MULTI-AGENT LEARNING IN COVERAGE CONTROL GAMES

by

Mohammadhosein Hasanbeig

A thesis submitted in conformity with the requirements
for the degree of Master of Science
Graduate Department of Electrical and Computer Engineering
University of Toronto

Abstract

Multi-agent Learning in Coverage Control Games

Mohammadhosein Hasanbeig

Master of Science

Graduate Department of Electrical and Computer Engineering

University of Toronto

2016

Multi-agent systems have found a variety of industrial applications from economics to robotics. With the increasing complexity of multi-agent systems, multi-agent control has become a challenging problem in many areas. While studying multi-agent systems is not identical to studying game theory, there is no doubt that game theory can be a key tool to manage such complex systems. Game theoretic multi-agent learning is one of relatively new solutions to the complex problem of multi-agent control. In such learning scheme, each agent eventually discovers a solution on his own. The main focus of this thesis is on enhancement of multi-agent learning in game theory and its application in multi-robot control. Each algorithm proposed in this thesis, relaxes and imposes different assumptions to fit a class of multi-robot learning problems. Numerical experiments are also conducted to verify each algorithm's robustness and performance.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisor, Professor Lacra Pavel, for her deep insight, wisdom, invaluable guidance, and limitless patience during my graduate study at the University of Toronto. Her never-ending support over the past two years not only made this work possible, but also let me appreciate the rigorous approach and systematic attitude towards research. I have been extremely fortunate to have her as my advisor.

I would like to extend my gratitude to the Electrical and Computer Engineering Department, School of Graduate Studies and Rogers Family for their generous financial support throughout the course of my study in Canada.

Last but not least, I would like to thank my caring, loving family who encouraged me to make it through. You have been there for me my entire life and your constant love and support are immeasurable. Thank you.

Contents

List of Figures

List of Symbols

α^i	player i's pure action
η	inverse temperature $(1/\tau)$
$\mathcal{A}^i$	player i's pure strategy space
$\mathcal{I}$	set of all players
$\mathcal{X}^i$	player i's mixed strategy space
$\mu^{‘}$	discretization step size
$C_{(a(n))}$	the worth of the covered area by all agents
E^i_{move}	movement energy consumption of agent i
h	penalty function
p^i_α	score variable of the pure action α for player i
R	neighborhood radius
rp^i	player i's revision probability
$SBR(p)$	smoothed best response map
$x^i_{\alpha^i}$	player i's mixed strategy probability corresponding to the pure action α^i
$\alpha^i(n)$	location of agent i at time step n
λ	discount rate

μ_j	the mean vector of jth Gaussian component
ω_j	the weight of jth Gaussian component
Φ	potential function
Σ_j	the covariance of jth Gaussian component
τ	Boltzmann sampling temperature
AIC	Akaike Information Criterion
$f(l)$	the worth at coordinate l
J_{com}	communication cost
J_{merge}	merge criterion for two Gaussian components
J_{split}	split criterion for a Gaussian component
M	number of the targets
m	EM repetition regulation factor
N_R^i	collection of all neighbors of agent i
u^i	player i's utility function

Chapter 1

Introduction

In the last few decades, multi-agent systems, i.e. systems composed of multiple interacting entities, have been used to handle complex problems that are difficult or even impossible for an individual agent system to solve. Such complex problems have to be solved in a distributed manner, either because a central controller can not be used or because the resources are highly distributed. Applications of multi-agent systems cover a variety of domains including transportation, networking, information gathering systems, economics and project management. Thus, an agent in a multi-agent system can be a software agent, a human, a robot, etc.

Comparing to a monolithic system, it is a challenging task to organize and control each agent's behavior in a multi-agent system. This control task becomes more complicated if agents' "environment", i.e., the space in which agents operate, imposes limiting constraints on agents. Hence, the main concern in the multi-agent control is to provide principles and mechanisms for coordination of individual agent's behaviors while the problem's constraints are considered [1].

One of the most well-known examples of multi-agent control is multi-robot control problem in which the goal is to coordinate each robot's behavior toward achieving a global objective. A global objective is a desirable outcome that we expect the whole group of robots to reach. This objective is either maximizing the system's productivity or minimizing the consumption of resources, or both. There are various applications with different objectives for multi-robot systems, from search-and-rescue to area mapping.

1.1 Motivation

Recently, robot coverage control has received researchers' attention with various backgrounds as a fundamental problem in robotics. In a robot coverage problem, we seek to optimally cover an environment either by employing a set of robots or by using an individual robot system. However, due to the distributed nature of the coverage problem, a multi-robot setup has several potential advantages over a single-robot system [2]:

- A multi-robot system can exhibit robustness and fault-toleration due to extensive information sharing and data fusion among the robots.
- Comparing to a single-robot system, a multi-robot system can have a lower cost. Employing a number of robots that are simple and cheap can be more cost-efficient than using a single complex and expensive robot.
- A heterogeneous group of robots with diverse abilities can join together to form a multi-agent setup and to accomplish complex tasks which are difficult for each of them to deal with.
- Due to a better spatial distribution, a multi-robot system benefits from flexibility and scalability when the objective is to cover an area.

Multi-robot Coverage Control (MCC) problem covers an extensive range of applications, both with static and dynamic manners. An offline or static study of MCC problem optimizes the location of immobile sensors. The main applications of the static version include efficient sensor placement for area surveillance (e.g., [3]) and environmental parameter estimation (e.g., [4] and [5]). On the other hand, the main focus in dynamic studies is on optimal motion control of mobile sensors, i.e. robots, in various applications such as tracking problems [6], odor detection [7], search and rescue [8], home automation [9] and collision avoidance [10].

In each of these applications, a set of constraints are applied to the MCC problem. One of these constraints can be the robots' lack of knowledge about their environment. Another practical constraint is that the robots may not have an unconditional freedom to move within the environment (e.g. search and rescue). Alternatively, the robots may not be able to monitor each other's decisions and to freely communicate with other robots.

The major challenge in MCC is to design an efficient coordination between the robots which enables them to cover an area while the environment constraints are considered. The techniques from game theory have drawn significant attentions in the last few years to provide an efficient coordination between the robots in MCC systems. Game theory is able not only to describe the problem setting for MCC but also provides the tools to analyze its outcome [11]. Most of the studies done on MCC within the game theory framework are in the class of potential games (e.g. [12], [13], [14]). Potential games are an important class of games that are most suitable for optimization and modeling large-scale decentralized systems.

The concept of "learning" in potential games is a critical notion which can guarantee a potential game to reach the desired point, i.e. Nash equilibrium in game theory. Learning schemes assume that players learn over time about the environment and also about the behavior of other players. Such learning mechanisms agree with the social and biological situations in which players imitate, reinforce or update their belief. There are a variety of approaches to learning in games depending on the number of players, the information players are expected to know and the importance of the communication with other players [15].

In this thesis, we consider a MCC problem in which the objective is to cover an area toward localizing a set of unknown targets while the energy consumption is minimized. Our main focus in this thesis is on the development of learning schemes in games and their applications in MCC. Through a variety of MCC numerical experiments we show that agents can perform successfully under our proposed learning schemes, in an environment that is partially or fully unknown and in an environment that applies various constraints on the robots.

1.2 Literature Review

Design of algorithms that fit the MCC problem is closely related to a type of potential games called non-cooperative potential games. A non-cooperative game is a game in which there exists a competition between players while in a cooperative game players collaborate. However, both cooperative (e.g., [16]) and non-cooperative (e.g., [12]) schemes have been studied in MCC. In [16], a distributed cooperative coverage control algorithm is used in which the coverage problem is formulated in terms of sensing cost and communication cost. The non-cooperative approach studied in [12] is based on the notion of the potential game and learning in games. Learning in a non-cooperative potential game guarantees that the learning converges to a Nash equilibrium despite the fact that there is no coordination between the robots.

Fictitious Play (FP) is one of the earliest learning models studied in games [17]. In FP learning scheme, each player "believes" that the opponents are playing with certain strategies so each player responds accordingly. FP flaws if players happen not to play strategies that are expected. Each player's belief is actually the frequency with which his opponents selected their strategies until the current moment in the game [15].

A different version of FP is adaptive play learning which is originally introduced in [18]. Adaptive play is basically similar to FP but with an extra assumption on players' limited access to the whole history of past actions. As another variant of FP, Joint Strategy Fictitious Play (JSFP) in which players monitor the joint actions is recently introduced in [19]. With limited observational and computational capabilities, JSFP converges to a Nash equilibrium in a generalized ordinal potential game [19].

Another type of learning which is one of the simplest learning strategies in game theory is Best Response (BR) in which players select an action that yielded the highest reward in the past. Classical best response dynamics are used in [21] to achieve a solution for MCC with a network of mobile sensors. However, in BR, the slightest benefit to one action will result in that action to be played with probability one. This may not occur in practice as agents may not act fully rational due to a variety of reasons. In a modified version of BR known as Smoothed Best Response (SBR), players select each action with a probability corresponding to that action's reward. SBR is then used to decrease the probability jump in the standard BR [22].

In both BR and SBR, players need to know the reward of their future actions. Incorporation of model-based planning is a relatively new development in learning (e.g. [12]) which tries to relax the assumption on agents' initial knowledge about the reward of their future actions. A model is basically an approximation of how the environment (or a part of it) behaves prior to any interaction. Hence, by using a model players are able to predict the possible outcome of their future actions. However, a model-based learning requires additional assumptions on the structure of the environment.

A different type of learning, which is originally derived from behaviorist psychology and the notion of stimulus-response, is Reinforcement Learning (RL). The main idea in RL is that players tend to use strategies that worked well in the past. In RL, players keep an aggregate of their past interactions with their environment to respond in future situations and produce the most favorable outcome. Thus, agents' future strategies are directed by their past information about the game. Among learning techniques, RL has shown great potentials for self-learned autonomous control. RL-based autonomous systems iteratively learn to adapt themselves and to reach their objective. Due to its extensive potentials, RL is studied in many areas such as game theory, automatic control, artificial intelligence, optimization, information theory, genetic analysis algorithms and statistics. Recently, a high-order aggregation scheme is proposed in [23], which improves agents' perception of their environment.

Unlike a distributed control scheme in which each player has a level of autonomy, MCC problem can also be solved via centralized approaches (e.g., [24], [25]). In a centralized control method, there exists a control base that correlates robots in their task. In this method, the computations are usually carried out by the base and the robots rely on this base. Due to this dependency, fault rejection and system's stability are significantly reduced comparing to a distributed scheme. Furthermore, there is a critical need for an always online and area-wide communication between the base and agents [26].

There are a variety of solutions other than game theory and learning that can be used to solve the MCC problem. A decentralized gradient-search-based algorithm is proposed in [26] where some unsupervised cooperative behaviors such as group formation are reported. The gradient-type control can be combined with trajectory tracking control in some cases of multi-agent systems to improve the algorithm's performance (e.g., [27]). An interesting swarm-based dynamic control algorithm is developed in [28] where a group of mobile agents moves as a swarm to explore their environment. In [5], a learned Gaussian process is developed to optimize the sensors' location using a rough initial estimation of the model and sensor placement strategy.

There are also empirical issues involved in the MCC problem, such as robot's limited computational capacity and communication capabilities. To address such experimental difficulties, a game-theory-based navigation study for multi-robot systems is presented in [29] and [30]. Interesting methods such as event-driven discretization and decision buffer are proposed in [29] to reduce computation burden of each agent. In [33], measurement and modeling uncertainties, due to the environment and system model unobservabilities, are also investigated.

1.3 Contribution

This thesis presents an effort in enhancing multi-agent learning algorithms in games and studying applications of theses algorithms in the MCC problem. The following is an overview of the contributions of this thesis:

- While many of learning algorithms in games assume one-agent-one-time learning, it is more efficient if all agents can learn simultaneously. Such simultaneous learning decreases the algorithm's run-time. Towards achieving this goal, we introduce a synchronous learning algorithm which allows the robots to explore autonomously. Note that through this thesis, a synchronous learning process is a process that involves all the agents, while in an asynchronous learning process only one player is allowed to learn at each iteration. However, in both cases, all players are aware of the process common clock. Convergence analysis of the proposed algorithm is based on Markov decision process and the theory of resistance trees presented in [18]. Through a numerical experiment in a MCC setup, we show that both the convergence rate and the equilibrium worth are improved.

- One approach to analyzing learning processes is the stochastic approximation of the governing differential equation. Through this approximation, we can relate the behavior of stochastic learning model to an ordinary differential equation. The use of diminishing learning step size allows us to use stochastic approximation results, but in general leads to slow convergence time. Thus, we use constant learning step size in RL in order to increase the convergence rate of our algorithm. Additionally, we propose a double aggregation process in this algorithm which deepens robots' insight about the environment. The performance of the proposed algorithm is evaluated via a numerical experiment in a MCC setup.
- Finally, we propose a predictive RL algorithm and we provide conditions for convergence of this algorithm to Nash equilibrium, in potential games. In standard RL algorithms, players use an integration of past events to form a memory and to understand the actual outcome of each action. In this algorithm, we introduce an anticipatory term incorporated with the integration scheme in order to reuse past experiences to predict the Q-values trend which increases the algorithm's adaptiveness and responsiveness.

1.4 Organization

The thesis organization is as follows:

- Chapter 2 reviews basic concepts of game theory and how it fits the MCC problem.

Additionally, we introduce variants of learning schemes that can be used in game theory and we study the applications of these schemes in MCC.

- In Chapter 3, the technical details in our MCC setup are presented. We formulate a potential game in which players use an asynchronous learning process to find the game's Nash equilibrium. The case of unknown environment is also investigated in this chapter, for which we show that extra adaptation is required for the learning process to be able to converge to the game's Nash equilibrium.
- Chapter 4 introduces a synchronous learning algorithm and studies the convergence of this algorithm in potential games. We show that the performance of the proposed algorithm is improved comparing to the asynchronous learning algorithm in Chapter 3.
- In Chapter 5, a high-order RL algorithm is proposed in which the learning step size is constant. We show that the performance of this algorithm is comparable to the performance of the asynchronous learning algorithm in Chapter 3. However, the high-order RL algorithm does not need the extra adaptation which the asynchronous learning algorithm needs.
- Finally, Chapter 6 proposes a predictive RL scheme in order to increase RL's convergence rate. Similar to the high-order RL, the proposed RL algorithm in Chapter 6 is independent from the extra adaptation that is required in Chapter 3's asynchronous learning.

1.5 Publications by the author

The material presented in this thesis has appeared in international conference proceedings or peer-reviewed journals:

• *Hasanbeig, M. and Pavel, L., "On Synchronous Binary Log-linear Learning and Second Order Q-learning", International Federation of Automatic Control, 2017. [31]*

• *Hasanbeig, M. and Pavel, L., "Distributed Coverage Control by Robot Networks in Unknown Environments Using a Modified EM Algorithm", International Journal of Computer, Electrical, Automation, Control and Information Engineering, 2017. [32]*

Chapter 2

Background

In this chapter we present preliminary concepts essential for formulating our MCC setup and learning algorithms. Section 2.1 review fundamental game theoretic concepts in which we introduce a class of games which fits the MCC problem. We then move forward to the notion of learning in games and we review two main types of learning in games.

2.1 Potential Games

The concept of potential game is first introduced in [34] as a useful tool to analyze equilibrium properties in games. In a potential game each player's strategy change is expressed using a single global function, i.e. potential function. Thus, a potential game is characterized by the potential function, which specifies the players' global preference over the outcome of their actions.

Let $\mathcal{G}(\mathcal{I}, \mathcal{A}, u)$ be a game where $\mathcal{I}$ is the set of players and $|\mathcal{I}| = N$, $\mathcal{A} = \bigtimes_i \mathcal{A}^i$ is the action space, where $\mathcal{A}^i$ is the finite set of actions of player i, and $u^i : \mathcal{A} \to \mathbb{R}$ is player i's utility function. Player i's (pure) action is denoted by $\alpha^i \in \mathcal{A}^i$, with $\alpha^{-i} = (\alpha^1, ..., \alpha^{i-1}, \alpha^{i+1}, ..., \alpha^N)$ denoting action profile for players other than i. With this notation, we may write a joint action profile $\alpha = (\alpha^1, ..., \alpha^N) \in \mathcal{A}$ as $\alpha = (\alpha^i, \alpha^{-i}) \in \mathcal{A}$. In this thesis, t is the continuous time and n is referred to the learning iteration, i.e. the discrete time with which agents play.

In a repeated version of the game $\mathcal{G}$, at each time t (or at every iteration n), each player $i \in \mathcal{I}$ selects an action $\alpha^i(t) \in \mathcal{A}^i$ (or $\alpha^i(n) \in \mathcal{A}^i$) and receives a utility $u^i(\alpha)$ which in general is a function of the joint action α. Each player i chooses action $\alpha^i(n)$ (or $\alpha^i(n)$) according to the information and observations available to player i up to t (or iteration n). Both the action selection process and the available information depend on the learning process. It is important to emphasize that the learning process considered in this thesis is online, which means that as the game goes on players eventually learn how to play.

In a repeated game, a Best Response (BR) correspondence $BR(\alpha^{-i})$ is defined as the set of optimal strategies for player i against the strategy profile of its opponents, i.e.

$$BR(\alpha^{-i}) = \arg\max_{\alpha^i \in \mathcal{A}^i} u^i(\alpha^i, \alpha^{-i}) \tag{2.1}$$

Definition 1. A game $\mathcal{G}$ is a potential game if for any agent $i \in \mathcal{I}$, for every $\alpha^{-i} \in \mathcal{A}^{-i}$ and for any $\alpha^i_1, \alpha^i_2 \in \mathcal{A}^i$ there exists a potential function $\varPhi : \mathcal{A} \mapsto \mathbb{R}$ such that:

$$\varPhi(\alpha^i_2, \alpha^{-i}) - \varPhi(\alpha^i_1, \alpha^{-i}) = u^i(\alpha^i_2, \alpha^{-i}) - u^i(\alpha^i_1, \alpha^{-i}), \tag{2.2}$$

where $u^i : \mathcal{A} \mapsto \mathbb{R}$ is the player i's utility function [35].

As in (2.2), a change in the potential equals the same change in utility of player i when player i switches its action. This means that the utility function of each agent i is aligned with the potential function, for all possible deviations from all action pairs. Naturally, a player's goal is to maximize its own utility against the choices of its opponents. Thus, in potential games, each player's utility improvement is equal to the same improvement in the potential function.

An improvement path $\varOmega$ in a potential game is then defined as a sequence of action profiles $\{\alpha_1 \rightarrow \alpha_2 \rightarrow ... \rightarrow \alpha_m\}$ such that in each sequence $\alpha_k \rightarrow \alpha'_k$ a player i makes a change in its action and receives a strictly higher utility, i.e. $u^i(\alpha'_k) > u^i(\alpha_k)$. An improvement path terminates at action profile α_* if no further improvement can be obtained. A game $\mathcal{G}$ is said to have the finite improvement property if every improvement path in $\mathcal{G}$

is finite.

Theorem 2.1 *Every improvement path in a potential game is finite [34].*

Proof: Since $\mathcal{A}$ is a finite set, every improvement path $\Omega = \{\alpha_1 \rightarrow \alpha_2 \rightarrow ...\}$ must be finite.

$\square$

This means that there exist a point α_* such that no player in $\mathcal{I}$ can improve its utility and the global potential function by deviating from this point. In other words

$$u^i(\alpha_*^i, \alpha_*^{-i}) \geq u^i(\alpha^i, \alpha_*^{-i}), \ \ \forall \alpha^i \in \mathcal{A}^i, \ \ \forall i \in \mathcal{I}. \tag{2.3}$$

The strategy profile α_* is called pure Nash equilibrium of the game.

Definition 2. Given a game $\mathcal{G}$, a strategy profile $\alpha_* = (\alpha_*^i, \alpha_*^{-i})$ is a pure Nash equilibrium of $\mathcal{G}$ if and only if

$$u^i(\alpha_*^i, \alpha_*^{-i}) \geq u^i(\alpha^i, \alpha_*^{-i}), \ \ \forall \alpha^i \in \mathcal{A}^i, \ \ \forall i \in \mathcal{I}. \tag{2.4}$$

John F. Nash's theorem on Nash equilibrium has had the biggest impact on game theory and it is not limited to potential games. In a Nash equilibrium no player has a motivation to unilaterally deviate from its current state. This means that no player has any ground for regret if all pure choices of its opponents are revealed. In other words, all players in a Nash equilibrium play their best response [15].

Remark 1. Note that the potential function Φ is player-independent, i.e. global. Thus, in a potential game, the set of pure Nash equilibria can be found by identifying the local optima of the potential function.

A mixed strategy for player i is defined when player i randomly chooses between its actions in $\mathcal{A}^i$. Let $x^i_{\alpha^i}$ be the probability that player i selects action $\alpha^i \in \mathcal{A}^i$. Hence, player i's mixed strategy is $x^i \in \mathcal{X}^i$ where $x^i = (x^i_{\alpha^i_1}, ..., x^i_{\alpha^i_{|\mathcal{A}^i|}})$ and $\mathcal{X}^i$ is a unit $|\mathcal{A}^i|$-dimensional simplex $\mathcal{X}^i = \{x \in \mathbb{R}^{|\mathcal{A}^i|} \ s.t. \ \sum_{\alpha \in S} x_\alpha = 1, \ x_\alpha \geq 0\}$. Likewise, we denote the mixed-strategy profile of all players by $x = (x^1, ..., x^N) \in \mathcal{X}$ where the mixed strategy space is denoted by $\mathcal{X} = \bigtimes_i \mathcal{X}^i$.

The concept of the mixed-strategy Nash equilibrium is first introduced in [36] for a particular case of games. A mixed-strategy Nash equilibrium is an N-tuple such that each player's mixed strategy maximizes his expected payoff if the strategies of the others are held fixed. Thus, each player's strategy is optimal against his opponents'. If the expected utility of player i is given as

$$u^i(x) = \sum_{\alpha \in \mathcal{A}} (\prod_{s \in \mathcal{I}} x^s_{\alpha^s}) u^i(\alpha^i, \alpha^{-i}), \tag{2.5}$$

then a mixed strategy profile $x_* \in \Delta$ is a mixed strategy Nash equilibrium if for all players we have

$$u^i(x^i_*, x^{-i}_*) \geq u^i(x^i, x^{-i}_*), \ \ \forall x^i \in \Delta^i, \ \ \forall i \in \mathcal{I}, \ \ x^i \neq x^i_*.$$

Such a Nash equilibrium is a fixed-point of the mixed-strategy best-response, or in other words, all players in a Nash equilibrium play their best response

$$x^i_* \in BR(x^{-i}_*), \ \ \forall i \in \mathcal{I}$$

where $BR(x^{-i}) = \{x^i_* \in \Delta^i | u^i(x^i_*, x^{-i}) \geq u^i(x^i, x^{-i}), \ \forall x^i \in \Delta^i\}$, is the best response set.

Remark 2. In the case when (2.3) and (2.5) hold strictly, the Nash equilibrium is called a strict Nash equilibrium.

2.2 Learning in Games

Learning in games tries to relax assumptions of classical game theory on players' initial knowledge and belief. Learning game theory basically assumes that players adapt and learn over time about the game and the environment. Therefore, instead of immediately playing a perfect action, players adapt their strategies based on the outcomes of their past interactions. In the following, we review two novel class of learning in games: reinforcement learning and model-based learning.

2.2.1 Reinforcement Learning

Reinforcement learning discusses how to map players' action to actions' reward so as to maximize the reward. Players are not told which actions to take but instead they have to discover which actions yield the most reward by "exploring" the environment [37]. RL only requires players to observe their own ongoing payoffs, so they do not need to monitor their opponents' strategies or calculate payoffs of action that they did not play. However, a RL learning is more like a trial and error process in which all the actions or states need to be examined to determine their corresponding reward. This naturally slows down the algorithm's convergence rate. In the following we present a technical background on RL and its application in multi-agent systems.

2.2.1.1 Aggregation

Each player in RL uses a score variable as a memory to store, i.e. aggregate, a track of past events. It is common to assume that the game rewards can be stochastic, i.e. an action profile does not always result in the same deterministic utility. Therefore, actions are needed to be sampled, i.e. aggregated, repeatedly or continuously. We denote the player i's score vector by $p^i \in \mathcal{P}^i$ where $\mathcal{P}^i$ is the player i's score space. A common form of continuous aggregation rule is the exponential discounted model:

$$p^i_\beta(t) = p^i_\beta(0)\lambda^t + \int_0^t \lambda^{t-s} u^i(\beta, \alpha^{-i}) ds, (\beta, \alpha^{-i}) \in \mathcal{A}, \tag{2.6}$$

where p^i_β is the action β's aggregated score and $\lambda > 0$ is the model's discount rate. λ can be alternatively defined via $T = log(1/\lambda)$. Clearly, if $p^i_\beta(0) = 0$ then

- For $\lambda \in (0, 1)$, the model in (2.6) allocates more weight (exponentially) to the recent observations ($T > 0$),

- If $\lambda = 1$, we have a uniform aggregation: all past observations are treated uniformly ($T = 0$),
- For $\lambda > 1$, the model considers more weight for older observations ($T < 0$).

As discussed in [38] the choice of the value of T affects the learning dynamics and the process of finding the estimated Nash equilibrium. The discount rate has a double role in RL: (1) It determines the weight that players give to their past observations. (2) T reflects the rationality of the players in choosing their actions and consequently the accuracy of the players' stationary points in being the true Nash equilibrium. Additionally, discounting implies that the score variable $p^i_\beta(t)$ will remain bounded which consequently prevent the agents' mixed strategies from approaching the boundaries $\mathcal{X}$. By differentiating (2.6), and assuming $p^i_\beta(0) = 0$, we obtain the following score dynamics

$$\dot{p}^i_\beta = u^i(\beta, \alpha^{-i}) - T p^i_\beta. \tag{2.7}$$

Remark 3. Discrete-Time Analysis: By applying the first-order Euler discretization on (2.7) we obtain:

$$P^i_\beta(n+1) = P^i_\beta(n) + \mu(n)[u^i(\beta, \alpha^{-i}) - T P^i_\beta(n)], \tag{2.8}$$

where n is the iteration number and $\mu(n)$ is the discretization step size. $P^i_\beta(n)$ is the discrete equivalents of $p^i_\beta(t)$. A stochastic approximation of the discrete dynamics requires step size to satisfy $\sum_n \mu(n) = \infty$ and $\sum_n \mu(n)^2 < \infty$ [39].

Remark 4. The received payoff may be perturbed by some noise signal. Let $\alpha^i(n)$ be the action of player i at the nth time step and assume the corresponding observed payoff is $u^i(\alpha^1(n), ..., \alpha^N(n))$. The perturbed observed payoff is of the form

$$\tilde{u}^i(n) = u^i(\alpha^1(n), ..., \alpha^N(n)) + \xi^i(n), \tag{2.9}$$

where ξ^i is a bounded noise signal with $\xi^i(n)$ independent of $\alpha^i(n)$.

2.2.1.2 Choice Map

In action selection step players decide how to "exploit" the score variable to choose a strategy against the environment. One natural method is to select a strategy that maximizes the expected score via the mapping

$$BR(p^i) = \arg\max_{x^i \in \mathcal{X}^i} \sum_{\beta \in \mathcal{A}^i} x^i_\beta p^i_\beta, \tag{2.10}$$

where p^i is the player i's score vector. However, this simple best response map may raise several problems: (1) Equation (2.10) can become a multi-valued mapping if at least two score variables $p^i_\alpha, p^i_\beta \in \mathcal{P}^i$ happen to be equal. (2) A simple best response generally forces RL to converge to pure strategies while we may be interested in non-pure equilibria. (3) Such a choice map may lead to a discontinuous trajectories of play due to the sudden step jumps in the best response correspondence [38].

To overcome these issues, a smooth strongly convex penalty function ($h^i : \mathcal{X}^i \mapsto R$) is usually added to (2.10) to regularize the choice map:

$$SBR(p^i) = \arg\max_{x^i \in \mathcal{X}^i} \sum_{\beta \in \mathcal{A}^i} [x^i_\beta p^i_\beta - h^i(x^i)]. \tag{2.11}$$

This choice model is often called "Smoothed Best Response (SBR) map" or "quantal response function". The choice map (2.11) actually discourages player i from choosing an action from boundaries of $\mathcal{X}^i$, i.e. from choosing pure strategies. The penalty function h^i in (2.11) has the following properties [38]:

1. h^i is finite except on the relative boundaries of $\mathcal{X}^i$,
2. h^i is continuous on $\mathcal{X}^i$, smooth on relative interior of $\mathcal{X}^i$ and $|dh^i(x^i)| \to +\infty$ as x^i approaches to the boundaries of $\mathcal{X}^i$,
3. h^i is convex on $\mathcal{X}^i$ and strongly convex on relative interior of $\mathcal{X}^i$.

Some of the famous penalty functions are:

1. The Gibbs entropy: $h(x) = \sum_\beta x_\beta \log x_\beta$,
2. The Tsallis entropy: $h(x) = (1-q)^{-1} \sum_\beta (x_\beta - x_\beta^q),\ \ 0 < q \leq 1$,
3. The Burg entropy: $h(x) = -\sum_\beta x \log x_\beta$.

The most prominent SBR map is the "logit" map which is derived by implementing Gibbs entropy in (2.11),

$$x_\alpha^i = \Big[SBR(p^i)\Big]_\alpha = \frac{\exp(p_\alpha^i)}{\sum_{\beta \in \mathcal{A}^i}\ \exp\big(p_\beta^i\big)}. \tag{2.12}$$

It is not always easy to write a close form of the choice map and the Gibbs entropy is an exception [38].

Remark 5. By applying the first-order Euler discretization on (2.11) we obtain:

$$X^i(n+1) = SBR(P^i(n+1)), \tag{2.13}$$

where $X^i(n)$ is the discrete equivalents of $x^i(t)$.

Remark 6. The SBR of the players can be sharpened by adding parameter η^i to (2.11)

$$x^i = SBR(\eta^i\, p^i) = \arg\max_{x^i \in \mathcal{X}^i} \sum_{\beta \in \mathcal{A}^i} [x_\beta^i \eta^i p_\beta^i - h^i(x^i)],\ \ \eta^i > 0, \tag{2.14}$$

where the parameter η^i is the player-specific inverse "temperature" $(1/\tau^i)$ of the choice map. Clearly, as $\eta^i \to \infty$, the choice map in (2.14) becomes a simple best response map. Conversely, as $\eta^i \to 0$, player i disregards the score vector (p^i) and chooses its next strategy uniformly randomly.

Remark 7. The steepness of the penalty function ensures that mixed strategies of play, that is following the game's differential equation, remain in the interior of $\mathcal{X}$. However if the boundaries of $\mathcal{X}$ is of interest, one can replace h^i with an arbitrary non-steep, strongly convex penalty function as discussed in [22]. In this case the trajectories of play can enter and exit the boundaries of $\mathcal{X}$ and the image of the choice map becomes the entire $\mathcal{X}$ rather than only its interior.

Remark 8. An alternative derivation method of the logit map is introduced in [40]: a distributed perturbation variable with a zero mean is applied on the score variable vector and a simple best response is used as the choice map. Thus, the logit map can be either derived from the SBR (penalty function) or by perturbing the stochastic model.

Remark 9. In the case of $T = 0$ and employing the Gibbs entropy function as the penalty function in (2.11), the RL dynamics reduces to

$$\begin{aligned} \dot{p}^i_\beta &= u^i(\beta, \alpha^{-i}), \\ x^i_\beta &= \frac{\exp\left(p^i_\beta\right)}{\sum_{\beta' \in \mathcal{A}^i} \exp\left(p^i_{\beta'}\right)}. \end{aligned} \tag{2.15}$$

By differentiating x^i_β with respect to time and substituting into (2.15)

$$\dot{x}^i_\beta = x^i_\beta \, [u^i(\beta, \alpha^{-i}) - \sum_{\beta' \in \mathcal{A}^i} x^i_{\beta'} \, u^i(\beta', \alpha^{-i})]. \tag{2.16}$$

It is interesting to see that the evolution of mixed strategies under dynamics of (2.16) agrees with the well known replicators dynamics of evolutionary games. There are several RL algorithms that lead to the replicator dynamics [22].

2.2.1.3 High-order RL

As discussed in Section 2.2.1.1, RL dynamics in games typically use first order aggregation where actions' payoffs determine the growth rate of the players' strategy:

$$\dot{p}^i_\beta = u^i(\beta, \alpha^{-i}) - T p^i_\beta.$$

In this section, we review what happens beyond first order by using payoffs as higher order forces of change. To that end we first introduce the notion of strictly dominated strategy and weakly dominated strategy:

- strategy $\alpha \in \mathcal{A}^i$ is a strictly dominated strategy by strategy $\beta \in \mathcal{A}^i$ if for all $x \in \mathcal{X}$

we have $u^i_\alpha(x) < u^i_\beta(x)$,

- strategy $\alpha \in \mathcal{A}^i$ is a weakly dominated strategy by strategy $\beta \in \mathcal{A}^i$ if for all $x \in \mathcal{X}$ we have $u^i_\alpha(x) \leq u^i_\beta(x)$.

For a pure strategy $\beta \in \mathcal{A}^i$, extinction means as $t \to \infty$ the strategy $x^i_\beta \to 0$. Alternatively, a mixed strategy $q^i \in \mathcal{X}^i$ becomes extinct along $x(t)$ if $D_{kl}(q^i||x^i) = \sum_\beta q^i_\beta \log\big(q^i_\beta / x^i_\beta\big) \to \infty$ as $t \to \infty$.

The evolution of the system's dynamic and the efficiency of players' learning, crucially depend on the how players aggregate their past observation and update their score variable. As proposed in [23], an n-fold, i.e. high-order, aggregation scheme is

$$p^{i\,(n)}_\beta(t) = u^i(\beta, \alpha^{-i}). \tag{2.17}$$

Theorem 4.1 in [23] proves that in the high order learning, dominated strategies become extinct. Furthermore, Theorem 4.3 in [23] shows that unlike the first order dynamics in which weakly dominated strategies may survive, in the high-order dynamics ($n \geq 2$) weakly dominated strategies become extinct. However, one of the basic assumptions in Theorem 4.3 is that the players start at rest ($\dot{x}(0) = \ddot{x}(0) = ... = x^{(n-1)}(0) = 0$).

Needless to say, an n-th order aggregation scheme looks deeper into the past observations. In addition to the Gibbs entropy function, and as in (2.14), the parameter η^i can be added to the choice map of high-order RL, to control the model's sensitivity to the received payoff. Thus, the choice map $SBR^i : \mathbb{R}^{\mathcal{A}^i} \to \mathcal{X}^i$ is of the form

$$x^i_\beta = \Big[SBR^i(\eta^i\, p^i)\Big]_\beta = \frac{\exp\big(\eta^i\, p^i_\beta\big)}{\sum_{\beta' \in \mathcal{A}^i}\, \exp\big(\eta^i\, p^i_{\beta'}\big)}. \tag{2.18}$$

Remark 10. For $n = 1$, the dynamics reduce to the standard replicator dynamics of (2.16)

$$\dot{x}^i_\beta = \eta^i x^i_\beta\, [u^i(\beta, \alpha^{-i}) - \sum_{\beta' \in \mathcal{A}^i} x^i_{\beta'}\, u^i(\beta', \alpha^{-i})], \tag{2.19}$$

and for $n = 2$, it yields

$$\ddot{x}^i_\beta = \eta^i x^i_\beta \,[u^i(\beta, \alpha^{-i}) - \sum_{\beta' \in \mathcal{A}^i} x^i_{\beta'}\, u^i(\beta', \alpha^{-i})] + x^i_\beta \,[\dot{x}^i_\alpha{}^2 / x^i_\alpha{}^2 - \sum_{\beta' \in \mathcal{A}^i} \dot{x}^i_{\beta'}{}^2 / x^i_{\beta'}{}^2]. \tag{2.20}$$

2.2.2 Model-based Learning

In a model-based learning algorithm, we assume that players can eventually form a model of the environment by constantly interacting with the environment. Thus, when the environment is unknown, by using the environment model, players are able to estimate the payoff of the actions that they did not play. In a model-based learning, each player i possesses an unbiased and bounded estimate $\hat{u}^i_\alpha$ for the payoff of each action $\alpha \in \mathcal{A}^i$ (including the actions that he did not play) such that

$$\mathbb{E}[\hat{u}^i_\alpha(n+1)|H_n] = u^i_\alpha(n), \tag{2.21}$$

where H_n is the history of the process up to the iteration n.

Clearly, it reduces the critical need for the trial and error process over all actions if the estimation is accurate enough. Thus, in a model-based learning scheme, we expect a faster convergence in comparison to RL. In the following, we review a class of learning algorithms in games which can be incorporated with a model of the environment and appropriately fits the MCC problem.

2.2.2.1 Log Linear Learning

In this section we discuss the well-known log linear learning (LLL) algorithm which is originally introduced in [41]. In LLL algorithm, at each time step, only "one" random player, e.g. player i, is allowed to alter his action. Player i chooses a trial action from its "entire" action set $\mathcal{A}^i$ and according to its mixed strategy X^i.

In LLL, player i's mixed strategy probability for action $\alpha \in \mathcal{A}^i$ is updated by a sharp

SBR on u^i:

$$X^i_\alpha = \frac{\exp(1/\tau\ u^i_\alpha)}{\sum_{\beta\in\mathcal{A}^i}\exp\big(1/\tau\ u^i_\beta\big)}. \tag{2.22}$$

Note that in (2.22), player i needs to know the utility value of any action $\beta \in \mathcal{A}^i$. However, in the case of unknown environment, utility values of actions in $\mathcal{A}^i$ that are not played are not available to player i. Thus, in a model-based LLL scheme, utility values of unknown actions have to be replaced by player's estimation model.

Definition 3. As discussed in [18], if the frequency or the probability with which an action β is played is Pr^ϵ_β in the associated perturbed Markov process, then the action β is a stochastically stable state if:

$$\lim_{\epsilon\to 0} Pr^\epsilon_\beta = Pr_\beta$$

where Pr_β is the corresponding probability in the unperturbed Markov process.

The following proposition shows that LLL guarantees that only actions that maximize the potential function are stochastically stable.

Proposition 2.1 *In a finite N-player potential game if all players adhere to log-linear learning then the stochastically stable states are the set of potential maximizers [35].*

Proof: see Appendix I

LLL can be modeled as a perturbed Markov chain where the unperturbed Markov process is a best reply process. The theory of resistance trees which is presented in the following, is used in [35] to analyze the convergence of the Markov process associated with LLL.

2.2.2.2 Theory of Resistance Trees

The following is a brief review of the theory of resistance trees presented in [18]. Consider a Markov chain over a state space Z. A complete directed graph with $|Z|$ vertices can represent the Markov chain over the space Z. Let P_0 be the unperturbed probability transition matrix for this Markov chain and let $P_0(z \to z')$ denote the probability of transition from state z to z'.

Conversely, in a perturbed process the probability transition matrix is denoted by P_ϵ and the size of the perturbation on P_0 is indexed by $\epsilon > 0$. In the following, we will also use P_ϵ to refer to the process and the graph associated with the transition matrix P_ϵ. Each directed edge $z \to z'$ in the graph P_ϵ has the weight of $R(z \to z')$ which is called the resistance of the edge $z \to z'$. The resistance of the edge $z \to z'$ is a relative "measure" of how sup-optimal the transition $z \to z'$ is, with respect to the defined utility distribution over Z. The process P_ϵ is a perturbed Markov process of P_0 if for sufficiently small ϵ and for any two states $z \in Z$ and $z' \in Z$,

$$\lim_{\epsilon \to 0^+} P_{\epsilon(z \to z')} = P_{0(z \to z')}, \tag{2.23}$$

and

$$P_{\epsilon(z \to z')} > 0 \Rightarrow 0 < \lim_{\epsilon \to 0^+} \frac{P_{\epsilon(z \to z')}}{\epsilon^{R(z \to z')}} < \infty, \tag{2.24}$$

where $R(z \to z') > 0$ is the resistance of the transition $z \to z'$.

A tree T rooted at z is a set of $|Z| - 1$ directed edges such that from every vertex other than z there exist a unique directed path in T to z. The sum of the resistances over the $|Z| - 1$ edges is the resistance of the tree T. The stochastic potential of the state z_j is denoted by φ_j and is defined as the minimum resistance over all the trees rooted at z_j. Thus a minimum resistance tree is a tree, rooted at any state z_i, for which φ_i is minimum among all the trees rooted at other states.

Proposition 2.1 shows that the stochastic stable states are the states with minimum stochastic potential (see Appendix I). Therefore, as in Lemma 1 in [18], z_* is a stochastically stable state if and only if there exists a tree rooted at z_* for which the resistance is minimum among all the trees rooted at other states.

2.2.2.3 Synchronous Learning

Recall that one of the basic assumptions in standard LLL is that only one random player is allowed to alter its action at each step. In synchronous log linear learning (SLLL), which is introduced in [35], a group of players $G \subset \mathcal{I}$ is selected to update its action based on the probability distribution $rp \subset \Delta(2^{\mathcal{I}})$; rp^G is defined as the probability that group G will be chosen and rp^i is defined as the probability that player $i \in \mathcal{I}$ updates its action. The set of all groups with $rp^G > 0$ is denoted by $\bar{G}$. This algorithm is called SLLL with revision process rp. A SLLL scheme with revision process rp leads to a perturbed Markov process while the equivalent unperturbed process is the SLLL under best replies. SLLL changes the stationary distribution (as $t \to \infty$) and also affects the Markov process stable states. It is shown in [35] that the resistance of a feasible transition $\alpha_0 \to \alpha_1 = (\alpha_1^i, \alpha_0^{-i})$ in SLLL can be defined as

$$R(\alpha_0 \to \alpha_1) = \min_{G \in \bar{G}(\alpha_0, \alpha_1, rp)} \sum_{i \in G} \max_{\alpha_*^i \in \mathcal{A}^i} u^i(\alpha_*^i, \alpha_0^{-i}) - u^i(\alpha_1^i, \alpha_0^{-i}). \tag{2.25}$$

Note that $R(\alpha_0 \to \alpha_1)$ in (2.25) is not unique and determines a relative measurement of how sub-optimal α_1 is, comparing to α_0. For any finite weakly acyclic game under best replies, [35] shows that if all players adhere to SLLL with a revision probability rp, and if all Nash equilibria are strict then stochastically stable states are contained in the set of Nash equilibria.

In an independent revision process, each player independently decides whether to revise his strategy by LLL rule. At each time, the probability that each player i select a new action based on Boltzmann choice map is $\omega \in \mathbb{R}^+$ and the probability that each player i selects his previous action is $1 - \omega$. The following theorem shows that in a finite potential game where all players adhere to SLLL, the stochastically stable states are the set of potential maximizer.

Theorem 2.2 *Consider any finite N-player potential game where all players adhere to synchronous log-linear learning with an independent revision probability and with an up-*

date parameter $\omega \in R^{+}$. If $\omega = (e^{-\frac{1}{\tau}})^{m}$ then for sufficiently large m, the stochastically stable states are the set of potential maximizers [35].

Proof: see Appendix II

2.2.2.4 Constrained Action Set

The standard LLL requires each player i to have access to all available actions in $\mathcal{A}^i$. In the case when player i has no free access to every action in $\mathcal{A}^i$, his action set is "constrained" and is denoted by $\mathcal{A}^i_c$. With a constrained action set, players may be trapped in local sub-optimal equilibria since the entire $\mathcal{A}^i$ is not available to player i at each move.Thus stochastically stable states may not be potential maximizers. Binary log linear learning (BLLL) is a variant of standard LLL which provides a solution for the issues caused by constrained action sets.

In a BLLL scheme, at each time n, only one random player i is allowed to alter his action while all the other players must repeat their current actions. Player i selects one trial action α^i_T uniformly randomly from his constrained action set $\mathcal{A}^i_c(\alpha^i(n))$. Player i's mixed strategy is then played by

$$X^i_{\alpha^i(n)}(n) = \frac{\exp\left(\frac{1}{\tau}u^i(\alpha^i(n), \alpha^{-i}(n))\right)}{\exp\left(\frac{1}{\tau}u^i(\alpha^i(n), \alpha^{-i}(n))\right) + \exp\left(\frac{1}{\tau}u^i(\alpha^i_T, \alpha^{-i}(n))\right)}, \tag{2.26}$$

$$X^i_{\alpha^i_T}(n) = \frac{\exp\left(\frac{1}{\tau}u^i(\alpha^i_T, \alpha^{-i}(n))\right)}{\exp\left(\frac{1}{\tau}u^i(\alpha^i(n-1), \alpha^{-i}(n))\right) + \exp\left(\frac{1}{\tau}u^i(\alpha^i_T, \alpha^{-i}(n))\right)}, \tag{2.27}$$

where $X^i_{\alpha^i(n)}$ is the probability that player i remains on his current action $\alpha^i(n)$ and $X^i_{\alpha^i_T}$ is the probability that player i selects the trial action α^i_T.

Assumption 2.1 *For each player i and for any action pair $\alpha^i(1), \alpha^i(m) \in \mathcal{A}^i$ there exists a sequence of actions $\alpha^i(1) \rightarrow \alpha^i(2) \rightarrow ... \rightarrow \alpha^i(m)$ satisfying*

$$\alpha^i(k) \in \mathcal{A}_c^i(\alpha^i(k-1)),\ \forall k \in \{2, ..., m\}.$$

Assumption 2.2 *For each player i and for any action pair $\alpha^i(1), \alpha^i(2) \in \mathcal{A}^i$,*

$$\alpha^i(2) \in \mathcal{A}_c^i(\alpha^i(1)) \ \Leftrightarrow\ \alpha^i(1) \in \mathcal{A}_c^i(\alpha^i(2)).$$

The following theorem studies the convergence of BLLL in potential games under Assumptions 2.1 and 2.2.

Theorem 2.3 *In a finite N-player potential game satisfying Assumptions 2.1 and 2.2 and with potential function $\phi : \mathcal{A} \mapsto R$, if all players adhere to BLLL, then the stochastically stable states are the set of potential maximizers [35].*

Proof: see Appendix III.

2.3 Summary

In this chapter, we presented a background on potential games and learning algorithms that can guarantee potential games to converge to a Nash equilibrium. In RL scheme players only observe the utility of the actions that they play. On the other hand, in model-based learning players form a model of the environment as the game goes on. This model is further used to help players decide how to efficiently play against the environment.

Chapter 3

Multi-robot Coverage Control

3.1 Introduction

Multi-robot Coverage Control (MCC) algorithms are concerned with the design of rules for coordinating a set of robots' action. In this thesis, the main objective of agents is to efficiently cover an area and to learn about a set of stationary targets. We assume targets are distributed randomly in the area and each target emits a signal in a form of a symmetric Gaussian function which can be sensed by nearby agents. The environment in this thesis is defined as a limited, two-dimensional, rectangular area over which a probabilistic function is defined that represents the probability of targets' existence (e.g., [12], [16] and [5]). Therefore, the probability of targets' existence at each point is proportional to the cumulative strength of the signals detected at that point.

There are a variety of different approaches to model the search area. One approach is to model the environment as a two-dimensional polyhedron over which an event density function is defined to determine targets' distribution (e.g., [16]). Alternatively, the area can be divided into Voronoi regions where the objective of agents in this approach is to converge to the centroids of their assigned regions (e.g., [42], [43] and [44]).

During the search for the targets, the mobile sensors regularly collect data from the environment by sensing the signal strength at different locations. The robots use the col-

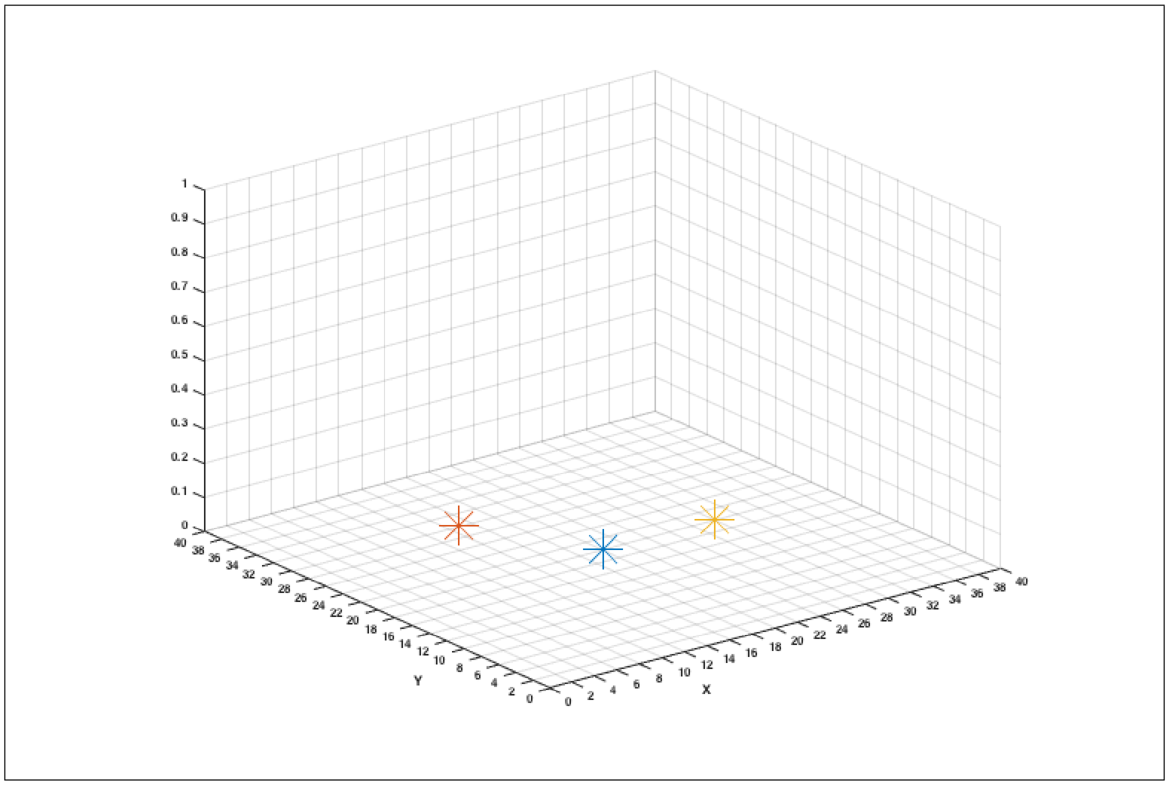

Figure 3.1. Two dimensional environment and the set of targets

lected data to "reinforce" their knowledge about the area or to "model" the environment. This knowledge will further support agents' decision-making process. Clearly, the overall systems' performance is directly related to agents' location in the environment. Thus, the coverage control problem can be reduced to the problem of optimally locating the robots based on data collected from the area. In the following we present technical details in formulation of our MCC setup.

3.2 Problem Formulation

Consider a finite number of robots as a network of mobile sensors. The environment is a two-dimensional area, divided into a $L \times L$ square grid. Each cell on this grid is a 1×1 square and the coordinates of its center are $l = (l_x, l_y) \in \mathcal{L}$, where $\mathcal{L}$ is defined as the collection of the centroids of all lattices. The agents can only move between these centroids (Fig 3.1).

The location of agent $i \in \mathcal{I}$ at iteration n is $\alpha^i(n) := (l_x^i(n), l_y^i(n)) \in \mathcal{L}$ and is defined as his action at time step n. Let the set of agent i's neighbor cells be defined as $N_R^i(\alpha(n)) := \{l \in \mathcal{L} | \, |\alpha^i(n) - l| \leq R\}$ where R is the neighborhood radius. The motion of agents is limited to their adjacent lattices and we use $\mathcal{A}_c^i(n)$ to denote this constrained action set for agent i at time step n.

The worth of each cell is defined as the probability that a target exists at the center of that cell and is denoted by $f : \mathcal{L} \to [0, 1]$. The agents can determine $f(l)$ when they are at the coordinate l. The area covered by agent i at time step n is $C^i : \mathcal{A}^i \mapsto \mathbb{R}$ and is defined as $C^i(\alpha^i(n)) := \sum_{l \in N_\delta^i(n)} f(l)$ where δ is the covering range.

In our setup, each agent i lays a flag at each cell as he observes that cell; the trace of his flags is denoted by $\varUpsilon^i$ and is detectable by other agents from the maximum distance of 2δ. This technique allows the agents to have access to the observed areas locally without any need to have access to each other's observation vectors.

3.2.1 Game Formulation

An appropriate utility function has to be defined to specify the game's objective and to properly model the correlations between the agents. In our case, we have to define a utility function that takes the reward of sensing worthwhile areas. Inspired by [12], we also consider the cost of movement of the robots: the movement energy consumption of agent i from time step $n-1$ to n is denoted by $E_{move}^i = K^i(|\alpha^i(n) - \alpha^i(n-1)|)$, where $K^i > 0$ is a regulating coefficient. The coverage problem can now be formulated as a game by defining the utility function:

$$u^i(\alpha^i(n), \alpha^i(n-1)) = \varrho^i[C^i(\alpha^i(n)) - C_n^i(\alpha^i(n))] - K^i(|\alpha^i(n) - \alpha^i(n-1)|), \quad (3.1)$$

where $C_n^i(\alpha^i(n)) = \sum_{j \in \mathcal{I} \setminus i} \sum_{l \in N_j^\delta \cap N_i^\delta} f(l)$ and ϱ^i is defined as:

$$\varrho^i = \begin{cases} 1 & : \alpha^i \notin \varUpsilon^j, j \neq i \\ 0 & : otherwise \end{cases}$$

Parameter ϱ^i prevents players to pick their next actions from the areas that are previously observed by other agents. As in (3.1), $C^i(\alpha^i(n)) - C^i_n(\alpha^i(n))$ is the worth of area that is only covered by agent i. Hence, u^i only depends on the actions of player i, i.e. u^i is "separable" from the actions of other players.

Definition 4. Player i's utility function u^i is called separable if u^i only depends on player i's action α^i.

Lemma 3.1 *The coverage game $\mathcal{G} := \langle \mathcal{I}, \mathcal{A}, U_{cov} \rangle$ is a potential game where U_{cov} is the collection of all utilities $U_{cov} = \{u_i, i = 1, ..., N\}$ and the potential function is:*

$$\Phi(\alpha(n), \alpha(n-1)) = \sum_{j=1}^{N} u^j(\alpha^j(n), \alpha^j(n-1)). \tag{3.2}$$

Proof: We have to show that for any agent $i \in \mathcal{I}$, for every $\alpha^{-i}(n) \in \mathcal{A}^{-i}$ and for any $\alpha^i_1(n), \alpha^i_2(n) \in \mathcal{A}^i$ there exists a potential function $\Phi : \mathcal{A} \mapsto \mathbb{R}$ such that:

$$\begin{aligned} &\Phi(\alpha^i_2(n), \alpha^{-i}(n), \alpha(n-1)) - \Phi(\alpha^i_1(n), \alpha^{-i}(n), \alpha(n-1)) = \\ &u^i(\alpha^i_2(n), \alpha^{-i}(n), \alpha(n-1)) - u^i(\alpha^i_1(n), \alpha^{-i}(n), \alpha(n-1)), \end{aligned} \tag{3.3}$$

By (3.2) we have:

$$\begin{aligned} &\Phi(\alpha^i_2(n), \alpha^{-i}(n), \alpha(n-1)) = [\Sigma^N_{j=1,j\neq i} u^j(\alpha^j(n), \alpha^j(n-1))] + u^i(\alpha^i_2(n), \alpha^i(n-1)) = \\ &\Big[\Sigma^N_{j=1,j\neq i} \varrho^j [C^j(\alpha^j(n)) - C^j_n(\alpha^j(n))] - \Sigma^N_{j=1,j\neq i}[K^i(|\alpha^j(n) - \alpha^j(n-1)|)]\Big] + \\ &\Big[\varrho^i[C^i(\alpha^i_2(n)) - C^i_n(\alpha^i_2(n))] - K^i(|\alpha^i_2(n) - \alpha^i(n-1)|)\Big], \end{aligned}$$

and

$$\begin{aligned} &\Phi(\alpha^i_1(n), \alpha^{-i}(n), \alpha(n-1)) = [\Sigma^N_{j=1,j\neq i} u^j(\alpha^j(n), \alpha^j(n-1))] + u^i(\alpha^i_1(n), \alpha^i(n-1)) = \\ &\Big[\Sigma^N_{j=1,j\neq i} \varrho^j [C^j(\alpha^j(n)) - C^j_n(\alpha^j(n))] - \Sigma^N_{j=1,j\neq i}[K^i(|\alpha^j(n) - \alpha^j(n-1)|)]\Big] + \\ &\Big[\varrho^i[C^i(\alpha^i_1(n)) - C^i_n(\alpha^i_1(n))] - K^i(|\alpha^i_1(n) - \alpha^i(n-1)|)\Big], \end{aligned}$$

Hence,

$$\varPhi(\alpha_2^i(n), \alpha^{-i}(n), \alpha(n-1)) - \varPhi(\alpha_1^i(n), \alpha^{-i}(n), \alpha(n-1)) =$$
$$\varrho^i[C^i(\alpha_2^i(n)) - C_n^i(\alpha_2^i(n))] - K^i(|\alpha_2^i(n) - \alpha^i(n-1)|) - \varrho^i[C^i(\alpha_1^i(n)) - C_n^i(\alpha_1^i(n))] -$$
$$K^i(|\alpha_1^i(n) - \alpha^i(n-1)|) = u^i(\alpha_2^i(n), \alpha^{-i}(n), \alpha(n-1)) - u^i(\alpha_1^i(n), \alpha^{-i}(n), \alpha(n-1)),$$

and (3.3) follows. □

3.2.2 Gaussian Model

Targets are assumed to emit a signal in a Gaussian form that decays proportionally to the distance from the target; the mobile sensor should be close enough to sense the signal. Thus, the function $f(l)$ is basically a worth distribution in the form of Gaussian Mixture over the environment as in [12]. A Gaussian Mixture Model (GMM) is a weighted sum of Gaussian components (i.e. single Gaussian functions). In our game GMM is defined as:

$$GMM := f(l) = \sum_{j=1}^{M} \omega_j \, g(l|\mu_j, \varSigma_j), \tag{3.4}$$

where M is the number of components (i.e. targets), ω_j is the weight (signal strength) of jth component and

$$g(l|\mu_j, \varSigma_j) = \frac{1}{2\pi|\varSigma_j|^{1/2}} \, exp[\frac{-1}{2}(l-\mu_j)^T \varSigma_j^{-1}(l-\mu_j)], \tag{3.5}$$

is a two-variable Gaussian function where l is the location vector; μ_j is the mean vector (i.e. location of the target j); and $\varSigma_j$ is the covariance matrix of component j. Note that each component represent one target's presence probability and the whole GMM is a mixture of these components. To ensure that the Gaussian distribution is representing the probability map over the area, the summation of all weight coefficients must be equal to one (i.e. $\sum_{j=1}^{M} \omega_j = 1$). GMM parameters can be summarized into the set $\lambda :=$

$\{\omega_j, \mu_j, \sigma_j | j = 1, ..., M\}$.

3.3 Learning Process

Recall that a learning scheme in a potential game can provide the finite improvement property for the game to efficiently reach a Nash equilibrium. This section reviews BLLL as a model-based learning scheme that is used in the literature (e.g. [12]) to solve the MCC problem. In practice a robot in our MCC setup is not able to move rapidly from its current lattice to any arbitrary lattice. In other words, the range of its movement is bounded, i.e. constrained. Recall that a BLLL scheme is a modified version of standard LLL which allows players to have a constrained action set in a potential game. Therefore, it is reasonable to assume each player i's available action set at time n is a constrained set $\mathcal{A}_c^i(n)$ so the suitable learning process is BLLL. It is shown that in a potential game, the game will stochastically converge to the potential maximizer if players adhere to BLLL [35].

By employing BLLL in our setup, at each time step n, one agent $i = (n \ mod \ N) + 1$ is selected to change its action while other agents repeat their action (i.e. $\alpha^{-i}(n+1) = \alpha^{-i}(n)$). Next, agent i will choose a trial action α_T^i uniformly randomly from its available action set $\mathcal{A}_c^i(n)$. In our case, the available action set for player i is the set of his neighbor lattices. Finally, based on the Boltzmann sampling method, the agent i chooses between selecting the action α_T^i as its next action or remaining on its current action $\alpha^i(n)$:

$$X_{\alpha^i(n)}^i = \frac{\exp\left(\frac{1}{\tau} u^i(\alpha^i(n), \alpha^{-i}(n), \alpha(n-1))\right)}{\exp\left(\frac{1}{\tau} u^i(\alpha^i(n), \alpha^{-i}(n), \alpha(n-1))\right) + \exp\left(\frac{1}{\tau} u^i(\alpha_T^i, \alpha^{-i}(n), \alpha(n))\right)}, \tag{3.6}$$

$$X^i_{\alpha^i_T} = \frac{\exp\left(\frac{1}{\tau}u^i(\alpha^i_T, \alpha^{-i}(n), \alpha(n))\right)}{\exp\left(\frac{1}{\tau}u^i(\alpha^i(n), \alpha^{-i}(n), \alpha(n-1))\right) + \exp\left(\frac{1}{\tau}u^i(\alpha^i_T, \alpha^{-i}(n), \alpha(n))\right)}, \tag{3.7}$$

$X^i_{\alpha^i(n)}$ is the probability of agent i repeating its action, $X^i_{\alpha^i_T}$ is the probability of agent i selecting α^i_T as its next move, and the coefficient τ specifies how likely it is that the agents choose the sub-optimal action based on (3.6) and (3.7).

Note that each agent must have a posteriori knowledge about the utility distribution, i.e. GMM, to be able to calculate their future utility function $u^i(\alpha^i_T, \alpha^{-i}(n), \alpha(n))$ in (3.6) and (3.7). Hence in the case of unknown environment, it is not possible to use BLLL scheme, unless we replace $u^i(\alpha^i_T, \alpha^{-i}(n), \alpha(n))$ with an estimation from the model $\hat{u}^i(\alpha^i_T, \alpha^{-i}(n), \alpha(n))$. This raises the need for an environmental model in the BLLL algorithm.

Remark 11. Exploration in learning algorithms is defined as the ability of the agents to adequently discover their environment to ensure that the best solution has been found. Exploitation is referred to optimally reacting to the environment according to the past interactions with the environment. One of the challenges that arise in learning is maintaining the balance between exploration and exploitation. This balance is essential since without enough exploration the algorithm may fail to find the global optimum and then converge to a local optimum. However, most of the studies done for balancing exploitation and exploration have strong assumptions on agents' prior knowledge about the actions' rewards or assumptions on the utility function itself [37].

In the following we introduce an estimation model algorithm which relaxes the assumption of agents' prior knowledge about the utility distribution.

3.4 Unknown Utility Model

It is common in practice to assume that the game being played is unknown to the agents, i.e. agents do not have any prior information about the utility function and do not know the expected reward that will result from playing a certain action [11]. It is because the environment may be difficult to assess, and the utility distribution may not be completely known. In the following, inspired by [12] we introduce an estimation which can provide a model of the environment in model-based learning.

3.5 Estimation Maximization

Assume the environment is a GMM as (3.4). Agent i's estimation of the mixture parameters $\lambda := \{\omega_j, \mu_j, \sigma_j | j = 1, ..., M\}$ is denoted by $\hat{\lambda}^i = \{\hat{\omega}^i_j, \hat{\mu}^i_j, \hat{\sigma}^i_j | j = 1, ..., M\}$. The estimation model is:

$$\widehat{GMM} := \hat{f}(l) = \sum_{j=1}^{M} \hat{\omega}_j \, g(l | \hat{\mu}_j, \hat{\Sigma}_j). \tag{3.8}$$

During the searching task, the robots will keep their observations of sensed regions in their memories. This information will be further used to form an estimation model of the environment.

Let agent i's observation vector at iteration n be a sequence of its sensed values from time step 1 to n and is denoted by $O^i = \{O^i_1, O^i_2, O^i_3, ..., O^i_n\}$ where O^i_n is defined as the corresponding coordinates of the sensed lattice by agent i at time step n. Estimation Maximization (EM) algorithm is used to find maximum likelihood parameters of a statistical model when there are missing data points from the model. The EM algorithm can estimate the GMM parameters $\hat{\lambda}$ through an iterative algorithm [45]:

$$
\begin{aligned}
\hat{\omega}_j^i &= \frac{1}{n}\sum_{\tau=1}^{n} P^i(j|O_\tau^i, \hat{\lambda}^i), \\
\hat{\mu}_j^i &= \frac{\sum_{\tau=1}^{n} P^i(j|O_\tau^i, \hat{\lambda}^i) O_\tau^i}{\sum_{\tau=1}^{n} P^i(j|O_\tau^i, \hat{\lambda}^i)}, \\
\hat{\sigma}_j^{i\ 2} &= \frac{\sum_{\tau=1}^{n} P^i(j|O_\tau^i, \hat{\lambda}^i)(O_\tau^i - \hat{\mu}_j^i)(O_\tau^i - \hat{\mu}_j^i)^T}{\sum_{\tau=1}^{n} P^i(j|O_\tau^i, \hat{\lambda}^i)}, \\
P^i(j|O_\tau^i, \hat{\lambda}^i) &= \frac{\hat{\omega}_j^i\, g(O_\tau^i|\hat{\mu}_j^i, \hat{\sigma}_j^i)}{\sum_{k=1}^{M} \hat{\omega}_k^i\, g(O_\tau^i|\hat{\mu}_k^i, \hat{\sigma}_k^i)},
\end{aligned}
\tag{3.9}
$$

where $P^i(j|O_\tau^i, \hat{\lambda}^i)$ is often referred to as posteriori probability of the observation vector O_τ^i of agent i for the jth component of Gaussian distribution. Through (3.9), given the current estimated parameters, the EM algorithm estimates the likelihood that each data point belongs to each component. Next, the algorithm maximizes the likelihood to find new parameters of the distribution.

Recall that O^i is the sequence of agent i's observed coordinates within the area. Thus, (3.9) only considers the coordinates of the observed lattice l and disregards its corresponding signal strength $f(l)$. This issue is pointed out in [12] and the proposed solution as introduced in [46] is to repeat the iterative algorithm for m times in worthwhile areas and m is chosen as:

$$
m = \begin{cases} 1 + V\ round(\dfrac{f(l)}{f_{mode}}) & : f(l) \geq f_{mode} \\ 1 & : f(l) < f_{mode} \end{cases}
$$

where $f(l)$ is the worth at coordinate of l, f_{mode} is the threshold above which the signal is considered worthwhile for EM repetition and V is the correction factor which regulates the number of algorithm repetitions for the worthy lattices. The algorithm will run once for areas with a worth value of less than the threshold.

The variation rate of the estimation parameters $\hat{\lambda}^i$ is relatively low due to the rate of update of the observation vector and the nature of the EM algorithm. The case of slow variation of Gaussian distribution is investigated in [47]. By using binary log-linear learning and assuming a slow changing distribution, [47] showed that the agents' estimation error $|\hat{\lambda}(n) - \lambda(n)|$ is decreased by extending the observations vector; consequently, the agents will stochastically converge to a Nash equilibrium. If we assume that the number of the targets (M) is a known parameter, we can use the iterative algorithm discussed in this section.

3.6 EM with Split and Merge

In this section, we present a new modified estimation algorithm based on the EM algorithm. Recall that we assumed the agents do not have any prior knowledge about the environment. This lack of knowledge includes the probability distribution function and also the number of targets. However, as we discussed in Section 3.2, we need to know the value of M (i.e. number of the targets) to use the EM algorithm in (3.9). To overcome this issue, we can also estimate the number of the targets. The Akaike information criterion (AIC) introduced in [48] is considered as the main verification method to help agents select the best estimation for the number of the targets. The Akaike information criterion is a measure of the relative quality of a distribution model for a set of data points. Given a set of models for the same data, AIC estimates the quality of each model, relative to other statistical models. The AIC value of each model is:

$$AIC = 2k - 2ln(L), \tag{3.10}$$

where L is the maximized value of the likelihood and k is the number of estimated parameters in the model. Given a collection of statistical models, the one with minimum AIC value is the best model [48]. As in (3.10) the Akaike criterion considers the goodness of the fit as the likelihood function L and penalty of complexity of the model as k (i.e. the number of model parameters). In our study, different Gaussian distributions with different number of components (i.e. number of targets) have been fitted over the gathered data from the environment for every T_{AIC} iterations. Then, the Akaike information criterion is applied over the collection of the statistical models (that have different number of components) and the model with the lowest AIC has been selected. This procedure starts after a certain number of iterations (t_{AIC}) to let the observation vector expand enough.

Since the number of the components are changing every T_{AIC} iterations, the next step is determining an appropriate method for merging and splitting Gaussian components. The method proposed in [49] incorporates the split and merge operations into the EM algorithm for Gaussian mixture estimations. Moreover, efficient criteria have been proposed in [49] to decide which components should be merged or split. Despite the fact that the number of Gaussian components are changing over time the shape of estimation distribution is not changed significantly since the set of data points is not increasing so fast. In the following the merge and split algorithms are briefly discussed:

3.6.1 Merge

In order to reduce the number of components in a Gaussian distribution, some of the components should be merged together. However, an effective criterion is needed to pick the optimal pairs to merge.

a) Criterion: The posteriori probability of a data point gives a good estimation about to which Gaussian component that data point belongs. If for many data points, the posteriori probabilities are almost equal for two different components, it can be perceived that the components are mergeable. To mathematically implement this, the following criterion for jth and j'th Gaussian components is proposed in [49]:

$$J_{merge}(j, j'; \hat{\lambda}) = \mathbf{P}_j(\hat{\lambda})^T \mathbf{P}_{j'}(\hat{\lambda}), \tag{3.11}$$

where $\mathbf{P}_j(\hat{\lambda}) = (P(j|O_1, \hat{\lambda}), P(j|O_2, \hat{\lambda}), ..., P(j|O_t, \hat{\lambda}))^T$ is a N-dimensional vector consisting of posteriori probabilities of all data points for jth Gaussian component. The criterion $J_{merge}(j, j'; \hat{\lambda})$ must be calculated for all possible pairs and the pair with the largest value is a candidate for the merge.

b) Merging Procedure: In order to merge two Gaussian components, the distribution model parameters must be re-estimated. A modified EM algorithm is proposed in [49] that re-estimates Gaussian parameters based on the former distribution parameters ($\hat{\lambda}$). If the merged Gaussian from the pair of j and j' is denoted by j'' then the initial parameters for the modified EM algorithm is:

$$\begin{aligned}
\omega^0_{j''} &= \omega_j + \omega_{j'}, \\
\mu^0_{j''} &= \frac{\omega_j \mu_j + \omega_{j'} \mu_{j'}}{\omega_j + \omega_{j'}}, \\
\Sigma^0_{j''} &= \frac{\omega_j \Sigma_j + \omega_{j'} \Sigma_{j'}}{\omega_j + \omega_{j'}}.
\end{aligned} \tag{3.12}$$

The initial parameter values calculated by (3.12) are often poor. Hence, the newly generated Guassians should be first processed by fixing the other Gaussians through the modified EM. An EM iterative algorithm then run to re-estimate the distribution parameters. The main steps are the same as (3.9) except the posteriori probability:

$$P(j''|O_\tau, \hat{\lambda}) = \frac{\hat{\omega}_{j''}\; g(O_\tau|\hat{\mu}_{j''}, \hat{\sigma}_{j''})\; \sum_{k=j,j'} P(k|O_\tau, \hat{\lambda})}{\sum_{k=j''} \hat{\omega}_k\; g(O_\tau|\hat{\mu}_k, \hat{\sigma}_k)}. \tag{3.13}$$

By using this modified EM algorithm, the parameters of j''th Gaussian are re-estimated without affecting the other Gaussian components. In our study, the merging algorithm could be repeated for several times if more than one merge step was needed.

3.6.2 Split

In case of a need to increase the number of components in the estimation distribution, we use the split algorithm to split one or more Gaussians. As for the merging process, an appropriate criterion is necessary.

a) Criterion: As the split criterion of kth component, the local Kullback-Leibler divergence is proposed in [49]:

$$J_{split}(k;\hat{\lambda}) = \int p_k(x,\hat{\lambda}) \log\left(\frac{p_k(x,\hat{\lambda})}{g(x|\hat{\mu_k},\hat{\Sigma_k})}\right) dx, \tag{3.14}$$

where $p_k(x,\hat{\lambda})$ is the local data density around kth component and is defined as:

$$p_k(x,\hat{\lambda}) = \frac{\sum_{n=1}^{t} \delta(x-x_n)P(k|x_n,\hat{\lambda})}{\sum_{n=1}^{t} P(k|x_n,\hat{\lambda})}. \tag{3.15}$$

Equation (3.14) actually represents the distance between two Gaussian components where the kth Gaussian is characterized by $\hat{\mu_k}$ and $\hat{\Sigma_k}$. The split criterion, $J_{split}(k,\hat{\lambda})$ must be applied over all candidates and the one with the largest value will be selected.

b) Splitting Procedure: A modified EM algorithm is proposed in [49] to re-estimate the Gaussian parameters. If the split candidate is the kth Gaussian component and the two resulting Gaussians are denoted by j' and k' the initial conditions are calculated as follows:

$$\begin{aligned} \omega_{j'}^0 &= \omega_{k'}^0 = \frac{1}{2}\,\omega_k, \\ \Sigma_{j'}^0 &= \Sigma_{k'}^0 = \det(\Sigma_k)^{1/d}\, I_d, \end{aligned} \tag{3.16}$$

where I_d is the d-dimensional unit matrix and d is the dimension of Gaussian function $g(x|\mu_k,\Sigma_k)$. The mean vectors $\mu_{j'}^0$ and $\mu_{k'}^0$ are determined by applying random perturbation vector ϵ_m, $m = 1,2$ on μ_k as $\mu_{j'}^0 = \mu_k + \epsilon_1$ and $\mu_{k'}^0 = \mu_k + \epsilon_2$ where $||\epsilon_m|| \ll ||\mu_k||$ and $\epsilon_1 \neq \epsilon_2$. The parameters re-estimation for j' and k' can be done by a modified EM algorithm similar to the merge EM algorithm where the modified posteriori probability is

$$P(m'|O_\tau,\hat{\lambda}) = \frac{\hat{\omega}_{m'}\, g(O_\tau|\hat{\mu}_{m\prime},\hat{\sigma}_{m'})\, \sum_{l=k} P(l|O_\tau,\hat{\lambda})}{\sum_{l=j',k'} \hat{\omega}_l\, g(O_\tau|\hat{\mu}_l,\hat{\sigma}_l)}, \tag{3.17}$$

where $m' = j', k'$. The parameters of j' and k' are re-estimated without affecting other Gaussians. Splitting algorithm will be repeated if more than one split was necessary according to the Akaike criterion. By means of AIC and split-merge technique, the agents are able to estimate the number of targets.

3.7 Communication

In this section a mutual information-sharing mechanism is investigated to improve the robots' knowledge about the environment. In this setting each agent can only communicate with its neighbors, i.e. within N_R^i. The agents are picked pairwise to check if they are within each other's communication radius. Thus, all the neighbor robots have the chance to make a contact with other robots in their neighborhood and share their information. The communication cost between agent i and agent i' is defined as:

$$J_{com} = K_{com}\, D_{trans}(i, j, n)\, (|\alpha^i(n) - \alpha^{i'}(n)|), \tag{3.18}$$

where K_{com} is the regulation coefficient and $D_{trans}(i, i', n)$ is the length of transmitted data vector between agent i and agent i' at time step n. This communication cost is modeled as the power consumption needed to transmit data between the sensor nodes in the robotic network based on a wireless communication setting. At each iteration, agents that are in each other's communication radius (R_{com}) are able to contact each other. In order to make a more efficient communication mechanism, a robot pair (i and i') are allowed to communicate with each other if the following conditions are satisfied in addition to being in each other's communication radius:

1. One of the agents (i) has a larger cumulative covered worth (with a certain ratio) than the other member of the communication pair (i'). In other words:

$$\sum_{\tau=1}^{n} \sum_{k=1}^{\hat{M}} \omega_k\, g(O_\tau^i | \mu_k, \Sigma_k) \geq K'_{com} \sum_{\tau=1}^{n} \sum_{k=1}^{\hat{M}} \omega_k\, g(O_\tau^{i'} | \mu_k, \Sigma_k), \tag{3.19}$$

where K'_{com} is a regulating coefficient (ratio) and $\hat{M}$ is the estimation of the number of the Gaussian components.

2. The length of the observation vector of agent i is larger than that of agent i' with an specified ratio:

$$num^i \geq K''_{com}\, num^{i'}, \tag{3.20}$$

where K''_{com} is the ratio coefficient and num^l is the number of unique observations of agent l and $l = i, i'$.

Based on the above conditions we can claim that the agent i is more experienced than agent i' and a communication between agent i and agent i' could improve the knowledge of agent i' about the area which results in an enhancement in the overall performance of the whole system. During communication, agent i' that is inexperienced and possesses a lower data volume, transmits its observation vector along with its distribution estimation parameters (i.e. $\hat{\omega}^{i'}$,$\hat{\mu}^{i'}$ and $\hat{\Sigma}^{i'}$) to agent i. Hence, in this scenario, agent i carries out the computation burden. The computation consists of re-estimation of the Gaussian distribution based on the observation vectors of agent i and agent i' together; the aggregated vector of O^i and $O^{i'}$ is denoted by $O^{i+i'}$. The order of elements in $O^{i+i'}$ does not matter since the re-estimation algorithm, which will be explained in the following chapter, treats $O^{i+i'}$ as a set of data points. Hence, for simplicity we only add $O^{i'}$ at the bottom of O^i. The re-estimation algorithm is basically an EM algorithm with the following initial condition:

$$\hat{\Theta}^r_0 = \frac{E^i \hat{\Theta}^i + E^{i'} \hat{\Theta}^{i'}}{E^i + E^{i'}}, \tag{3.21}$$

where Θ could be ω, μ or σ, and E^l is defined as:

$$E^l = num^l \sum_{\tau=1}^{n} \sum_{k=1}^{M} \omega_k\, g(O^l_\tau | \mu_k, \Sigma_k),\ l = i, i'. \tag{3.22}$$

Equation (3.21) actually represents a weighted average of the estimation parameters of agent i and agent i' which takes the experience of each agent into account. The initial conditions are calculated based on (3.21) and the observation vector which is fed into the estimation algorithm is $O^{i+i'}$. Finally, the resulting parameters $\hat{\omega}^r$, $\hat{\mu}^r$ and $\hat{\sigma}^r$ will replace

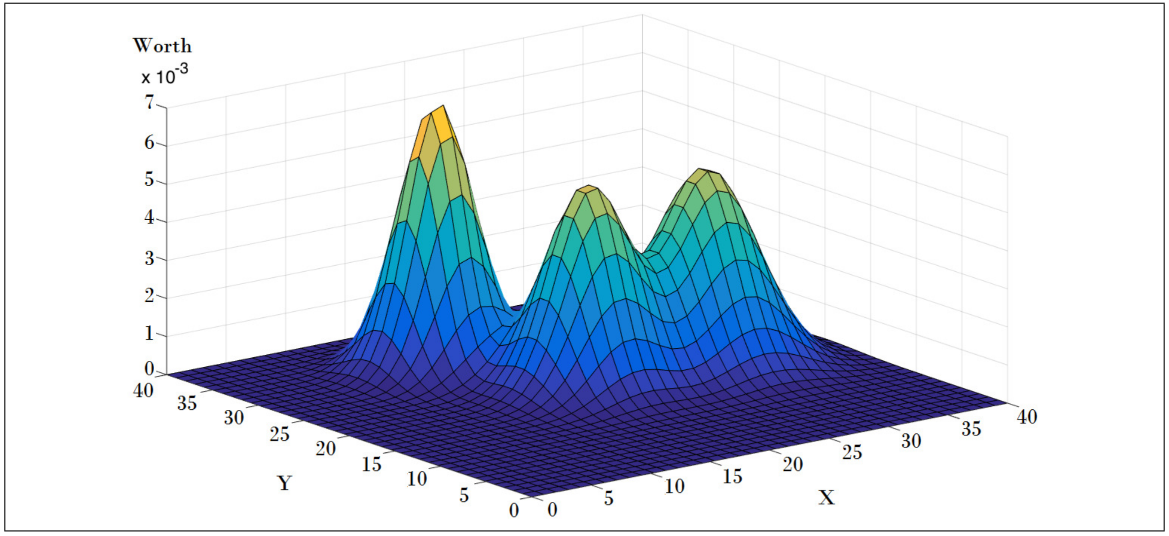

Figure 3.2. Gaussian distribution in the environment (signal strengths are exaggerated for illustrative purposes)

old estimation parameters of both agent i and agent i'. In this thesis, the communication cost is not implemented in the agents' utility function and is only considered as a performance comparison mean between different methods.

3.8 Simulation Results

In this section we provide a numerical example for the MCC problem formulated in this chapter, to verify performance of the proposed EM algorithm. A 40×40 square area with 1×1 lattices has been considered as the environment. The Gaussian distribution in this area has been chosen randomly with different but reasonable weight, mean and covariance values (Fig.3.2). The number of targets (Gaussian components) is between 1 to 5 and robots have no prior knowledge about this number.

A group of five robots ($N = 5$) scatter through the area to maximize their utility function (Fig 3.3). Recall that players' utility functions are separable. Therefore, with a set of stationary targets, utility distribution remains stationary for each agent as it is not changed by the actions of other players.

Note that the number of the targets could be higher than the number of the robots

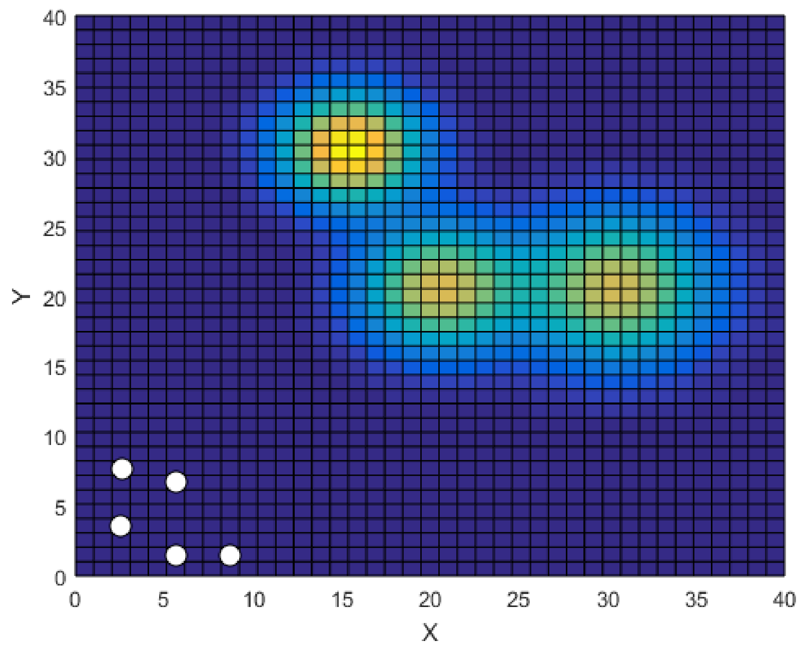

Figure 3.3. Gaussian distribution in the environment (signal strengths are exaggerated for illustrative purposes)

but the robots need more time to explore the environment and gather enough data. The objective of mobile agents is to maximize the cover of the worthwhile areas as well as to minimize the energy consumption regarding relocation.

The simulation parameters are chosen as $K_i = 3 \times 10^{-5}$ for $i = 1, ..., N$, $\tau = 5 \times 10^{-4}$, $f_{mode} = 10^{-5}$, $V = 0.1$, $\delta = 1.5$, $T_{AIC} = t_{AIC} = 100$, $K_{com} = 1$, $K'_{com} = K''_{com} = 1.5$ and $R_{com} = 5$. Note that an increase in K_i results agents not to significantly deviate from their current location. Parameter τ controls how rational the robots are, in their learning process. The importance of worth distribution from agents' point of view is regulated by f_{mode} and V. We need reasonable values of T_{AIC} and t_{AIC} to let the agents collect enough data to accurately estimate the number of targets. Communication parameters can be regulated according to the on-board power limitations. However, it is obvious that the wider the communication the more accurate the estimation. The following is the algorithm's pseudo-code:

```
set $n = 1$
for each robot $i \in \mathcal{I}$ do
initialize $\alpha^i(1) \in \mathcal{L}$ randomly
while the covered worth $\sum_{i\in\mathcal{I}} C^i(\alpha^i)$ is not in a steady state do
    if $T_{AIC}$ is a sub-multiple of $n$ then
        process merge or split
    $i \leftarrow (n \, mod \, N) + 1$
    robot $i$ chooses a trial action $\alpha^i_T$ randomly from $\mathcal{A}^i_c$
    calculate $X^i_{\alpha^i(n)}$ and $X^i_{\alpha^i_T}$ by using both $GMM$ and $\widehat{GMM}$ (see (3.6) and (3.7))
    $\alpha^i(n+1) \leftarrow \alpha^i(n)$ with probability $X^i_{\alpha^i(n)}$ or $\alpha^i(n+1) \leftarrow \alpha^i_T$ with probability $X^i_{\alpha^i_T}$
    $\alpha^{-i}(n+1) \leftarrow \alpha^{-i}(n)$
    if $\alpha^i(n+1)$ is $\alpha^i_T$ then
        add $\alpha^i(n+1)$ to $O^i$
        run EM and update $\widehat{GMM}$
    $n \leftarrow n + 1$
```

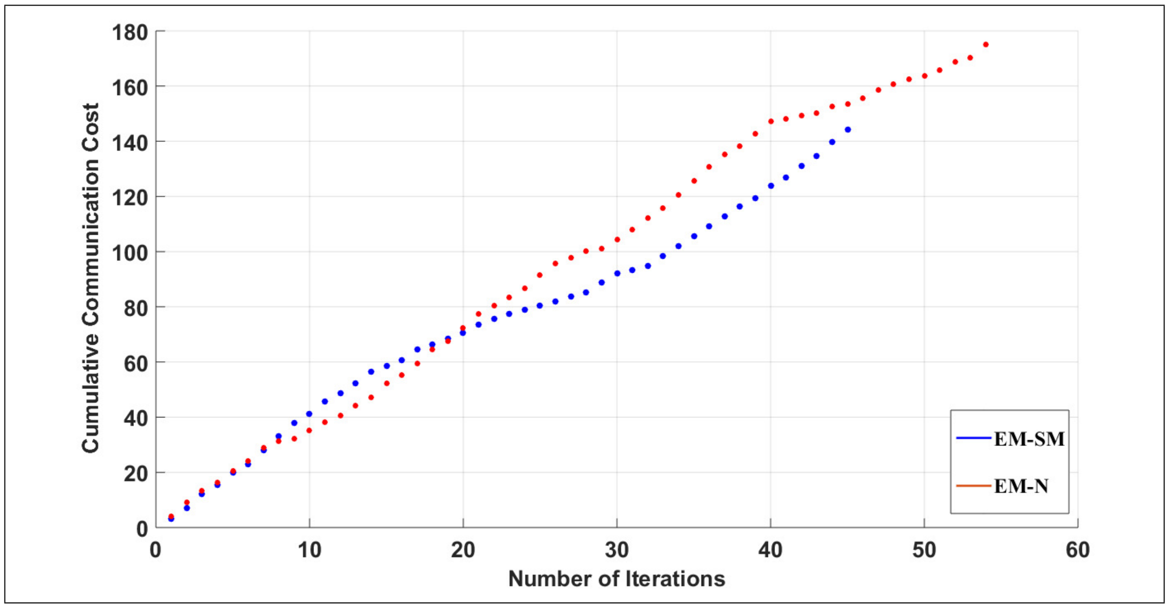

Figure 3.4. Cumulative communication cost

In this chapter, the model-based learning algorithm with the standard EM is called EM_N and the model-based learning algorithm with the proposed EM is called EM_SM. The worth of covered area using a normal EM algorithm (EM_N, Section 3.5) and our proposed algorithm (EM_SM, Section 3.6) is illustrated in Fig.3.7. In EM_N method agents

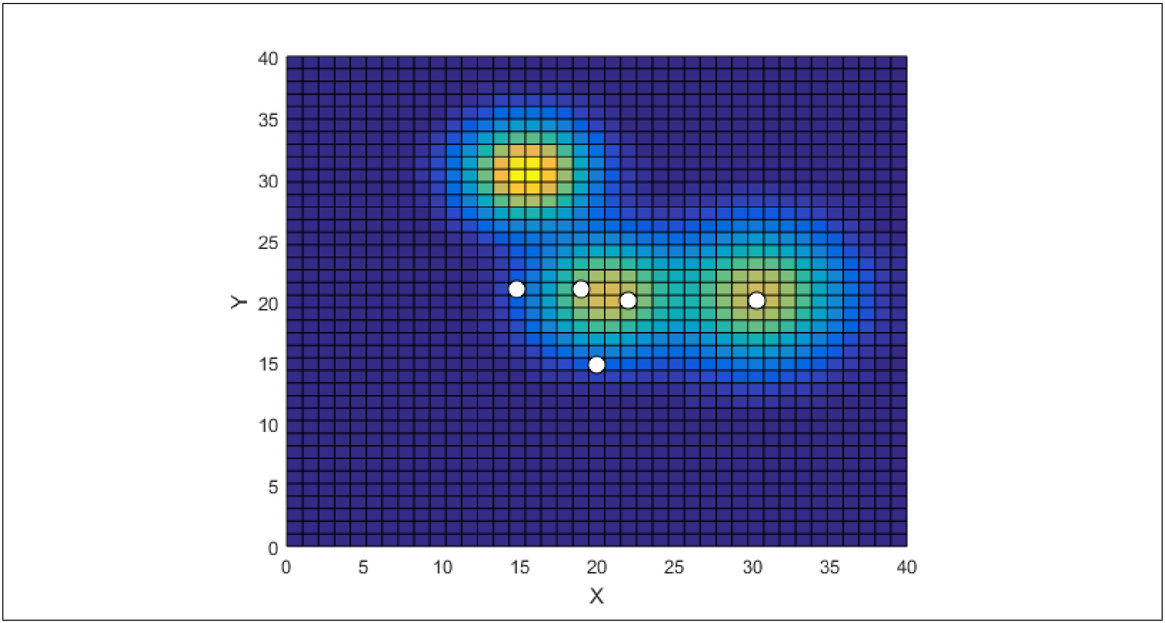

Figure 3.5. Final configuration of the robots in EM_N

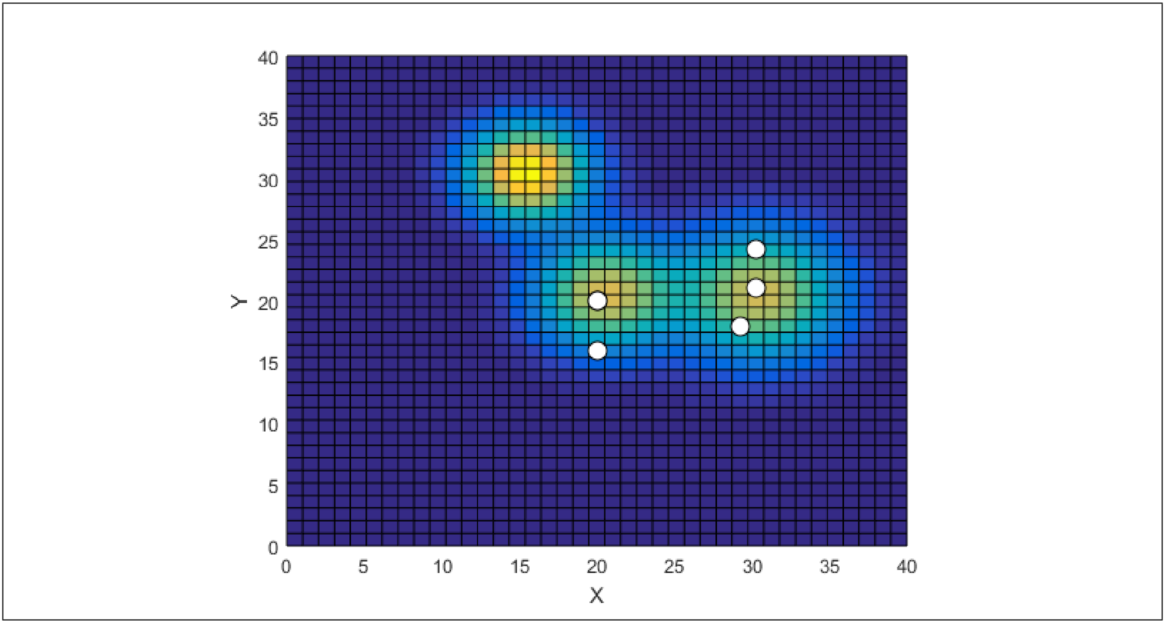

Figure 3.6. Final configuration of the robots in EM_SM

do not have a prior knowledge about the location of targets but the exact number of the targets is known to the agents. In addition to the lack of knowledge about the targets' location, the number of the targets is also unknown in EM_SM.

As in Fig.3.7, both algorithms reached the same steady state point with almost the same rate. The algorithms ran for 10 times with different initial conditions and for each algorithm Fig.3.7 presents a bound and a mean for the coverage worth. The final configuration of EM_N and EM_SM are presented in Fig 3.5 and Fig 3.6 respectively. Although the agents located two targets in the area they decided not to explore for the third target. It could be due to the trade off between the coverage worth and the energy consumption.

The communication cost of both methods (EM_N and EM_SM) is shown in Fig.3.4. In EM_SM, agents attempted to communicate with each other 45 times while in EM_N it was 54. One interpretation is that all the agents gained a uniform average experience through EM_SM; so the number of communication attempts is less than EM_N. Furthermore, the

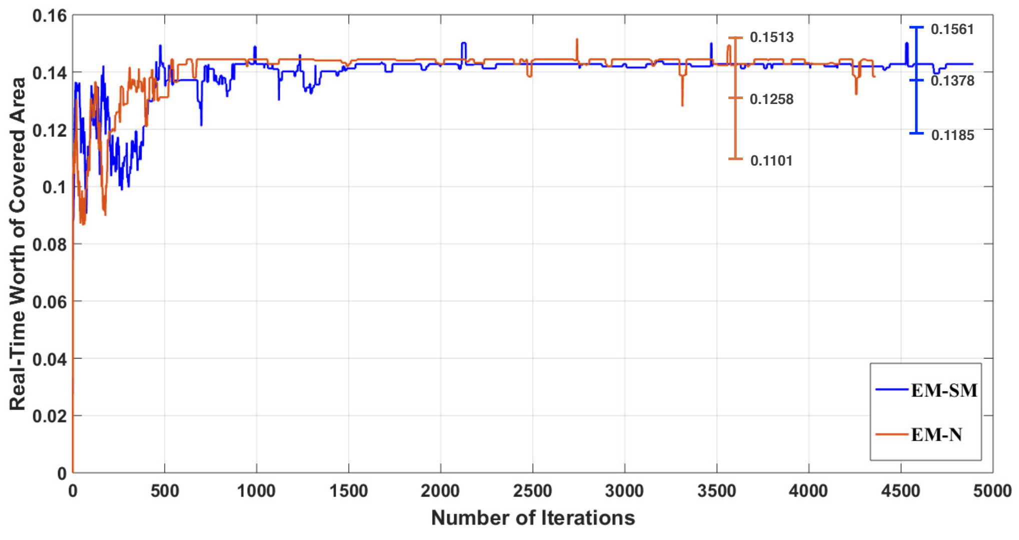

Figure 3.7. The worth of the covered area

amount of data bits transferred in EM_SM is less than EM_N which means the agents consume less energy to send data bits in EM_SM while maintained their level of collaboration (data sharing) in an acceptable range.

3.9 Discussion

In this chapter, a MCC problem has been investigated in which a finite number of robots seek to cover the areas with the highest probability of existence of targets. The robots have no familiarity with the environment as they do not have any information either about the location or the number of the targets. A direct application of this setup is search-and-rescue in which a number of victims have to be found in an unknown area. The signal distribution in the area could be victims' audio signal or other vital signals such as body temperature.

A potential game was formulated to relate the robots together and to optimize agents' actions. The reward of sensing valuable areas is considered in the utility function as well as the penalty of energy consumption due to the agents' movement. Update of agents' action profile is based on BLLL algorithm in which agents need to know an estimation of the outcome of their future actions. Hence, an estimation algorithm has been utilized to assist agents in anticipating the probability of the targets' existence in undiscovered areas

A modified EM algorithm has been proposed to estimate the number of the targets as well as other parameters of the probability distribution. We also discussed a communication scheme and the cost of the communication (proportional to the volume of transmitted data and agents distance).

Recall that in BLLL only one robot is selected at each time and allowed to update its action. We guess that this one-time-one-agent update rule reduces the whole algorithm's convergence rate. Furthermore, in this update rule, agents are selected randomly to update their actions, which is not very efficient. In Chapter 4, we relax these limitations in BLLL.

Chapter 4

Synchronous Binary Log Linear Learning

4.1 Introduction

In this chapter, we present a modified LLL algorithm in which both asynchrony and complete action set assumptions are relaxed. This means that in a Synchronous Binary Log Linear Learning (SBLLL) scheme agents can learn simultaneously while their available action sets are constrained. This simultaneous learning feature presumably increases the BLLL learning rate in the MCC problem, described in Chapter 3. The convergence analysis of the proposed algorithm is based on Markov decision process and the theory of resistance trees presented in [18].

4.2 The Proposed Algorithm

Recall that in BLLL algorithm, at each iteration n, one random player is selected to try an alternative action. In SBLLL algorithm, we propose that at each time n, a set of players $S(n) \subseteq \mathcal{I}$ independently update their actions according to each player i's revision probability rp^i. The revision probability rp^i, which depends on agent i's action at time n, is described by the probability with which agent i wakes up to update its action. All the other players $\mathcal{I} \setminus S(n)$ must repeat their current actions.

Each player $i \in S(n)$ selects one trial action α_T^i uniformly randomly from his constrained action set $\mathcal{A}_c^i(\alpha^i(n))$. Let α_T be the action profile for which each player $i \in S(n)$ updates its action to α_T^i and all the other players $\mathcal{I} \setminus S(n)$ repeat their actions. Then, player i's mixed strategy is

$$X^i_{\alpha^i(n)}(n) = \frac{\exp\left(\frac{1}{\tau}u^i(\alpha(n))\right)}{\exp\left(\frac{1}{\tau}u^i(\alpha(n))\right) + \exp\left(\frac{1}{\tau}u^i(\alpha_T)\right)}, \tag{4.1}$$

$$X^i_{\alpha^i_T}(n) = \frac{\exp\left(\frac{1}{\tau}u^i(\alpha_T)\right)}{\exp\left(\frac{1}{\tau}u^i(\alpha(n))\right) + \exp\left(\frac{1}{\tau}u^i(\alpha_T)\right)}, \tag{4.2}$$

where $X^i_{\alpha^i(n)}$ is the probability that player i remains on his previous action $\alpha^i(n)$ and $X^i_{\alpha^i_T}$ is the probability that player i selects the trial action α^i_T. In the following, we analyze the convergence of the proposed algorithm.

4.3 Convergence Analysis

This section studies the convergence of the SBLLL by using the theory of resistance trees reviewed in Chapter 2. From this theory, we use the useful relationship between stochastically stable states and potential maximizing states.

Assumption 4.1 *For each player i, and for each action α^i, the revision probability probability rp^i must be bounded and* $0 < rp^i(\alpha^i(n)) < 1$.

Assumption 4.2 *For each player i and for any action pair $\alpha^i_1, \alpha^i_2 \in \mathcal{A}^i$,*

$$\alpha^i_2 \in \mathcal{A}^i_c(\alpha^i_1) \;\Leftrightarrow\; \alpha^i_1 \in \mathcal{A}^i_c(\alpha^i_2).$$

Lemma 4.1 *Under Assumption 4.1, SBLLL induces a perturbed Markov process where the resistance of any feasible transition $\alpha_1 \in \mathcal{A} \to \alpha_2 \in \mathcal{A}$ with deviating set of players S is*

$$R(\alpha_1 \to \alpha_2) = \sum_{i \in S} \max\{u^i(\alpha_1), u^i(\alpha_2)\} - u^i(\alpha_2). \tag{4.3}$$

Proof: Let P_ϵ denote the perturbed transition matrix. The probability of the transition from α_1 to α_2 is

$$P_{\epsilon(\alpha_1 \to \alpha_2)} = \prod_{i \in S} \frac{rp^i(\alpha_1^i)}{|\mathcal{A}_c^i(\alpha_1^i)|} \prod_{j \in \mathcal{I} \setminus S} (1 - rp^j(\alpha_1^j)) \prod_{i \in S} \frac{\epsilon^{-u^i(\alpha_2)}}{\epsilon^{-u^i(\alpha_1)} + \epsilon^{-u^i(\alpha_2)}}, \tag{4.4}$$

where $\epsilon := e^{-1/\tau}$. The first term $\prod_{i \in S} \dfrac{rp^i(\alpha_1^i)}{|\mathcal{A}_c^i(\alpha_1^i)|}$ represents the probability that all the players in S, wake up to change their actions from α_1 to α_2. The second term $\prod_{j \in \mathcal{I} \setminus S}(1 - rp^j(\alpha_1^j))$ is the probability that the players in $\mathcal{I} \setminus S$ stay asleep. The last term $\prod_{i \in S} \dfrac{\epsilon^{-u^i(\alpha_2)}}{\epsilon^{-u^i(\alpha_1)} + \epsilon^{-u^i(\alpha_2)}}$ is the binary SBR over α_1 and α_2.
Next define the maximum utility of player i for any two action profiles α_1 and α_2 as

$$V^i(\alpha_1, \alpha_2) = \max\{u^i(\alpha_1), u^i(\alpha_2)\}.$$

By multiplying the numerator and denominator of (4.4) by $\prod_{i \in S} \epsilon^{V^i(\alpha_1, \alpha_2)}$ we obtain

$$P_{\epsilon(\alpha_1 \to \alpha_2)} = \prod_{i \in S} \frac{rp^i(\alpha_1^i)}{|\mathcal{A}_c^i(\alpha_1^i)|} \prod_{j \in \mathcal{I} \setminus S} (1 - rp^j(\alpha_1^j)) \prod_{i \in S} \frac{\epsilon^{V^i(\alpha_1,\alpha_2) - u^i(\alpha_2)}}{\epsilon^{V^i(\alpha_1,\alpha_2) - u^i(\alpha_1)} + \epsilon^{V^i(\alpha_1,\alpha_2) - u^i(\alpha_2)}}. \tag{4.5}$$

Dividing (4.5) by $\epsilon^{\sum_{i \in S} V^i(\alpha_1,\alpha_2) - u^i(\alpha_2)}$ follows:

$$\begin{aligned} &\frac{P_{\epsilon(\alpha_1 \to \alpha_2)}}{\epsilon^{\sum_{i \in S} V^i(\alpha_1,\alpha_2) - u^i(\alpha_2)}} = \\ &\prod_{i \in S} \frac{rp^i(\alpha_1^i)}{|\mathcal{A}_c^i(\alpha_1^i)|} \prod_{j \in \mathcal{I} \setminus S} (1 - rp^j(\alpha_1^j)) \prod_{i \in S} \frac{1}{\epsilon^{V^i(\alpha_1,\alpha_2) - u^i(\alpha_1)} + \epsilon^{V^i(\alpha_1,\alpha_2) - u^i(\alpha_2)}}. \end{aligned} \tag{4.6}$$

According to (2.23) and (2.24), if $0 < \lim_{\epsilon \to 0} \frac{P_{\epsilon(\alpha_1 \to \alpha_2)}}{\epsilon^{\sum_{i \in S} V^i(\alpha_1,\alpha_2) - u^i(\alpha_2)}} < \infty$ then our guess about $R(\alpha_1 \to \alpha_2)$ in (4.3) is true. Considering the definition of $V^i(\alpha_1, \alpha_2)$, we know that for each player i, either $V^i(\alpha_1, \alpha_2) - u^i(\alpha_1)$ or $V^i(\alpha_1, \alpha_2) - u^i(\alpha_2)$ is zero and the other one is a positive real number. Thus, by taking the limit, $\prod_{i \in S} \frac{1}{\epsilon^{V^i(\alpha_1,\alpha_2) - u^i(\alpha_1)} + \epsilon^{V^i(\alpha_1,\alpha_2) - u^i(\alpha_2)}}$ in (4.6) approaches one and

$$\lim_{\epsilon \to 0} \frac{P_{\epsilon(\alpha_1 \to \alpha_2)}}{\epsilon^{\sum_{i \in S} V^i(\alpha_1,\alpha_2) - u^i(\alpha_2)}} = \prod_{i \in S} \frac{rp^i(\alpha_1^i)}{|\mathcal{A}_c^i(\alpha_1^i)|} \prod_{j \in \mathcal{I} \setminus S} (1 - rp^j(\alpha_1^j)). \tag{4.7}$$

From Assumption 4.1, $\prod_{i \in S} \frac{rp^i(\alpha_1^i)}{|\mathcal{A}_c^i(\alpha_1^i)|}$ and $\prod_{j \in \mathcal{I} \setminus S} (1 - rp^j(\alpha_1^j))$ are finite positive real numbers. Hence,

$$0 < \lim_{\epsilon \to 0} \frac{P_{\epsilon(\alpha_1 \to \alpha_2)}}{\epsilon^{\sum_{i \in S} max\{u^i(\alpha_1), u^i(\alpha_2)\} - u^i(\alpha_2)}} < \infty. \tag{4.8}$$

Therefore, the process P_ϵ is a perturbed Markov process where the resistance of the transition α_1 to α_2 is

$$R(\alpha_1 \to \alpha_2) = \sum_{i \in S} \max\{u^i(\alpha_1), u^i(\alpha_2)\} - u^i(\alpha_2).$$

$\square$

Lemma 4.2 *Consider any finite N-player potential game where all the players adhere to SBLLL and the potential function is defined as $\phi : \mathcal{A} \to \mathbb{R}$. Assume that players' utility functions are separable. For any feasible transition $\alpha_1 \in \mathcal{A} \to \alpha_2 \in \mathcal{A}$ with deviating set of players S*

$$R(\alpha_1 \to \alpha_2) - R(\alpha_2 \to \alpha_1) = \phi(\alpha_1) - \phi(\alpha_2). \tag{4.9}$$

Proof: From Lemma 4.1,

$$R(\alpha_1 \to \alpha_2) = \sum_{i \in S_{12}} \max\{u^i(\alpha_1), u^i(\alpha_2)\} - u^i(\alpha_2),$$

and

$$R(\alpha_2 \to \alpha_1) = \sum_{i \in S_{21}} \max\{u^i(\alpha_2), u^i(\alpha_1)\} - u^i(\alpha_1),$$

where S_{12} is the set of deviating players during the transition $\alpha_1 \to \alpha_2$ and S_{21} is the set of deviating players in the transition $\alpha_2 \to \alpha_1$. By Assumption 4.2, if the transition $\alpha_1 \to \alpha_2$ is possible then there exist a reverse transition $\alpha_2 \to \alpha_1$. Clearly, the same set of deviating players is needed for both $\alpha_1 \to \alpha_2$ and $\alpha_2 \to \alpha_1$, i.e. $S_{12} = S_{21} = S$. Therefore:

$$\begin{aligned}
&R(\alpha_1 \to \alpha_2) - R(\alpha_2 \to \alpha_1) = \\
&\Big[\sum_{i \in S_{12}} \max\{u^i(\alpha_1), u^i(\alpha_2)\} - u^i(\alpha_2)\Big] - \Big[\sum_{i \in S_{21}} \max\{u^i(\alpha_2), u^i(\alpha_1)\} - u^i(\alpha_1)\Big] = \\
&\sum_{i \in S} \Big[\max\{u^i(\alpha_1), u^i(\alpha_2)\} - u^i(\alpha_2) - \max\{u^i(\alpha_2), u^i(\alpha_1)\} + u^i(\alpha_1)\Big].
\end{aligned} \tag{4.10}$$

By canceling identical terms $max\{u^i(\alpha_2), u^i(\alpha_1)\}$,

$$R(\alpha_1 \to \alpha_2) - R(\alpha_2 \to \alpha_1) = \sum_{i \in S} u^i(\alpha_1) - u^i(\alpha_2). \tag{4.11}$$

Since players' utility functions are separable, for any player i, $u^i(\alpha_1) - u^i(\alpha_2) = u^i(\alpha_1^i, \alpha_1^{-i}) - u^i(\alpha_2^i, \alpha_1^{-i})$. From the assumption of potential game and from (2.2)

$$u^i(\alpha_1^i, \alpha_1^{-i}) - u^i(\alpha_2^i, \alpha_1^{-i}) = \phi(\alpha_1^i, \alpha_1^{-i}) - \phi(\alpha_2^i, \alpha_1^{-i}). \tag{4.12}$$

By (4.12), it is easy to show that

$$\sum_{i \in S} u^i(\alpha_1) - u^i(\alpha_2) = \phi(\alpha_1) - \phi(\alpha_2). \tag{4.13}$$

$\square$

Next, we prove that in separable games, i.e. games in which the utility function of each player only depends on his action, the stable states of the algorithm maximize the potential function. In other words, the stable states are the optimal states.

Proposition 4.1 *Consider any finite N-player potential game with potential function $\phi : \mathcal{A} \to \mathbb{R}$ where all players adhere to SBLLL. If the utility functions for all players are separable, then the stochastically stable states are the set of potential maximizers.*

Proof: We first show that for any path Ω, defined as a sequence of action profiles $\Omega := \{\alpha_0 \to \alpha_1 \to ... \to \alpha_m\}$ and its reverse path $\Omega^R := \{\alpha_m \to \alpha_{m-1} \to ... \to \alpha_0\}$, the resistance difference is

$$R(\Omega) - R(\Omega^R) = \phi(\alpha_0) - \phi(\alpha_m).$$

where

$$R(\Omega) := \sum_{k=0}^{m-1} R(\alpha_k \to \alpha_{k+1}) \tag{4.14}$$

$$R(\Omega^R) := \sum_{k=0}^{m-1} R(\alpha_{k+1} \to \alpha_k) \tag{4.15}$$

Since players' utilities are separable, each transition $\alpha_k \to \alpha_{k+1}$ in these paths can be considered as a sequence of sub-edges in which only one agent is deviating. Assuming S as the set of deviating players for the edge $\alpha_k \to \alpha_{k+1}$ in Ω, the expanded form of this edge can be written as

$$\alpha_k \to \alpha_{k+1} = \{\alpha_k = \alpha_{k_0} \to \alpha_{k_1} \to \alpha_{k_2} \to ... \to \alpha_{k_{q-1}} \to \alpha_{k_q} = \alpha_{k+1}\},$$

where $\alpha_{k_i} \to \alpha_{k_{i+1}}$ is a sub-edge in which only one player is deviating and $q = |S|$. From Lemma 4.2

$$R(\alpha_{k_i} \to \alpha_{k_{i+1}}) - R(\alpha_{k_{i+1}} \to \alpha_{k_i}) = \phi(\alpha_{k_i}) - \phi(\alpha_{k_{i+1}}).$$

Note that $|S| = 1$ for each sub-edge. Consequently for each edge $\alpha_k \to \alpha_{k+1} = \{\alpha_k =$

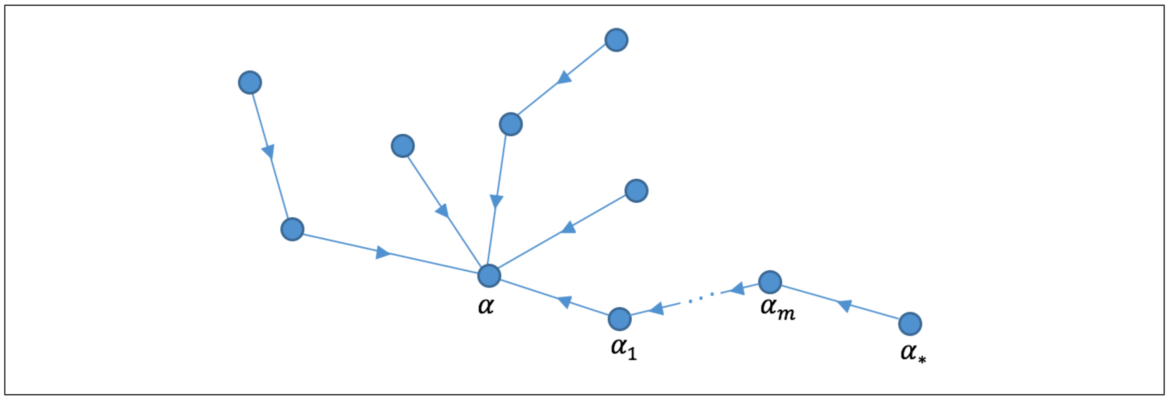

Figure 4.1. Tree T rooted at α

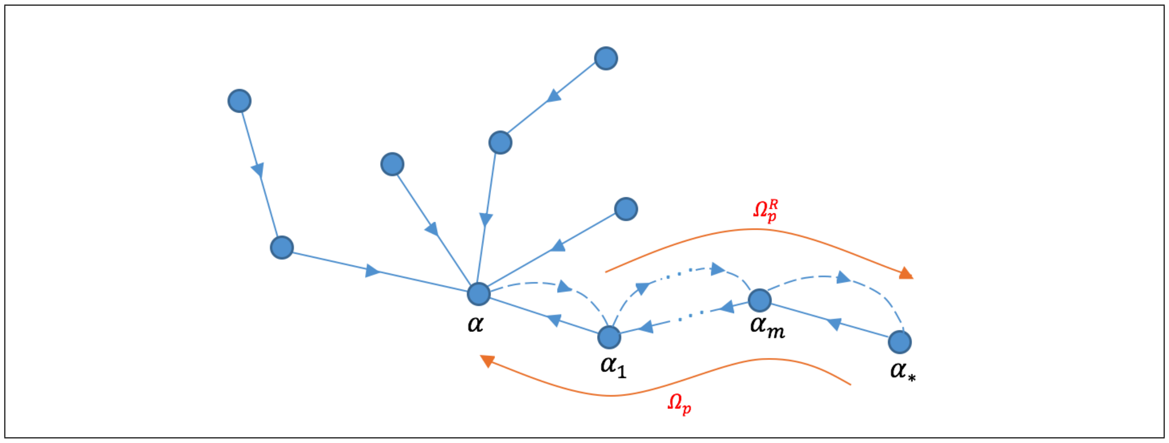

Figure 4.2. Ω_p and its reverse path Ω_p^R

$\alpha_{k_0} \to \alpha_{k_1} \to \alpha_{k_2} \to \ldots \to \alpha_{k_{q-1}} \to \alpha_{k_q} = \alpha_{k+1}\}$ we obtain

$$R(\alpha_k \to \alpha_{k+1}) - R(\alpha_{k+1} \to \alpha_k) = \phi(\alpha_{k_0}) - \phi(\alpha_{k_1}) + \phi(\alpha_{k_1}) - \phi(\alpha_{k_2}) + \ldots + \phi(\alpha_{k_{q-1}}) - \phi(\alpha_{k_q}).$$

By canceling the identical terms,

$$R(\alpha_k \to \alpha_{k+1}) - R(\alpha_{k+1} \to \alpha_k) = \phi(\alpha_{k_0}) - \phi(\alpha_{k_q}) = \phi(\alpha_k) - \phi(\alpha_{k+1}). \tag{4.16}$$

Comparing (4.16) and (4.9) implies that if the utility functions of all players are separable, the number of deviating players does not affect the resistance change between forward and backward transitions. Finally, we sum up the resistance difference in (4.16) for all pairs

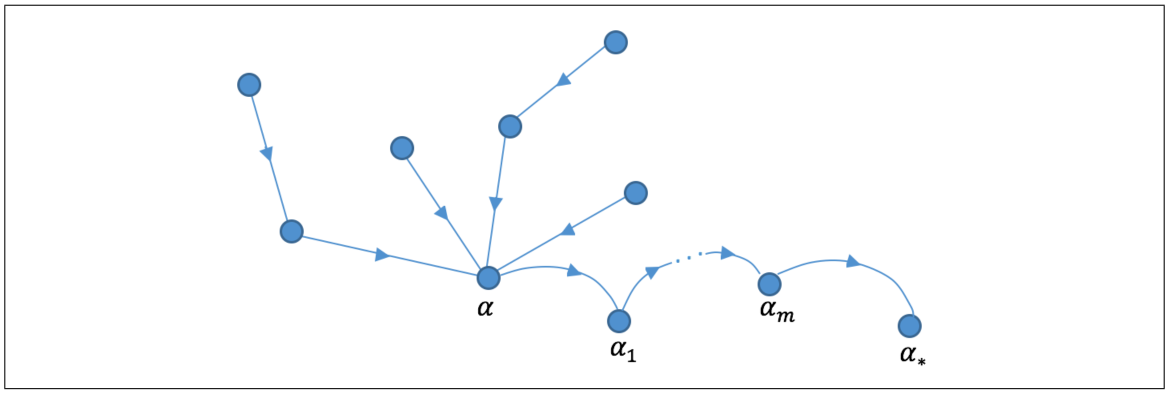

Figure 4.3. Tree T' rooted at α_*

$(\alpha_k, \alpha_{k+1}),\ k \in \{0, 1, ..., m-1\}$:

$$\sum_{k=0}^{m-1} R(\alpha_k \to \alpha_{k+1}) - R(\alpha_{k+1} \to \alpha_k) = \sum_{k=0}^{m-1} \phi(\alpha_k) - \phi(\alpha_{k+1}), \tag{4.17}$$

or equivalently

$$\sum_{k=0}^{m-1} R(\alpha_k \to \alpha_{k+1}) - \sum_{k=0}^{m-1} R(\alpha_{k+1} \to \alpha_k) = \sum_{k=0}^{m-1} \phi(\alpha_k) - \phi(\alpha_{k+1}), \tag{4.18}$$

From (4.14) $\sum_{k=0}^{m-1} R(\alpha_k \to \alpha_{k+1})$ is the resistance over the path Ω and from (4.15) $\sum_{k=0}^{m-1} R(\alpha_{k+1} \to \alpha_k)$ is the resistance over the reverse path Ω^R. Furthermore, it is easy to show $\sum_{k=0}^{m-1} \phi(\alpha_k) - \phi(\alpha_{k+1}) = \phi(\alpha_0) - \phi(\alpha_m)$. Consequently,

$$R(\Omega) - R(\Omega^R) = \phi(\alpha_0) - \phi(\alpha_m). \tag{4.19}$$

Now assume that an action profile α is a stochastically stable state. Therefore, there exist a tree T rooted at α (Fig. 4.1) for which the resistance is minimum among the trees rooted at other states (see Section 2.2.2.2 or Lemma 1 in [18]). We use contradiction to prove our claim. As the contradiction assumption, suppose the action profile α does not maximize the potential and let α_* be the action profile that maximizes the potential function. Since T is rooted at α, there exist a path Ω_p from α_* to α as $\Omega_p = \{\alpha_* \to \alpha_1 \to ... \to \alpha\}$.

Consider the reverse path Ω_p^R from α to α_* (Fig. 4.2)

$$\Omega_p^R = \{\alpha \to \alpha_m \to ... \to \alpha_*\}.$$

We can construct a new tree T' rooted at α_* by adding the edges of Ω_p^R to T and removing the edges of Ω_p (Fig. 4.3). The resistance of the new tree T' is

$$R(T') = R(T) + R(\Omega_p^R) - R(\Omega_p).$$

By (4.19),

$$R(T') = R(T) + \phi(\alpha) - \phi(\alpha_*).$$

Recall that we assumed $\phi(\alpha_*)$ is the maximum potential. Hence $\phi(\alpha) - \phi(\alpha_*) < 0$ and consequently $R(T') < R(T)$. Therefore T is not a minimum resistance tree among the trees rooted at other states, which is in contrast with the basic assumption about the tree T. Hence, the supposition is false and α is the potential maximizer. We can use the above analysis to show that all action profiles with maximum potential have the same stochastic potential (φ) which means all the potential maximizers are stochastically stable. □

In the following we analyze the algorithm's convergence for the general case of non-separable game under extra assumptions.

Assumption 4.3 *For any two player i and j, if $\phi(\alpha_2^i, \alpha^{-i}) \geq \phi(\alpha_1^i, \alpha^{-i})$ then:*

$$u^i(\alpha_2^i, \alpha^{-i}) \geq u^j(\alpha_1^i, \alpha^{-i}).$$

Theorem 4.1 *Consider a finite potential game where all players adhere to SBLLL with independent revision probability. Assume that the potential function is concave and Assumption 4.3 holds. If an action profile is stochastically stable then it is potential function maximizer.*

Proof: As reviewed in Chapter 2, from the theory of resistance trees, the state α is stochastically stable if and only if there exists a minimum resistance tree T rooted at α (see Theorem 3.1 in [35]). Recall that a minimum resistance tree rooted at α has a minimum resistance among the trees rooted at other states. We use contradiction to prove that α also maximizes the potential function. As the contradiction assumption, suppose the action profile α does not maximize the potential. Let α_* be any action profile that maximizes the potential ϕ. Since T is rooted at α, there exists a path from α_* to α: $\mathit{\Omega} = \{\alpha_* \to \alpha\}$. We can construct a new tree T', rooted at α_* by adding the edges of $\mathit{\Omega}^R = \{\alpha \to \alpha_*\}$ to T and removing the edges of $\mathit{\Omega}$. Hence,

$$R(T') = R(T) + R(\mathit{\Omega}^R) - R(\mathit{\Omega}). \tag{4.20}$$

If the deviator at each edge is only one player, then the algorithm reduces to BLLL (see Appendix IV). Suppose there exist an edge $\alpha_q \to \alpha_{q+1}$ in $\mathit{\Omega}^R$ with multiple deviators. The set of deviators is denoted by S. If we show $R(\mathit{\Omega}^R) - R(\mathit{\Omega}) \leq 0$ then T is not the minimum resistance tree and therefore, supposition is false and α is actually the potential maximizer. Note that since players' utilities may not be separable, the utility of each player depends on the actions of other players.
From Lemma 4.1, for any transition $\alpha_1 \to \alpha_2$ in $\mathit{\Omega}^R$,

$$R(\alpha_1 \to \alpha_2) = \sum_{i \in S_{12}} \max\{u^i(\alpha_1), u^i(\alpha_2)\} - u^i(\alpha_2),$$

and

$$R(\alpha_2 \to \alpha_1) = \sum_{i \in S_{21}} \max\{u^i(\alpha_2), u^i(\alpha_1)\} - u^i(\alpha_1),$$

where S_{12} is the set of deviating players during the transition $\alpha_1 \to \alpha_2$. By Assumption

4.2, $S_{12} = S_{21} = S$. Therefore:

$$\begin{aligned}
&R(\alpha_1 \to \alpha_2) - R(\alpha_2 \to \alpha_1) = \\
&\Big[\sum_{i\in S_{12}} \max\{u^i(\alpha_1), u^i(\alpha_2)\} - u^i(\alpha_2)\Big] - \Big[\sum_{i\in S_{21}} \max\{u^i(\alpha_2), u^i(\alpha_1)\} - u^i(\alpha_1)\Big] = \\
&\sum_{i\in S} \Big[\max\{u^i(\alpha_1), u^i(\alpha_2)\} - u^i(\alpha_2) - \max\{u^i(\alpha_2), u^i(\alpha_1)\} + u^i(\alpha_1)\Big].
\end{aligned} \tag{4.21}$$

By canceling identical terms $max\{u^i(\alpha_2), u^i(\alpha_1)\}$,

$$R(\alpha_1 \to \alpha_2) - R(\alpha_2 \to \alpha_1) = \sum_{i\in S} u^i(\alpha_1) - u^i(\alpha_2). \tag{4.22}$$

Now divide $\alpha_1 \to \alpha_2$ into a sequence of sub-edges in which only one agent is deviating. Assuming $S = \{i, i+1, i+2, ..., i+|S|\}$ as the set of deviating players for this transition, the expanded from of this edge can be written as

$$\alpha_1 \to \alpha_2 = \{\alpha_1 = \alpha_{k_0} \to (\alpha_2^i, \alpha_1^{-i}) = \alpha_{k_1} \to \alpha_{k_2} \to ... \to \alpha_{k_{q-1}} \to \alpha_{k_q} = \alpha_2\},$$

where $\alpha_{k_j} \to \alpha_{k_{j+1}}$ is a sub-edge in which only player $i+j$th is deviating. Note that $q = |S|$. We can rewrite (4.22) as

$$\begin{aligned}
&R(\alpha_1 \to \alpha_2) - R(\alpha_2 \to \alpha_1) = \sum_{i\in S} \boldsymbol{u^i(\alpha_1)} - u^i(\alpha_{k_1}) + u^i(\alpha_{k_1}) - u^{i+1}(\alpha_{k_2}) + u^{i+1}(\alpha_{k_2}) \\
&- ... + u^{i+|S|-1}(\alpha_{k_{q-1}}) - \boldsymbol{u^i(\alpha_2)}.
\end{aligned} \tag{4.23}$$

By concavity of potential function and by knowing that $\alpha_1 \to \alpha_2$ is in $\Omega^R = \{\alpha \to \alpha_*\}$

$$\phi(\alpha_1) = \phi(\alpha_{k_0}) \leq \phi(\alpha_{k_1}) \leq \phi(\alpha_{k_2}) \leq ... \leq \phi(\alpha_{k_q}) = \phi(\alpha_2).$$

Thus, by Assumption 4.3:

$$\boldsymbol{u^i}(\boldsymbol{\alpha_1}) - u^i(\alpha_{k_1}) \leq 0, u^i(\alpha_{k_1}) - u^{i+1}(\alpha_{k_2}) \leq 0, ..., u^{i+|S|-1}(\alpha_{k_{q-1}}) - \boldsymbol{u^i}(\boldsymbol{\alpha_2}) \leq 0.$$

This means that $R(\alpha_1 \to \alpha_2) - R(\alpha_2 \to \alpha_1) \leq 0$. Since $\alpha_1 \to \alpha_2$ is an arbitrary transition in $\varOmega^R$ and $\alpha_2 \to \alpha_1$ is its reverse transition in $\varOmega$, it is obvious that

$$R(\varOmega^R) - R(\varOmega) \leq 0.$$

Therefore, from (4.20), T is not a minimum resistance tree among the trees rooted at other states, which is in contrast with the basic assumption about T. Hence, the supposition is false and α is the potential maximizer. □

4.4 Simulation Results

To verify the performance of SBLLL, in this section, we provide a numerical experiment on the MCC setup similar to the one introduced in Chapter 3. Recall that in our MCC problem, there exist a number of robots, playing a potential game within an unknown GMM environment and try to maximize their potential using the BLLL learning scheme. In this section, we only replace the BLLL learning algorithm of Chapter 3 with our proposed learning algorithm SBLLL. All the other mechanisms such as model estimation and agents' communication remain unchanged. The GMM in this chapter is also the same as the GMM in Chapter 3.

We considered a two variable function for the revision probability of each agent. The corresponding probability of each player's revision probability depends on the player's action, i.e. player's location in our MCC example. Each player i can determine the outcome of his action as the signal strength $f^i(l)$ of his location $\alpha^i = l$. Furthermore, each agent can sense a local gradient of the worth denoted by $g^i(l)$. At each iteration, $f^i(l)$ and $g^i(l)$ are normalized based on the player's maximum observed $f^i(l)$ and $g^i(l)$. For each agent i,

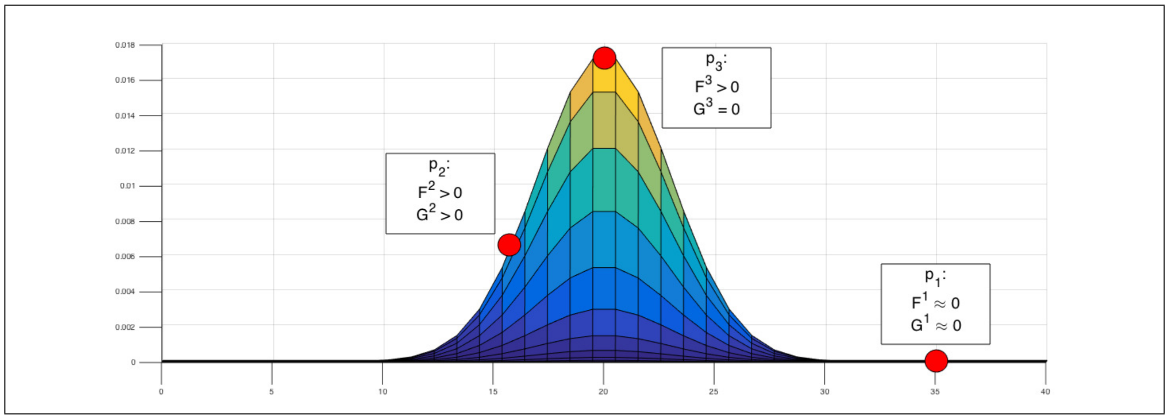

Figure 4.4. The logic behind the revision probability

let F^i and G^i be the normalized versions of $f^i(l)$ and $g^i(l)$ respectively.

We desire a revision probability function $(rp^i(F^i, G^i))$ for which the probability that players wake up depends on each player's situation. In this thesis and as in Fig.4.4, we consider three situations which a robot can fall in:

- *Situation p_1 where player's received signal and gradient is almost zero:* In such situation, player has to wake up to update its action, i.e. explore, and to move toward worthwhile areas. Let a_1 be the probability that player in p_1 wakes to update its action.

- *Situation p_2 where player's received signal and gradient is relatively high:* In this situation, player has entered a worthwhile region. Since the player already entered a worthwhile area, the need for exploration reduces comparing to the player in p_1, i.e. $a_2 < a_1$ where a_2 is the probability that player in p_2 wakes up.

- *Situation p_3 where player's received signal is relatively high but the gradient is almost zero:* This situation happens when player reaches a high local value. Clearly, player in p_3 has to remain on its position, i.e. remain asleep. Hence, $a_3 < a_2 < a_1$ where a_3 is the probability that player in p_3 wakes up.

To match with the shape of Gaussian function, we prefer to have an exponential drop as player i's normalized signal strength F^i increases. Hence, when $G^i = 0$, the form of

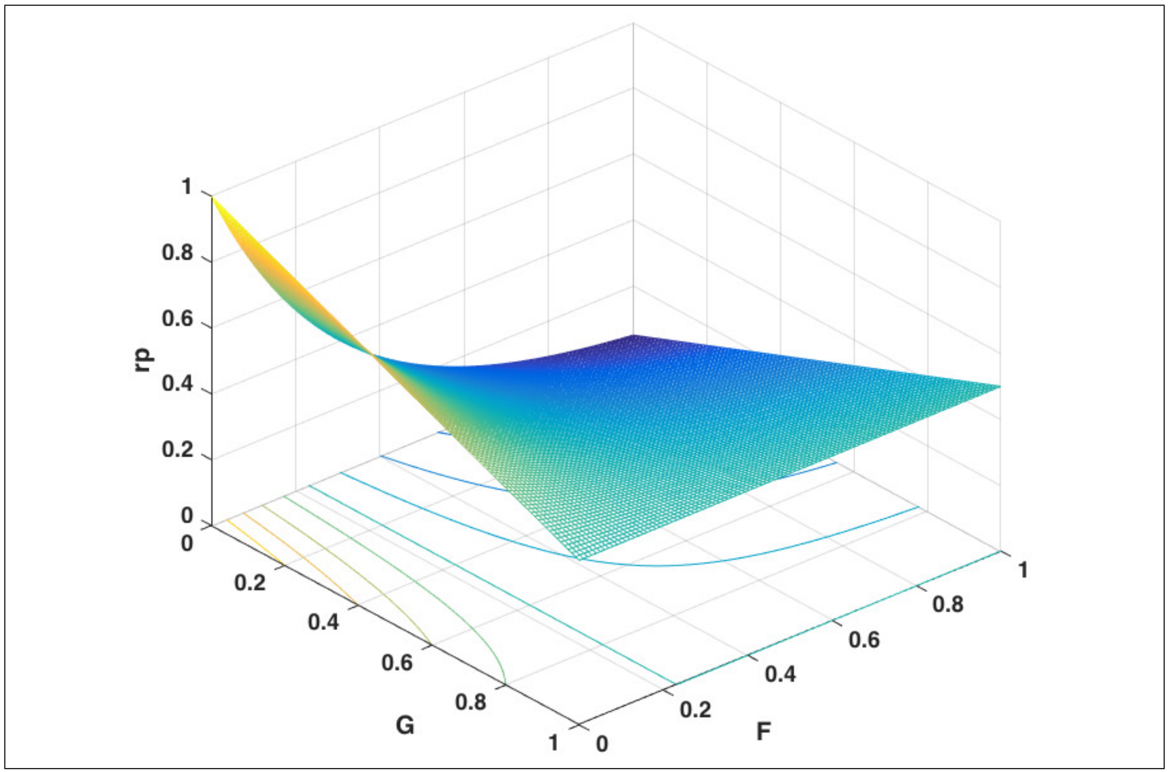

Figure 4.5. Revision probability function

revision probability function is as follow:

$$rp^i(F^i, G^i = 0) = e^{-k(F^i - c)},$$

where k is the drop rate and $c := \dfrac{\ln(a_1)}{k}$. For the sake of simplicity, we consider a linear change in revision probability function as G increases. As G reaches 1, the value of the revision probability function must be equal to a_2. Therefore, for a constant F, the slope of the line from $G = 0$ to $G = 1$ is $a_2 - e^{-k(F^i - c)}$. Furthermore, y-intercept for this line is $e^{-k(F^i - c)}$. Thus the complete function is of the form

$$rp^i(F^i, G^i) = (a_2 - e^{-k(F^i - c)})G + e^{-k(F^i - c)}. \tag{4.24}$$

As in Fig. 4.5, the function behavior is linear versus G and exponential versus F. With this independent revision, each player decides based on its status whether it is a good time to wake up or not. We believe it is more efficient than a random selection as in BLLL.

Furthermore, it reduces the need for unnecessary trial and errors.

Assume that all agents adhere to SBLLL and Assumptions 4.1 and 4.2 hold for the MCC problem. Players' utility function is separable and from Proposition 4.1, we know that the stochastically stable states are the set of potential maximizers. The simulation parameters are chosen as $K^i = 3 \times 10^{-5}$ for $i = 1, ..., N$, $\delta = 1.5$, $R_{com} = 56$, $a_1 = 1$, $a_2 = 0.5$ and $a_3 = 0.1$. The algorithm's is as follows:

```
set $n = 1$
**for** each robot $i \in \mathcal{I}$ **do**
initialize $\alpha^i(1) \in \mathcal{L}$ randomly
**while** the covered worth $\sum_{i\in\mathcal{I}} C^i(\alpha^i)$ is not in a steady state **do**
    **if** $T_{AIC}$ is a sub-multiple of $n$ **then**
        process merge or split
    **for** each robot $i \in \mathcal{I}$ **do**
        determine $rp^i$ (see (4.24))
        robot $i$ wakes up with probability $rp^i$
        (let $S(n)$ be the set of robots that are awake at $n$)
    **for** each robot $i \in S(n)$ **do**
        robot $i$ chooses a trial action $\alpha^i_T$ randomly from $\mathcal{A}^i_c$
        calculate $X^i_{\alpha^i(n)}$ and $X^i_{\alpha^i_T}$ by using both $GMM$ and $\widehat{GMM}$ (see (4.1) and (4.2))
        $\alpha^i(n+1) \leftarrow \alpha^i(n)$ with probability $X^i_{\alpha^i(n)}$
        or
        $\alpha^i(n+1) \leftarrow \alpha^i_T$ with probability $X^i_{\alpha^i_T}$
        **if** $\alpha^i(n+1)$ is $\alpha^i_T$ **then**
            add $\alpha^i(n+1)$ to $O^i$
            run EM and update $\widehat{GMM}$
    **for** each robot $j \in \mathcal{I} \setminus S(n)$ **do**
        $\alpha^j(n+1) \leftarrow \alpha^j(n)$
    $n \leftarrow n+1$
```

The worth of covered area using SBLLL and Chapter 3's BLLL is shown in Fig.4.6. Clearly, the covered worth is higher and the convergence rate is faster in SBLLL. Comparing to BLLL, in SBLLL each player autonomously decides, based on its situation, to update its action while in BLLL only one random player is allowed to do the trial and error. The algorithms ran for 10 times with different initial conditions and for each algorithm

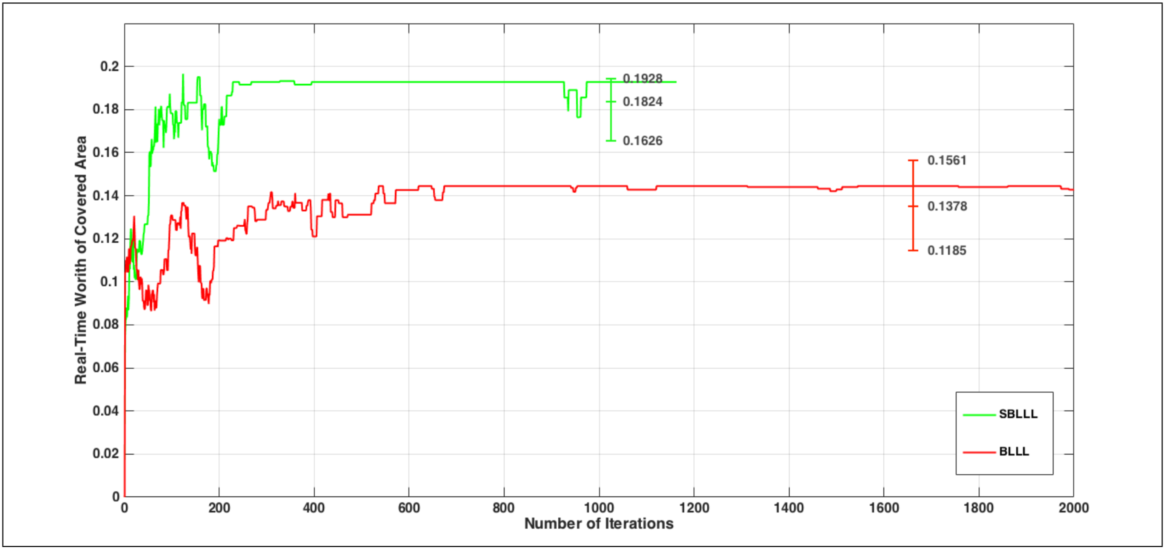

Figure 4.6. The real-time worth of the covered area

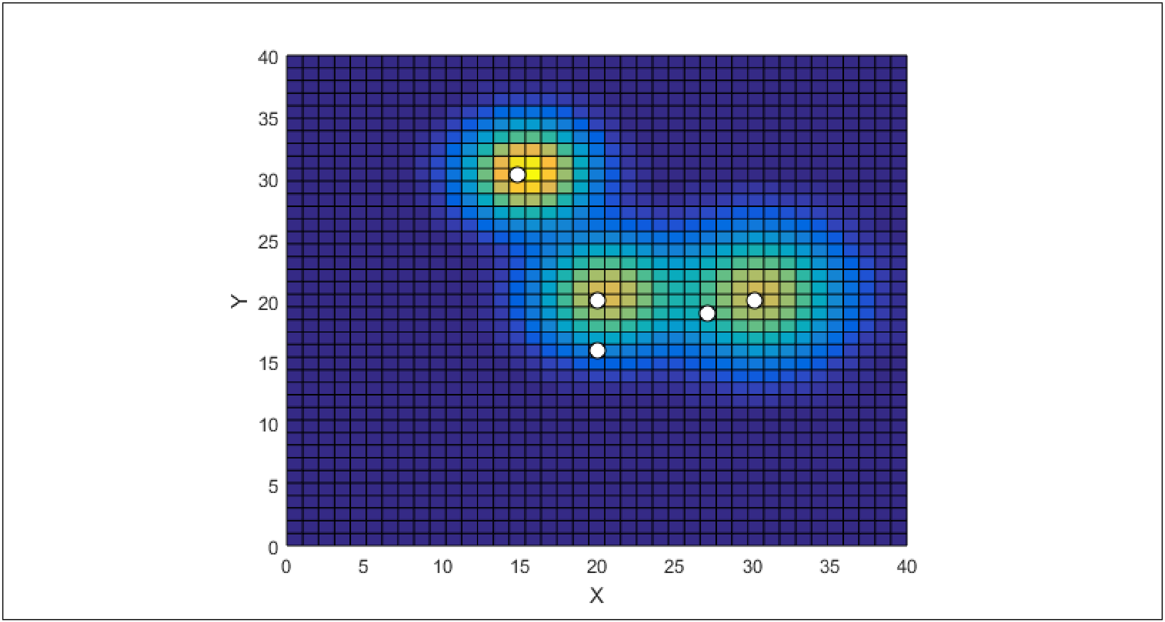

Figure 4.7. The final configuration of agents in SBLLL

Fig.4.6 presents a bound and a mean for the coverage worth. The final configuration of the agents are shown in Fig 4.7. We can see that the agents found all the targets. Although three agents have the targets in their sensing range, the two other agents tried to optimize their configuration with respect to the signal distribution to maximize the coverage worth.

4.5 Discussion

SBLLL is a type of learning algorithm in which each agent's action set is constrained. Furthermore, unlike BLLL in which only one random agent take a trial action, all the agents are allowed to explore the environment.

SBLLL demonstrated an excellent performance comparing to BLLL in Chapter 3, in terms of convergence rate and the covered worth. A higher convergence rate, as it was expected, is due to the synchronous learning in SBLLL. Furthermore, in SBLLL, agents have a wider opportunity to explore the environment as they are able to interact with the environment simultaneously.

Another valuable feature of SBLLL is that each agent's exploration process can be precisely regulated by rp^i. Thus, each agent can decide independently, and not randomly, whether it is the time for taking a trial action or it is better to remain on the current state. This will certainly reduce redundant explorations in SBLLL comparing to BLLL.

Despite the above-mentioned improvements, SBLLL still relies on an estimation model of the environment. This means that players in SBLLL have a prior knowledge about the structure of the utility distribution so they can apply an appropriate model on the environment. In practice, this may not be an acceptable assumption. In order to relax this assumption, we turned to RL algorithms which do not require a model of the environment. In the next two chapters, we will focus on RL and its application in the MCC problem.

Chapter 5

Second Order Q-Learning

5.1 Introduction

Recall from Chapter 2 that RL algorithms are model free and players only need to monitor the sequence of their own actions' outcome. This relaxes the need for an estimation model in our MCC problem. Thus, a MCC setup with RL can be employed when the utility distribution is not a GMM or more generally, when the utility structure is unknown. However, due to the lack of model from the environment, RL needs more time to explore it.

Q-learning, as a sub-class of RL algorithms, is widely used to optimally solve Markov decision processes. In the Q-learning algorithm the actions' expected utilities can be compared without requiring a knowledge of the utility function. Furthermore, Q-learning can be used in processes with stochastic transitions, without requiring any adaptations.

In this chapter we propose a modified Q-learning algorithm in which we incorporated a second-order reinforcement in the aggregation step. Such high-order reinforcement provides more momentum towards the strategies that tend to perform better. We call this algorithm Second Order Q-Learning (SOQL).

5.2 The Proposed Algorithm

Recall that in a RL algorithm, the score variable characterizes the relative utility of a particular action. In the Q-learning algorithm, Q actually represents the score variable in RL. The Q-value corresponding to the selected action will be updated by the generated reinforcement signal once that action is performed. In the standard Q-learning, the update rule for Q-value is

$$\begin{aligned} u^i_\beta(n) &= \mathbb{1}_{\{\alpha^i(n)=\beta\}}\, u^i(n), \\ Q^i_\beta(n+1) &= Q^i_\beta(n) + \mu(n)\, [u^i_\beta(n) - Q^i_\beta(n)], \end{aligned} \tag{5.1}$$

where $u^i(n)$ is the player i's realized utility at time step n, $Q^i_\beta(n)$ is the player i's Q-value corresponding to action β, and $\mu(n)$ is the step size assumed to be diminishing. In the action selection step, at each time each player selects his action according to a Q-value based SBR:

$$X^i_\beta(n+1) = \frac{\exp\bigl(1/\tau Q^i_\beta(n)\bigr)}{\sum_{\beta' \in \mathcal{A}^i} \exp\bigl(1/\tau Q^i_{\beta'}(n)\bigr)}. \tag{5.2}$$

The use of diminishing step size allows us to use stochastic approximation results, but in general leads to slow convergence time. Thus, in our proposed algorithm we use an update rule, with constant step size μ. Additionally, we propose a double aggregation process for updating Q-value as in Section 2.2.1.3. Undoubtedly, a higher order aggregation looks deeper into the player's observed utility:

$$\begin{aligned} u^i_\beta(n) &= \mathbb{1}_{\{\alpha^i(n)=\beta\}}\, u^i(n), \\ P^i_\beta(n+1) &= P^i_\beta(n) + \mu\, [u^i_\beta(n) - P^i_\beta(n)], \\ Q^i_\beta(n+1) &= Q^i_\beta(n) + \mu\, [P^i_\beta(n) - Q^i_\beta(n)], \end{aligned} \tag{5.3}$$

where $u^i(n)$ is the player i's realized utility at time step n, $Q^i_\beta(n)$ is the player i's Q-value corresponding to action β, and μ is the step size. To update players' mixed strategies, we use the following greedy action selection:

$$X^i(n+1) = (1-\vartheta)X^i(n) + \vartheta BR^i(Q^i(n)), \tag{5.4}$$

where the constant coefficient ϑ is the action selection step size satisfying $\vartheta < \mu < 1$ and player i's best response correspondence $BR^i(Q^i(n))$ is defined as

$$BR^i(Q^i(n)) = \{e^i_{\alpha^i_*} \mid \alpha^i_* \in \mathcal{A}^i,\ Q^i_{\alpha^i_*}(n) = \max_{\alpha \in \mathcal{A}^i} Q^i_\alpha(n)\}, \tag{5.5}$$

where $e^i_{\alpha^i_*}$ is player i's pure strategy or the unit vector corresponding to α^i_*.

5.3 Convergence Analysis

In the following we give conditions under which our proposed Q-learning algorithm converges to a pure strategy Nash equilibrium almost surely.

Proposition 5.1 *If an action profile $\alpha(n) = (\beta, \alpha^{-i})$ is repeatedly played for $m > 0$ iterations, i.e. $\alpha(n+c) = \alpha(n)$ for all $1 \leq c < m$, then:*

$$P^i_\beta(n+m) = (1-\mu)^m P^i_\beta(n) + (1-(1-\mu)^m)u^i_\beta(n), \tag{5.6}$$

and

$$\begin{aligned} Q^i_\beta(n+m) &= m(1-\mu)^{m-1}Q^i_\beta(n+1) - (m-1)(1-\mu)^m Q^i_\beta(n) \\ &+ \left[(m-1)(1-\mu)^m - m(1-\mu)^{m-1} + 1\right] u^i_\beta(n). \end{aligned} \tag{5.7}$$

Proof: From recursively using (5.3) it is easy to show that (5.6) and (5.7) hold. $\square$

Corollary 5.1 *For sufficiently large m if $0 < \mu < 1$:*

$$lim_{m\to\infty}\, Q^i_\beta(n+m) = u^i_\beta(n). \tag{5.8}$$

Proof: Corollary 5.1 can be easily verified by taking the limit $m \to \infty$ in (5.7).

$$lim_{m\to\infty} m(1-\mu)^{m-1} = 0,$$

$$lim_{m\to\infty} (m-1)(1-\mu)^{m} = 0,$$

$$lim_{m\to\infty} (m-1)(1-\mu)^{m} - m(1-\mu)^{m-1} + 1 = 1.$$

□

Remark 12. Note that in Corollary 5.1, during the m iterations, other actions $\beta' \neq \beta,\ \beta' \in \mathcal{A}^i$ are never played. In other words $Q^i_{\beta'}(n+m) = Q^i_{\beta'}(n)$.

Proposition 5.2 *Let $\alpha_* = (\beta_*, \alpha_*^{-i})$ be the action profile that is played at time step n and $n+1$ where $e^i_{\beta_*} = BR^i(Q^i(n))$ and for every player $i \in \mathcal{I}$ and for every action $\beta \in \mathcal{A}^i,\ \beta \neq \beta_*$ we have $Q^i_{\beta_*}(n+1) > Q^i_{\beta_*}(n) > Q^i_{\beta}(n)$ and $u^i_{\beta_*}(n+1) = u^i_{\beta_*}(n) > Q^i_{\beta}(n)$. We show that if $0 < \mu < 1$ then at any following mth iteration, with the probability of at least $\prod_{c=1}^{m} \left(1-(1-\vartheta)^c\right)^N$,*

$$\begin{aligned} Q^i_{\beta_*}(n+m+1) &= (m+1)(1-\mu)^m Q^i_{\beta_*}(n+1) - (m)(1-\mu)^{m+1} Q^i_{\beta_*}(n) \\ &\quad + \left[m(1-\mu)^{m+1} - (m+1)(1-\mu)^m + 1\right] u^i_{\beta_*}(n), \end{aligned} \tag{5.9}$$

and

$$X^i(n+m+1) = (1-\vartheta)^{m+1} X^i(n) + (1-(1-\vartheta)^{m+1}) e^i_{\beta_*}. \tag{5.10}$$

Proof: We use induction to prove our claim. For $m = 1$ since $Q^i_{\beta_*}(n+1) > Q^i_{\beta}(n) = Q^i_{\beta}(n)$ it follows that at time step $n+2$, α_* is still the estimated best response. By (5.4) and considering the fact that $BR^i(Q^i(n)) = e^i_{\beta_*}$ then at time $n+2$

$$X^i(n+2) = (1-\vartheta) X^i(n+1) + \vartheta e^i_{\beta_*},$$

Since $e^i_{\beta_*}$ is a unit vector, $X^i_{\beta_*}(t) > \vartheta$ and it is true for all other players. Therefore, at time step $n+2$, the action profile α_* is played with the probability of at least ϑ^N. From the assumption that for player i, $u^i_{\beta_*}(n) > Q^i_{\beta_*}(n)$, we have:

$$\mu^2 u^i_{\beta_*}(n) > \mu^2 Q^i_{\beta_*}(n). \tag{5.11}$$

Now, consider the condition $Q^i_{\beta_*}(n+1) > Q^i_{\beta_*}(n)$. For $0 < \mu < 1$ it is easy to show that

$$2(1-\mu)Q^i_{\beta_*}(n+1) > 2(1-\mu)Q^i_{\beta_*}(n). \tag{5.12}$$

By combining (5.11) and (5.12)

$$2(1-\mu)Q^i_{\beta_*}(n+1) + \mu^2 u^i_{\beta_*}(n) > 2(1-\mu)Q^i_{\beta_*}(n) + \mu^2 Q^i_{\beta_*}(n).$$

By subtracting $(1-\mu)^2 Q^i_{\beta_*}(n)$ from the both sides,

$$\begin{aligned} 2(1-\mu)Q^i_{\beta_*}(n+1) + \mu^2 u^i_{\beta_*}(n) - (1-\mu)^2 Q^i_{\beta_*}(n) > \\ 2(1-\mu)Q^i_{\beta_*}(n) + \mu^2 Q^i_{\beta_*}(n) - (1-\mu)^2 Q^i_{\beta_*}(n). \end{aligned} \tag{5.13}$$

From (5.7), $2(1-\mu)Q^i_{\beta_*}(n+1) + \mu^2 u^i_{\beta_*}(n) - (1-\mu)^2 Q^i_{\beta_*}(n)$ is equivalent to $Q^i_{\beta_*}(n+2)$ and therefore:

$$Q^i_{\beta_*}(n+2) > Q^i_{\beta_*}(n) > Q^i_{\beta}(n) \tag{5.14}$$

Recall Remark 12 explaining that other actions $\beta \in \mathcal{A}^i$ are not played during the m iterations, i.e. $Q^i_\beta(n) = Q^i_\beta(n+1) = Q^i_\beta(n+2)$. By substituting $Q^i_\beta(n)$ with $Q^i_\beta(n+2)$ in (5.14) we get $Q^i_{\beta_*}(n+2) > Q^i_\beta(n+2)$ which means that β_* is still the estimated best response for player i at iteration $n+2$. By repeating the argument above for all players we show that at time step $n+2$, α_* remains the estimated best response and the claim in (5.9) follows for $m = 1$.

Now assume, at every iteration c where $1 \leq c \leq m-1$, a_* is played with probability $\prod_{c=1}^{m-1}\left(1-(1-\vartheta)^c\right)^N$. At time step m, X^i is updated as:

$$X^i(n+m+1) = (1-\vartheta)^{m+1}\, X^i(n) + (1-(1-\vartheta)^{m+1})e^i_{\beta_*}. \tag{5.15}$$

By Proposition 5.1,

$$\begin{aligned} Q^i_{\beta_*}(n+m+1) &= (m+1)(1-\mu)^m Q^i_{\beta_*}(n+1) - (m)(1-\mu)^{m+1} Q^i_{\beta_*}(n) \\ &+ \left[m(1-\mu)^{m+1} - (m+1)(1-\mu)^m + 1\right] u^i_{\beta_*}(n), \end{aligned} \tag{5.16}$$

Since for player i any action $\beta \neq \beta_*$ is not played, it follows that $Q^i_\beta(n+m+1) = Q^i_\beta(n)$ (Remark 12). Moreover, $Q^i_{\beta_*}(n+m+1) > Q^i_{\beta_*}(n+m) > Q^i_\beta(n+m)$ and $u^i_{\beta_*}(n+m+1) > Q^i_\beta(n+m)$. From (5.15), at time step $n+m+2$, α_* with probability of at least $\left(1-(1-\vartheta)^n\right)^N$ and from Proposition 5.1,

$$Q^i_\beta(n+m+2) = 2(1-\mu)Q^i_\beta(n+m+1) - (1-\mu)^2 Q^i_\beta(n+m) + \mu^2 u^i_\beta(n+m). \tag{5.17}$$

With the same argument as for $n=1$,

$$Q^i_{\beta_*}(n+m+2) > Q^i_\beta(n+m+2).$$

Therefore, the estimated best response for player i is not changed. In other words

$$BR^i(Q^i(n+m+2)) = BR^i(Q^i(n)) = e^i_{\beta_*}.$$

By substituting (5.15) into (5.4)

$$X^i(n+m+2) = (1-\vartheta)^{m+2} X^i(t) + (1-(1-\vartheta)^{m+2})e^i_{\beta_*}. \tag{5.18}$$

This completes the induction argument.

□

Corollary 5.2 *If for sufficiently large $m > 0$ the conditions of Proposition 5.2 hold, then for every player i the following holds with probability $\prod_{c=1}^{\infty}[\big(1-(1-\vartheta)^c\big)^N$:*

$$\lim_{m\to\infty} Q^i_{\beta_*}(n+m) = u^i(\alpha_*), \tag{5.19}$$

$$\lim_{m\to\infty} X^i(n+m) = e^i_{\beta_*}. \tag{5.20}$$

Theorem 5.1 *For sufficiently large m, if the conditions in Proposition 5.2 hold, then $X(n+m)$ converges to a neighborhood of α_* with probability one.*

Proof: From Proposition 5.2 we know that α_* is played with probability $\prod_{c=1}^{\infty}[\big(1-(1-\vartheta)^c\big)^N$. Inspired by Proposition 6.1 in [50], we show that this probability is strictly positive. The product $\prod_{c=1}^{\infty}(1-(1-\vartheta)^c)$ is non-zero if and only if $\sum_{c=1}^{\infty}\log(1-(1-\vartheta)^c) > -\infty$. Alternatively we can show

$$-\sum_{c=1}^{\infty}\log(1-(1-\vartheta)^c) < \infty. \tag{5.21}$$

By using the limit comparison test and knowing $0 < 1-\vartheta < 1$:

$$\lim_{c\to\infty}\frac{-\log(1-(1-\vartheta)^c)}{(1-\vartheta)^c} = \lim_{c\to\infty}\frac{1}{1-(1-\vartheta)^c)} = 1.$$

Therefore, (5.21) holds if and only if $\sum_{c=1}^{\infty}\ (1-\vartheta)^c < \infty$. The latter holds since

$$\sum_{c=1}^{\infty}\ (1-\vartheta)^c = \frac{1}{1-(1-\vartheta)} = \frac{1}{\vartheta}\ <\ \infty.$$

By (5.21)

$$\lim_{m\to\infty}\prod_{c=1}^{m}(1-(1-\vartheta)^m) > 0,$$

so we can conclude that

$$\forall \eta,\ 0 < \eta < 1,\ \exists M\ s.t.\ (1 - (1 - \vartheta)^m) \geq \eta\ \forall m \geq M.$$

In other words after M iterations $X(n+M)$ enters a ball $B_{1-\eta}(\alpha_*)$ with probability $\prod_{c=1}^{M}(1 - (1 - \vartheta)^c)^N$. As discussed in [13], and following the proof of Theorem 3.1 in [19], for $M \geq \log_{1-\vartheta}(1 - \eta) > 0$, trajectories of $X(n+M)$ enters a neighborhood of α_* with probability $\prod_{c=1}^{\infty}(1 - (1 - \vartheta)^c)^N$; hence, converges to α_* almost surely.

□

5.3.1 Perturbation

We proved that the SOQL converges to a neighborhood of α_*, i.e. $B_{1-\eta}(\alpha_*)$, almost surely. Recall that in a learning algorithm it is essential that the action space is well explored; otherwise, it is likely that the estimated equilibrium does not converge to the true Nash equilibrium. Thus, the action selection procedure should allow players to explore new actions even if the new actions are sub-optimal. In this section we discuss the conditions under which the estimated equilibrium can reach an actual Nash equilibrium. The perturbation functions are usually carefully designed to suit this need. A perfect perturbation function is decoupled from dynamics of the learning algorithm, adjustable and has minimum effect on perturbing the optimal solution [13].

As in [51], in the standard Q-learning scheme, the Boltzmann action selection map has already incorporated the exploration feature by using the temperature parameter τ. In order to implement such exploration feature in the modified algorithm, we use a perturbation function. The perturbed mixed strategy for player i is defined as

$$\tilde{X}_j^i = (1 - \rho^i(X^i, \xi))X_j^i + \rho^i(X^i, \xi)\ \mathbb{1}_j^i / |\mathcal{A}_i|. \tag{5.22}$$

where $\rho^i(X^i, \xi)$ is the perturbation function. The perturbation function $\rho^i : \mathcal{X}^i \times [\bar{\epsilon}, 1] \rightarrow [0, 1]$ is a continuously differentiable function where for some $\xi \in (0, 1)$ sufficiently close to one, ρ^i satisfies the following properties:

- $\rho^i(X^i, \xi) = 0, \ \forall X^i$ such that $\forall \xi \geq \bar{\epsilon} : \ |X^i|_\infty < \zeta$,
- $\lim_{|X^i|_\infty \to 1} \ \rho^i(X^i, \xi) = \xi$,
- $\lim_{|X^i|_\infty \to 1} \frac{\partial \rho^i(X^i,\xi)}{\partial X^i_j} \ |_{\xi=0} = 0, \ \forall j \in \mathcal{A}^i$.

As in (5.22), each player i selects a random action with a small probability ρ_i and selects the action with highest Q value with the probability $(1 - \rho_i)$.

Assumption 5.1 *Step sizes in Q update rule are adjusted based on:*

$$\mu^i_j(n) = (1 - \tilde{X}^i_j(n)).$$

Assumption 5.2 *When all the players enter the perturbation zone, i.e.* $|X^i|_\infty > \zeta, \ \forall i \in \mathcal{I}$, *no more than one player chooses a sub-optimal action at each iteration (asynchronous perturbation).*

Assumption 5.3 *The utility map for each player* i, $u^i \ : \ \mathcal{A}^i \ \to \ \mathbb{R}$, *is continuous and smooth.*

Theorem 5.2 *If the conditions in Proposition 5.2 hold for a sufficiently large* m, *then under Assumption 5.1, Assumption 5.2 and Assumption 5.3, the estimated equilibrium* α_* *would converge to a Nash equilibrium almost surely.*

Proof: From Corollary 5.2

$$\lim_{m \to \infty} \ Q^i_{\beta_*}(n + m) = u^i(\alpha_*),$$

$$\lim_{m \to \infty} \ X^i(n + m) = e^i_{\beta_*},$$

where $\alpha_* = (\beta_*, \alpha_*^{-i})$. Assume that perturbation becomes active at some large time step $\bar{n} + 1$ and player i chooses an action $\beta \neq \beta_*$. From (5.22) we know that such perturbation happens with the probability of at least $\xi/|\mathcal{A}^i|$. From (5.3) and Assumption 5.1, player i updates its action by

$$Q^i_\beta(\bar{n}+2) = 2\bar{X}^i_\beta(\bar{n})Q^i_\beta(\bar{n}+1) - \bar{X}^i_\beta(\bar{n})^2 Q^i_\beta(\bar{n}) + (1-\bar{X}^i_\beta(\bar{n}))^2 u^i(\beta(\bar{n}), \alpha_*^{-i}(\bar{n})). \tag{5.23}$$

Consider the following two cases:

- $u^i(\beta, \alpha_*^{-i}(\bar{n})) < u^i(\alpha_*(\bar{n}))$: In this case player i failed to improve the utility and will stay at the estimated equilibrium α_* almost surely.

- $u^i(\beta, \alpha_*^{-i}(\bar{n})) = u^i(\alpha_*(\bar{n}))$: With $\bar{X}^i_\beta$ sufficiently close to zero the Q-values for the actions β is updated to a value sufficiently close to $u^i(\beta, \alpha_*^{-i}(\bar{n})) = u^i(\alpha_*(\bar{n}))$. Hence, in the worst case, when $\alpha_*(\bar{n})$ is only played once, the Q-values for the actions β and β_* may become equal. Therefore, in the action selection stage, the set of best response in (5.5) may contain two pure actions. Consequently, the mixed strategies for both actions β and β_* is updated by (5.4) and player i chooses one of them randomly in the next time step.

- $u^i(\beta, \alpha_*^{-i}(\bar{n})) > u^i(\alpha_*(\bar{n}))$: In this case player i found a response that is better than α_*. Since the action β has small probability $\bar{X}^i_\beta$, i.e. sufficiently close to 0, then $(1-\bar{X}^i_\beta)$ is sufficiently close to 1. Thus from (5.23), Q^i_β is updated to a value sufficiently close to $u^i(\beta, \alpha_*^{-i}(\bar{n}))$ and the action profile $(\beta, \alpha_*^{-i}(\bar{n}))$ becomes the new estimated best response, i.e. $\alpha_*(\bar{n}+1) := (\beta, \alpha_*^{-i}(\bar{n}))$. Note that player's utility is improved by $u^i(\alpha_*(\bar{n}+1)) - u^i(\alpha_*(\bar{n}))$, and by (2.2) the potential would also increase by the same amount. Recall that from finite improvement property, such improvements in potential function is not limitless and will terminate at the potential maximum, i.e. Nash equilibrium in potential games. Hence, the estimated equilibrium α_* will eventually converge to an actual Nash equilibrium.

□

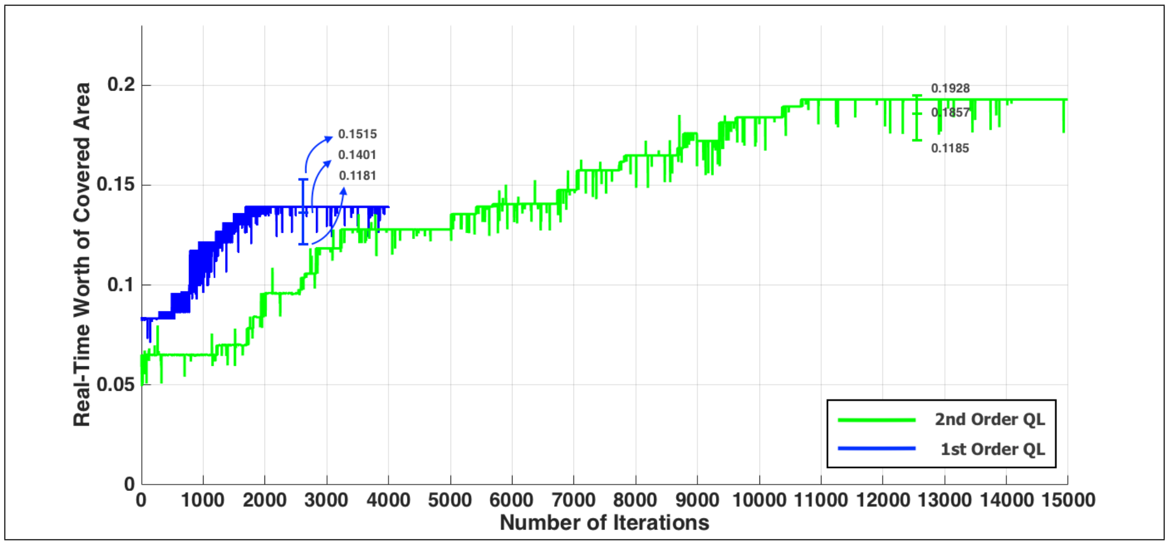

Figure 5.1. The real-time worth of the covered area

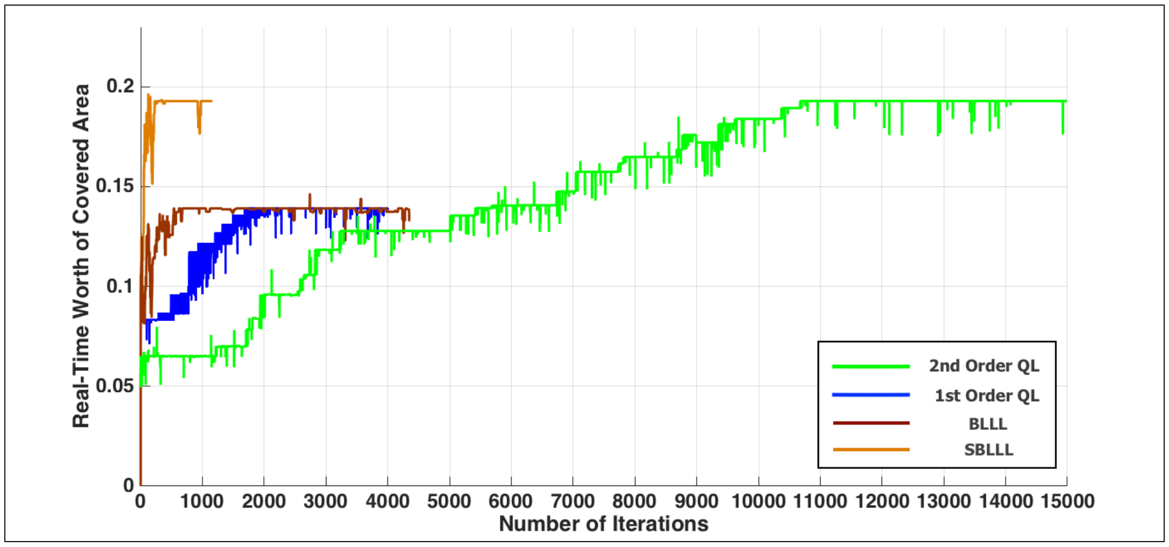

Figure 5.2. The real-time worth of the covered area

5.4 Simulation Results

In this section, we present simulation results of SOQL algorithm in a MCC setup. The environment is a 40×40 square area over which a GMM utility distribution is defined. The GMM distribution in SOQL simulation is the same as the GMM in Chapter 3 and Chapter 4. The robots also have no initial information about the utility distribution. However, comparing to BLLL and SBLLL simulation setup, there is no communication between agents.

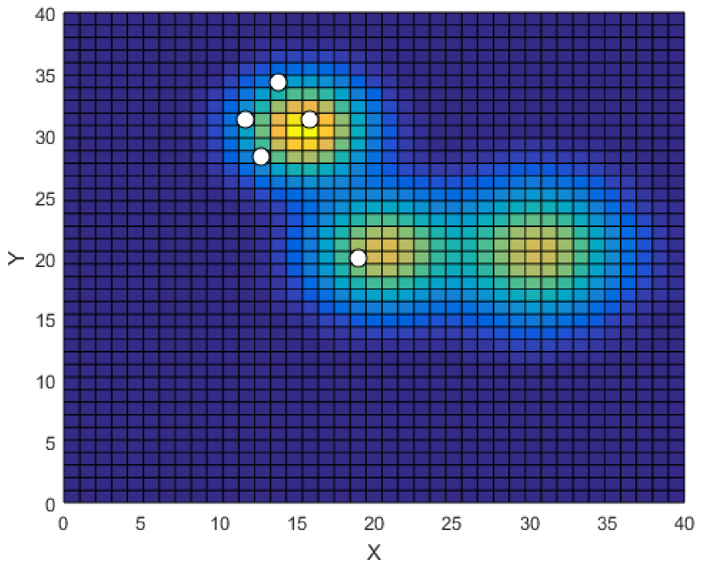

Figure 5.3. The final configuration of agents in standard Q-learning

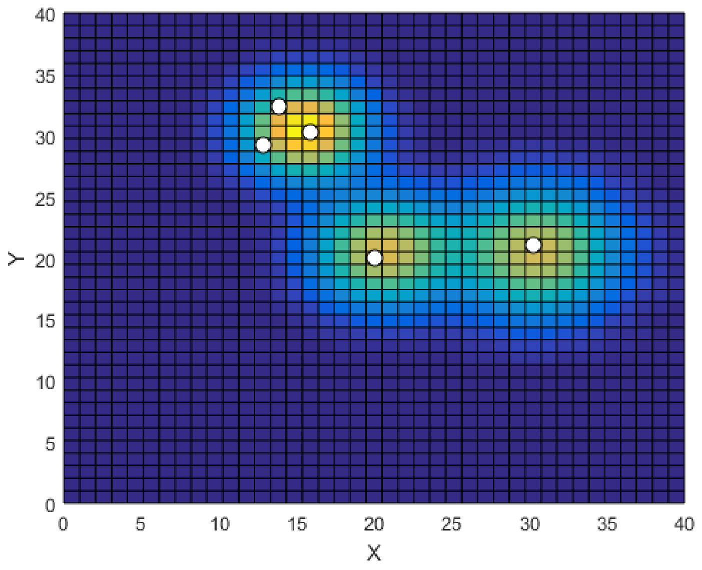

Figure 5.4. The final configuration of agents in SOQL

A group of five robots ($N = 5$) scatter through this GMM to maximize their utility function. In SOQL, the robots sample the environment at each time to create a memory of the payoff of the actions that they played. When the environment is explored enough, this memory can help the robots to find the game's global optimum. Recall that players' utility functions are separable. Therefore, with a set of stationary targets, utility distribution remains stationary for each agent as it is not changed by the actions of other players.

The simulation parameters are chosen as $K^i = 3 \times 10^{-5}$ for $i = 1, ..., N$, $\delta = 1.5$, $\mu = 0.97$, $\vartheta = 0.5$, $\zeta = 0.9999$ and $\xi = 0.01$. The following is the algorithm's pseudo-code:

```
set $n = 1$
**for** each robot $i \in \mathcal{I}$ **do**
initialize $\alpha^i(1) \in \mathcal{L}$ randomly
initialize $Q^i(0)$ and $Q^i(1) \in \mathbb{R}^{|\mathcal{L}|}$
initialize $X^i \in \mathbb{R}^{|\mathcal{L}|}$
**while** the covered worth $\sum_{i\in\mathcal{I}} C^i(\alpha^i)$ is not in a steady state **do**
    **for** each player $i \in \mathcal{I}$ **do**
        perform an action $\alpha^i(n) = \beta^i$ from $\mathcal{A}^i_c$ based on $X^i(n)$
        receive $u^i(n)$
        $u^i_\beta(n) \leftarrow \mathbb{1}_{\{\alpha^i(n)=\beta\}}\, u^i(n)$
        $Q^i_\beta(n+1) \leftarrow 2(1-\mu^i)Q^i_\beta(n) - (1-\mu^i)^2 Q^i_\beta(n-1) + {\mu^i}^2 u^i_\beta(n)$
        **for** each $\beta' \in \mathcal{A}^i, \beta' \neq \beta$ **do**
            $Q^i_{\beta'}(n+1) \leftarrow Q^i_{\beta'}(n)$
        $X^i(n+1) \leftarrow (1-\vartheta)X^i(n) + \vartheta BR^i(Q^i(n))$
    $n \leftarrow n+1$
```

The worth of covered area using SOQL and first order Q-learning algorithms is shown in Fig.5.1. As in Fig.5.1, the convergence rate is lower for SOQL algorithm comparing to first order algorithm. However, the covered worth is higher comparing to the first order case. The algorithms ran for 5 times with different initial conditions and for each algorithm Fig.5.1 presents a bound and a mean for the coverage worth.

Comparing to the model-based SBLLL and BLLL, the convergence rate of SOQL is lower (Fig.5.2). It is due to the need for a wide exploration in a RL scheme. However, in

SOQL algorithm, the equilibrium worth is the same to SBLLL's equilibrium worth while in SOQL we have no auxiliary estimation algorithm, such as EM.

5.5 Discussion

In this chapter we considered a RL solution to our MCC problem. In this RL algorithm, we proposed a second order reinforcement to increase the algorithm's aggregation depth. SOQL's performance as a model-free RL algorithm is comparable to the model-based SBLLL in terms of equilibrium worth. However, in terms of the convergence rate, SOQL can not reach SBLLL's performance.

In order to increase the Q-learning convergence rate, in the next chapter, we introduce a differentiation in Q values update rule. We expect that such differentiation will increase the algorithm's convergence rate by considering the Q values trending.

Chapter 6

Predictive Q-Learning

6.1 Introduction

This chapter proposes a predictive Q-leaning algorithm and investigates its application in the MCC problem. As discussed in Chapter 5, one of the drawbacks of RL algorithm is its low convergence rate. Thus, in this chapter we propose a modified Q-learning to increase RL's convergence rate. We propose to add an anticipatory term to the aggregation step which provides further adaptiveness to the algorithm by considering the derivative, i.e. the variation slope of the reinforced signal. We expect that such anticipation increases the algorithm's convergence rate.

6.2 The Proposed Algorithm

Recall the update rule for the standard Q-learning in (5.1) which can be seen to act as an integration for Q is:

$$[Q_j^i(n+1) - Q_j^i(n)]/\mu^i = u^i(\alpha(n)) - Q_j^i(n). \tag{6.1}$$

We propose to incorporate the following anticipatory term into the aggregation update rule:

$$Q_j^i(n+1) = \frac{[u^i(\alpha(n)) - Q_j^i(n)] - [u^i(\alpha(n-1)) - Q_j^i(n-1)]}{\mu^i}. \tag{6.2}$$

This anticipation is nothing more than a differentiation over the Q and utility difference.

Remark 13. Note that in the case of stationary utility, (6.2) reduces to a differentiation over Q_j^i, i.e.

$$Q_j^i(n+1) = \frac{Q_j^i(n-1) - Q_j^i(n)}{\mu^i}. \tag{6.3}$$

Note that unlike the integration component, the anticipation needs to have access to iteration $n-1$ in order to determine $Q_j^i(n+1)$. This may raise issues when player i is at its very first step or if it switch its action from j to action $j' \neq j$. To resolve this issue, we introduce ν^i as we combine (6.2) with (6.1):

$$\begin{aligned} &Q_j^i(n+1) = K_i\Big[(1-\mu^i)Q_j^i(n) + \mu^i u^i(\alpha(n))\Big] + \\ &\frac{\nu^i K_d}{\mu^i}\Big[u^i(\alpha(n)) - u^i(\alpha(n-1)) + Q_j^i(n-1) - Q_j^i(n)\Big], \end{aligned} \tag{6.4}$$

where $K_i > 0$ and $K_d > 0$ are the integrator gain and the differentiator gain respectively and ν^i is defined as

$$\nu^i = \begin{cases} 1 & : \alpha^i(n-1) = e_j^i \\ 0 & : otherwise \end{cases}$$

To update each player i's strategy, we use the same greedy update rule as in Chapter 5:

$$X^i(n+1) = (1-\vartheta)X^i(n) + \vartheta BR^i(Q^i(n)), \tag{6.5}$$

where the constant coefficient ϑ is the action selection step size satisfying $\vartheta < \mu < 1$ and player i's best response correspondence $BR^i(Q^i(n))$ is defined as

$$BR^i(Q^i(n)) = \{e^i_{\alpha^i_*} \mid \alpha^i_* \in \mathcal{A}^i,\ Q^i_{\alpha^i_*}(n) = \max_{\alpha \in \mathcal{A}^i} Q^i_\alpha(n)\}, \tag{6.6}$$

and $e^i_{\alpha^i_*}$ is player i's pure strategy or the unit vector corresponding to α^i_*. We call this algorithm Predictive Q-learning (PQL).

6.3 Convergence Analysis

In the following we give conditions under which PQL converges to a pure strategy Nash equilibrium almost surely.

Proposition 6.1 *If an action profile $\alpha(n-1) = (\beta, \alpha^{-i})$ is repeatedly played for $m+1 > 0$ iterations, i.e. $\alpha(n+c) = \alpha(n)$ for all $-1 \leq c < m$, then from (6.4):*

$$\begin{aligned} Q^i_\beta(n+m) =& \Big[(K_i - K_d/\mu + K_i\mu)a(m-1) + (K_d/\mu) \times a(m-2)\Big] Q^i_\beta(n) + \\ & \Big[(K_i - K_d/\mu + K_i\mu) \times b(m-1) + (K_d/\mu) b(m-2)\Big] Q^i_\beta(n-1) + \\ & \Big[(K_i - K_d/\mu + K_i\mu) c(m-1) + (K_d/\mu) \times c(m-2) + K_i\mu\Big] u^i_\beta(n), \end{aligned} \tag{6.7}$$

where $a(m)$, $b(m)$ and $c(m)$ are recursive sequences defined as follows:

$$a(m) = (K_i - K_d/\mu + K_i\mu)a(m-1) + (K_d/\mu)a(m-2),$$

$$a(0) = K_d/\mu,\ a(1) = K_i - K_d/\mu + K_i\mu, \tag{6.8}$$

$$b(m) = (K_i - K_d/\mu + K_i\mu)b(m-1) + (K_d/\mu)b(m-2),$$

$$b(0) = 0,\ b(1) = K_d/\mu, \tag{6.9}$$

$$c(m) = (K_i - K_d/\mu + K_i\mu)c(m-1) + (K_d/\mu)c(m-2) + K_i\mu,$$

$$c(0) = 0,\ c(1) = K_i\mu. \tag{6.10}$$

Proof: From recursively using (6.4) it can be shown that (6.7) holds. Note that $\nu^i = 1$ during $m + 1$ iterations. □

Proposition 6.2 *The sequences $a(m)$, $b(m)$ and $c(m)$ are convergent if $K_i = \dfrac{1}{1 + 2\mu}$ and $K_d < \dfrac{\mu + \mu^2}{1 + 2\mu}$.*

Proof: Let $K_1 := K_i - K_d/\mu + K_i\mu$ and let $K_2 := K_d/\mu$ and $K_3 := K_i\mu$. The recursive equations (6.8), (6.9) and (6.10) are linear constant-coefficient difference equations. While (6.8) and (6.9) are homogeneous difference equations, (6.10) is a non-homogeneous difference equation.

The trial solution for (6.8) or (6.9) is $y_c(m) = r^m$. The variable r, must satisfy the characteristic equation

$$r^2 - K_1 r - K_2 = 0. \tag{6.11}$$

Each distinct root of the characteristic equation gives rise to a distinct solution of the homogeneous difference equation. Suppose there are two distinct roots r_1 and r_2. The general solution $y_g(m)$ is then the linear combination

$$y_g(m) = c_1 r_1^m + c_2 r_2^m.$$

The roots of (6.11) are

$$\begin{aligned} r_1 &= \frac{1}{2}(K_1 + \sqrt{K_1^2 + 4K_2}), \\ r_2 &= \frac{1}{2}(K_1 - \sqrt{K_1^2 + 4K_2}). \end{aligned} \tag{6.12}$$

By substituting $K_i = 1/(1 + 2\mu)$ and considering the condition $K_d < \dfrac{\mu + \mu^2}{1 + 2\mu}$, it is easy to show that both $|r_1|$ and $|r_2|$ are less than one. Thus, both $a(m)$ and $b(m)$ converge to zero.

Unlike (6.8) and (6.9), (6.10) is a non-homogeneous equation and its general solution is consist of a common solution and a particular solution:

$$y'_g(m) = c'_1 r_1^m + c'_2 r_2^m + y'_p(m),$$

where r_1 and r_2 are the roots of (6.11) and $y'_p(m)$ is the particular solution. Since K_3 is a constant, the initial guess for $y'_p(m)$ is of the form $Am + B$. By substituting $Am + B$ in (6.10), we find $y'_p(m) = 1$. Therefore, $c(m)$ converges to one. $\square$

Corollary 6.1 *Under assumptions of Proposition 1 and Proposition 2, for sufficiently large m,*

$$\lim_{m \to \infty} Q^i_\beta(n+m) = u^i_\beta(n). \tag{6.13}$$

Proof: By taking the limit $m \to \infty$ from the general expressions of each sequence we have:

$$\lim_{m \to \infty} a(m) = \lim_{m \to \infty} b(m) = 0$$

$$\lim_{m \to \infty} c(m) = 1$$

Hence, (6.13) follows from (6.7). $\square$

Remark 14. Note that in Corollary 1, during the m iterations, other actions $\beta' \neq \beta,\ \beta' \in \mathcal{A}^i$ are never played. In other words $Q^i_{\beta'}(n+m) = Q^i_{\beta'}(n)$.

Proposition 6.3 *Let $\alpha_* = (\beta_*, \alpha_*^{-i})$ be the action profile that is played at time step $n-1$ and n where $e^i_{\beta_*} = B^i(Q^i(n-1))$. Assume that for every player $i \in \mathcal{I}$ and for every action $\beta \in \mathcal{A}^i,\ \beta \neq \beta_*$ we have $Q^i_{\beta_*}(n) > Q^i_{\beta_*}(n-1) > Q^i_\beta(n-1)$ and $u^i_{\beta_*}(n) = u^i_{\beta_*}(n-1) > Q^i_\beta(n-1)$. Then under assumptions of Proposition 1 and Proposition 2, at any following mth iteration, with the probability of at least $\prod_{c=1}^m \left(1-(1-\vartheta)^c\right)^N$ the following holds*

$$Q^i_{\beta_*}(n+m) = a(m)Q^i_{\beta_*}(n) + b(m)Q^i_{\beta_*}(n-1) + c(m)u^i_{\beta_*}(n), \tag{6.14}$$

$$X^i(n+m) = (1-\vartheta)^m \, X^i(n) + (1-(1-\vartheta)^m) \, e^i_{\beta_*}. \tag{6.15}$$

Proof: We use induction to prove our claim. For $m = 1$, since $Q^i_{\beta_*}(n) > Q^i_{\beta}(n-1) = Q^i_{\beta}(n)$, it follows that at time step $n+1$, α_* is the estimated best response where β_* is the player i's action. By (6.5) and considering the fact that $B^i(Q^i(n)) = e^i_{\beta_*}$ it follows that at time step n

$$X^i(n+1) = (1-\vartheta)X^i(n) + \vartheta e^i_{\beta_*},$$

Since $e^i_{\beta_*}$ is a unit vector, $X^i_{\beta_*}(n) > \vartheta$ and it is true for all other players. Therefore, at time step $n+1$, the action profile α_* is played with the probability of at least ϑ^N. Hence β_* is played by player i at time step $n+1$. From the assumption $u^i_{\beta_*}(n) > Q^i_{\beta}(n-1)$,

$$K_i \mu u^i_{\beta_*}(n) > K_i \mu Q^i_{\beta}(n-1), \tag{6.16}$$

and from the assumption $Q^i_{\beta_*}(n-1) > Q^i_{\beta}(n-1)$,

$$\frac{K_d}{\mu} Q^i_{\beta_*}(n-1) > \frac{K_d}{\mu} Q^i_{\beta}(n-1). \tag{6.17}$$

Note that $K_i - K_d/\mu + K_i\mu > 0$, since $K_i = \dfrac{1}{1+2\mu}$ and $K_d < \dfrac{\mu+\mu^2}{1+2\mu}$. Thus

$$(K_i - K_d/\mu + K_i\mu) Q^i_{\beta_*}(n) > (K_i - K_d/\mu + K_i\mu) Q^i_{\beta}(n-1). \tag{6.18}$$

By combining (6.16), (6.17) and (6.18)

$$Q^i_{\beta_*}(n+1) > (K_i + 2K_i\mu) Q^i_{\beta}(n-1).$$

By substituting $K_i = \dfrac{1}{1+2\mu}$,

$$K_i + 2K_i\mu = \frac{1}{1+2\mu} + \frac{2\mu}{1+2\mu} = 1. \tag{6.19}$$

Therefore,

$$Q^i_{\beta_*}(n+1) > Q^i_{\beta}(n-1). \tag{6.20}$$

Recall Remark 14 explaining that other actions $\beta \in \mathcal{A}^i$ are not played, i.e. $Q^i_\beta(n-1) = Q^i_\beta(n) = Q^i_\beta(n+1)$. By substituting $Q^i_\beta(n-1)$ with $Q^i_\beta(n+1)$ in (6.20) we get $Q^i_{\beta_*}(n+1) > Q^i_\beta(n+1)$ which means that β_* is still the estimated best response for player i at iteration $n+1$. By repeating the argument above for all players we show that at time step $n+1$, α_* remains the estimated best response and the claim follows for $m=1$.

Now assume, at every iteration c where $1 \leq c \leq m-1$, α_* is played with probability $\prod_{c=1}^{m-1}\left(1-(1-\vartheta)^c\right)^N$. At time step $n+m$, X^i is updated as,

$$X^i(n+m) = (1-\vartheta)^m X^i(n) + (1-(1-\vartheta)^m)e^i_{\beta_*}. \tag{6.21}$$

By Proposition 1

$$Q^i_{\beta_*}(n+m) = a(m)Q^i_{\beta_*}(n) + b(m)Q^i_{\beta_*}(n-1) + c(m)u^i_{\beta_*}(n). \tag{6.22}$$

Since for player i any action $\beta \neq \beta_*$ is not played, it follows that $Q^i_\beta(n+m) = Q^i_\beta(n)$ (Remark 14). Moreover, $Q^i_{\beta_*}(n+m) > Q^i_{\beta_*}(n+m-1) > Q^i_\beta(n+m-1)$ and $u^i_{\beta_*}(n+m) > Q^i_\beta(n+m-1)$. From (6.21), at time step $n+m+1$, α_* is played with probability of at least $(1-(1-\vartheta)^m)^N$. Hence, from proposition 1,

$$Q^i_{\beta_*}(n+m+1) = a(1)Q^i_{\beta_*}(n+m) + b(1)Q^i_{\beta_*}(n+m-1) + c(1)u^i_{\beta_*}(n+m).$$

With the same argument as for $m=1$,

$$Q^i_{\beta_*}(n+m+1) > Q^i_\beta(n+m+1).$$

Therefore, the estimated best response for player i is not changed. In other words

$$B^i(Q^i(n+m+1)) = B^i(Q^i(n)) = e^i_{\beta_*}.$$

By substituting (6.21) into (6.5) and setting $B^i(Q^i(n+m)) = e^i_{\beta_*}$

$$X^i(n+m+1) = (1-\vartheta)^{m+1}X^i(n) + (1-(1-\vartheta)^{m+1})e^i_{\beta_*}. \tag{6.23}$$

This completes the induction argument.

□

Corollary 6.2 *If for sufficiently large $m > 0$ the conditions of Proposition 3 hold, then for every player i the following holds with probability $\prod_{c=1}^{\infty}[\big(1-(1-\vartheta)^c\big)^N$:*

$$\lim_{m\to\infty} Q^i_{\beta_*}(n+m) = u^i(\alpha_*), \tag{6.24}$$

$$\lim_{m\to\infty} X^i(n+m) = e^i_{\beta_*}. \tag{6.25}$$

Theorem 6.1 *For sufficiently large m, if the conditions in Proposition 3 hold, then $X(n+m)$ converges to a neighborhood of α_* with probability one.*

Proof: The proof is the same to the proof of Theorem 5.1.

6.3.1 Perturbation

With the same perturbation function introduced in Section 5.3.1, consider the following assumptions:

Assumption 6.1 *Step sizes in Q update rule are adjusted based on:*

$$\mu^i_j(n) = (1-\chi^i_j(n)).$$

Assumption 6.2 *When all the players enter the perturbation zone, i.e. $|X^i|_\infty > \zeta$, $\forall i \in \mathcal{I}$, no more than one player chooses a sub-optimal action at each iteration (asynchronous perturbation).*

Assumption 6.3 *The utility map for each player i, $u^i : \mathcal{A}^i \to \mathbb{R}$, is continuous and smooth.*

Theorem 6.2 *If the conditions in Proposition 3 hold for a sufficiently large m, then under Assumption 6.1, Assumption 6.2 and Assumption 6.3, the estimated equilibrium α_* would converge to a Nash equilibrium almost surely.*

Proof: From Corollary 2

$$\lim_{m\to\infty} Q^i_{\beta_*}(n+m) = u^i(\alpha_*),$$

$$\lim_{m\to\infty} X^i(n+m) = e^i_{\beta_*},$$

where $\alpha_* = (\beta_*, \alpha_*^{-i})$. Assume that perturbation becomes active at some large time step $\bar{n}$ and player i chooses an action $\beta \neq \beta_*$. From Section 5.3.1 we know that such perturbation happens with the probability of at least $\xi/|\mathcal{A}^i|$. From (6.4) and Assumption 1, player i updates its action by

$$Q^i_\beta(\bar{n}+1) = K_i\Big[\chi^i_\beta(\bar{n})Q^i_\beta(\bar{n}) + (1-\chi^i_\beta(\bar{n}))u^i(\beta, \alpha_*^{-i})\Big] \tag{6.26}$$

Note that $\nu^i = 0$ in (6.4) since player i changed its action from β_* to β. Consider the following two cases:

- $u^i(\beta, \alpha_*^{-i}(\bar{n})) < u^i(\alpha_*(\bar{n}))$: In this case player i failed to improve the utility and will stay at the estimated equilibrium α_* almost surely.
- $u^i(\beta, \alpha_*^{-i}(\bar{n})) = u^i(\alpha_*(\bar{n}))$: With $\bar{X}^i_\beta$ sufficiently close to zero the Q-values for the actions β is updated to a value sufficiently close to $u^i(\beta, \alpha_*^{-i}(\bar{n})) = u^i(\alpha_*(\bar{n}))$. Hence, in the worst case, when $\alpha_*(\bar{n})$ is only played once, the Q-values for the actions β and β_* may become equal. Therefore, in the action selection stage, the

set of best response in (5.5) may contain two pure actions. Consequently, the mixed strategies for both actions β and β_* is updated by (5.4) and player i chooses one of them randomly in the next time step.

- $u^i(\beta, \alpha_*^{-i}(\bar{n})) > u^i(\alpha_*(\bar{n}))$: In this case player i found a response that is better than α_*. Since the action β has small probability $\bar{X}^i_\beta$, i.e. sufficiently close to 0, then $(1 - \bar{X}^i_\beta)$ is sufficiently close to 1. Thus from (5.23), Q^i_β is updated to a value sufficiently close to $K_i u^i(\beta, \alpha_*^{-i}(\bar{n}))$ and the action profile $(\beta, \alpha_*^{-i}(\bar{n}))$ becomes the new estimated best response, i.e. $\alpha_*(\bar{n}+1) := (\beta, \alpha_*^{-i}(\bar{n}))$. Note that player's utility is improved by $u^i(\alpha_*(\bar{n}+1)) - u^i(\alpha_*(\bar{n}))$, and by (2.2) the potential would also increase by the same amount. Recall that from finite improvement property, such improvements in potential function is not limitless and will terminate at the potential maximum, i.e. Nash equilibrium in potential games. Hence, the estimated equilibrium α_* will eventually converge to an actual Nash equilibrium.

□

6.4 Simulation Results

In this section, we present simulation results of PQL algorithm in MCC. The environment is a 40×40 square area over which a GMM utility distribution is defined. The GMM distribution in PQL simulation is the same as the GMM in Chapter 3, Chapter 4 and Chapter 5. The robots also have no initial information about the utility distribution. Unlike BLLL and SBLLL simulation setup, and similar to SOQL setup there is no communication between agents.

A group of five robots ($N = 5$) scatter through this GMM to maximize their utility function. In PQL, the robots sample the utility distribution at each time to create a memory of the payoff of the actions that they played. This memory can help the robots to find the game's global optimum if the environment is explored enough. Recall that players' utility functions are separable. Therefore, with a set of stationary targets, utility distribution remains stationary for each agent as it is not changed by the actions of other players.

The simulation parameters are chosen as $K^i = 3 \times 10^{-5}$ for $i = 1, ..., N$, $\delta = 1.5$, $\mu = 0.97$, $\vartheta = 0.5$, $\zeta = 0.9999$, $\xi = 0.01$ and $K_i = \dfrac{1}{1+2\mu}$. The algorithm's pseudo-code is as follows:

```
set $n = 1$
**for** each robot $i \in \mathcal{I}$ **do**
    initialize $\alpha^i(1) \in \mathcal{L}$ randomly
    initialize $Q^i(0)$ and $Q^i(1) \in \mathbb{R}^{|\mathcal{L}|}$
    initialize $X^i \in \mathbb{R}^{|\mathcal{L}|}$
**while** the covered worth $\sum_{i \in \mathcal{I}} C^i(\alpha^i)$ is not in a steady state **do**
    **for** each player $i \in \mathcal{I}$ **do**
        perform an action $\alpha^i(n) = \beta^i$ from $\mathcal{A}_c^i$ based on $X^i(n)$
        receive $u^i(n)$
        $u^i_\beta(n) \leftarrow \mathbb{1}_{\{\alpha^i(n)=\beta\}}\, u^i(n)$
        $Q^i_\beta(n+1) \leftarrow K_i\Big[(1-\mu^i)Q^i_\beta(n) + \mu^i u^i_\beta(n)\Big] + \dfrac{\nu^i K_d}{\mu^i}\Big[Q^i_\beta(n-1) - Q^i_\beta(n)\Big]$
        **for** each $\beta' \in \mathcal{A}^i, \beta' \neq \beta$ **do**
            $Q^i_{\beta'}(n+1) \leftarrow Q^i_{\beta'}(n)$
        $X^i(n+1) \leftarrow (1-\vartheta)X^i(n) + \vartheta BR^i(Q^i(n))$
    $n \leftarrow n+1$
```

Simulation results of PQL are shown in Fig. 6.1. By increasing K_d from 0 to $0.4K_{dmax}$ the algorithm's convergence rate increases. As we continue to increase K_d from $0.4K_{dmax}$ to $0.8K_{dmax}$, the algorithm's convergence rate decreases. It is interesting that as we cross the K_d's limit, i.e. K_{dmax}, the algorithm fails to converge to a single value. This implies that the proposed bound $K_d < (\mu + \mu^2)/(1+2\mu)$ could be the tightest limit for K_d.

Fig.6.2 studies the performance of PQL at its maximum efficiency and the performance of the model-based BLLL in Chapter 3. The performance of the model-based learning is almost the same to PQL in terms of convergence rate and the covered worth. Recall that the EM algorithm in Chapter 3 predicts the GMM's component parameters in (3.4), i.e. ω^k, λ^k and Σ^k. Clearly, dependence of the model-based learning on EM narrows down its application to the environment with Gaussian mixture. Meanwhile, PQL, a payoff-based RL, performs identically in any unknown erratic environment. The final configuration of agents for PQL with $K_d = 0.4K_{dmax}$ is presented in Fig 6.3. Agents located two targets while the third target remained unexplored. It could be due to the trade-off between the

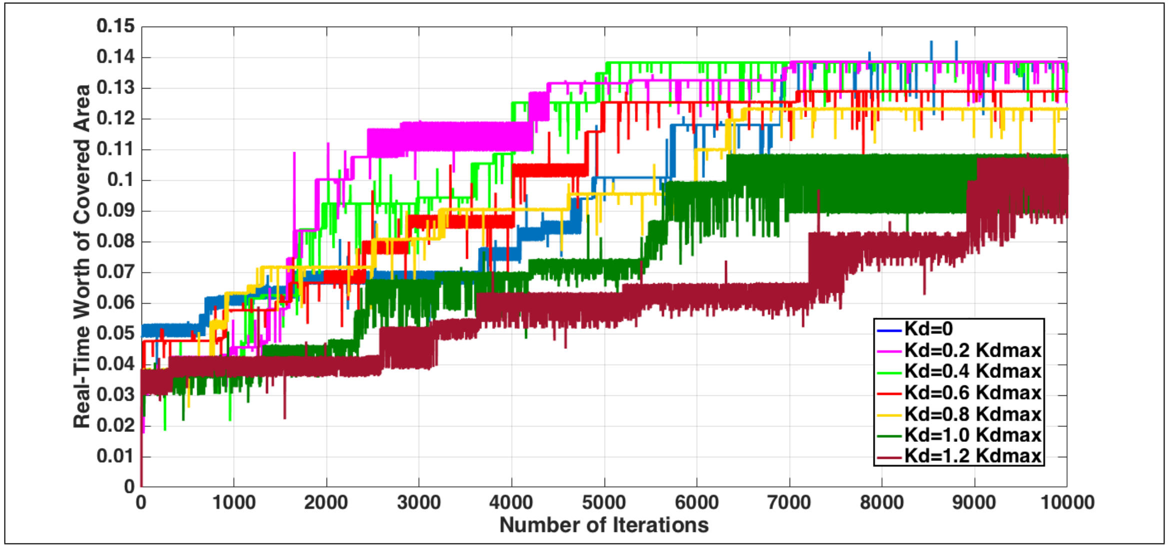

Figure 6.1. Performance varies as K_d increases

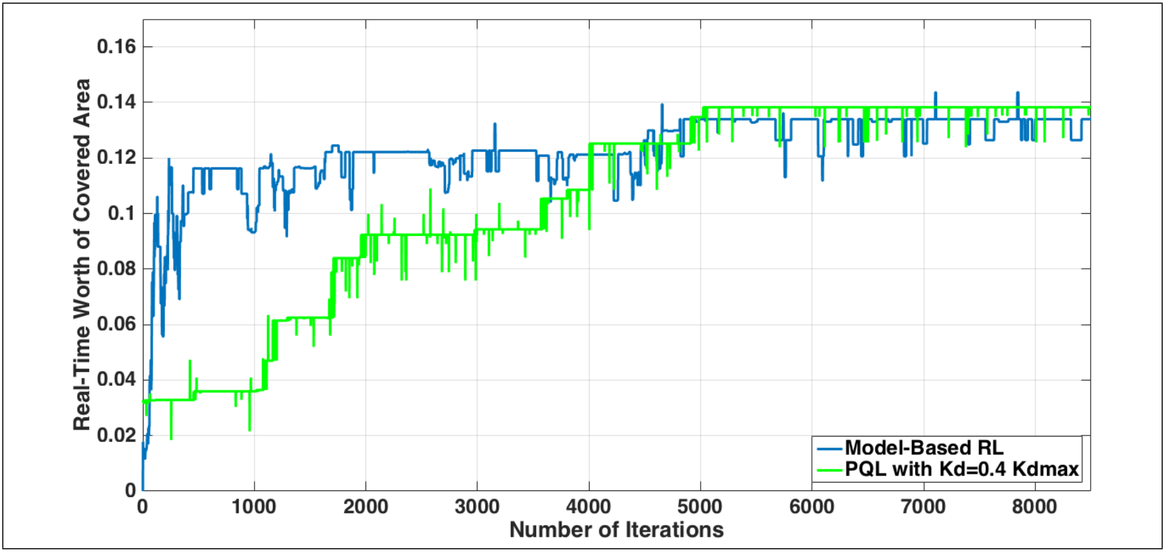

Figure 6.2. Performance comparison between PQL with $K_d = 0.4K_{dmax}$ and the model-based BLLL as in Chapter 3

energy consumption and the coverage worth. This means that we can either decrease the importance of energy consumption in agents' utility or increase the perturbation.

6.5 Discussion

In this chapter we proposed a modified Q-learning in which we introduced a differentiator term in the aggregation step. We presented an effort to increase the convergence rate of the Q-learning algorithm. The differentiator term predicts the evolution of the Q value versus the utility value which increases the algorithm's performance. However, it is shown that the differentiation effect must be carefully regulated.

PQL is a comparable learning algorithm to the model-based BLLL, in terms of convergence rate and also the equilibrium worth. However, PQL does not require an auxiliary estimation algorithm to form a model of the environment. Thus, PQL can be applied to MCC problems where the environment is not a GMM. Although PQL succeed to reach its primary goal, i.e. a faster convergence comparing to SOQL, PQL is not comparable to the model-based SBLLL both in convergence rate and equilibrium worth.

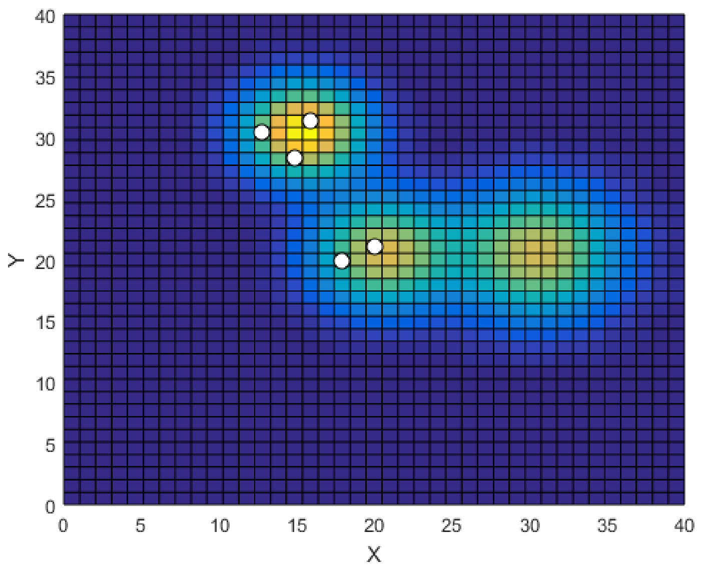

Figure 6.3. The final configuration of agents in PQL

Chapter 7

Conclusions and Future Work

In this thesis we investigated a MCC problem in unknown environments and we addressed a game theoretic learning approach as a solution to this problem. The MCC problem is concerned with the design of rules for coordinating a set of robots' action within an environment. In MCC, the main objective of robots is to cover an area towards achieving a particular goal.

The main focus of our study in this thesis was to improve and introduce new learning algorithms in games and to investigate their application in the MCC problem. Each learning algorithm proposed, relaxed some assumptions in learning to fit a class of MCC problems. Each algorithm, had its own advantages and limitations which we have summarized in the following:

- SBLLL in Chapter 4 enhanced the way the group of agents interact with their environment. We showed that comparing to BLLL in Chapter 3, SBLLL's performance is excellent in a model-based learning scheme. Comparing to BLLL, a higher convergence rate, as it was expected, is because in SBLLL players learn in parallel. Furthermore, because of this simultaneous learning in SBLLL, agents can widely explore the environment.

  Another valuable feature of SBLLL is that each agent's exploration is dependent on the agent's situation in the environment. Thus, each agent can autonomously decide whether it is a good time for exploration or it is better to remain on its current state. This will certainly reduce redundant explorations in SBLLL comparing to BLLL.

Despite these improvements, SBLLL still needed an estimation model of the environment. This means that players in SBLLL have to know the structure of the utility distribution so they can apply an appropriate model on the environment. In practice, this may not be an acceptable assumption. In order to relax this assumption, we turned to RL algorithms which do not require a model of the environment and players only need to monitor the sequence of their own actions' outcome. This relaxes the need for an estimation model in SBLLL.

- A RL scheme can be employed when the utility distribution is not a GMM or more generally, when the utility structure is unknown. However, due to the lack of model from the environment, RL needs more time to explore it. Additionally, in a RL approach, the targets can move within the environment, i.e. the utility distribution can be non-stationary. A RL scheme is also communication-free which means that the implemented control mechanism in a RL scheme can be highly distributed.

  SOQL in Chapter 5 is a RL solution to the MCC problem. In SOQL we proposed a second order reinforcement to increase the algorithm's aggregation depth. SOQL's performance as a model-free RL algorithm is comparable to the model-based SBLLL in terms of equilibrium worth. However, in terms of the convergence rate, SOQL can not reach SBLLL's performance. In order to increase the learning rate of RL in the MCC problem, in Chapter 6, we introduced a differentiation in the aggregation step.

- Chapter 6 proposed a predictive RL algorithm and provided conditions for convergence to a pure Nash equilibrium in potential games. In standard RL algorithms, score values are updated by an integration scheme. This integration is further used as a memory to understand the actual outcome of the states. In PQL, we introduced an anticipatory term incorporated with the integration component. Such anticipation reuses past experiences to predict the Q-values trend which increases the algorithm's learning rate. However, it is shown that the differentiation effect must be carefully regulated.

PQL's performance is comparable to the model-based BLLL, in terms of convergence rate and also the equilibrium worth. However, PQL does not require an auxiliary estimation algorithm to form a model of the environment. Thus, PQL can be applied to MCC problems where the environment is not a GMM. However, PQL is not comparable to the model-based SBLLL both in convergence rate and equilibrium worth.

It is worthwhile to emphasize that in comparison to non-learning approaches (e.g., [42], [43] and [44]), in learning schemes the utility distribution can be unknown. The assumption of unknown utility holds in many of practical applications. Thus, an algorithm which can satisfy this assumption is more applicable on practical setups.

7.1 Future Work

There are several ways to improve the results of this work that we will address and suggest for future research in MCC problems:

- The convergence rate of SBLLL can be improved if the trial action is selected based on the sensed gradient of the utility. With this trial action selection scheme, one can actually incorporate the gradient of the utility to increase the adaptability of the algorithm to the environment. SBLLL's convergence rate can also be characterized in terms of the algorithm's parameters

- SOQL's performance can be investigated with a third or higher order aggregation. We believe that by increasing the order of aggregation, the algorithm's performance improves in terms of convergence rate and stability of the equilibrium.

- The differentiator term in PQL predicts the evolution of the Q value versus the utility value which increases the algorithm's performance. However, it is showed that the differentiation effect must be carefully regulated. Thus, the future work is to determine the optimum differentiator coefficient.

- The communication cost can also be incorporated in the utility function. In practice, robots' on-board resources are limited. Thus incorporation of the communication cost can optimize the energy consumption.

Appendices

Appendix I

Proposition 3.1 in [35]: In a finite N-player potential game if all players adhere to log-linear learning then the stochastically stable states are the set of potential maximizers.

Proof: By Lemma 3.1 in [35], LLL induces a perturbed Markov process. Thus an action profile $\alpha \in \mathcal{A}$ is stochastically stable if and only if there exists a minimum resistance tree rooted at α.

We use contradiction to prove our claim. Consider a minimum resistance tree T which is rooted at α and assume α does not maximize the potential function. Let α_* be the action profile that maximizes the potential function. To reach a contradiction, we show that T is not a minimum resistance tree and there exists a tree rooted at α_* with a resistance less than T (see Section 2.2.2.2).

As T is rooted at α, there exists a path $\mathit{\Omega}$ from α_* to α, i.e. $\{\alpha_* \rightarrow \alpha_1 \rightarrow ... \rightarrow \alpha_m \rightarrow a\}$. Consider the reverse path $\mathit{\Omega}^R$ as $\{\alpha \rightarrow \alpha_m \rightarrow ... \rightarrow \alpha_1 \rightarrow \alpha_*\}$. We construct a new tree T' rooted at α_* by adding the edges of $\mathit{\Omega}^R$ to T and removing the edges of $\mathit{\Omega}$. Clearly, the resistance of the new tree is:

$$R(T') = R(T) + R(\mathit{\Omega}^R) - R(\mathit{\Omega}).$$

We then have (see Proposition 5.1 in Chapter 5):

$$R(T') = R(T) + \phi(\alpha) - \phi(\alpha_*).$$

Recall we assumed that α_* is the action profile that maximizes the potential function. Thus $\phi(\alpha) < \phi(\alpha_*)$ and $\phi(\alpha) - \phi(\alpha_*) < 0$. We then conclude that $R(T') < R(T)$ and T is not the tree with a minimum resistance among all the trees rooted at other states. This is in contrast with our assumption about the resistance of the tree T.

By repeating the above analysis we can show that all action profiles that maximize the potential have the same stochastic potential, i.e. all potential maximizers are stochastically stable.

Appendix II

Theorem 4.2 in [35]: Consider any finite N-player potential game where all players adhere to synchronous log-linear learning with an independent revision probability and with an update parameter $\omega \in R^+$. If $\omega = (e^{-\frac{1}{\tau}})^m$ then for sufficiently large m, the stochastically stable states are the set of potential maximizers.

Proof: Consider a transition from $\alpha \to \alpha_1$. The probability of this transition is:

$$\sum_{S \subseteq \mathcal{I}: G \subseteq S} \epsilon^{m|S|} \left(1-\epsilon^m\right)^{|\mathcal{I} \setminus S|} \prod_{i \in S} \frac{\epsilon^{u^i(\alpha_1^i, \alpha^{-i})}}{\sum_{\alpha_k^i \in \mathcal{A}^i} \epsilon^{u^i(\alpha_k^i, \alpha^{-i})}},$$

where $G = \{i \in \mathcal{I} \; s.t. \; \alpha^i \neq \alpha_1^i\}$ and $\epsilon = e^{-\frac{1}{\tau}}$. SLLL with independent revision probability is a perturbed Markov process and the resistance of any transition $\alpha \to \alpha_1$ is

$$R(\alpha \to \alpha_1) = m|G| + \sum_{i \in G} \Big(\max_{\alpha_*^i \in \mathcal{A}^i} \; u^i(\alpha_*^i, \alpha^{-i}) - u^i(\alpha_1^i, \alpha^{-i}) \Big),$$

where the unperturbed process corresponds to the situation where players never do the trial process, i.e. $\omega = 0$.

We then normalize $\phi(\alpha)$ to be between 0 and 1 for all $\alpha \in \mathcal{A}$. By using this normalization we have:

$$m|G| + n \geq R(\alpha \to \alpha_1) \geq m|G|. \tag{1}$$

Let T be a minimum resistance tree rooted at α' where α' is the stochastically stable state. If there is a single deviator for each edge of T the argument will reduce to Proposition 3.1 in Appendix I.

Assume that there exists an edge $\alpha \to \tilde{\alpha}$ with multiple deviators. Let $G = \{i \in \mathcal{I} : \alpha^i \neq \tilde{\alpha}^i\}$. The minimum resistance for this transition is $m|G|$, .i.e $R(\alpha \to \tilde{\alpha}) \geq m|G|$. For this transition we can consider a path $\Omega = \{\alpha = \alpha_0 \to \alpha_1 \to ... \to \alpha_{|G|} = \tilde{\alpha}\}$ in which every sub-transition $\alpha_k \to \alpha_{k+1}$ is an unilateral deviation by some player $i \in G$. By (1) we know that the resistance of each edge is at most $m + n$, i.e. $R(\alpha_{k-1} \to \alpha_k) \leq m + n$.

Thus the maximum resistance of the path $\mathit{\Omega}$ is $|G|(m+n)$. Since $|G| \leq n$, we have $|G|(m+n) \leq m|G| + n^2$. Now consider a new tree T' rooted at α_* which can be easily constructed by adding the edges of $\mathit{\Omega}$ to T and removing the edges of $\mathit{\Omega}^R$. The total resistance of $\mathit{\Omega}^R$ is at least $m|G|+m(|G|-1)$. Hence the following holds for the resistance of the tree T'.

$$R(T') = R(T) + R(\mathit{\Omega}) - R(\mathit{\Omega}^R) \leq R(T) + m|G| + n^2 - \Big[m|G| + m(|G|-1)\Big].$$

Recall that we assumed that the edge $\alpha \to \tilde{\alpha}$ has multiple deviators, i.e. $|G| \geq 2$. Therefore, if $m \geq n^2$ then $R(T') < R(T)$. This implies that that only potential maximizers are stochastically stable. The argument above can be repeated for every edge with multiple deviators.

Appendix III

Theorem 5.1 in [35]: In a finite N-player potential game satisfying Assumptions 5.1 and 5.2 and with potential function $\phi : \mathcal{A} \mapsto R$, if all players adhere to BLLL, then the stochastically stable states are the set of potential maximizers.

Proof: Lemma 5.1 in [35], shows that in BLLL algorithm, the resistance of any transition $\alpha_0 \to \alpha_1 = (\alpha_1^i, \alpha_0^{-i})$ where $\alpha_1^i \in \mathcal{A}_c^i(\alpha_0^i)$ is

$$R(\alpha_0 \to \alpha_1) = \max_{\alpha_*^i \in \{\alpha_1^i, \alpha_0^i\}} u^i(\alpha_*^i, \alpha_0^{-i}) - u^i(\alpha_1).$$

With a same approach to Proposition 5.1, it is easy to show that for any feasible path

$$\mathit{\Omega} := \{\alpha_0 \to \alpha_1 \to ... \to \alpha_m\}$$

and its reverse path

$$\mathit{\Omega}^R := \{\alpha_m \to \alpha_{m-1} \to ... \to \alpha_0\}$$

the resistance difference is

$$R(\mathit{\Omega}) - R(\mathit{\Omega}^R) = \phi(\alpha_0) - \phi(\alpha_m). \tag{2}$$

Now assume that α is the stochastically stable state and there exist a minimum resistance tree T rooted at α for which the resistance is minimum among all the trees rooted at other states. As the contradiction assumption, suppose the minimum resistance tree T does not maximize the potential function and let α_* be the action profile that maximizes the potential function. Since T is rooted at α there exist a path $\mathit{\Omega}_p$ from α_* to α as

$$\mathit{\Omega}_p = \{\alpha_* \to \alpha_1 \to ... \to \alpha\}.$$

Consider the reverse path $\mathit{\Omega}_p^R$ from α to α_*

$$\mathit{\Omega}_p^R = \{\alpha \rightarrow \alpha_m \rightarrow ... \rightarrow \alpha_*\}.$$

We can now construct a new tree T' rooted at α_* by adding the edges of $\mathit{\Omega}_p^R$ to T and removing the edges of $\mathit{\Omega}_p$. The resistance of the new tree is

$$R(T') = R(T) + R(\mathit{\Omega}_p^R) - R(\mathit{\Omega}_p).$$

By (2),

$$R(T') = R(T) + \phi(\alpha) - \phi(\alpha_*).$$

Recall that we assumed $\phi(\alpha_*)$ is the maximum potential. Hence $\phi(\alpha) - \phi(\alpha_*) < 0$ and consequently $R(T') < R(T)$. Therefore T is not a minimum resistance tree among the trees rooted at other states, which is in contrast with the basic assumption about the tree T. Hence, the supposition is false and α is the potential maximizer.

References

[1] P. Stone, M. Veloso, “Multi-agent systems: a survey from a machine learning perspective”, Autonomous Robots, pp. 345-383, 2000.

[2] Zhi Yan, Nicolas Jouandeau, and Arab Ali Cherif, “A survey and analysis of multi-robot coordination”, International Journal of Advanced Robotic Systems, 10(399):1-18, 2013.

[3] S. Dhillon, K. Chakrabarty, S. Iyengar, “Sensor Placement for Grid Coverage under Imprecise Detections”, Proceedings of international conference on Information Fusion, pp. 1571-1587, 2002.

[4] D. Ucinski, “Measurement Optimization for Parameter Estimation in Distributed Systems”, CRC Press, 1999.

[5] C. Guestrin, A. Krause, and A. P. Singh, “Near-optimal sensor placements in Gaussian processes”, Proceedings of the 22nd International Conference on Machine Learning, New York, pp. 265-272, 2005.

[6] J. R. Spletzer and C. J. Taylor, “Dynamic Sensor Planning and Control for Optimally Tracking Targets”, International Journal of Robotics Research, vol. 22, no. 1, pp. 7-20, 2003.

[7] T. Nakamoto, H. Ishida, and T. Moriizumi, “Active odor sensing system”, Proceedings of international symposium on Industrial Electronics, vol. 1, pp. SS128-SS133, 1997.

[8] J.R. Frost and L.D. Stone, "Review of Search Theory: Advances and Applications to Search and Rescue Decision Support", Technical Report CG-D-15-01, U.S. Coast Guard Research and Development Center, Gronton, CT, 2001.

[9] W. Chen and W. Tseng, "A Novel Multi-agent Framework for the Design of Home Automation", Fourth International Conference on Information Technology (ITNG'07), pp. 277-281, 2007.

[10] T. Heng, Y. Kuno, and Y. Shirai, "Active sensor fusion for collision avoidance", in Proc of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, vol. 3, pp. 1244-1249, 1997.

[11] A. Nowe, P. Vrancx, and Y.-M. D. Hauwere, "Game theory and multi-agent reinforcement learning", Reinforcement Learning: State-of-the-Art, chapter 14, Springer, 2012.

[12] Rahili, S., Ren, W., "Game theory control solution for sensor coverage problem in unknown environment", 53rd IEEE Conference on Decision and Control (CDC), pp.1173-1178, 2014.

[13] W. Yatao and L. Pavel, "A Modified Q-Learning Algorithm for Potential Games", Proceedings of the 19th IFAC World Congress, vol.19, pp. 8710-8718, 2014.

[14] J. Marden and A. Wierman, "Distributed welfare games with applications to sensor coverage", 47th IEEE Conference on Decision and Control, pp. 1708-1713, 2008.

[15] L. Pavel, "Game Theory For Control of Optical Networks", Springer Science and Business Media, 2012.

[16] Li, W., Cassandras, C.G., "Distributed Cooperative coverage Control of Sensor Networks", 44th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC 2005), pp. 2542-2547, 2005.

[17] Brown, G. W., “Iterative solution of games by Fictitious Play”, Activity Analysis of Production and Allocation, pp. 374-376, Wiley: New York, 1951.

[18] Young, H.P., “ The evolution of conventions”, Econometrica, pp. 57-84, 1993.

[19] Marden, J.R., Arslan, G., and Shamma, J.S., “Joint strategy fictitious play with inertia for potential games.”, IEEE Transactions on Automatic Control, 54, pp. 208-220, 2009.

[20] Hasanbeig M, Pavel L., “From game-theoretic multi-agent log linear learning to reinforcement learning”, arXiv preprint, arXiv:1802.02277.

[21] Cortes, A., Martinez, S., “Self-triggered best-response dynamics for mobile sensor deployment”, American Control Conference (ACC), pp. 6370-6375, 2013.

[22] P. Mertikopoulos and W. H. Sandholm, “Regularized best responses and reinforcement learning in games”, Mathematics of Operations Research, 2014.

[23] R. Laraki and P. Mertikopoulos, “Higher order game dynamics”, Journal of Economic Theory, vol. 148, no. 6, pp. 2666-2695, 2013.

[24] H. Bayram and H. Bozma, “Multirobot communication network topology via centralized pairwise games”, IEEE International Conference on Robotics and Automation, 2013.

[25] K. Srivastava and M. Spong, “Multi-agent coordination under connectivity constraints”, American Control Conference, 2008.

[26] Chung, T.H., Gupta, V., Burdick, J.W., Murray, R.M., “On a decentralized active sensing strategy using mobile sensor platforms in a network”, 43rd IEEE Conference on Decision and Control (CDC), pp.1914-1919, Vol.2, 2004.

[27] G. Atinc, D. Stipanovic, P. Voulgaris and M. Karkoub, “Supervised coverage control with guaranteed collision avoidance and proximity maintenance”, 52nd IEEE Conference on Decision and Control, pp. 3463-3468, 2013.

[28] G. Atinc, D. Stipanovic, P. Voulgaris and M. Karkoub, “Swarm-based dynamic coverage control”, 53rd IEEE Conference on Decision and Control, pp. 6963-6968, 2014.

[29] Y. Meng, “Multi-Robot Searching using Game- Theory Based Approach”, International Journal of Advanced Robotic Systems, p. 350, 2008.

[30] R. Cui, J. Guo and B. Gao, “Game theory-based negotiation for multiple robots task allocation”, Robotica, vol. 31, no. 6, pp. 923-934, 2013.

[31] Hasanbeig M, Pavel L., “On synchronous binary log-linear learning and second order Q-learning”, International Federation of Automatic Control (IFAC), pp. 8987-8992, 2017.

[32] Hasanbeig M, Pavel L., “Distributed coverage control by robot networks in unknown environments using a modified EM algorithm”, International Journal of Computer and Information Engineering, vol. 11, no. 7, pp. 815-823, 2017.

[33] L. Mihaylova, T. Lefebvre, H. Bruyninckx, and K. Gadeyne, “Active sensing for robotics-a survey”, Proceedings of international conference on Numerical Methods and Applications, Borovets, Bulgaria, pp. 316-324, 2002.

[34] Monderer, D., Shapley, L.S., “Potential Games”, Games and Economic Behavior, pp. 124-143, 1996.

[35] J. Marden and J. Shamma, “Revisiting log-linear learning: Asynchrony, completeness and payoff-based implementation”, 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton), 2010.

[36] Von Neumann, J., O. Morgenstern, “Theory of Games and Economic Behavior”, Princeton: Princeton University Press, 1944.

[37] Sutton, R. S., and Barto, A. G., “Reinforcement Learning: An Introduction”, MIT Press, 1988.

[38] P. Coucheney, B. Gaujal and P. Mertikopoulos, “Penalty-Regulated Dynamics and Robust Learning Procedures in Games”, Mathematics of Operations Research, vol. 40, no. 3, pp. 611-633, 2015.

[39] Benaïm, M.,“Dynamics of stochastic approximation algorithms”, Seminare de probabilites de Strasbourg, 1999.

[40] McKelvey, R. D., Palfrey, T. R., “Quantal Response Equilibria for Normal Form Games”, Games and Economics Behavior, pp. 6-38, 1995.

[41] L. Blume, “The statistical mechanics of strategic interaction”, Games and Economic Behavior, pp. 387- 424, 1993.

[42] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, “Coverage control for mobile sensing networks”, IEEE Transactions on Robotics and Automation, vol. 20, no. 2, pp. 243-255, 2004.

[43] J. Cortes, S. Martinez and F. Bullo, “Spatially-distributed coverage optimization and control with limited-range interactions”, ESAIM: Control, Optimisation and Calculus of Variations, vol. 11, no. 4, pp. 691-719, 2005.

[44] A. Kwok and S. Martinez, “Deployment algorithms for a power-constrained mobile sensor network”, IEEE International Conference on Robotics and Automation, 2008.

[45] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm”, Journal of the Royal Statistical Society Series B, vol. 39, no. 1, pp. 1-38, 1977.

[46] V. Lakshmanan and J. S. Kain, “A Gaussian Mixture Model Approach to Forecast Verification”, Weather and Forecasting, vol. 25, pp. 908-920, 2010.

[47] Y. Lim and J. S. Shamma, “Robustness of stochastic stability in game theoretic learning”, American Control Conference, pp. 6145-6150, 2013.

[48] Akaike, H., "A new look at the statistical model identification", IEEE Transactions on Automatic Control, vol.19, no.6, pp. 716-723, 1974.

[49] N. Ueda, R. Nakano, Z. Ghahramani and G. Hinton, "Split and Merge EM Algorithm for Improving Gaussian Mixture Density Estimates", The Journal of VLSI Signal Processing, vol. 26, no. 12, pp. 133-140, 2000.

[50] Chasparis, G.C., Shamma, J.S., and Rantzer, A., "Perturbed learning automata in potential games", Proceedings of the IEEE Conference on Decision and Control, pp. 2453-2458, 2011.

[51] D. Leslie and E. Collins, "Individual Q -Learning in Normal Form Games", SIAM Journal on Control and Optimization, vol. 44, no. 2, pp. 495-514, 2005.

[52] Erev, Ido, Alvin E. Roth, "Predicting how people play games: Reinforcement learning in experimental games with unique, mixed strategy equilibria", American Economic Review, pp. 848-881, 1998.

[53] R. Cominetti, E. Melo and S. Sorin, "A payoff-based learning procedure and its application to traffic games", Games and Economic Behavior, vol. 70, no. 1, pp. 71-83, 2010.

[54] E. Hopkins and M. Posch, "Attainability of boundary points under reinforcement learning", Games and Economic Behavior, vol. 53, no. 1, pp. 110-125, 2005.

[55] J. Yao, "On Recursive Estimation in Incomplete Data Models", Statistics, vol. 34, no. 1, pp. 27-51, 2000.