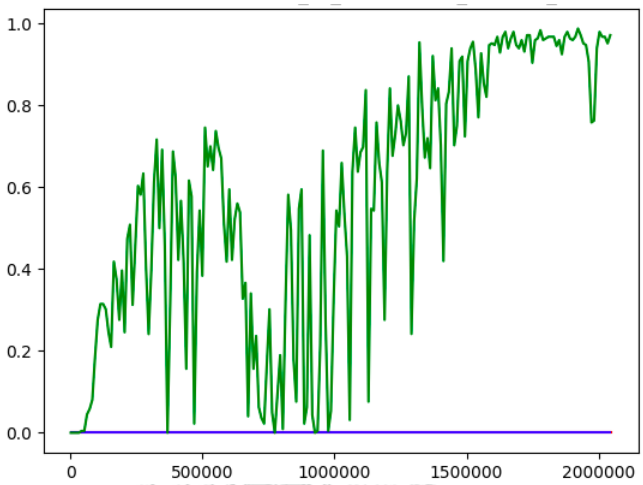

Figure 55: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 2s_vs_1sc with Clipping Ratio = 3.0

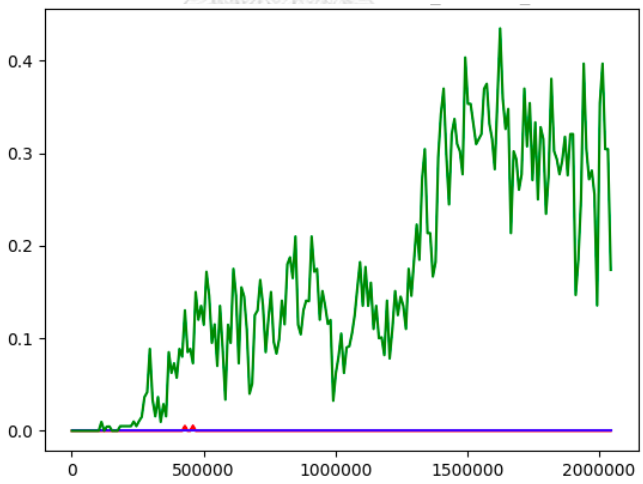

Figure 56: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 2s3z with Clipping Ratio = 3.0

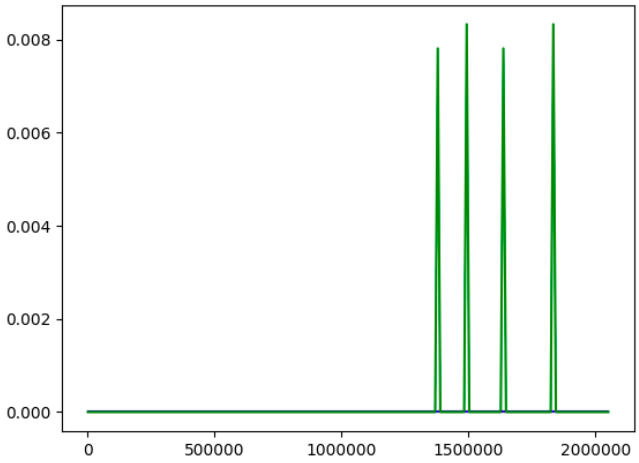

Figure 57: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 3s5z with Clipping Ratio = 3.0

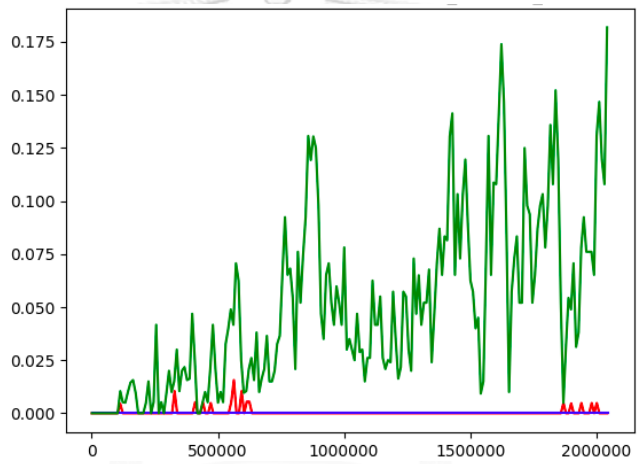

Figure 58: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 1c3s5z with Clipping Ratio = 3.0

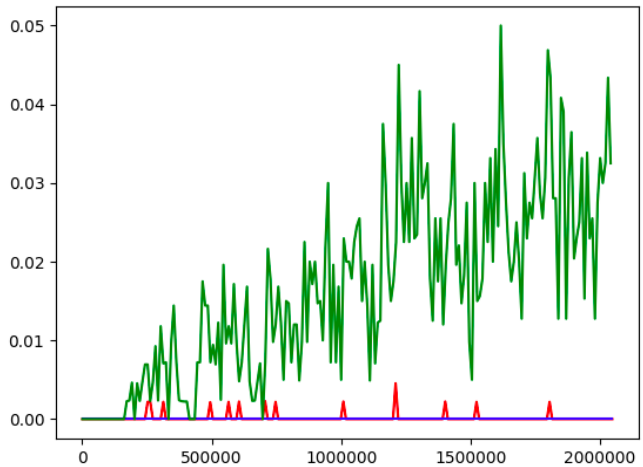

Figure 59: Win rate in percentage (Y-axis) vs Number of Episodes (X-axis) of 10m_vs_11m with Clipping Ratio = 3.0

In case of clipping ratio=3.0, the graphs (Figure 55-59) confirmed that both architectures, CIMA and IIMA is not performing at a very high clipping ratio. Interestingly that the decay of RND by itself may not be enough to clear out intrinsic components before two million episodes.

Chapter 5

Conclusion

The research had explored the performance of Intrinsic Motivation on Multi-agent reinforcement learning in 2 different dimensions, architecture, and clipping ratio. The former introduced the new architecture while the latter enforces the limit on optimization magnitude. In detail of the first dimension, the intrinsic motivation of the multi-agent architecture is constructed in two different architectures: Individual Intrinsic Motivation Architecture (IIMA) and Centralized Intrinsic Motivation Architecture (CIMA). On the other hand, due to unknown characteristics of magnitude of intrinsic motivation from many agents, the limit which is in forms of clipping ratio based on the extrinsic reward is introduced to limit the excessive magnitude that may cause instability to the optimization. In this case, we experiment on 8 clipping ratios which are: 0, 0.1, 0.2, 0.5, 0.8, 1, 2 and 3 to find the optimal solution for the architectures.

Table 3: Best Results in percentage of different architectures and clipping ratio compared with the baselines [32]

	COMA (Baseline)	No Intrinsic Motivation (COMA)	IIMA	IIMA Clipping Ratio	CIMA	CIMA Clipping Ratio
2s_vs_1sc	98	98	99	0.2	98	0.1
2s3z	43	55	70	0.1	65	0.2
3s5z	1	1	10	0.1	2.5	0.2
1c3s5z	31	17	20	0.2	17	0.1
10m_vs_11m	7	4	5	0.1	5	0.5

Note: The Baseline performance shows the final median performance (maximum median across the testing intervals within the last 250k of training) of the algorithms tested.

The results in table 3 show the win rate improvement over the benchmark counterfactual multi-agent policy gradient (COMA) in the many complex

environments. The more complex, the better value for intrinsic motivation as it helps improve exploration on an unknown state which sometimes allows it to move out of the local optimal path. In this case, the CIMA achieved almost 70%-win rate on 2s3z with 0.2 clipping ratio and the IIMA could even reach 70% point with 0.1 clipping ratio while the benchmark record is only 43% [32]. In addition, in case of very complex environment of 3s5z, the IIMA could potentially ramp win-rate up to 14% while the baseline reaches the 5%-win rate mark. This illustrate the potential of intrinsic Motivation on very complex environment which can be extended to multi-agent setup. Although the optimal clipping ratio is not yet explored, we do know that the appropriate scale of intrinsic advantage could help improve overall long run performance of the algorithm while the inappropriate magnitude or clipping ratio shows the opposite.

It is known that the main optimization objective relies on the designed reward function in terms of extrinsic motivation. Hence, it is intuitive that the magnitude of extrinsic motivation should be more dominant compared to intrinsic motivation in the long run, especially at the end. Although the random network distillation supports the decay of Intrinsic Motivation based on the curiosity characteristic, the characteristic alone may still not be enough. The evidence supports that high clipping ratio is not appropriate but low clipping ratio shows an improvement over the baseline. This confirms the value of clipping ratio in such a high complexity environment such as a multi-agent environment.

To illustrate in more detail, the value of clipping ratio greater than one always shows a significant drop in performance for both CIM and IIM because it allows intrinsic motivation to have a greater magnitude than extrinsic motivation which makes noise bigger than the main objective function. Whereas setting intrinsic ratio to zero in either CIM or IIM means the case of none of intrinsic motivation. Based on the results, a small amount of intrinsic motivation on MARL is the sweet spot to improve exploration. Even Though the optimal solution is yet to be explored

on the Multi-agent setup, the findings help lay the foundation of how to apply and scale intrinsic motivation for other research in the future.

Further investigation on other clipping ratios or some other aspects rather than win rate may reveal more underlying insight of how to apply such a novel information-seeking method on different categories of multi-agent setup. Other research on distributed architecture and the study on impact of cooperative or competitive agents with intrinsic motivation may further advance the research in the area significantly.

REFERENCES

[1] C. Liu, X. Xu, and D. Hu, "Multiobjective reinforcement learning: A comprehensive overview," *IEEE Transactions on Systems, Man, and Cybernetics: Systems,* vol. 45, no. 3, pp. 385-398, 2014.

[2] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," *arXiv preprint arXiv:1801.01290,* 2018.

[3] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 16-17.

[4] "Markov decision process " Wikipedia. https://en.wikipedia.org/wiki/Markov_decision_process (accessed 1 July 2020).

[5] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[6] Y. Li, "Deep reinforcement learning: An overview," *arXiv preprint arXiv:1701.07274,* 2017.

[7] S. Liang and R. Srikant, "Why deep neural networks for function approximation?," *arXiv preprint arXiv:1610.04161,* 2016.

[8] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," *Neural Networks,* vol. 107, pp. 3-11, 2018.

[9] S. Singh, M. James, and M. Rudary, "Predictive state representations: A new theory for modeling dynamical systems," *arXiv preprint arXiv:1207.4167,* 2012.

[10] T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever, "Evolution strategies as a scalable alternative to reinforcement learning," *arXiv preprint arXiv:1703.03864,* 2017.

[11] "Temporal-Difference Methods in Reinforcement Learning." Medium. https://towardsdatascience.com/my-journey-into-reinforcement-learning-part-5-temporal-difference-learning-d0cae79e850 (accessed 1 July, 2020).

[12] A. Nair, B. McGrew, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Overcoming exploration in reinforcement learning with demonstrations," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018: IEEE, pp. 6292-6299.

[13] M. Andrychowicz *et al.*, "Hindsight experience replay," in *Advances in neural information processing systems*, 2017, pp. 5048-5058.

[14] D. Trendafilov and D. Polani, "Information-theoretic Sensorimotor Foundations of Fitts' Law," in *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*, 2019, pp. 1-6.

[15] C. Colas, P. Fournier, M. Chetouani, O. Sigaud, and P.-Y. Oudeyer, "CURIOUS: intrinsically motivated modular multi-goal reinforcement learning," in *International conference on machine learning*, 2019, pp. 1331-1340.

[16] M. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, and R. Munos, "Unifying count-based exploration and intrinsic motivation," in *Advances in neural information processing systems*, 2016, pp. 1471-1479.

[17] A. N. Burnetas and M. N. Katehakis, "Optimal adaptive policies for Markov decision processes," *Mathematics of Operations Research,* vol. 22, no. 1, pp. 222-255, 1997.

[18] G. Brockman *et al.*, "Openai gym," *arXiv preprint arXiv:1606.01540,* 2016.

[19] O. Vinyals *et al.*, "Starcraft ii: A new challenge for reinforcement learning," *arXiv preprint arXiv:1708.04782,* 2017.

[20] A. G. Barto, "Intrinsic motivation and reinforcement learning," in *Intrinsically motivated learning in natural and artificial systems*: Springer, 2013, pp. 17-47.

[21] A. Aubret, L. Matignon, and S. Hassas, "A survey on intrinsic motivation in reinforcement learning," *arXiv preprint arXiv:1908.06976,* 2019.

[22] H. Tang *et al.*, "# exploration: A study of count-based exploration for deep reinforcement learning," in *Advances in neural information processing systems*, 2017, pp. 2753-2762.

[23] Y. Burda, H. Edwards, D. Pathak, A. Storkey, T. Darrell, and A. A. Efros, "Large-scale study of curiosity-driven learning," *arXiv preprint arXiv:1808.04355,* 2018.

[24] Y. Burda, H. Edwards, A. Storkey, and O. Klimov, "Exploration by random network

distillation," *arXiv preprint arXiv:1810.12894,* 2018.

[25] C. Yang, J. Yang, X. Wang, and B. Liang, "Control of Space Flexible Manipulator Using Soft Actor-Critic and Random Network Distillation," in *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2019: IEEE, pp. 3019-3024.

[26] M. Jaderberg *et al.*, "Reinforcement learning with unsupervised auxiliary tasks," *arXiv preprint arXiv:1611.05397,* 2016.

[27] S. B. Thrun, "Efficient exploration in reinforcement learning," 1992.

[28] B. Baker *et al.*, "Emergent tool use from multi-agent autocurricula," *arXiv preprint arXiv:1909.07528,* 2019.

[29] G. OpenAI, "A toolkit for developing and comparing reinforcement learning algorithms," ed.

[30] R. Houthooft, X. Chen, Y. Duan, J. Schulman, F. De Turck, and P. Abbeel, "Vime: Variational information maximizing exploration," in *Advances in Neural Information Processing Systems*, 2016, pp. 1109-1117.

[31] K. Charoenpitaks and Y. Limpiyakorn, "Curiosity-Driven Exploration Effectiveness on Various Environments," in *Proceedings of the 3rd International Conference on Vision, Image and Signal Processing*, 2019, pp. 1-6.

[32] M. Samvelyan *et al.*, "The starcraft multi-agent challenge," *arXiv preprint arXiv:1902.04043,* 2019.

[33] T. Rashid *et al.*, "The StarCraft Multi-Agent Challenge," 2019.

[34] D. Bahdanau *et al.*, "An actor-critic algorithm for sequence prediction," *arXiv preprint arXiv:1607.07086,* 2016.

[35] A. Bhardwaj, W. Di, and J. Wei, *Deep Learning Essentials: Your hands-on guide to the fundamentals of deep learning and neural network modeling*. Packt Publishing Ltd, 2018.

[36] J. N. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Thirty-second AAAI conference on artificial intelligence*, 2018.

VITA

NAME	Korawat Charoenpitaks
DATE OF BIRTH	2 April 1989
PLACE OF BIRTH	Bangkok
INSTITUTIONS ATTENDED	Bachelor of Engineering: Electrical Engineering Curtin University Master of Commerce: Applied Finance Curtin University
HOME ADDRESS	118/11 Chalermprakiat Ror 9 Rd. Nongbon Prawet Bangkok 10250
PUBLICATION	1) K. Charoenpitaks and Y. Limpiyakorn, "Curiosity-Driven Exploration Effectiveness on Various Environments," in Proceedings of the 3rd International Conference on Vision, Image and Signal Processing, 2019, pp. 1-6. 2) Chaoenpitaks, K., and Limpiyakorn, Y, “Multi-Agent Reinforcement Learning with Clipping Intrinsic Motivation,” International Journal of Machine Learning and Computing (IJMLC), to appear.