# OPTIMAL CONTROL STRATEGIES AND REINFORCEMENT LEARNING FOR DYNAMICAL MULTIAGENT SYSTEMS IN GRAPHICAL GAMES

by

VICTOR GABRIEL LOPEZ MEJIA

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON

August 2019

## ACKNOWLEDGEMENTS

I wish to express my gratitude to all the people that helped me complete the road of a doctoral program. I deeply thank my thesis adviser and friend, Dr. Frank L. Lewis, for his teachings, his guidance and the motivation he provided me during all these years. It has been my honor to be his student. With the same emphasis I thank my coadviser, Dr. Yan Wan, for her constant support and assistance to perform my research. Her passion for her work and her attention to detail are exemplary among all researchers.

I also want to thank the members of my dissertation committee, Dr. William Dillon, Dr. Ramtin Madani and Dr. Manfred Huber, for their helpful comments, their interesting insights and their unreserved willingness to devote some of their time to evaluate my research.

With my whole heart I thank all of my friends and collaborators at UTA, people from all around the world with goals similar to mine and whose support and friendship were invaluable throughout this experience. It would be unfair not to explicitly mention my two closest friends, Mushuang Liu and Patrik Kolaric. During these years, you guys made my worst moments bearable and my best moments possible.

The pillar of my career will always be my family, to which I owe every bit of success I may achieve. I thank my parents, Gabriel and Victoria, and my siblings, Melchor and Gabriela, for their words of advice and encouragement, and the unconditional support I always get from them. It is much easier to move forward when I know they have my back.

I finally thank the Mexican Council of Science and Technology (Conacyt) for the economic support provided to perform my PhD studies. This research was also partially supported by ONR grant N00014-17-1-2239, ONR Grant N00014-18-1-2221, NSF Grant ECCS-1839804.

June 05, 2019

ABSTRACT

# OPTIMAL CONTROL STRATEGIES AND REINFORCEMENT LEARNING FOR DYNAMICAL MULTIAGENT SYSTEMS IN GRAPHICAL GAMES

VICTOR G. LOPEZ MEJIA

The University of Texas at Arlington, 2019

Suppervising professor: Frank L. Lewis

As the number of artificial autonomous agents increases in industrial and urban areas, the development of formal protocols to analyze their behavior as they interact with each other becomes of central interest in control systems research. Each agent in this setting is interested in completing a specific task with considerations of an optimal performance. Game theory has become one of the most useful tools in multiagent systems analysis due to its rigorous mathematical representation of optimal decision making. The analysis of dynamical systems has been developed in the branch of game theory regarded as differential games. The set of graphical games consider also limited sensing capabilities among the agents, such that they can only measure the state of their closest neighbors.

This dissertation presents the formulation of different solutions for differential graphical games. The proposed solutions represent various scenarios for the interactions of multiagent systems, on which the agents face different conditions in their environments, their goals or their ability to speculate about the behavior of their neighbors. First, Bayesian Games are formulated to describe the case on which an agent is uncertain about the intentions of its neighbors. Conditions for Bayes-Nash equilibrium are provided. Then, Minmax strategies are analyzed for graphical games as an alternative for Nash equilibrium. Stability and robustness properties are thoroughly investigated. We prove that Minmax strategies improve the robustness properties of the single-agent LQR controller. As a particular application of the applicability of Minmax strategies, Pursuit-Evasion Games are then analyzed. In these games, different behaviors are obtained between both multiagent teams by varying the individual performance indices. Finally, Minmax Regret and Projection Strategies are proposed as two additional solution concepts that allow the agents to make assumptions about the information available to their neighbors.

TABLE OF CONTENTS

## List of illustrations

# Chapter 1

# INTRODUCTION

Game theory has become one of the most useful tools in multiagent systems analysis due to its rigorous mathematical representation of optimal decision making [1]-[3]. In a game-theoretic setting, a group of players, here regarded as agents, must decide the appropriate actions they must perform to complete specific tasks with considerations of maximizing their rewards or minimizing their costs. Most applications of game theory consider static agents pondering which action from a discrete set is optimal [1]. In contrast, differential game theory is the area of mathematics that studies the interactions of agents with continuous-time dynamics, and that obtain from their behavior a payoff that evolves along time [4], [5].

The modern approach to differential game theory has been described in [5]. It is shown that the agents must solve a set of coupled partial differential equations, regarded as the Hamilton-Jacobi (HJ) equations, to reach a Nash equilibrium. Nash equilibrium is the most important solution concept in game theory. If this equilibrium is achieved, no agent can unilaterally modify its control policy without negatively affect its own performance. This characteristic provides an incentive for the stability of the equilibrium.

The two main classifications of differential games are the zero-sum and nonzero-sum games [5]-[10]. In the former, any improvement in the performance of an agent is obtained at the cost of a decrease in the performance, in the same magnitude, by another agent. This represents strictly conflictive goals between the players. In the more general nonzero-sum games, the agents do not necessarily require someone else's decreased performance to achieve a better payoff in the game. Thus, nonzero-sum games allow the presence of cooperative goals among the agents.

The usual formulation of nonzero-sum games, however, assumes that all agents possess global information about the states of interest in the game. This assumption does not hold in many practical applications of multiagent systems, where the access to state information can be restricted. Consider, for example, practical applications of multiagent systems like platoons of self-driving cars or swarms of unmanned aerial vehicles (UAVs). Each of these autonomous vehicles is provided with multiple sensors to detect and recognize objects in their surroundings. Such sensors can only provide a limited area of sensing, allowing the vehicle to interact only with its closest neighbors. The agents must now use their limited information about the world to complete their tasks with an optimal performance.

Graphical games are the branch of game theory that considers limited sensing capabilities among the agents, such that they can only measure the state of a subset of the players in the game. In graphical games, the agents are taken as nodes in a communication graph with a well defined topology. This graph represents the flow of information from an agent to another, such that their sensing abilities are appropriately modeled. The literature available for control of networked agents is extensive (see [11]-[19] and references therein). Game-theoretic approaches have been recently proposed to provide optimality to the cooperative ([20], [21]) and non-cooperative ([22], [23]) interactions of networked agents.

The limited state information addressed by graphical games is not the only practical consideration required to properly model real-world interactions. In reality, any autonomous agent performing a task must consider the possibility to face uncertain environments, and it is desirable to provide it with the ability to succeed at achieving its goals as best as possible. An important source of uncertainty is presented when an individual needs to interact with an unknown agent. Consider the example of a UAV delivering a package and finding a second UAV approaching it. Our agent would be expected to ask itself the following questions: ‘Is that individual an adversary trying to steal the package? Or is it an uninterested agent heading to its own destination?’ The optimal behavior of the UAV will, of course, depend on the answer to these questions. The assumption made in most game theoretic models is that an agent has precise knowledge about the ojective functions of all other agents. As the number of autonomous agents increases in urban areas, it becomes likely that the true objectives or intentions of other agents will be unknown.

A solution proposed in the literature to address the uncertain objectives in a multiagent system consists of the formulation of Bayesian games [24]-[27]. In Bayesian games, or games with incomplete information, the agents are allowed to have beliefs about the possible objectives of the other players. The optimal strategies of each agent are then computed using their beliefs about the game being played. In this dissertation, the Bayesian games formulation is incorporated into the differential graphical games control structure, providing the agents with optimal strategies in scenarios with different sources of uncertainty or lack of knowledge.

The rest of this dissertation addresses a different challenge in differential graphical games. The concept of Nash equilibrium requires that every agent performs its optimal strategy with respect to the optimal strategies of all other agents. Thus, individual variations from the Nash solution are by definition unfavorable for the performance of an agent. Clearly, an agent can only compute its optimal strategy by simultaneously computing also the optimal strategies of the players around it. This fact reveals a significant disadvantage of graphical games. If an agent can only use its local information to design its control policy, then it will not be able to determine the best strategy of its neighbors. Such neighbors will be using their own local information, unavailable to anyone else. As it is analyzed in subsequent chapters, Nash equilibrium is in general not attainable in graphical games.

Different solution concepts can be defined that can be reached using the limited state information available in differential graphical games. The main solution here proposed is regarded as Minmax strategies. The concept of minmax strategies has been thoroughly studied for zero-sum games, where its application is natural due to the dynamics of the game [1], [22], [28]. The formulation presented in this dissertation, however, is applied to general non-adversarial games. In minmax strategies, an agent decides to ignore the actual objectives of its neighbors, and assumes that they will perform their worst-case behavior against it. That is, an agent attempts to minimize a cost function that its neighbors try to maximize.

The resulting control protocol obtained from minmax strategies is related to the $H_\infty$ formulation for disturbance attenuation [28]-[32]. The $H_\infty$ controller is designed to prepare a system against the worst-case disturbances that it may face. If the system is well-prepared against the worst disturbances, then it will also address properly less severe perturbations. The $H_\infty$ design is thus well-known for its robustness properties. In the minmax formulation here presented, each agent treats the influence of its neighbors as a disturbance to be rejected. This control scheme provides each individual player with a robust policy against the unknown behavior of its neighbors. The obtained value of the game for an agent represents a guaranteed performance value, and an agent can expect its real value to be improved from this upper bound.

Besides minmax strategies, three other solution concepts are defined for multiagent systems in graphs. First, the concept of regret of an agent is explored in the so-called Minmax Regret strategies. It is said that an agent regrets its decision when it uses a control policy that is not optimal given the actual behavior of its neighbors. In graphical games, the behavior of the neighbors is in general unknown, and an agent can expect to have a degree of regret from any control policy it selects. Minmax regret strategies consist in making an agent select the control policy with which it avoids the worst possible regret.

The last two solutions proposed in this dissertation are regarded as Projection strategies. The local information of the neighbors is unknown to an agent in graphical games, but it can attempt to minimize a cost function by making assumptions about those neighbors. The projection strategies are obtained when an agent makes the assumption that its neighbors possess similar information as itself. In one case, an agent assumes that its neighbors possess its same state information, thus solving a nonzero-sum game against them. In the second scenario, the agent assumes that its neighbors desire to move in a similar trajectory as itself. Any of these assumptions allow the agents to successfully determine their corresponding solution of the game.

An additional requirement for a practical solution of graphical games is the consideration of uncertainties in the system dynamics. The usual solutions for differential games and optimal control designs require complete knowledge of the mathematical model of the system. Various reinforcement learning (RL) algorithms have been proposed to solve multiagent optimal control problems with partial or no knowledge of the system dynamics [6], [33]-[35]. Off-policy RL algorithms have been proposed in the literature to solve control optimization problems in an adaptive and efficient manner, without any knowledge of the model of the system [36]-[38]. These capabilities of off-policy algorithms are used in this dissertation to determine the optimal policies of the agents without knowledge of the neighbor control policies. Thus, the analysis performed about the proposed solutions guarantee the existence of the solutions to the HJ equations, and then RL is employed to determine those solutions.

The rest of the dissertation is organized as follows.

- In Chapter 2, preliminary definitions and notations that are used throughout the dissertation are presented. In particular, the mathematical notion of game theory is first described. Then, the basic definitions for graph theory are introduced. Finally, the state variables and the system dynamics for the multiagent systems studied in this research are presented.
- In Chapter 3, Bayesian games are presented to solve differential graphical games with incomplete information. An agent in these games can possess individual information, known only to itself. This formulation leads to the definition of epistemic types. To solve the Bayesian graphical games, the concept of Bayes-Nash equilibrium is introduced for dynamical systems, and this solution is shown to be obtained by solving a set of HJB equations that

include the epistemic beliefs of the agents as a parameter. The beliefs of the agents are constantly updated throughout the game using the Bayesian rule to incorporate new evidence to the individual current estimates of the types. Two belief update methodologies that do not require the full knowledge of the graph topology are developed. The first of these approaches is a direct application of the Bayesian rule, and the second is a modification regarded as a non-Bayesian update.

- In Chapter 4, minmax strategies are proposed to solve non-adversarial differential graphical games as an alternative to Nash equilibrium. Minmax strategies are proven to provide distributed control policies under mild conditions in the system dynamics and the performance functions. The conditions for stability of the global multiagent system when all agents use their minmax policies are studied. Robustness of the control policy of each agent is also analyzed. The gain and phase margins of the minmax policies are determined, and these properties are shown to improve the corresponding characteristics of the linear quadratic regulator (LQR). Finally, an off-policy RL algorithm is designed to solve the minmax problem without any knowledge of the system dynamics.
- Chapter 5 presents the analysis of multiagent pursuit-evasion games in communication graphs. First, a different formulation of the graph topology is introduced. Three communication graphs are presented to represent, respectively, the interactions among the pursuers, interactions among the evaders, and interactions between pursuers and evaders. Conditions for Nash equilibrium with respect to individual performance indices are stated. Then, a natural formulation of minmax strategies is obtained due to the intrinsic characteristics of the pursuit-evasion game. A novel analysis of emergent

behaviors, obtained by making particular modifications to the cost functions of the agents, is finally presented. Finite-time interception and asymptotic behaviors are studied as variations of the individual goals of an agent with respect to its teammates and opponents. The containment control problem with static and moving leaders is also solved as a special case of the MPE games.

- Chapter 6 formulates additional solution concepts for differential graphical games to cope with the general unattainability of Nash equilibrium. Minmax regret strategies are obtained by defining the regret of an agent for not using its optimal control policy against its neighbors. Then, two projection strategies are introduced, on which each agent projects some of its personal information onto its neighbors. First, the agents project their information about their neighbors' states. Alternatively, the agents project their local error information. The conditions for stability of these strategies are provided, and their robustness properties are studied.
- Chapter 7 presents the conclusion of this work and describes present and future lines of research that can be considered to extend and improve the results in this dissertation.
- An Appendix is included at the end of the dissertation to describe the obtained results about the use of reinforcement learning for multiobjective optimization problems. These results are relevant to obtain the solutions of Bayesian games that, as explained in Chapter 3, can be determined from the convex combination of the possible cost functions of the agents.

The publications resulted from this dissertation are listed below:

[1] V.G. Lopez and F. L. Lewis, “Dynamical multiobjective control for continuous-time systems using reinforcement learning,” Accepted for publication in IEEE Transactions on Automatic Control. Online early access available.

[2] V.G. Lopez, Y. Wan and F. L. Lewis, “Bayesian graphical games for synchronization in dynamical networks,” Accepted for publication in IEEE Transactions on Control of Network Systems, 2019.

[3] V. G. Lopez, F. L. Lewis, Y. Wan, M. Liu, G. Hewer and K. Estrabidris, “Stability and robustness analysis of minmax solutions for differential graphical games”. Submitted to Automatica, 2019.

[4] V. G. Lopez, F. L. Lewis, Y. Wan, E. N. Sanchez and L. Fan, “Solutions for multiagent pursuit-evasion games on communication graphs: finite-time capture and asymptotic behaviors”. Submitted to IEEE Transactions on Automatic Control, 2018.

# Chapter 2

# PRELIMINARIES

## 2.1. Introduction

In this chapter, relevant background subjects are reviewed in preparation for the main contents of the dissertation. This is the current state-of-the-art of the topics of interest found in the literature on which this work is based. First, the main concepts of graph theory are described, along with the definition of variables that are used throughout this research. Then, the formal definitions of the state variables for multiagent dynamical systems are presented. This includes the description of local error variables and synchronization objectives. A brief review on game theory is then presented, for both static and differential games. The results presented in this dissertation are extensions to the differential game theory concepts presented in this chapter. Finally, miscellaneous definitions and notations, useful for our purposes in the remaining chapters of this work, are introduced.

## 2.2. Graph Theory

A graph is a pair $\mathcal{G} = (V, E)$ with $V = \; v_1, \cdots, v_N$ a set of $N$ nodes or vertices and $E$ a set of edges or arcs. Let the set $V$ describe a set of $N$ agents, each agent being represented by a node $v_i \in V$ of $\mathcal{G}$. The set $E \subseteq V \times V$ expresses the flow of information from one agent to another. The notation $(v_i, v_j) \in E$ means that there is flow of information from node $v_i$ to node $v_j$. Graphically, this interaction is represented by an arrow with tail at $v_i$ and head at $v_j$ (see Fig. 2.1). The graph is assumed to have no self-loops, that is, $(v_i, v_i) \notin E$.

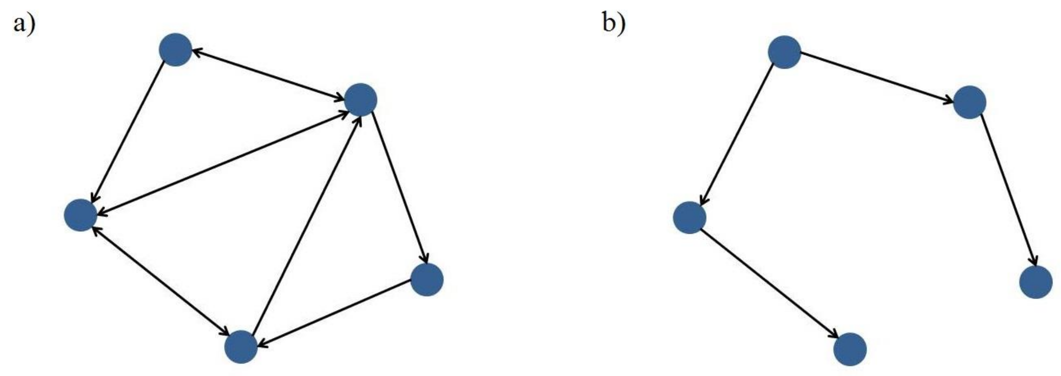

Fig. 2.1. Examples of graph topologies. a) A directed graph. b) A subset of the edges of the first graph forms a directed tree.

The edge weights of the graph are represented as $a_{ij}$, with $a_{ij} > 0$ if $(v_j, v_i) \in E$ and $a_{ij} = 0$ otherwise. Notice the order of the subindices; $a_{ij} > 0$ if agent $i$ receives information from agent $j$. Because the graph has no self-loops, we have $a_{ii} = 0$ for all $i = 1, \ldots, N$. The set of neighbors of node $v_i$ is $\mathcal{N}_i = \ v_j : a_{ij} > 0 \ $. The set of all nodes that are not neighbors of node $v_i$ is represented as $\mathcal{N}_{-i}$. The weighted in-degree of node $i$ is defined as the sum of the edge weights incoming to $v_i$, i.e., $d_i = \sum_{j=1}^{N} a_{ij}$. Define the graph adjacency matrix as $\mathcal{A} = \left[ a_{ij} \right]$, and the in-degree matrix of the graph as $D = \mathrm{diag}_i \ d_i \ $. The graph Laplacian matrix is $L = D - \mathcal{A}$. Matrix $L$ has all row sums equal to zero, and many properties of the graph $\mathcal{G}$ can be studied by analyzing its Laplacian matrix.

A graph is said to be undirected if $a_{ij} = a_{ji}$ for all nodes $i$, $j$. This means that all edges are bi-directional and, if agent $i$ can receive information from agent $j$, then agent $j$ can also receive information from $i$. If the graph $\mathcal{G}$ is not undirected, then it is said to be a directed graph, or a digraph. A directed path is a sequence of nodes $v_1, v_2, \ldots, v_r$ such that $(v_i, v_{i+1}) \in E$ for $i \in \{1, \ldots, r-1\}$. Node $v_i$ is said to be connected to node $v_j$ if there is a directed path from

$v_i$ to $v_j$. A graph is strongly connected if $v_i$ and $v_j$ are connected for all nodes $v_i, v_j \in V$. A graph is said to have a spanning tree if a subset of its edges forms a directed tree that connects all the nodes of the graph (Fig. 2.1-b).

Further information about graph theory, as well as a detailed analysis of multiagent systems can be found in [13].

## 2.3. Game Theory

Game theory is the mathematical analysis of optimal decision making. Its rigorous representation of the interaction of many agents makes the game-theoretical framework a fitting structure in our search for optimal behavior in dynamical multiagent systems. This section presents a brief review on the basic concepts of games in normal form. Many of the results in this dissertation can be seen as extensions of these concepts to differential graphical games.

Game theory was formally introduced by the work of von Newman and Morgenstern with the particular interest of solving economic decision problems [39]. Few years later, the analysis performed by John Nash yielded the most significant solution concept in game theory to date, now regarded as Nash equilibrium [2], [3], [40]-[42]. Besides economics, game theory has now found applications in many areas of research including logic, computer science, social analysis, evolutionary biology and control theory.

The most common formulation of a game consists on a set of $N$ players that wish to select their most appropriate actions in order to achieve a specific goal. The goals of a player are represented by means of a utility function which may be designed to represent selfish objectives, collective objectives, or a combination of both. The final payoff of a player usually depends on its own actions and the actions of all other players in the game. Thus, game theory studies the reasoning an agent must perform to select its optimal action against the other players to maximize its utility function. The formal definition of a game in normal form is now presented. Most of the definitions in this section are taken from [1], with some notational modifications to make them coherent with the rest of the dissertation.

**Definition 2.1 (Game in normal-form).** An $N$-person normal-form game is a tuple $(\mathcal{N}, U, J)$, where:

- $\mathcal{N}$ is a finite set of $N$ players;
- $U = U_1 \times \cdots \times U_N$, where $U_i$ is a finite set of actions available to player $i$;
- $J = J_1, \ldots, J_N$, where $J_i : U \to \mathbb{R}$ is a real-valued utility function for player $i$.

□

A *strategy* in a game is the policy selected by an agent to determine the action from the set $U_i$ to play at each instance of the game. For example, an agent can choose a specific action and play it without further operations; this is regarded as a *pure strategy*. In contrast, an agent can apply a randomization procedure to determine the action to play. This is called a *mixed strategy*. The object of study in game theory is the design of strategies that allow the agents achieve their goals.

Depending on the objectives of the players, two main distinctions of games can be described. A *zero-sum* game is defined such that $J_1 + \cdots + J_N = 0$ for any strategies played by the agents. The dynamics of a zero-sum game can be clearly noticed for the case of two-player games, on which we have $J_1 + J_2 = 0$, or $J_1 = -J_2$. This means that the utility received by an agent is necessarily lost by the other. These games describe strictly conflictive goals among the players. A game that allows for cooperative objectives between the agents is regarded as a *non-zero sum* game.

The *solution concepts* in a game are the set of strategies that provide useful behaviors for the players. Many solution concepts have been proposed in game theory that depend on the assumptions made by the agents, the information they have available, or the priorities they assign to each outcome. As stated before, the most important solution concept in game theory is Nash equilibrium, defined as follows.

Let $s_i$ represent any strategy available for agent $i$, and let $s_{-i}$ be the set of strategies for all players except $i$. The best response of agent $i$ to the strategies $s_{-i}$ is defined as the strategy $s_i^*$ such that $J_i(s_i^*, s_{-i}) \geq J_i(s_i, s_{-i})$ for all strategies $s_i$. This implies that the best response of agent $i$ is the strategy that provides it with a utility at least as good as any other strategy. Nash equilibrium is obtained in the game if all players play their best strategies simultaneously.

**Definition 2.2 (Nash equilibrium).** A strategy profile $s = (s_1, \ldots, s_N)$ is a Nash equilibrium if, for all agents $i$, $s_i$ is a best response to $s_{-i}$; that is,

$$J_i(s_i^*, s_{-i}^*) \geq J_i(s_i, s_{-i}^*) .$$

□

An important characteristic of Nash equilibrium is the stability it provides to the behavior of the agents, in the sense that no agent can individually modify its own strategy without decreasing its utility obtained in the game. Therefore, all agents are compelled to use their Nash strategies as long as the equilibrium is maintained.

A different solution concept, particularly suitable to study strictly conflictive games, is known as the maxmin strategy. This solution is obtained when agent $i$ decides to maximize its worst-case payoff. Such worst-case scenario occurs when all other players have as objective to cause the greatest harm to $i$. The maxmin strategy is defined below.

**Definition 2.3 (Maxmin strategy).** The maxmin strategy for player $i$ is obtained as

$$s_i^{\mathrm{maxmin}} = \arg\max_{s_i} \min_{s_{-i}} J_i(s_i, s_{-i}) .$$

□

The maxmin strategy provides a minimum guaranteed performance, such that the agent that employs it can expect its utility to be larger than or equal to the maxmin value. It can be easily shown that in two-player zero-sum games, the maxmin strategies are exactly the same as the Nash equilibrium strategies.

A final solution concept described in this review is known as the *Minimax regret* strategy. Define the regret of agent $i$ for playing action $u_i$ when the other players perform the actions $u_{-i}$ as

$$\mathcal{R}_i(u_i, u_{-i}) = \left[\max_{u'_i \in U_i} J_i(u'_i, u_{-i})\right] - J_i(u_i, u_{-i}). \tag{2.1}$$

That is, the regret is the amount of payoff lost by an agent for not using its best response against the strategies of the other agents. The minimax regret policy can now be defined.

**Definition 2.4 (Minimax regret).** The minimax regret policy of agent $i$ is defined as

$$u_i^{\text{regret}} = \arg\min_{u_i} \max_{u_{-i}} \mathcal{R}_i(u_i, u_{-i})$$

with $\mathcal{R}_i$ as in (2.1). □

This section is concluded with the definition of a game that allows the consideration of uncertainties in the environment of the players. Many practical applications of game-theoretic models require considering players with incomplete knowledge about the payoff they will receive after playing a particular action. The category of games that studies this scenario is regarded as Bayesian games, or games with incomplete information.

In a Bayesian game, the players are presented with a set of possible games, one of which is actually being played. Being aware of their lack of knowledge, the agents must define a probability distribution over the set of all possible games they may be engaged on. We call these probabilities the *beliefs* of an agent. At the beginning of the game, the agents possess some personal information, only known by themselves, and regarded as their *epistemic type*. The objective of an agent during the game depends on its current type and the types of the other agents. A Bayesian game is formally defined as follows.

**Definition 2.5 (Bayesian games).** An $N$-person Bayesian game is defined as a tuple $(\mathcal{N}, U, \Theta, P, J)$, where:

- $\mathcal{N}$ is a finite set of $N$ players;
- $U = U_1 \times \cdots \times U_N$, where $U_i$ is a finite set of actions available to player $i$;
- $\Theta = \Theta_1 \times \cdots \times \Theta_N$, with $\Theta_i = \{\theta_i^1, \ldots, \theta_i^{M_i}\}$ the type space of player $i$;
- $P : \Theta \rightarrow [0,1]$ expresses the probability of finding every agent $i$ in type $\theta_i^k$ for all possible types $k$;
- $J = J_1, \ldots, J_N$, where $J_i : U \rightarrow \mathbb{R}$ is a real-valued utility function for player $i$.

□

In the following section, the extensions of these game theoretic concepts for dynamical agents are described.

## 2.4. Differential Graphical Games

The canonical leader-follower synchronization game is described in this section. Let each agent in the graph $\mathcal{G}$ defined in Section 2.2 be a dynamical system with linear dynamics as

$$\dot{x}_i = Ax_i + Bu_i \tag{2.2}$$

for all agents $i = 1, \ldots, N$, where $x_i(t) \in \mathbb{R}^n$ is the vector of state variables and $u_i \in \mathbb{R}^m$ is the control input vector of agent $i$. The pair $(A, B)$ is assumed to be stabilizable throughout this dissertation. Consider also an additional node, regarded as the leader or target node, with state dynamics

$$\dot{x}_0 = Ax_0. \tag{2.3}$$

The leader is connected to the other nodes by means of the pinning gains $g_i \geq 0$. This dissertation studies the behavior of the agents with the general objective of achieving synchronization with the leader node $x_0$.

Each agent is assumed to observe the full state vector of its neighbors in the graph. The local synchronization error for agent $i$ is defined as

$$\delta_i = \sum_{j=1}^{N} a_{ij}(x_i - x_j) + g_i(x_i - x_0) \tag{2.4}$$

and the local error dynamics are

$$\begin{aligned} \dot{\delta}_i &= \sum_{j=1}^{N} a_{ij}(\dot{x}_i - \dot{x}_j) + g_i(\dot{x}_i - \dot{x}_0) \\ &= A\delta_i + (d_i + g_i)Bu_i - \sum_{j=1}^{N} a_{ij}Bu_j \end{aligned} \tag{2.5}$$

where the dynamics (2.2) - (2.3) have been incorporated.

Each agent $i$ expresses its objective in the game by defining a performance index as

$$J_i(\delta_i, \delta_{-i}, u_i, u_{-i}) = \int_0^\infty r_i(\delta_i, \delta_{-i}, u_i, u_{-i})dt \tag{2.6}$$

where $r_i(\delta_i, \delta_{-i}, u_i, u_{-i})$ is selected as a positive definite scalar function of the variables expected to be minimized by agent $i$, with $\delta_{-i}$ and $u_{-i}$ the local errors and control inputs of the neighbors of agent $i$, respectively. For synchronization games, $r_i$ can be selected as

$$r_i(\delta_i, \delta_{-i}, u_i, u_{-i}) = \sum_{j=0}^{N} a_{ij}\left(\bar{\delta}_{ij}^T Q_{ij} \bar{\delta}_{ij} + u_i^T R_{ii} u_i + u_j^T R_{ij} u_j\right) \tag{2.7}$$

with $Q_{ij} = Q_{ij}^T \geq 0$, $R_{ii} = R_{ii}^T > 0$, $a_{i0} = g_i$, $\bar{\delta}_{i0} = \begin{bmatrix} \delta_i^T & 0^T \end{bmatrix}^T$, $\bar{\delta}_{ij} = \begin{bmatrix} \delta_i^T & \delta_j^T \end{bmatrix}^T$ for $j \neq 0$, and $u_0 = 0$. Function $r_i$ can also presented in a simplified form,

$$r_i(\delta_i, u_i, u_{-i}) = \delta_i^T Q_i \delta_i + u_i^T R_{ii} u_i + \sum_{j=1}^{N} a_{ij} u_j^T R_{ij} u_j \tag{2.8}$$

which is widely employed in the differential graphical games literature [20], [33].

The dependence of $J_i$ on $\delta_{-i}$ and $u_{-i}$ does not imply that the optimal control policy, $u_i^*$, requires these variables to be computed by agent $i$ [20]. The definition of $J_i$, therefore, can yield a valid distributed control policy as solution of the game.

The best response of agent $i$ for fixed neighbor policies $u_{-i}$ is defined as the control policy $u_i^*$ such that the inequality $J_i(\delta, u_i^*, u_{-i}) \le J_i(\delta, u_i, u_{-i})$ holds for all policies $u_i$. Nash equilibrium is achieved if every agent plays its best response with respect to all its neighbors, that is,

$$J_i(\delta, u_i^*, u_{-i}^*) \le J_i(\delta, u_i, u_{-i}^*) \tag{2.9}$$

for all agents $i = 1, \ldots, N$.

A differential version of the performance index (2.6) is regarded as the Bellman equation, obtained by means of the Hamiltonian function $H_i$ as

$$0 = r_i(\delta, u_i, u_{-i}) + \dot{V}_i(\delta) \triangleq H_i \tag{2.10}$$

where $V_i(\delta)$ is a scalar function regarded as the value function of the game. The optimal control policy for agent $i$ can now be obtained by means of the stationary condition $\frac{\partial H_i}{\partial u_i} = 0$. The following assumption provides a condition to obtain distributed control policies for the agents.

**Assumption 2.1.** Let the value functions $V_i(\delta)$ in (2.10) be distributed, in the sense that they contain only local information, i.e., $V_i(\delta) = V_i(\delta_i)$.

It is proven in [20] that, if Assumption 2.1 holds, the best response of agent $i$ with cost function (2.6) and function $r_i$ as in (2.8) is given by

$$u_i^* = -\frac{1}{2}(d_i + g_i) R_{ii}^{-1} B^T \nabla V_i(\delta_i). \tag{2.11}$$

Substituting the control policies (2.11) for all agents $i = 1, \ldots, N$ in the Bellman equations (2.10), yields the set of coupled partial differential equations

$$r_i(\delta_i, u_i, u_{-i}) + \nabla V_i^T \left( A\delta_i + (d_i + g_i) B u_i^* - \sum_{j=1}^{N} a_{ij} B u_j^* \right) = 0 \tag{2.12}$$

known as the Hamilton-Jacobi (HJ) equations. The procedure to obtain the optimal control inputs $u_i^*$ for the agents now consists in solving for the value functions $V_i(\delta_i)$ from the coupled HJ equations (2.12), and substituting these functions in the policies (2.11). When all agents simultaneously use their best policies (2.11), Nash equilibrium is achieved in the game.

## 2.5. Further Notations

This chapter is concluded with few additional definitions that will simplify the notation in subsequent chapters [43], [44].

The space $\mathcal{L}_2^n$ is defined as the set of all piecewise continuous functions $x : [0, \infty) \to \mathbb{R}^n$ such that

$$\|x\|_{\mathcal{L}_2} = \left( \int_0^\infty x^T(t) x(t) \right)^{1/2} dt < \infty \tag{2.13}$$

that is, the space $\mathcal{L}_2^n$ defines the set of all square-integrable functions $x(t)$.

The extended space $\mathcal{L}_{2e}^n$ is defined by

$$\mathcal{L}_{2e}^n = \; x \mid x_\tau \in \mathcal{L}_2^n, \; \forall \tau \geq 0$$

where $x_\tau$ is a truncation of $x$ defined by

$$x_\tau(t) = \begin{cases} x(t), & 0 \leq t \leq \tau \\ 0, & t > \tau \end{cases} \tag{2.14}$$

Define the inner-product in space $\mathcal{L}_2^n[0, \infty)$ as

$$\langle x, y \rangle = \int_0^\infty x^T(t) y(t) dt \tag{2.15}$$

where $x, \, y \in \mathcal{L}_2^n[0, \infty)$.

A mapping $H : \mathcal{L}_{2e}^n \to \mathcal{L}_{2e}^n$ is finite-gain $\mathcal{L}_2$ stable if there exist nonnegative constants $\gamma$ and $\beta$ such that

$$\left\|(Hu)_\tau\right\|_{\mathcal{L}_2} \le \gamma \left\|u_\tau\right\|_{\mathcal{L}_2} + \beta \tag{2.16}$$

for all $u \in \mathcal{L}_{2e}^m$ and $\tau \in [0, \infty)$.

When inequality (2.16) is satisfied for some $\gamma \ge 0$, the mapping is said to have $\mathcal{L}_2$ gain less than or equal to $\gamma$.

# Chapter 3

# EPISTEMIC BELIEFS AND BAYESIAN GRAPHICAL GAMES

## 3.1. Introduction

The general approach to differential games in the current literature is to expand the single-agent optimal control techniques to groups of agents with both common and conflicting interests. It is proven in [5] that, if the solutions of the set of coupled partial differential equations known as the Hamilton-Jacobi (HJ) equations exist, then Nash equilibrium is achieved in the game and no agent can unilaterally change its control policy without producing a decreased performance for itself. A more general case has been described with the study of graphical games [15], [20], [33], [34], [46], on which, as described in Chapter 2, the agents are taken as nodes in a communication graph, such that each agent can only measure the state of the agents connected to it through the graph links, regarded as its neighbors.

A downside of these standard differential games solutions is the assumption that all agents are fully aware of all the aspects of the game being played. In complex practical applications, the agents operate in fast-evolving and uncertain environments which provide them with incomplete information about the game. A dynamical agent facing other agents for the first time, for example, may not be certain of their real intentions or objectives.

Bayesian games [1], [24]-[27], also called games with incomplete information, describe the situation on which the agents participate in an unspecified game. The true intentions of the other players may be unknown, and each agent must adjust its objectives accordingly. The initial information of each agent about the game, and the personal experience gained during its interaction with other agents, form the basis for the epistemic analysis of the dynamical system. The agents must collect the evidence provided by their environment and use it to update their beliefs about the state of the game. Thus, the aim is to develop belief assurance protocols, distributed control protocols and distributed learning mechanisms to induce optimal behaviors with respect to an *expected cost* function.

In [1], [24] and [25], Bayesian games are defined for static agents and it is shown that the solution of the game consist on the selection of specific actions with a given probability. In this work, Bayesian games are defined for dynamic systems and the optimal control policies vary as the belief of the agents change. The ex post stability in Bayesian games is studied in [26] and [27], consisting in a solution that would not change if the agents were fully aware of the conditions of the game. Our results are shown not to be ex post stable because we allow the agents to improve their policies as they collect new information. Different learning algorithms for static agents in Bayesian games have been studied [42], [47]-[49], but not for differential graphical games.

Potential applications for the proposed Bayesian games for dynamical systems include collision avoidance in automatic transport systems, sensible decision making against possibly hostile agents and optimal distribution of tasks in cooperative environments. As the number of autonomous agents increase in urban areas, the formulation of optimal strategies for unknown scenarios becomes a necessary development.

The rest of the chapter is structured as follows. Section 3.2 presents the formal mathematical definitions of Bayesian games and differential graphical games, which form the basis for the rest of the chapter. Section 3.3 presents the formulation of Bayesian games for dynamical systems in a graph topology; the best response strategies for the minimization of the expected costs and the minmax strategies of every agent are obtained. Section 3.4 is focused on the Bayesian methodology for the belief updates, while Section 3.5 presents the application of a non-Bayesian methodology. Finally, a simulation of the proposed control scheme is presented in Section 3.6, and Section 3.7 includes a brief conclusion.

## 3.2. Bayesian Graphical Games for Dynamical Systems

In Section 2.3, Bayesian games where defined for normal-form, static games. This section presents our main results on the formulation of Bayesian games for multiagent dynamical systems connected by a communication graph and the analysis of the conditions to achieve Bayes-Nash equilibrium in the game.

### *3.2.1. Game formulation*

Consider a system of $N$ agents with linear dynamics (2.2) distributed on a communication graph $\mathcal{G}$, and a leader with state dynamics (2.3). The local synchronization errors are defined as in (2.4) - (2.5).

The desired objectives of an agent can vary depending on its current type and the types of its neighbors. This condition can be expressed by defining the performance index of agent $i$ as

$$J_i^{\theta}(\delta_i,u_i,u_{-i})=\int_0^{\infty} r_i^{\theta}(\delta_i,u_i,u_{-i})dt \tag{3.1}$$

where $\theta$ refers to the set of current types of all the agents in the game, $\theta=\theta_1\times\cdots\times\theta_N$, as defined in Section 2.3, and each function $r_i^{\theta}$ is defined for that particular combination of types. With this information, we define a new category of game as follows.

**Definition 3.1 (Bayesian graphical game).** A Bayesian graphical game for dynamical systems is defined as a tuple $(\mathcal{N},X,U,\Theta,P,J)$ where:

- $\mathcal{N}$ is the set of agents in the game;
- $X=X_1\times\cdots\times X_N$, with $X_i$ the set of reachable states of agent $i$;
- $U=U_1\times\cdots\times U_N$, with $U_i$ the set of admissible controllers for agent $i$,
- $\Theta=\Theta_1\times\cdots\times\Theta_N$, with $\Theta_i$ the type space of player $i$;
- $P:\Theta\to[0,1]$ is the common prior over types that describes the probability of finding every agent $i$ in type $\theta_i^k\in\Theta_i$, $k=1,\ldots,M_i$, at the beginning of the game.
- $J=(J_1,\ldots,J_N)$ are the performance indices, where $J_i:X\times U\times\Theta\to\mathbb{R}$, is the cost paid by agent $i$ for using a given control policy while the game is in a state

value and a combination of types.

□

Define the set $\Delta_i = X_1^i \times \cdots \times X_N^i$, where $X_j^i$ is the set of possible states of the $j$th neighbor of agent $i$; that is, $\Delta_i$ represents the set of states that agent $i$ can observe from the graph topology.

It is assumed that the sets $\mathcal{N}$, $X$, $U$, $P$ and $J$ are of common prior for all the agents before the game starts. However, the set of states $\Delta_i$ and the actual type $\theta_i$ are known only by agent $i$. The objective of every agent in the game is now to use their (limited) knowledge about $\delta_i$ and $\theta$ to determine the control policies $u_i^*(\delta_i,\theta)$, such that every agent expects to minimize the cost he pays during the game according to the cost functions (3.1).

To fulfill this objective, a different cost index formulation is required to allow the agents to determine their optimal policies according to their current beliefs about the global type $\theta$. This requirement is addressed by defining the *expected cost* of agent $i$*,* as studied in the following subsection.

## *3.2.2. Expected cost*

In the Bayesian games' literature, three different concepts of expected cost are usually defined, namely the *ex post*, the *ex interim*, and the *ex ante* expected costs, that differ in the information available for their computation [1], [26], [27].

The *ex post* expected cost of agent $i$ considers the actual types of all agents of the game. For a given Bayesian game $(N,X,U,\Theta,P,J)$, where the agents play with policies $u_i$ and the global type is $\theta$, the *ex post* expected utility is defined as

$$EJ_i(\delta_i,u_i,u_{-i},\theta) = J_i^\theta(\delta_i,u_i,u_{-i}) \tag{3.2}$$

The *ex interim* expected cost of agent $i$ is computed when $i$ knows its own type, but the types of all other agents are unknown to it. Note that this case applies if the agents calculate their expected costs once the game has started. Given a Bayesian game $(N, X, U, \Theta, P, J)$, where the agents play with policies $u_i$ and the type of agent $i$ is $\theta_i$, the *ex interim* expected cost is

$$EJ_i(\delta_i, u_i, u_{-i}, \theta_i) = \sum_{\theta\in\Theta} p(\theta \mid \theta_i) J_i^{\theta}(\delta_i, u_i, u_{-i}) \tag{3.3}$$

where $p(\theta \mid \theta_i)$ is the probability of having global type $\theta$ given that agent $i$ has type $\theta_i$, and the summation index $\theta \in \Theta$ indicates that all possible combination of types in the game must be considered.

We can finally define the *ex ante* expected cost for the case when agent $i$ is ignorant of the type of every agent, including itself. This can be seen as the expected cost that is computed before the game starts, such that the agents do not know their own types. For a given Bayesian game $(N, X, U, \Theta, P, J)$ and given the control policies $u_i$ for all the agents, the *ex ante* expected cost for agent $i$ is defined as

$$EJ_i(\delta_i, u_i, u_{-i}) = \sum_{\theta\in\Theta} p(\theta) J_i^{\theta}(\delta_i, u_i, u_{-i}). \tag{3.4}$$

Throughout this dissertation we use the *ex interim* expected cost as the objective for minimization of every agent, such that they can compute it during the game.

In the following subsection, the optimal control policy $u_i^*$ for every agent is obtained, and conditions for Bayes-Nash equilibrium are provided.

### *3.2.2. Best response policy and Bayes-Nash equilibrium*

The best response of an agent in a Bayesian game for given fixed neighbor strategies $u_{-i}$, is defined as the control policy that makes the agent pay the minimum expected cost. Formally, agent $i$'s best response to control policies $u_{-i}$ are given by

$$u_i^* = \arg\min_{u_i} EJ_i(\delta_i, u_i, u_{-i}, \theta) \tag{3.5}$$

Now, it is said that a Bayes-Nash equilibrium is reached in the game if each agent plays a best response to the strategies of the other players during a Bayesian game. The Bayes-Nash equilibrium is the most important solution concept in Bayesian graphical games for dynamical systems. Definition 3.2 formalizes this idea.

**Definition 3.2 (Bayes-Nash equilibrium).** A Bayes-Nash equilibrium is a set of control policies $u = u_1 \times \cdots \times u_N$ that satisfies $u_i = u_i^*$, as in (3.5), for all agents $i$, such that

$$EJ_i(\delta_i, u_i^*, u_{-i}^*) \le EJ_i(\delta_i, u_i, u_{-i}^*) \tag{3.6}$$

for any control policy $u_i$. □

Following an analogous procedure to single-agent optimal control, define the value function of agent $i$, given the types of all agents $\theta$, as

$$V_i^\theta(\delta_i, u_i, u_{-i}) = \int_t^\infty r_i^\theta(\delta_i, u_i, u_{-i})\, d\tau \tag{3.7}$$

with $r_i^\theta$ as defined in (3.1). The *expected value function* for a control policy $u_i$ is defined as

$$EV_i(\delta_i, u_i, u_{-i}, \theta) = \sum_{\theta \in \Theta} p(\theta \mid \theta_i) V_i^\theta(\delta_i, u_i, u_{-i}) \tag{3.8}$$

where agent $i$ knows its own epistemic type.

Function (3.8) can be used to define the *expected Hamiltonian* of agent $i$ as

$$EH_i(\delta_i, u, \theta) = \sum_{\theta \in \Theta} p(\theta \mid \theta_i) \Big[ r_i^\theta(\delta_i, u) + \nabla V_i^{\theta T} \Big( A\delta_i + (d_i + g_i) B u_i - \textstyle\sum_{j=1}^N a_{ij} B u_j \Big) \Big]. \tag{3.9}$$

The expected Hamiltonian (3.9) is now employed to determine the best response control policy of agent $i$, by computing its derivative with respect to $u_i$ and equating it to zero. This procedure yields the optimal policy

$$u_i^* = -\frac{1}{2}(d_i + g_i) \left[ \sum_{\theta \in \Theta} p(\theta \mid \theta_i) R_{ii}^\theta \right]^{-1} \sum_{\theta \in \Theta} p(\theta \mid \theta_i) B^T \nabla V_i^\theta. \tag{3.10}$$

As in the deterministic multiplayer nonzero-sum games [5], the functions $V_i^\theta(\delta_i)$ are the solutions of a set of coupled partial differential equations. For the setting of Bayesian games, we introduce the novel concept of the Bayes-Hamilton-Jacobi (BHJ) equations, given by

$$\sum_{\theta\in\Theta} p(\theta\,|\,\theta_i)\left[ r_i^{\theta}(\delta_i,u^*) + \nabla V_i^{\theta T}\left( A\delta_i + (d_i+g_i)Bu_i^* - \textstyle\sum_{j=1}^{N} a_{ij}Bu_j^*\right)\right] = 0. \tag{3.11}$$

**Remark 3.1.** The optimal control policy (3.10) establishes for the first time, the relation between belief and distributed control in multi-agent systems with unawareness. Each agent should compute its best response by observing only its immediate neighbors. This is distributed computation with bounded rationality imposed by the communication network.

**Remark 3.2.** Notice that the probability terms in (3.10) have the properties $0 \le p(\theta\,|\,\theta_i) \le 1$ and $\sum_{\theta\in\Theta} p(\theta\,|\,\theta_i) = 1$. Therefore, Equation (3.9) is a convex combination of the Hamiltonian functions defined for each performance index (3.1) for agent $i$, and (3.10) is the solution of a multiobjective optimization problem using the weighted sum method (see Appendix) [50], [51].

**Remark 3.3.** The solution obtained by means of the minimization of the expected cost, does not represent an increase in complexity when compared to the optimization of a single performance index. Only the number of sets of coupled HJ equations increases according to the total number of combination of types of the agents.

**Remark 3.4.** If there is a time $t_f$ at which agent $i$ is convinced of the global type $\theta$ with probability 1, then the problem is reduced to a single objective optimization problem and the solution is given by the deterministic control policy

$$u_i^* = -\frac{1}{2}\left(R_{ii}^{\theta}\right)^{-1} B^T \nabla V_i^{\theta}(\delta_i).$$

In the particular case when the value function associated with each $J_i^{\theta}$ has the quadratic form

$$V_i^{\theta} = \delta_i^T P_i^{\theta} \delta_i \tag{3.12}$$

the optimal policy (3.10) can be written in terms of the states of agent $i$ and its neighbors as

$$u_i^* = -(d_i+g_i)\left[\sum_{\theta\in\Theta} p(\theta\,|\,\theta_i) R_{ii}^{\theta}\right]^{-1} \sum_{\theta\in\Theta} p(\theta\,|\,\theta_i) B^T P_i^{\theta}\delta_i. \tag{3.13}$$

The next technical lemma shows that the Hamiltonian function for general policies $u_i$,

$u_{-i}$ can be expressed as a quadratic form of the optimal policies $u_i^*$ and $u_{-i}^*$ defined in (3.10).

**Lemma 3.1**. Given the expected Hamiltonian function (3.9) for agent $i$ and the optimal control policy (3.10), then

$$EH_i(\delta_i,u_i,u_{-i}) = EH_i(\delta_i,u_i^*,u_{-i}) + \sum_{\theta\in\Theta} p(\theta\,|\,\theta_i)(u_i-u_i^*)^T R_{ii}^{\theta}(u_i-u_i^*). \tag{3.14}$$

*Proof*. The proof is similar to the proof of Lemma 10.1-1 in [5], performed by completing the squares in (3.9) to obtain

$$\begin{aligned} EH_i(\delta_i,u,\theta) = \sum_{\theta\in\Theta} p(\theta\,|\,\theta_i)\Bigg[ & \delta_i^T Q_i^{\theta}\delta_i + u_i^T R_{ii}^{\theta}u_i + \sum_{j=1}^{N} a_{ij}u_j^T R_{ij}^{\theta}u_j + u_i^{*T}R_{ii}^{\theta}u_i^* - u_i^{*T}R_{ii}^{\theta}u_i^* \\ & + (d_i+g_i)\nabla V_i^{\theta T}Bu_i^* - (d_i+g_i)\nabla V_i^{\theta T}Bu_i^* + \nabla V_i^{\theta T}\left(A\delta_i + (d_i+g_i)Bu_i - \sum_{j=1}^{N} a_{ij}Bu_j\right)\Bigg] \end{aligned} \tag{3.15}$$

and conducting algebraic operations to obtain (3.14). □

The following theorem extends the concept of Bayes-Nash equilibrium to differential Bayesian games, and shows that this Bayes-Nash equilibrium is achieved by means of the control policies (3.10). The proof is performed using the quadratic cost functions as in (2.8), but it can easily be extended to other functions as (2.7).

**Theorem 3.1**. Consider a multiagent system on a communication graph, with agents' dynamics (2.2) and target node dynamics (2.3). Let $V_i^{\theta*}(\delta_i)$, $i=1,\ldots,N$, be the solutions of the BHJ equations (3.11). Define the control policy $u_i^*$ as in (3.10) and let Assumption 2.1 hold. Then, control inputs $u_i^*$ make the dynamics (2.5) asymptotically stable for all agents. Moreover, all agents are in Bayes-Nash equilibrium as defined in Definition 3.2, and the corresponding expected costs of the game are

$$EJ_i^* = V_i^{\theta*}(\delta_i(0)).$$

*Proof*. (Stability) Take the expected value function (3.8) as a Lyapunov function candidate. Its derivative is given by

$$E\dot{V}_i = \sum_{\theta\in\Theta} p(\theta\,|\,\theta_i)\dot{V}_i^{\theta} = \sum_{\theta\in\Theta} p(\theta\,|\,\theta_i)\nabla V_i^{\theta T}\dot{\delta}_i.$$

The BHJ equation (3.11) is a differential version of the value functions (3.8) using the optimal control policies (3.10) [5]. Then, as $V_i^\theta$ satisfies (3.11) we get

$$E\dot{V}_i = -\sum_{\theta\in\Theta} p(\theta \mid \theta_i)\left( \delta_i^T Q_i^\theta \delta_i + u_i^T R_{ii}^\theta u_i + \sum_{j=1}^{N} a_{ij} u_j^T R_{ij}^\theta u_j \right) < 0$$

and the dynamics (2.5) are asymptotically stable.

(Bayes-Nash equilibrium) Note that $V_i^\theta(\delta_i(\infty)) = V_i^\theta(0) = 0$ because of the asymptotic stability of the system. Now, the expected cost of the game for agent $i$ is expressed as

$$\begin{aligned} EJ_i &= \sum_{\theta\in\Theta} p(\theta \mid \theta_i)\int_0^\infty \left( \delta_i^T Q_i^\theta \delta_i + u_i^T R_{ii}^\theta u_i + \sum_{j=1}^{N} a_{ij} u_j^T R_{ij}^\theta u_j \right) dt + \sum_{\theta\in\Theta} p(\theta \mid \theta_i)\int_0^\infty \dot{V}_i^\theta dt + \sum_{\theta\in\Theta} p(\theta \mid \theta_i) V_i^\theta(\delta_i(0)) \\ &= \int_0^\infty EH_i(\delta_i, u_i, u_{-i}) dt + \sum_{\theta\in\Theta} p(\theta \mid \theta_i) V_i^\theta(\delta_i(0)). \end{aligned}$$

By Lemma 1, this expression becomes

$$EJ_i = \sum_{\theta\in\Theta} p(\theta \mid \theta_i) V_i^\theta(\delta_i(0)) + \int_0^\infty EH_i(\delta_i, u_i^*, u_{-i}) dt + \sum_{\theta\in\Theta} p(\theta \mid \theta_i)\int_0^\infty (u_i - u_i^*)^T R_{ii}(u_i - u_i^*) dt$$

for all $u_i$ and $u_{-i}$. Assume all the neighbors of agent $i$ are using their best response strategies $u_{-i}^*$. Then, as the BHJ equations (3.11) hold, we have

$$EJ_i = \sum_{\theta\in\Theta} p(\theta \mid \theta_i)\left[ \int_0^\infty (u_i - u_i^*)^T R_{ii}(u_i - u_i^*) dt + V_i^\theta(\delta_i(0)) \right]$$

We conclude that $u_i^*$ minimizes the expected cost of agent $i$ and the value of the game is $EV_i^\theta(\delta_i(0))$. □

It is of our interest to determine the influence of the graph topology in the stability of the synchronization errors given by the control policies (3.13). To make this analysis, define the matrix $K = \mathrm{diag}\{K_i\} \in \mathbb{R}^{Nn}$ with $K_i = (d_i + g_i) R_i^{-1} B^T P_i$.

Theorem 3.2 relates the stability properties of the game with the communication graph topology $\mathcal{G}$.

***Theorem 3.2***. Let the conditions of Theorem 3.1 hold. Then, the eigenvalues of matrix $\left[(I\otimes A)-((L+G)\otimes B)K\right]\in\mathbb{R}^{n(N+M)}$ have all negative real parts, i.e., for $k=1,\ldots,nN$,

$$\operatorname{Re}\{\lambda_k\left((I\otimes A)-((L+G)\otimes B)K\right)\}<0. \tag{3.16}$$

*Proof*. Define the vectors $\delta=\left[\delta_1^T,\cdots,\delta_N^T\right]^T$ and $u=\left[u_1^T,\cdots,u_N^T\right]^T$. Using the local error dynamics (2.5), we can write

$$\dot{\delta}=(I\otimes A)\delta+((L+G)\otimes B)u \tag{3.17}$$

Control policies (3.13) can be expressed as $u_i=-K_{pi}\delta_i$, with $K_i=(d_i+g_i)R_i^{-1}B^TP_i$. Now we get,

$$u=-K\delta. \tag{3.18}$$

Substitution of (3.18) in (3.17) yields the global closed-loop dynamics

$$\dot{\delta}=\left[(I\otimes A)-((L+G)\otimes B)K\right]\delta \tag{3.19}$$

Theorem 3.1 shows that if matrices $P_i$ satisfy (3.11) then the control policies (3.13) make the gents achieve synchronization with the leader node. This implies that the system (3.19) is stable, and the condition (3.16) holds. □

## 3.3. Bayesian Belief Updates

Two approaches to perform the belief update of the agents are developed in this dissertation. In this section we study the use of the Bayesian rule to compute a new estimate given the evidence provided by the states of the neighbors. Section 3.4 analyzes a non-Bayesian approach to perform the belief updates.

### *3.3.1. Epistemic type estimation*

Let every agent in the game to revise its beliefs every $T$ units of time. Then, using its knowledge about its type $\theta_i$, the previous states of its neighbors $x_{-i}(t)$, and the current state of the neighbors $x_{-i}(t+T)$, agent $i$ can perform its belief update at time $t+T$ using the Bayesian rule as

$$p(\theta \mid x_{-i}(t+T), x_{-i}(t), \theta_i) = \frac{p(x_{-i}(t+T) \mid x_{-i}(t), \theta)\, p(\theta \mid x_{-i}(t), \theta_i)}{p(x_{-i}(t+T) \mid x_{-i}(t), \theta_i)} \tag{3.20}$$

where $p(\theta \mid x_{-i}(t+T), x_{-i}(t), \theta_i)$ is agent $i$'s belief at time $t+T$ about the types $\theta$, $p(\theta \mid x_{-i}(t), \theta_i)$ is agent $i$'s beliefs at time $t$ about $\theta$, $p(x_{-i}(t+T) \mid x_{-i}(t), \theta)$ is the likelihood of the neighbors reaching the states $x_{-i}(t+T)$ $T$ time units after being in states $x_{-i}(t)$ given that the global type is $\theta$, and $p(x_{-i}(t+T) \mid x_{-i}(t), \theta_i)$ is the overall probability of the neighbors reaching $x_{-i}(t+T)$ from $x_{-i}(t)$ regardless of every other agent's types.

**Remark 3.5.** Although the agents know only the state of their neighbors, they need to estimate the type of all agents in the game, for this combination of types determines the objectives of the game being played.

**Remark 3.6.** In the previous sections we defined the Bayesian games using the probabilities $p(\theta \mid \theta_i)$. In this section, we make explicit the fact that agent $i$ uses the behavior of its neighbors as evidence of the global type $\theta$ by expressing the probabilities $p(\theta \mid x_{-i}(t), \theta_i)$.

It is of our interest to find an expression for the belief update (3.20) that explicitly displays distributed update terms for the neighbors and non-neighbors of agent $i$. In the following, such expressions are obtained for the three terms $p(\theta \mid x_{-i}(t), \theta_i)$, $p(x_{-i}(t+T) \mid x_{-i}(t), \theta)$ and $p(x_{-i}(t+T) \mid x_{-i}(t), \theta_i)$.

The likelihood function $p(x_{-i}(t+T) \mid x_{-i}(t), \theta)$ in the Bayesian belief update (3.20) can be expressed in terms of the individual positions of each neighbor of agent $i$ as the joint probability

$$p(x_{-i}(t+T) \mid x_{-i}(t), \theta) = p(x_1^i(t+T), \ldots, x_{N_i}^i(t+T) \mid x_{-i}(t), \theta) \tag{3.21}$$

where $x_j^i(t)$ is the state of the $j$th neighbor of $i$. Notice that $x_i(t+T)$ is dependent of $x_i(t)$ and of $x_{-i}(t)$ by means of the control input $u_i$, for all agents $i$. However, the current state value of agent $i$, $x_i(t+T)$, is independent of the current state value of its neighbors, $x_{-i}(t+T)$, for there has been no time for the values $x_{-i}(t+T)$ to affect the policy $u_i$. Independence of the state variables at time $t+T$ allows computing the joint probability (3.21) as the product of factors

$$p(x_{-i}(t+T)\mid x_{-i}(t),\theta)=\prod_{j\in N_i} p(x_j(t+T)\mid x_{-i}(t),\theta). \tag{3.22}$$

Using the same procedure, we can express the denominator of (3.20), $p(x_{-i}(t+T)\mid x_{-i}(t),\theta_i)$, as the product

$$p(x_{-i}(t+T)\mid x_{-i}(t),\theta_i)=\prod_{j\in N_i} p(x_j(t+T)\mid x_{-i}(t),\theta_i). \tag{3.23}$$

Notice that the value of $p(x_j(t+T)\mid x_{-i}(t),\theta_i)$ can be computed from the likelihood function $p(x_j(t+T)\mid x_{-i}(t),\theta)$ as

$$p(x_j(t+T)\mid x_{-i}(t),\theta_i)=\sum_{\theta\in\Theta} p(\theta\mid x_{-i}(t),\theta_i)\,p(x_j(t+T)\mid x_{-i}(t),\theta). \tag{3.24}$$

Finally, the term $p(\theta\mid x_{-i}(t),\theta_i)$ in (3.20) expresses the joint probability of the types of each individual agent, that is, $p(\theta\mid x_{-i}(t),\theta_i)=p(\theta_1,\ldots,\theta_N\mid x_{-i}(t),\theta_i)$. Two cases must be considered to compute the value of this probability. In the general case, the types of the agents are dependent on each other; in particular applications, the types of all agents may be independent and, therefore, the knowledge of an agent about one type does not affect its belief in the others.

*Dependent epistemic types.* If the type of an agent depends on the types of other agents, the term $p(\theta\mid x_{-i}(t),\theta_i)$ can be computed in terms of conditional probabilities using the chain rule

$$\begin{aligned} p(\theta\mid x_{-i}(t),\theta_i) &= p(\theta_1,\theta_2,\ldots,\theta_N\mid x_{-i}(t),\theta_i)\\ &= p(\theta_1\mid x_{-i}(t),\theta_i)\,p(\theta_2\mid x_{-i}(t),\theta_i,\theta_1)\cdots p(\theta_N\mid x_{-i}(t),\theta_i,\theta_1,\ldots,\theta_{N-1})\\ &= \prod_{j=1}^{N} p(\theta_j\mid x_{-i}(t),\theta_i,\theta_1,\ldots,\theta_{j-1}) \end{aligned} \tag{3.25}$$

We can further separate the products of (3.25) in terms of the neighbors and non-neighbors of agent $i$ as

$$\prod_{j=1}^{N} p(\theta_j \mid x_{-i}(t),\theta_i,\theta_1,\dots,\theta_{j-1}) = \prod_{j\in N_i} p(\theta_j \mid x_{-i}(t),\theta_i,\theta_1,\dots,\theta_{j-1}) \prod_{k\notin N_i} p(\theta_k \mid x_{-i}(t),\theta_i,\theta_1,\dots,\theta_{k-1}) \tag{3.26}$$

Using expressions (3.22), (3.23) and (3.26), the Bayesian update (3.20) can be written as

$$p(\theta \mid x_{-i}(t+T), x_{-i}(t),\theta_i) = \prod_{j\in N_i} \frac{p(x_j(t+T) \mid x_{-i}(t),\theta)\, p(\theta_j \mid x_{-i}(t),\theta_i,\theta_1,\dots,\theta_{j-1})}{p(x_j(t+T) \mid x_{-i}(t),\theta_i)} \times \prod_{k\notin N_i} p(\theta_k \mid x_{-i}(t),\theta_i,\theta_1,\dots,\theta_{k-1}) \tag{3.27}$$

where the belief update with respect to the position of each neighbor is explicitly expressed, as desired.

*Independent epistemic types.* In this case, agent $i$ updates its beliefs about the other agents' types based only on its local information about the states of its neighbors. Thus, we obtain the expression

$$\begin{aligned} p(\theta \mid x_{-i}(t),\theta_i) &= p(\theta_1,\theta_2,\dots,\theta_N \mid x_{-i}(t)) \\ &= p(\theta_1 \mid x_{-i}(t))\, p(\theta_2 \mid x_{-i}(t)) \cdots p(\theta_N \mid x_{-i}(t)) \end{aligned} \tag{3.28}$$

Again, using expressions (3.22), (3.23) and (3.28), the belief update of agent $i$ can be written as the product of the inference of each of its neighbors and its beliefs about its non-neighbors' types, as

$$p(\theta \mid x_{-i}(t+T), x_{-i}(t),\theta_i) = \prod_{j\in N_i} \frac{p(x_j(t+T) \mid x_{-i}(t),\theta)\, p(\theta_j \mid x_{-i}(t))}{p(x_j(t+T) \mid x_{-i}(t),\theta_i)} \times \prod_{k\notin N_i} p(\theta_k \mid x_{-i}(t)). \tag{3.29}$$

As equations (3.27) or (3.29) grow in number of factors, computing their value becomes computationally expensive. A usual solution to avoid this inconveniency is to calculate the log-probability to simplify the product of probabilities as the sum of their logarithms. This is expressed as

$$\log p(\theta \mid x_{-i}(t+T), x_{-i}(t),\theta_i) = \sum_{j\in N_i} \log \frac{p(x_j(t+T) \mid x_{-i}(t),\theta)\, p(\theta_j \mid x_{-i}(t))}{p(x_j(t+T) \mid x_{-i}(t),\theta_i)} + \sum_{k\notin N_i} \log p(\theta_k \mid x_{-i}(t))$$

for the independent types case (3.29). A similar result can be obtained for the dependent types version (3.27).

*3.3.2. Naïve likelihood approximation*

A significant difficulty in computing the value of the expression (3.29) is the limited knowledge of the agents due to the communication graph topology. It is desired to design a method to estimate the likelihood function (3.22) for agents that know only the state values of their neighbors, and are unaware of the graph topology except for the links that allow them to observe such neighbors.

From (3.22), agent $i$ needs to compute the probabilities $p(x_j(t+T)\,|\,x_{-i}(t),\theta)$ for all its neighbors $j$. This can be done if agent $i$ can predict the position $x_j(t+T)$ for each possible combination of types $\theta$ and given the current states $x_{-i}(t)$. However, $i$ does not know if the value $x_j(t+T)$ depends on the states of its neighbors $x_{-i}(t)$ because the neighbors of agent $j$ are unknown. The states of $i$'s neighbors may or may not affect $j$'s behavior.

Furthermore, the control policy (3.10) that agent $j$ uses at time $t$ depends not only on its type, but on its beliefs about the types of all other agents. The beliefs of agent $j$ are also unknown to agent $i$. Due to these knowledge constraints, agent $i$ must make assumptions about its neighbors to predict the state $x_j(t+T)$ using only local information.

Let agent $i$ make the naïve assumption that its other neighbors and itself are the neighbors of agent $j$. Thus, player $i$ tries to predict the state of its neighbor $j$ at time $t+T$ for the case where $i$ and $j$ have the same state information available. Besides, agent $i$ assumes that $j$ is certain (i.e., assigns probability one) of the combination of types in question, $\theta$.

Under these assumptions, agent $i$ estimates the local synchronization error of agent $j$ to be

$$\hat{\delta}_j^i = \sum_{k=1}^{N} a_{ik}(x_j - x_k) + g_i(x_j - x_0) + (x_j - x_i) \tag{3.30}$$

which means that $i$ expects the control policy of agent $j$ with types $\theta$ to be

$$E_i\left\{u_j^\theta\right\} = -\frac{1}{2}\left(R_{jj}^\theta\right)^{-1} B^T \nabla V_j^\theta\left(\hat{\delta}_j^i\right) \tag{3.31}$$

where the expected value operator is employed here in the sense that this is the value of $u_j^\theta$ that agent $i$ expects given its limited knowledge. If we consider a quadratic value function as in (3.12), then the expected policy (3.31) is written as

$$E_i\left\{u_j^\theta\right\} = -\frac{1}{2}\left(R_{jj}^\theta\right)^{-1} B^T P_j^\theta \hat{\delta}_j^i$$

with $\hat{\delta}_j^i$ defined in (3.30).

Now, the probabilities $p(x_j(t+T) \mid x_{-i}(t), \theta)$ can be determined by defining a probability distribution for the state $x_j(t+T)$. If a normal distribution is employed, then it is fully described by the mean $\mu_{ij}^\theta$ and the covariance $Cov_{ij}^\theta$, for neighbor $j$ and types $\theta$. In this case, the mean of the normal distribution function is the prediction of the state of agent $j$ at time $t+T$, that is

$$\mu_{ij}^\theta = \hat{x}_j^\theta(t+T) \tag{3.32}$$

where $\hat{x}_j^\theta(t+T)$ is the solution of the differential equation (2.2) for agent $j$ at time $t+T$, with control policy (3.31), i. e.,

$$\hat{x}_j^\theta(t+T) = e^{A(t+T)} x_j(t) + \int_t^{t+T} e^{-A(\tau-t-T)} B E_i\left\{u_j^\theta(\tau)\right\} d\tau.$$

The covariance $Cov_{ij}^\theta$ represents the lack of confidence of agent $i$ about the previous naïve assumptions, and is selected according to the problem in hand.

**Remark 3.7.** The intuition behind the naïve likelihood approximation for multiagent systems in graphs is inspired in the Naïve Bayes method for classification [52]. However, the proposed assumptions made by the agents in this chapter are different in nature and must not be confused.

Depending on the graph topology and the settings of the game, the proposed method for the likelihood calculation can differ considerably from reality. The effectiveness of the naïve likelihood approximation depends on the degree of accuracy of the assumptions made by the agents in a limited information environment. A measure of the uncertainty in the game is therefore useful in the analysis of the performance of the players.

In the following we introduce an uncertainty measure, the Bayesian game's *index of uncertainty* of agent $i$ with respect to its neighbor $j$. For simplicity, assume in this subsection that the graph weights are binary, i.e., $a_{ij} = 1$ if agents $i$ and $j$ are neighbors, and $a_{ij} = 0$ otherwise; the general case when $a_{ij} \geq 0$ can be obtained with few modifications. The index of uncertainty is defined by comparing the center of gravity of the true neighbors of agent $j$, and the neighbors that agent $i$ assumes for agent $j$.

Define the center of gravity of $j$'s neighbors as

$$c_j = \frac{\sum_{k=1}^{N} a_{jk} x_k}{\sum_{k=1}^{N} a_{jk}}. \tag{3.33}$$

When considering the *virtual* neighbors that agent $i$ assigned to agent $j$, we can acknowledge two mutually exclusive sets: the assigned *true* neighbors, which are actually neighbors of $j$, and the assigned *false* neighbors, which are not neighbors of $j$. Let the center of gravity of the assigned true neighbors be

$$\hat{c}_{ij}^{true} = \frac{\sum_{k=1}^{N} a_{ik} a_{jk} x_k + a_{ji} x_i}{\sum_{k=1}^{N} a_{ik} a_{jk} + a_{ji}}, \quad j \in N_i \tag{3.34}$$

and the center of gravity of the assigned false neighbors is

$$\hat{c}_{ij}^{false} = \frac{\sum_{k=1}^{N} a_{ik} (1 - a_{jk}) x_k + (1 - a_{ji}) x_i}{\sum_{k=1}^{N} a_{ik} (1 - a_{jk}) + (1 - a_{ji})}, \quad j \in N_i \tag{3.35}$$

Finally, let $\theta^*$ be the actual combination of types of the agents in the game, and $p_j(\theta^*)$ the belief of agent $j$ about $\theta^*$. The index of uncertainty is now defined as follows.

**Definition 3.4 (Index of uncertainty).** Define the index of uncertainty of agent $i$ about agent $j$ as

$$\upsilon_{ij} = \frac{1}{2}\frac{\left\| c_j - \hat{c}_{ij}^{true} + \hat{c}_{ij}^{false} \right\|}{\left\| \hat{c}_{ij}^{true} \right\|} + \frac{1}{2}\frac{1 - p_j(\theta^*)}{p_j(\theta^*)}. \tag{3.36}$$

□

Thus, index $\upsilon_{ij}$ measures how correct agent $i$ was about the beliefs and the states of the neighbors of agent $j$. The following lemma shows that the index of uncertainty is a nonnegative scalar, with $\upsilon_{ij} = 0$ if $i$ is absolutely correct about $j$'s neighbors and beliefs, and $\upsilon_{ij} \to \infty$ if the factors that influence $j$'s behavior are completely unknown to $i$.

**Lemma 3.2.** Let the index of uncertainty of agent $i$ about its neighbor, agent $j$, in a Bayesian game be as in (3.36). Then, $\upsilon_{ij} \in [0, \infty)$.

*Proof.* Notice that $c_j - \hat{c}_{ij}^{true}$ is a pseudo-center of gravity of all agents that are neighbors of agent $j$ but are not neighbors of $i$. Therefore, $\left\| c_j - \hat{c}_{ij}^{true} + \hat{c}_{ij}^{false} \right\|$ is a measure of all the agents that agent $i$ got wrong in its assumptions. If all of $i$'s assignments are true, then $\left\| c_j - \hat{c}_{ij}^{true} + \hat{c}_{ij}^{false} \right\| = 0$. On the contrary, if all alleged neighbors of $j$ were wrong, then $\left\| \hat{c}_{ij}^{true} \right\| = 0$.

Similarly, it can be seen that the second term in (3.36) is zero if $p_j(\theta^*) = 1$, and it tends to infinity if $p_j(\theta^*) = 0$. □

Theorem 3.4 uses the index of uncertainty (3.36) to determine a sufficient condition for the beliefs of an agent to converge to the actual types of the game $\theta^*$. Lemma 3.3 is used in the proof of this theorem.

***Lemma 3.3.*** Let $\theta^*$ be the actual combination of types in the game and consider the likelihood $p(x_{-i}(t+T) \mid x_{-i}(t), \theta)$ in (3.20). If the inequality

$$p(x_{-i}(t+T) \mid x_{-i}(t), \theta^*) > p(x_{-i}(t+T) \mid x_{-i}(t), \theta') \tag{3.37}$$

holds for every combination of types $\theta' \neq \theta^*$ at time instant $t+T$, then

$$p(\theta^* \mid x_{-i}(t+T), x_{-i}(t), \theta_i) > p(\theta^* \mid x_{-i}(t), \theta_i).$$

*Proof*. Let $\Gamma_i(\theta) = p(x_{-i}(t+T) \mid x_{-i}(t), \theta)$ be the likelihood of agent $i$ for types $\theta$. Because $\sum_{\theta\in\Theta} p(\theta \mid x_{-i}(t), \theta_i) = 1$, we have

$$\begin{aligned}
\Gamma_i(\theta^*) &= \Gamma_i(\theta^*) \sum_{\theta\in\Theta} p(\theta \mid x_{-i}(t), \theta_i) \\
&= \Gamma_i(\theta^*) p(\theta^1 \mid x_{-i}(t), \theta_i) + \cdots + \Gamma_i(\theta^*) p(\theta^M \mid x_{-i}(t), \theta_i) \\
&> \Gamma_i(\theta^1) p(\theta^1 \mid x_{-i}(t), \theta_i) + \cdots + \Gamma_i(\theta^M) p(\theta^M \mid x_{-i}(t), \theta_i) \\
&= \sum_{\theta\in\Theta} \Gamma_i(\theta) p(\theta \mid x_{-i}(t), \theta_i) \\
&= p(x_{-i}(t+T) \mid x_{-i}(t), \theta_i)
\end{aligned}$$

where inequality (3.37) was used in the third step, and the expression (3.24) was used in the last step. Now, from the Bayes rule (3.20) we can write

$$\begin{aligned}
p(\theta^* \mid x_{-i}(t+T), x_{-i}(t), \theta_i) &= \frac{\Gamma_i(\theta^*) p(\theta^* \mid x_{-i}(t), \theta_i)}{p(x_{-i}(t+T) \mid x_{-i}(t), \theta_i)} \\
&> p(\theta^* \mid x_{-i}(t), \theta_i)
\end{aligned}$$

which completes the proof. □

**Theorem 3.4.** Let the beliefs of the agents about the epistemic type $\theta$ be updated by means of the Bayesian rule (3.20), with the likelihood computed by means of a normal probability distribution with mean $\mu_{ij}^{\theta}$ as in (3.32), and covariance $Cov_{ij}^{\theta}$. Then, the beliefs of agent $i$ converge to the correct combination of types $\theta^*$ if the index of uncertainty (3.36) is close to zero for all its neighbors $j$.

*Proof.* Consider the case where $\upsilon_{ij} = 0$; this occurs when the actual neighbors of agent $j$ are precisely agent $i$ and agent $i$'s neighbors, and agent $j$ assigns probability one to the combination of types $\theta^*$. This implies that the state value $x_j(t+T)$ will be exactly the estimation $\hat{x}_j^{\theta^*}(t+T)$ and the highest probability of the normal distribution is obtained for the likelihood $p(x_j(t+T) \mid x_{-i}(t), \theta^*)$. By Lemma 3.3, the belief in type $\theta^*$ is increased at every time step $T$, converging to 1.

If $\upsilon_{ij}$ is an arbitrarily small positive number, then the center of gravity of the assigned neighbors is close to the center of gravity of the real neighbors of agent $j$. Furthermore, the beliefs of $j$ in the combination of types $\theta^*$ is close to 1. Now, the estimation of the state $\hat{x}_j^{\theta^*}(t+T)$ is arbitrarily close to the actual state $x_j(t+T)$, making the likelihood $p(x_j(t+T)\,|\,x_{-i}(t),\theta^*)$ larger than the likelihood of any other type $\theta$. Again, the conditions of Lemma 3.3 hold and the belief in the type $\theta^*$ converges to 1 at each iteration. □

**Remark 3.8.** A large value for the index of uncertainty expresses that an agent lacks enough information to understand the behavior of its neighbors. This implies that the beliefs of the agent cannot be corrected properly.

**Remark 3.9.** The index of uncertainty is defined for analysis purposes and is unknown to the agents during the game. It allows to determine if the agents have enough information to find the actual combination of types of the game.

## 3.4. Non-Bayesian Belief Updates

The Bayesian belief update method presented in the previous section starts with the assumption that every agent knows its own type at the beginning of the game. In some applications, however, an agent can be uncertain about its type, or the concept of type can be ill-defined. In these cases, it is still possible to solve the Bayesian graphical game problem if we allow more information to flow through the communication topology. In [47], a non-Bayesian belief update algorithm is shown to efficiently converge to the type of the game $\theta$. In this section, we use this method as an alternative to the Bayesian update in Section 3.3 when every agent can communicate its beliefs about $\theta$ to its neighbors.

Let the belief update of player $i$ to be computed as

$$p_i(\theta \mid x_{-i}(t+T), x_{-i}(t)) = b_{ii} p_i(\theta \mid x_{-i}(t)) \frac{p_i(x_{-i}(t+T) \mid x_{-i}(t), \theta)}{p_i(x_{-i}(t+T) \mid x_{-i}(t))} + \sum_{j=1}^{N} a_{ij} p_j(\theta) \tag{3.38}$$

where $p_j(\theta)$ are the beliefs of agent $j$ about $\theta$, and the constant $b_{ii} > 0$ is the weight that player $i$ gives to its own beliefs relative to the graph weights $a_{ij}$ assigned to its neighbors. Notice that it is required that $\sum_{j=1}^{N} a_{ij} + b_{ii} = 1$ for $p_i(\theta \mid x_{-i}(t+T), x_{-i}(t), \theta_i)$ to be a well-defined probability distribution.

Equation (3.38) expresses that the beliefs of agent $i$ at time $t+T$ is a linear combination of its own Bayesian belief update, and the beliefs of its neighbors at time $t$. This is regarded as a non-Bayesian belief update of the epistemic types.

Notice that (3.38) does not consider the knowledge of $\theta_i$ by agent $i$. The assumption that the agents can communicate their beliefs to their neighbors is meaningful if we consider the case when the agents are uncertain about their own types; otherwise they would be able to inform to their neighbors about their actual type through the communication topology.

Similar to (3.27), the factors in the first term of (3.38) can be decomposed in terms of the states and types of $i$ neighbors as and non-neighbors, such that

$$\begin{aligned} p_i(\theta \mid x_{-i}(t+T), x_{-i}(t)) = b_{ii} \prod_{j \in N_i} \frac{p_i(x_j(t+T) \mid x_{-i}(t), \theta)}{p_i(x_j(t+T) \mid x_{-i}(t))} p(\theta_j \mid x_{-i}(t), \theta_1, \ldots, \theta_{j-1}) \\ \times \prod_{k \notin N_i} p(\theta_k \mid x_{-i}(t), \theta_1, \ldots, \theta_{k-1}) + \sum_{j=1}^{N} a_{ij} p_j(\theta) \end{aligned} \tag{3.39}$$

where dependent epistemic types have been considered.

## 3.5. Simulation Results

In this section, two simulations are performed to show the behavior of the agents during a Bayesian graphical game using a Bayesian and a non-Bayesian belief updates. The solutions of the BHJ equations for Nash equilibrium are given.

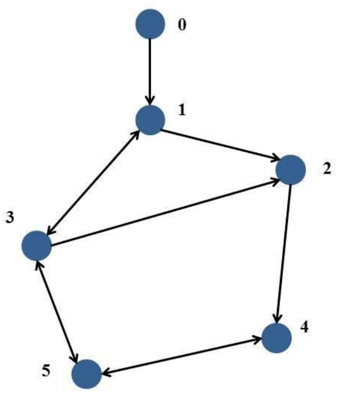

Fig. 3.1. Graph topology employed in simulation of Bayesian games.

### *3.5.1. Parameters for simulation*

Consider a multiagent system with 5 agents and one leader, connected in a directed graph as shown in Fig. 3.1. All agents are taken with single integrator dynamics, as

$$\dot{x}_i = \begin{bmatrix} \dot{x}_{i,1} \\ \dot{x}_{i,2} \end{bmatrix} = \begin{bmatrix} u_{i,1} \\ u_{i,2} \end{bmatrix}$$

In this game, only agent 1 has two possible types, and all other agents start with the same prior knowledge of the probabilities of each type. Let agent 1 have type 1 40% of the cases, and type 2 60% of the cases.

The cost functions of the agents are taken in the form (2.7), considering the same weighting matrices for all agents, that is, $Q_{ij}^{\theta} = Q_{kl}^{\theta}$, $R_{ii}^{\theta} = R_{jj}^{\theta}$ and $R_{ij}^{\theta} = R_{kl}^{\theta}$ for $i, j, k, l \in \{0,1,2,3,4,5\}$. For type $\theta_1$, the weighting matrices are taken as

$$Q_{ij}^{\theta_1} = \frac{4}{10}\begin{bmatrix} I & -I \\ -I & 2I \end{bmatrix},$$

$R_{ii}^{\theta_1} = 10I$ and $R_{ij}^{\theta_1} = -20I$ for $i \neq j$, where $I$ is the identity matrix. The matrices of the cost functions for type $\theta_2$ are taken as

$$Q_{ij}^{\theta_2} = \begin{bmatrix} 16I & -16I \\ -16I & 32I \end{bmatrix},$$

$R_{ii}^{\theta_2} = I$ for all agents $i$, and $R_{ij}^{\theta_2} = -2I$ for $i \neq j$.

To solve this game, we start by considering a general formulation for the value functions of the game, and then we show that the control policies of the agents are optimal and distributed. Propose a value function with the form $V_i^\theta = \sum_{j=0}^N a_{ij} \bar{\delta}_{ij}^T P_i^\theta \bar{\delta}_{ij}$, where $a_{i0} = g_i$, $\bar{\delta}_{i0} = \begin{bmatrix} \delta_i^T & 0^T \end{bmatrix}^T$ and $\bar{\delta}_{ij} = \begin{bmatrix} \delta_i^T & \delta_j^T \end{bmatrix}^T$ for $j \neq 0$, as solution for the cost function (2.6) - (2.7) for type $\theta$. Notice that this value function is not necessarily distributed because it depends on the local information of the neighbors of agent $i$. We prove below that, for type 1, matrix $P_i^{\theta_1}$ has the form

$$P_i^{\theta_1} = \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} \tag{3.40}$$

and, for type 2,

$$P_i^{\theta_2} = \begin{bmatrix} 2I & 0 \\ 0 & 0 \end{bmatrix} \tag{3.41}$$

for all agents, and hence distributed policies are obtained.

Express the expected Hamiltonian for agent $i$ as

$$EH_i = \sum_{\theta=1}^{2} \sum_{j=0}^{N} p(\theta) a_{ij} \left( \bar{\delta}_{ij}^T Q_{ij}^\theta \bar{\delta}_{ij} + u_i^T R_{ii}^\theta u_i + u_j^T R_{ij}^\theta u_j + 2\bar{\delta}_{ij}^T P_i^\theta \dot{\bar{\delta}}_{ij} \right)$$

where the derivative $\dot{\bar{\delta}}_{ij}$ when $j \neq 0$ is given by

$$\begin{bmatrix} \dot{\delta}_i \\ \dot{\delta}_j \end{bmatrix} = \begin{bmatrix} A\delta_i + (d_i + g_i) Bu_i - \sum_{k=1}^N a_{ik} Bu_k \\ A\delta_j + (d_j + g_j) Bu_j - \sum_{k=1}^N a_{jk} Bu_k \end{bmatrix}.$$

From the expected Hamiltonian, the optimal control policies are obtained as

$$u_i^* = -\left( \sum_{\theta=1}^{2} p(\theta) R_{ii}^\theta \right)^{-1} \sum_{j=0}^{N} \frac{a_{ij}}{d_i + g_i} \begin{bmatrix} (d_i + g_i) B^T & -a_{ji} B^T \end{bmatrix} \left( \sum_{\theta=1}^{2} p(\theta) P_i^\theta \right) \bar{\delta}_{ij} \tag{3.42}$$

which, as mentioned before, are not necessarily distributed. Using the policies $u_i^*$ for all agents, we find that the BHJ equations that must be solved by matrices $P_i^\theta$ are

$$\sum_{\theta=1}^{2}\sum_{j=1}^{N} p(\theta)a_{ij}\left(\bar{\delta}_{ij}^T Q_{ij}^\theta \bar{\delta}_{ij} + u_i^{*T} R_{ii}^\theta u_i^* + u_j^{*T} R_{ij}^\theta u_j^* + 2\bar{\delta}_{ij}^T P_i^\theta \dot{\bar{\delta}}_{ij}^*\right) = 0. \tag{3.43}$$

To show that (3.42) with $P_i^\theta$ as in (3.43) is the optimal policy for agent $i$, express the expected cost of agent $i$ as

$$EJ_i = \int_0^\infty \sum_{\theta\in\Theta}\sum_{j=1}^{N} p(\theta)a_{ij}\left(\bar{\delta}_{ij}^T Q_{ij}^\theta \bar{\delta}_{ij} + u_i^T R_{ii}^\theta u_i + u_j^T R_{ij}^\theta u_j\right)dt + \sum_{\theta\in\Theta} p(\theta)\int_0^\infty \dot{V}_i^\theta dt + \sum_{\theta\in\Theta} p(\theta)V_i^\theta(\delta(0)).$$

Similarly as in Lemma 3.1, it is easy to show that

$$\begin{aligned} EJ_i = &\int_0^\infty \sum_{\theta\in\Theta}\sum_{j=1}^{N} p(\theta)a_{ij}\left(\bar{\delta}_{ij}^T Q_{ij}^\theta \bar{\delta}_{ij} + u_i^{*T} R_{ii}^\theta u_i^* + u_j^T R_{ij}^\theta u_j\right)dt \\ &+ \sum_{\theta\in\Theta} p(\theta)\int_0^\infty \dot{V}_i^\theta dt + \sum_{\theta\in\Theta} p(\theta)\int_0^\infty (u_i - u_i^*)^T R_{ii}(u_i - u_i^*)dt + \sum_{\theta\in\Theta} p(\theta)V_i^\theta(\delta(0)) \end{aligned}$$

for all $u_i$ and $u_{-i}$. As (3.43) holds, if all neighbors of agent $i$ use their best strategies $u_{-i}^*$, then

$$EJ_i = \sum_{\theta\in\Theta} p(\theta)\left[\int_0^\infty (u_i - u_i^*)^T R_{ii}(u_i - u_i^*)dt + V_i^\theta(\delta(0))\right]$$

and $u_i^*$ in (3.42) is indeed the optimal strategy of agent $i$.

To show that (3.40) and (3.41) solve the equations (3.43) for all agents, substitute the matrices in the value functions $V_i^\theta$ and the policies $u_i^*$ of the agents. Thus, for type $\theta_1$, we can write $V_i^{\theta_1} = (d_i + g_i)\delta_i^T \delta_i$; for type $\theta_2$, $V_i^{\theta_2} = 2(d_i + g_i)\delta_i^T \delta_i$; and the optimal control policies are given by

$$u_i^* = -(d_i + g_i)\left(\sum_{\theta=1}^{2} p(\theta)R_{ii}^\theta\right)^{-1} B^T\left(p(\theta_1)I + 2p(\theta_2)I\right)\delta_i.$$

Notice that matrices (3.40) and (3.41) make $u_i^*$ distributed. Using the expressions (3.42) and (3.43) and the cost functions of the game, we obtain the following result for type $\theta_1$

$$
\begin{aligned}
&\sum_{j=0}^{N} a_{ij}\left(\bar{\delta}_{ij}^{T} Q_{ij}^{\theta_1}\bar{\delta}_{ij} + u_i^T R_{ii}^{\theta_1} u_i + u_j^T R_{ij}^{\theta_1} u_j\right) + \nabla V_i^{\theta T}\left(A\delta_i + (d_i+g_i)Bu_i - \sum\nolimits_{j=1}^{N} a_{ij} B u_j\right) \\
&= \sum_{j=0}^{N} a_{ij}\left(\frac{4}{10}\bar{\delta}_{ij}^{T}\begin{bmatrix} I & -I \\ -I & 2I\end{bmatrix}\bar{\delta}_{ij} + 10 u_i^T u_i - 20 u_j^T u_j\right) + 2\sum_{j=0}^{N} a_{ij}\left(\bar{\delta}_{ij}^{T}\begin{bmatrix} I & 0 \\ 0 & 0\end{bmatrix}\left(2u_i - \sum\nolimits_{j=1}^{N} a_{ij} u_j\right)\right) \\
&= \sum_{j=0}^{N} a_{ij}\left(\frac{4}{10}\delta_i^T\delta_i - \frac{8}{10}\delta_i^T\delta_j + \frac{8}{10}\delta_j^T\delta_j + 10 u_i^T u_i - 20 u_j^T u_j\right) + 2\sum_{j=0}^{N} a_{ij}\left(\delta_i^T\left((2u_i - \sum\nolimits_{j=1}^{N} a_{ij} u_j\right)\right)
\end{aligned}
$$

Substituting $u_i^*$ and $u_j^*$ we get

$$
\begin{aligned}
&\sum_{j=0}^{N} a_{ij}\left(\frac{4}{10}\delta_i^T\delta_i - \frac{8}{10}\delta_i^T\delta_j + \frac{8}{10}\delta_j^T\delta_j + \frac{4}{10}\delta_i^T\delta_i - \frac{8}{10}\delta_j^T\delta_j\right) - \sum_{j=0}^{N} a_{ij}\left(\frac{8}{10}\delta_i^T\delta_i + \frac{4}{10}\sum\nolimits_{j=1}^{N} a_{ij}\delta_i^T\delta_j\right) \\
&= \sum_{j=0}^{N} a_{ij}\left(\frac{4}{10}\delta_i^T\delta_i - \frac{8}{10}\delta_i^T\delta_j + \frac{8}{10}\delta_j^T\delta_j + \frac{4}{10}\delta_i^T\delta_i - \frac{8}{10}\delta_j^T\delta_j - \frac{8}{10}\delta_i^T\delta_i + \frac{8}{10}\sum\nolimits_{j=1}^{N} a_{ij}\delta_i^T\delta_j\right) = 0
\end{aligned}
$$

Similarly, for type $\theta_2$ we have

$$
\begin{aligned}
&\sum_{j=0}^{N} a_{ij}\left(\bar{\delta}_{ij}^{T} Q_{ij}^{\theta_2}\bar{\delta}_{ij} + u_i^T R_{ii}^{\theta_2} u_i + u_j^T R_{ij}^{\theta_2} u_j\right) + \nabla V_i^{\theta T}\left(A\delta_i + (d_i+g_i)Bu_i - \sum\nolimits_{j=1}^{N} a_{ij} B u_j\right) \\
&= \sum_{j=0}^{N} a_{ij}\left(\bar{\delta}_{ij}^{T}\begin{bmatrix} 16I & -16I \\ -16I & 32I\end{bmatrix}\bar{\delta}_{ij} + u_i^T u_i - 2 u_j^T u_j\right) + 2\sum_{j=0}^{N} a_{ij}\left(\bar{\delta}_{ij}^{T}\begin{bmatrix} 2I & 0 \\ 0 & 0\end{bmatrix}\left(2u_i - \sum\nolimits_{j=1}^{N} a_{ij} u_j\right)\right) \\
&= \sum_{j=0}^{N} a_{ij}\left(16\delta_i^T\delta_i - 32\delta_i^T\delta_j + 32\delta_j^T\delta_j + u_i^T u_i - 2u_j^T u_j\right) + 4\sum_{j=0}^{N} a_{ij}\left(\delta_i^T\left((2u_i - \sum\nolimits_{j=1}^{N} a_{ij} u_j\right)\right) \\
&= \sum_{j=0}^{N} a_{ij}\left(16\delta_i^T\delta_i - 32\delta_i^T\delta_j + 32\delta_j^T\delta_j + 16\delta_i^T\delta_i - 32\delta_j^T\delta_j\right) - \sum_{j=0}^{N} a_{ij}\left(32\delta_i^T\delta_i + 16\sum\nolimits_{j=1}^{N} a_{ij}\delta_i^T\delta_j\right) \\
&= \sum_{j=0}^{N} a_{ij}\left(16\delta_i^T\delta_i - 32\delta_i^T\delta_j + 32\delta_j^T\delta_j + 16\delta_i^T\delta_i - 32\delta_j^T\delta_j - 32\delta_i^T\delta_i + 32\sum\nolimits_{j=1}^{N} a_{ij}\delta_i^T\delta_j\right) \\
&= 0
\end{aligned}
$$

Finally, the BHJ equations for all agents, $i = 1,\ldots,5$, can be written as

$$
\begin{aligned}
&p(\theta_1)\left[\sum_{j=0}^{N} a_{ij}\left(\bar{\delta}_{ij}^{T} Q_{ij}^{\theta_1}\bar{\delta}_{ij} + u_i^T R_{ii}^{\theta_1} u_i + u_j^T R_{ij}^{\theta_1} u_j\right) + \nabla V_i^{\theta_1 T}\left(A\delta_i + (d_i+g_i)Bu_i - \sum\nolimits_{j=1}^{N} a_{ij} B u_j\right)\right] \\
&+ p(\theta_2)\left[\sum_{j=0}^{N} a_{ij}\left(\bar{\delta}_{ij}^{T} Q_{ij}^{\theta_2}\bar{\delta}_{ij} + u_i^T R_{ii}^{\theta_2} u_i + u_j^T R_{ij}^{\theta_2} u_j\right) + \nabla V_i^{\theta_2 T}\left(A\delta_i + (d_i+g_i)Bu_i - \sum\nolimits_{j=1}^{N} a_{ij} B u_j\right)\right] = 0
\end{aligned}
$$

Therefore, matrices $P_i^{\theta_1}$ and $P_i^{\theta_2}$ are the solutions of the game. As the control policies obtained from these matrices are distributed, this numerical example has shown a system for which Assumption 2.1 holds.

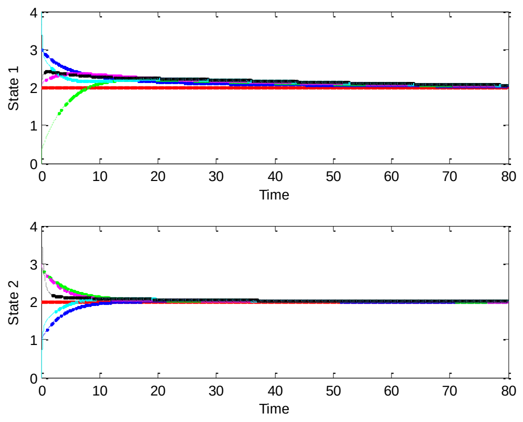

Fig. 3.2. Trajectories for both states of the five agents in Bayesian games.

### *3.5.2. Bayesian belief update*

With the exception of agent 1, let all players update their beliefs about the type $\theta$ every 0.1 seconds, using a Bayesian belief update (3.29) with naïve likelihood approximation, as described in Section 3.3. During this simulation, agent 1 is in type 1.

The state dynamics of the agents are shown in Fig. 3.2. In Fig. 3.3, the evolution of the beliefs of every agent is displayed. Note that all beliefs approach probability one for type $\theta_1$, and all agents end up playing the same game.

### *3.5.3. Non-Bayesian Belief Update*

The simulation is now repeated using (3.39) for the non-Bayesian belief update. Agent 1 is again in type 1, and agents 2 to 5 share their individual beliefs about $\theta_1$ with their neighbors according to the communication graph topology.

Figure 3.4 shows the convergence of the beliefs in type 1 of the four agents. Convergence is faster in this case, due to the additional flow of information about the beliefs of the agents.

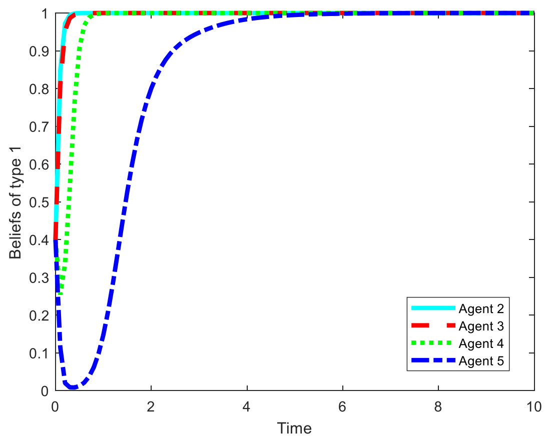

Fig. 3.3. Evolution of beliefs in type 1 of the agents with Bayesian update.

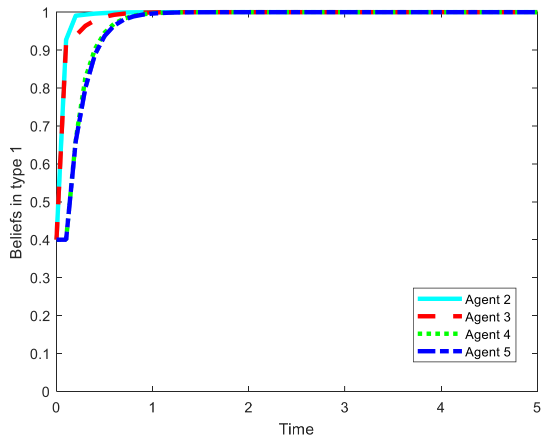

Fig. 3.4. Evolution of beliefs in type 1 of the agents with non-Bayesian update.

## 3.6. Conclusion

Multiagent systems analysis was performed for dynamical agents engaged on interactions with uncertain objectives. We reveal for the first time the tight relationship between the beliefs of an agent and its distributed best response control policy. Minmax strategies were shown to be a practical solution to overcome the lack of information of the agents to achieve Bayes-Nash equilibrium. Under appropriate circumstances, the proposed naïve likelihood approximation is a useful method to deal with the limited knowledge of the agents about the graph topology.

Simulations with two different belief update algorithms show the applicability of the proposed methods. The Bayesian belief update achieves convergence of the beliefs using solely measurements of the states of their neighbors. The non-Bayesian updates take advantage of supplementary information to achieve a faster and more robust convergence of the beliefs to the true type of the game.

Further lines of research about Bayesian graphical games include the proposition of new solution concepts that yield distributed control policies for the agents. Also, new belief update methods can be developed to improve their convergence in environments with incomplete information.

Chapter 4

# MINMAX SOLUTIONS FOR DIFFERENTIAL GRAPHICAL GAMES

## 4.1. Introduction

Every admissible solution for a graphical game requires the use of distributed control policies. This means that the agents are allowed to use only local information received through the communication graph to design their control strategies. The distributed-policy requirement, however, makes Nash equilibrium generally unattainable among the agents. This fact can be intuitively explained by noticing that an agent needs to know its neighbors' best strategies to determine its own best response against them, but the neighbors' best policies are unknown without global state information. Thus, the information restriction imposed by the graph topology prevents the multiagent system from reaching an equilibrium.

The unattainability of Nash equilibrium in graphical games can be addressed by proposing alternative solution concepts as objectives for the agents. Even with incomplete knowledge about their environment, the agents can be expected to determine a best policy for the information they have available. In this chapter, we analyze the behavior of the agents connected in a communication graph when they use minmax strategies [1], [28] to achieve their goals. Using minmax strategies, each agent assumes the worst possible behavior from its neighbors and prepares its best response to oppose them. This assumption is formalized by making every agent determine the control policy of its neighbors that would maximize its own cost function. From the perspective of an individual agent, the resulting formulation of this graphical game is the same as an $H_\infty$ control problem [29]. In turn, the $H_\infty$ formulation can be solved as the zero-sum game between a system and a disturbance term [5], [28]. The $H_\infty$ controller has been thoroughly studied due to its attractive robust characteristics [30]- [32]. Thus, minmax strategies provide robust control policies for the networked agents due to their conservative assumption about the worst neighbor behavior.

A useful method to solve $H_\infty$ problems has been proposed with the use of reinforcement learning (RL) algorithms [33]-[36]. An off-policy RL algorithm [36] is also proposed in this chapter to solve the minmax problem without any knowledge of the model of the agents, nor any knowledge about the neighbor control policies. Off-policy RL has been used to solve $H_\infty$ problems in [37], [38].

In the remainder of this chapter, Section 4.2 presents the minmax strategies problem and its solution is obtained. The stability of minmax policies is studied in Section 4.3. Section 4.4 analyzes the robustness of minmax strategies. An off-policy RL algorithm is presented in Section 4.5 to solve the minmax problem. Simulation studies are presented in Section 4.6.

## 4.2. Minmax Strategies for Graphical Games

In this section, we first formally present the disadvantages of the Nash equilibrium solution for games in communication graphs. Then, we remedy these inconveniences by defining the minmax strategies for differential graphical games.

### *4.2.1. Drawbacks in Nash equilibrium*

In Section 2.4, it was presented that in a graphical game with agent dynamics (2.2) - (2.3), and performance indices (2.6) - (2.8) Nash equilibrium is obtained by means of the control policies (2.11), repeated here for completeness,

$$u_i^* = -\frac{1}{2}(d_i + g_i)R_{ii}^{-1}B^T\nabla V_i(\delta_i) \tag{4.1}$$

where the value function $V_i(\delta_i)$ is obtained as the solution of the HJ equation (2.12). Substituting the policies (4.1) in (2.12) for all agents $i$, $j$ in the game, we obtain the explicit HJ equations

$$\begin{aligned} &\delta_i^T Q_i \delta_i + \nabla V_i^T A \delta_i - \frac{(d_i + g_i)^2}{4} \nabla V_i^T B R_i^{-1} B^T \nabla V_i \\ &\quad + \frac{1}{4} \sum_{j=1}^{N} a_{ij} (d_j + g_j)^2 \nabla V_j^T B R_j^{-1} B^T \nabla V_j + \frac{1}{2} \sum_{j=1}^{N} a_{ij} (d_j + g_j) \nabla V_i^T B R_j^{-1} B^T \nabla V_j = 0. \end{aligned} \tag{4.2}$$

A valid distributed control policy (4.1) requires that the value function for agent $i$ employs only local information. For this reason, we made Assumption 2.1 in Chapter 2, stating that $V_i(\delta) = V_i(\delta_i)$. Using this assumption, and rearranging the HJ equation (4.2), we obtain

$$\begin{aligned} &\frac{1}{4} \sum_{j=1}^{N} a_{ij} \left[ \nabla V_i + \nabla \bar{V}_j \right]^T \; B R_j^{-1} B^T \left[ \nabla V_i + \nabla \bar{V}_j \right] \\ &\quad = \frac{1}{4} \sum_{j=1}^{N} a_{ij} \nabla V_i^T B \;\; (d_i + g_i) R_i^{-1} + R_j^{-1} \;\; B^T \nabla V_i - \delta_i^T Q_i \delta_i - \nabla V_i^T A \delta_i \end{aligned} \tag{4.3}$$

where $\nabla \bar{V}_j = (d_j + g_j) \nabla V_j$. This equation has the form $f_1(\delta_i, \delta_{-i}) = f_2(\delta_i)$, with $\delta_{-i}$ the local errors of the neighbors of $i$, and in most cases it will not hold true for all possible neighbor trajectories $\delta_{-i}$.

In general, there may not exist a set of functions $V_i(\delta_i)$ that solve the HJ equations (4.2) and provide distributed control policies for the agents. This is an expected result due to the limited knowledge of the agents connected in the communication graph. If agent $i$ does not know the local information of its neighbors, $\delta_j$, then it cannot determine their best responses in the game and prepare its best strategy accordingly.

We are now in a position to propose minmax strategies as a practical alternative to the Nash equilibrium solution.

### *4.2.2. Formulation of minmax strategies*

Let agent $i$ prepare its minmax strategy by making the conservative assumption that the goal of its neighbors is to maximize its own performance index, $J_i$. The following definition formalizes the concept of minmax strategy employed in this dissertation.

**Definition 4.1 (Minmax strategies for graphical games).** In a differential graphical game, the minmax strategy of agent $i$ is given by

$$u_i^* = \arg\min_{u_i}\max_{u_{-i}} J_i \;\; \delta_i, u_i, u_{-i} \;\; . \tag{4.4}$$

□

The performance index (2.6) - (2.7) requires to be modified to formulate a zero-sum game between agent $i$ and its neighbors. To this end, define the function

$$J_i = \int_0^\infty \left( \delta_i^T Q_i \delta_i + (d_i + g_i) u_i^T R_i u_i - \gamma^2 \sum_{j=1}^{N} a_{ij} u_j^T R_j u_j \right) dt \tag{4.5}$$

where $Q_i \geq 0$, $R_i, R_j > 0$ and $\gamma$ is a positive scalar. To determine its minmax strategy, agent $i$ assumes that the goal of its neighbors is to maximize its cost function (4.5).

Define the Hamiltonian function associated with the cost index (4.5) as

$$\begin{aligned} H_i = {} & \delta_i^T Q_i \delta_i + (d_i + g_i) u_i^T R_i u_i - \gamma^2 \sum_{j=1}^{N} a_{ij} u_j^T R_j u_j \\ & + \nabla V_i^T \left( A\delta_i + (d_i + g_i) B u_i - \sum_{j=1}^{N} a_{ij} B u_j \right). \end{aligned} \tag{4.6}$$

If the value function $V_i$ has a quadratic form as

$$V_i(\delta_i) = \delta_i^T P_i \delta_i \tag{4.7}$$

then (4.6) can be expressed as

$$\begin{aligned} H_i = {} & \delta_i^T Q_i \delta_i + (d_i + g_i) u_i^T R_i u_i - \gamma^2 \sum_{j=1}^{N} a_{ij} u_j^T R_j u_j \\ & + 2\delta_i^T P_i \left( A\delta_i + (d_i + g_i) B u_i - \sum_{j=1}^{N} a_{ij} B u_j \right). \end{aligned} \tag{4.8}$$

The optimal control policy for agent $i$ is now obtained by means of the stationary condition $\dfrac{\partial H_i}{\partial u_i} = 0$, which yields

$$u_i^* = -R_i^{-1} B^T P_i \delta_i. \tag{4.9}$$

Similarly, the worst-case policy of the neighbors of agent $i$ can be obtained as

$$v_j^* = -\frac{1}{\gamma^2} R_j^{-1} B^T P_i \delta_i. \tag{4.10}$$

Notice that $v_j^*$ is not necessarily the actual control policy employed by agent $j$, represented by $u_j$.

The HJ equation to be solved for matrix $P_i$ is finally obtained by substituting the policies (4.9) and (4.10) in (4.8), and equating it to zero. This procedure yields the algebraic Riccati equations (ARE)

$$Q_i + P_i A + A^T P_i - (d_i + g_i) P_i B R_i^{-1} B^T P_i + \frac{1}{\gamma^2} \sum_{j=1}^{N} a_{ij} P_i B R_j^{-1} B^T P_i = 0 \tag{4.11}$$

The following theorem shows that control policy (4.9) with $P_i$ obtained as the solution of (4.11) provides the minmax strategy for agent $i$. The proof of this theorem assumes stability of the error dynamics (2.5); such stability will be analyzed in the next section.

**Theorem 4.1.** Let the agents of a differential graphical game with dynamics (2.2) and a leader with dynamics (2.3) use the control policies (4.9) where matrices $P_i$ are the solutions of the AREs (4.11). Moreover, assume these control policies stabilize the local synchronization errors (2.5) for all agents $i$. Then, all agents follow their minmax strategies as defined in Definition 4.1, and the minmax value of the game is $V_i(\delta_i(0))$.

*Proof*. Consider the value function (4.7) and express the performance index (4.5) as

$$J_i = \int_0^\infty \left( \delta_i^T Q_i \delta_i + (d_i + g_i) u_i^T R_i u_i - \gamma^2 \sum_{j=1}^{N} a_{ij} u_j^T R_j u_j \right) dt - \int_0^\infty \dot{V}_i(\delta_i) dt$$
$$+ \int_0^\infty 2\delta_i^T P_i \left( A\delta_i + (d_i + g_i) B u_i - \sum_{j=1}^{N} a_{ij} B u_j \right) dt.$$

Using the inner-product notation (2.15), we can express $J_i$ as

$$J_i = \left\langle \delta_i, Q_i \delta_i \right\rangle + (d_i + g_i)\left\langle u_i, R_i u_i \right\rangle - \gamma^2 \sum_{j=1}^{N} a_{ij} \left\langle u_j, R_j u_j \right\rangle + V_i(\delta_i(0)) + 2\left\langle \delta_i, P_i A \delta_i \right\rangle$$
$$+ 2(d_i + g_i)\left\langle \delta_i, P_i u_i \right\rangle - 2\sum_{j=1}^{N} a_{ij} \left\langle \delta_i, P_i B u_j \right\rangle$$

where we have used the fact that $\int_0^\tau \dot{V}_i(\delta_i)dt = V_i(\delta_i(\tau)) - V_i(\delta_i(0))$, and that, because the system is stable, $V_i(\delta_i(\tau)) = 0$ in the limit as $\tau \to \infty$. As $P_i$ makes the ARE (4.11) hold, we get

$$\begin{aligned} J_i &= (d_i + g_i)\left\langle u_i^*, R_i u_i^* \right\rangle - \gamma^2 \sum_{j=1}^{N} a_{ij} \left\langle v_j^*, R_j v_j^* \right\rangle + (d_i + g_i)\left\langle u_i, R_i u_i \right\rangle \\ &\quad - \gamma^2 \sum_{j=1}^{N} a_{ij} \left\langle u_j, R_j u_j \right\rangle - 2(d_i + g_i)\left\langle u_i^*, R_i u_i \right\rangle + 2\gamma^2 \sum_{j=1}^{N} a_{ij} \left\langle v_j^*, R_j u_j \right\rangle \\ &\quad + V_i(\delta_i(0)) \\ &= (d_i + g_i)\left\langle u_i - u_i^*, R_i(u_i - u_i^*) \right\rangle - \gamma^2 \sum_{j=1}^{N} a_{ij} \left\langle u_j - v_j^*, R_j(u_j - v_j^*) \right\rangle \\ &\quad + V_i(\delta_i(0)) \end{aligned} \tag{4.12}$$

Therefore, $u_i^*$ in (4.9) with $P_i$ as in (4.11) is the minmax strategy of agent $i$, (4.10) represents the worst-case policies of the neighbors, and the value of the game is given by $V_i(\delta_i(0))$. □

**Remark 4.1.** Control policies (4.9) are always distributed, in contrast to the policies based in the Nash solution given by (4.1).

**Remark 4.2.** Equations in the form of (4.11) are known to have solutions for $P_i$ if $(A, \sqrt{Q_i})$ is observable, $(A, B)$ is stabilizable, and $(d_i + g_i)R_i^{-1} - \frac{1}{\gamma^2}\sum_{j=1}^{N} a_{ij} R_j^{-1} > 0$.

In the following section we analyze the stability properties of the minmax policies (4.9).

## 4.3. Stability of Minmax Strategies

Two stability concepts are analyzed in this section for minmax strategies. First, it is proven that the system (2.5) with control policies $u_i^*$ in (4.9) is finite-gain $\mathcal{L}_2$ stable. Then, conditions for asymptotic stability of the global multiagent system are provided.

### *4.3.1. $\mathcal{L}_2$ stability*

When using minmax strategies, an agent with error dynamics (2.5) considers the effect of its neighbors policies, $\sum_{j=1}^{N} a_{ij} B u_j$ , as a disturbance term to be rejected. The nominal system for agent $i$ can then be defined as

$$\dot{\bar{\delta}}_i = A\bar{\delta}_i + (d_i + g_i) B u_i. \tag{4.13}$$

This idea provides the foundation of the following stability analysis.

Define the performance output of agent $i$ as

$$\left\| z_i(t) \right\|^2 = \delta_i^T Q_i \delta_i + (d_i + g_i) u_i^T R_i u_i. \tag{4.14}$$

Similarly, the *disturbance* input produced by the neighbors of agent $i$ is defined as

$$\left\| \zeta_i(t) \right\|^2 = \sum_{j=1}^{N} a_{ij} u_j^T R_j u_j. \tag{4.15}$$

According to (2.16), the output (4.14) is $\mathcal{L}_2$ stable if

$$\left\| z_i(t) \right\|_{\mathcal{L}_2} \leq \gamma \left\| \zeta_i(t) \right\|_{\mathcal{L}_2} + \beta \tag{4.16}$$

for some $\gamma, \beta \geq 0$. The following theorem shows the $\mathcal{L}_2$ stability properties of the minmax policies (4.9).

**Theorem 4.2.** The system (2.5) with policy $u_i^*$ as in (4.9) and $P_i$ as the solution of (4.11) is $\mathcal{L}_2$ stable with $\mathcal{L}_2$-gain bounded by $\gamma$ according to (4.16).

*Proof*. In the proof of Theorem 4.1, the final step (4.12) showed that

$$\begin{aligned} J_i &= \left\langle \delta_i, Q_i \delta_i \right\rangle + (d_i + g_i) \left\langle u_i, R_i u_i \right\rangle - \gamma^2 \sum_{j=1}^{N} a_{ij} \left\langle u_j, R_j u_j \right\rangle \\ &= (d_i + g_i) \left\langle u_i - u_i^*, R_i (u_i - u_i^*) \right\rangle - \gamma^2 \sum_{j=1}^{N} a_{ij} \left\langle u_j - v_j^*, R_j (u_j - v_j^*) \right\rangle + V_i(\delta_i(0)). \end{aligned}$$

Because $\mathcal{L}_2$ stability must hold for all initial conditions, select $\delta_i(0)=0$. This implies $V_i(\delta_i(0))=0$. Let $u_i = u_i^*$ to obtain

$$\left\langle \delta_i, Q_i\delta_i \right\rangle + (d_i+g_i)\left\langle u_i, R_i u_i \right\rangle - \gamma^2 \sum_{j=1}^{N} a_{ij} \left\langle u_j, R_j u_j \right\rangle = -\gamma^2 \sum_{j=1}^{N} a_{ij} \left\langle u_j - v_j^*, R_j(u_j - v_j^*) \right\rangle \le 0$$

which implies

$$\left\langle \delta_i, Q_i\delta_i \right\rangle + (d_i+g_i)\left\langle u_i, R_i u_i \right\rangle \le \gamma^2 \sum_{j=1}^{N} a_{ij} \left\langle u_j, R_j u_j \right\rangle$$

Taking the square root of both sides of the inequality shows that (4.16) holds. □

The asymptotic stability of minmax strategies is studied in the following subsection.

### *4.3.2. Asymptotic stability*

It is now of our interest to determine the conditions for asymptotic stability of the global system, i.e., the simultaneous stability of the dynamics (2.5) for all agents $i=1,\ldots,N$. To analyze the influence of the neighbors of agent $i$ in the stability properties of the system, substitute the control policies of the form (4.9) in the error dynamics (2.5) to get

$$\dot{\delta}_i = \ A-(d_i+g_i)BR_i^{-1}B^TP_i \ \ \delta_i + \sum_{j=1}^{N} a_{ij}BR_j^{-1}B^TP_j\delta_j. \tag{4.17}$$

System (4.17) can be expressed in global form by defining the variable $\delta = \left[\delta_1^T,\cdots,\delta_N^T\right]^T$, such that

$$\dot{\delta} = \left[(I\otimes A)-(L\otimes B)\bar{R}^{-1}(I\otimes B)P\right]\delta \tag{4.18}$$

where $I$ is an identity matrix of appropriate dimensions, $\otimes$ represents the Kronecker product, $\bar{R}=\mathrm{diag}_i\{R_i\}$ and $P=\mathrm{diag}_i\{P_i\}$. Fig. 4.1 shows the block diagram of the feedback global system in a communication graph.

Two additional assumptions will be considered to complete the asymptotic stability analysis of the global system (4.18).

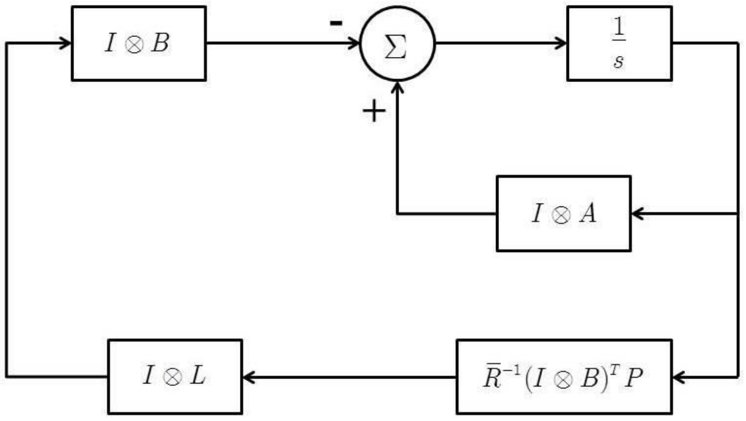

Fig. 4.1. Closed-loop multiagent system in graphs.

**Assumption 4.1.** The matrices $R_i$ in the performance indices (4.5) are selected such that $R_i = R_j = R$ for all agents $i$ and $j$.

**Assumption 4.2.** The graph weights $a_{ij}$ and $g_i$ have sufficiently small magnitudes for all paris $i, j$, such that

$$\lambda_{\min}(Q_i) \geq \left[\left(1 - \frac{1}{\gamma^2}\right)d_i + g_i\right]\lambda_{\max}\quad P_iBR^{-1}B^TP_i \tag{4.19}$$

with all matrices defined in the ARE (4.11), and where $\lambda_{\min}(\cdot)$ and $\lambda_{\max}(\cdot)$ are, respectively, the minimum and maximum eigenvalues of a matrix.

If Assumption 4.1 holds, then we can write $\bar{R}^{-1} = I \otimes R^{-1}$, and the global dynamics (4.18) can be expressed as

$$\dot{\delta} = \left[(I \otimes A) - (L \otimes BR^{-1}B^T)P\right]\delta. \tag{4.20}$$

Similarly, the ARE (4.11) can be written as

$$Q_i + P_iA + A^TP_i - \left[\left(1 - \frac{1}{\gamma^2}\right)d_i + g_i\right]P_iBR_i^{-1}B^TP_i. \tag{4.21}$$

The following lemmas are consequences of Assumptions 4.1 and 4.2, and will be used in our main proof of stability below.

**Lemma 4.1.** Let $L$ be the Laplacian matrix of a strongly connected graph, and define the matrix $W = \mathrm{diag}_i\{w_i\}$ where $w^T = \left[w_1, \cdots, w_N\right]$ is the left eigenvector of $L$ associated with the eigenvalue $0$. Let $R$ be a symmetric, positive definite matrix. Then,

$$WL \otimes BR^{-1}B^T + L^T W \otimes BR^{-1}B^T = S_1 \tag{4.22}$$

with $S_1 = S_1^T \geq 0$.

*Proof*. It is proven in [53] that $WL + L^T W = S \geq 0$ for strongly connected graphs.By properties of the Kronecker product [54] and the fact that $BR^{-1}B^T$ is a symmetric, positive semidefinite matrix, (4.22) holds. □

**Lemma 4.2.** If matrix $A$ has all its eigenvalues with non-positive real parts, matrix $P_i$ solves the ARE (4.21) and Assumption 4.2 holds, then

$$P_i A + A^T P_i = -S_2 \tag{4.23}$$

for some $S_2 = S_2^T \geq 0$.

*Proof*. Express the ARE (4.21) as

$$P_i A + A^T P_i = -Q_i + \left[\left(1 - \frac{1}{\gamma^2}\right) d_i + g_i\right] P_i B R_i^{-1} B^T P_i. \tag{4.24}$$

If (4.19) holds, it is clear that the right-hand side of (4.24) is a positive semidefinite matrix, which is a sufficient condition for (4.23). □

We are ready to prove that the minmax policies (4.9) make the global system (4.20) asymptotically stable. The following theorem is the main result in this section.

**Theorem 4.3.** Let the conditions in Theorem 4.1 hold, and make Assumptions 4.1 and 4.2. Furthermore, let the graph $\mathcal{G}$ have a spanning tree with the leader node as the root. Then, control policies (4.9) make the system (4.20) asymptotically stable.

*Proof*. The proof of this theorem is divided in two parts. First, we prove that asymptotic stability is achieved for strongly connected graphs. Then, we generalize the result for graphs with a spanning tree.

Thus, assume first that the graph $\mathcal{G}$ is strongly connected, and notice that Lemmas 4.1 and 4.2 hold from Assumptions 4.1 and 4.2. Premultiplying and postmultiplying both sides of the inequality (4.22) by $W^{-1} \otimes I$, we obtain

$$W^{-1}L^T \otimes BR^{-1}B^T + LW^{-1} \otimes BR^{-1}B^T = \bar{S}_1$$

where $\bar{S}_1 = (W^{-1} \otimes I)S_1(W^{-1} \otimes I)$. Following a similar procedure, from (4.23) we can obtain

$$w_i^{-1}AP_i^{-1} + w_i^{-1}P_i^{-1}A^T = \bar{S}_2$$

with $\bar{S}_2 = w_i^{-1}P_i^{-1}S_2P_i^{-1}$.

Define now the matrix $P_w = (W \otimes I)P = \mathrm{diag}_i\{w_iP_i\}$ with $w_i$ the $i$th element of the left eigenvector $w$ of $L$. Because the graph is strongly connected, $w_i > 0$ for all $i$ [16]. Now,

$$\begin{aligned}
&P_w^{-1}\left[(I \otimes A) - (L \otimes BR^{-1}B^T)P\right]^T + \left[(I \otimes A) - (L \otimes BR^{-1}B^T)P\right]P_w^{-1} \\
&= P_w^{-1}(I \otimes A^T) - (W^{-1} \otimes I)(L^T \otimes BR^{-1}B^T) + (I \otimes A)P_w^{-1} - (L \otimes BR^{-1}B^T)(W^{-1} \otimes I) \\
&= P_w^{-1}(I \otimes A^T) - (W^{-1}L^T \otimes BR^{-1}B^T) + (I \otimes A)P_w^{-1} - (LW^{-1} \otimes BR^{-1}B^T) \\
&= -\bar{S}_1 - \bar{S}_2 \leq 0
\end{aligned}$$

Notice that this is a Lyapunov equation and the system (4.20) is stable. Moreover, using LaSalle's extension and noticing that $\delta = 0$ only when synchronization has been achieved, we conclude that the system is asymptotically stable.

Consider now the case when the graph $\mathcal{G}$ has a spanning tree with the leader as a root. If $\mathcal{G}$ has a spanning tree but is not strongly connected, then the Laplacian matrix is reducible and can be expressed by means of a permutation transformation as the Forbenius form [13]

$$\begin{bmatrix} L_{11} & 0 & \cdots & 0 \\ L_{21} & L_{22} & \cdots & 0 \\ \vdots & & \ddots & \\ L_{M1} & L_{M2} & \cdots & L_{MM} \end{bmatrix}$$

where each submatrix $L_{kk}$ is irreducible. Irreducibility of matrix $L_{kk}$ implies that the subgraph connecting only the agents in the $k$ th block row of $L$ is strongly connected. This immediately shows, due to the first part of this proof, that the dynamics of the agents in the first block row are asymptotically stable.

We can now prove stability of the global system by induction. Assume that all agents in the block rows 1 to $k-1$ have stable dynamics. Thus, as $t \to \infty$, the influence of their local errors $\delta_j$ over the dynamics (4.17) of the agents in the $k$ th block row vanishes. This leaves only the strongly connected interaction of the agents in the $k$ th block row, which is proven to be stable. □

In the following section, we study the guaranteed robustness properties of minmax strategies.

## 4.4. Robustness Analysis for Minmax Strategies

In this section, we are particularly interested in determining the gain and phase margins of the agents provided by the minmax policies (4.9). Our approach to perform this analysis is to consider each individual agent using its minmax input (4.9) and determine how the neighbor policies, seen as a disturbance, affect its stability.

Let the perturbed version of the nominal system (4.13) be given by the dynamics

$$\dot{\hat{\delta}}_i = A\hat{\delta}_i + (d_i + g_i)(B\wp u_i) \tag{4.25}$$

where the disturbance $\wp$ is assumed to be a finite-gain operator with $\wp 0 = 0$, and $\hat{\delta}_i$ represents the state trajectories of the perturbed system. We let $\hat{\delta}_i(0) = \delta_i(0)$.

The subsequent robustness analysis follows a similar procedure as in [44]. The following lemma shows a sufficient condition on the disturbance $\wp$ that guarantees the stability of $\hat{\delta}_i$. Notice that guaranteeing the stability of $\hat{\delta}_i$ implies the stability of $\delta_i$.

**Lemma 4.3.** If the perturbation $\wp$ of the system (4.25) is such that

$$\left\langle u_i, [(d_i + g_i)(2\wp - I)R_i^{-1} + \frac{1}{\gamma^2}\sum\nolimits_{j=1}^{N} a_{ij} R_j^{-1}]u_i \right\rangle \geq 0$$

for all $u_i \in \mathcal{L}_2^m[0,\infty)$, then

$$\delta_i^T(0)P_i\delta_i(0) \geq \left\langle \hat{\delta}_i, Q_i\hat{\delta}_i \right\rangle. \tag{4.26}$$

If, additionally, $[A, \sqrt{Q_i}\,]$ is detectable, then $\hat{\delta}_i$ is asymptotically stable.

*Proof*. Using the definition of the perturbed system (4.25) and the ARE (4.21), we have for every $\tau$,

$$\begin{aligned}
\delta_i^T(0)P_i\delta_i(0) &= \hat{\delta}_i^T(\tau)P_i\hat{\delta}_i(\tau) - \int_0^\tau \frac{d}{dt}\ \hat{\delta}_i^T(t)P_i\hat{\delta}_i(t)\ dt \\
&\geq -2\left\langle \hat{\delta}_{i\tau}, P_i(A - (d_i + g_i)B\wp R_i^{-1}B^TP_i)\hat{\delta}_{i\tau} \right\rangle \\
&= \left\langle \hat{\delta}_{i\tau}, (-P_iA - A^TP_i)\hat{\delta}_{i\tau} \right\rangle + \left\langle \hat{\delta}_{i\tau}, 2(d_i + g_i)P_iB\wp R_i^{-1}B^TP_i\hat{\delta}_{i\tau} \right\rangle \\
&= \left\langle \hat{\delta}_{i\tau}, (d_i + g_i)P_iB(\wp - I)R_i^{-1}B^TP_i\hat{\delta}_{i\tau} \right\rangle + \left\langle \hat{\delta}_{i\tau}, \left(\frac{1}{\gamma^2}\sum\nolimits_{j=1}^{N} a_{ij}P_iB(\wp - I)R_i^{-1}B^TP_i + Q_i\right)\hat{\delta}_{i\tau} \right\rangle
\end{aligned}$$

where $\hat{\delta}_{i\tau}$ is the truncation of $\hat{\delta}_i$ as defined in (2.14). Define the expression $\Pi_i = (d_i + g_i)(2\wp - I)R_i^{-1} + (1/\gamma^2)\sum\nolimits_{j=1}^{N} a_{ij}R_j^{-1}$, and write

$$\begin{aligned}
\delta_i^T(0)P_i\delta_i(0) - \left\langle \hat{\delta}_{i\tau}, Q_i\hat{\delta}_{i\tau} \right\rangle &\geq \left\langle \hat{\delta}_{i\tau}, P_iB\Pi_iB^TP_i\hat{\delta}_{i\tau} \right\rangle \\
&= \left\langle B^TP_i\hat{\delta}_{i\tau}, \Pi_iB^TP_i\hat{\delta}_{i\tau} \right\rangle \\
&= \left\langle \bar{u}, \Pi_i\bar{u} \right\rangle \geq 0
\end{aligned}$$

where $\bar{u} = B^T P_i \hat{\delta}_{i\tau}$ and the last inequality holds by assumption. In the limit $\tau \to \infty$, it follows that $\delta_i^T(0) Q_i \delta_i(0) \geq \left\langle \hat{\delta}_i, Q_i \hat{\delta}_i \right\rangle$, which implies that $\left\langle \hat{\delta}_i, Q_i \hat{\delta}_i \right\rangle$ is bounded. As $[A, \sqrt{Q_i}\,]$ is detectable, $\hat{\delta}_i$ is square-integrable. Because $\wp$ has finite gain, so does the matrix $A - (d_i + g_i) B \wp R_i^{-1} B^T P_i$. From (4.25), we notice that $\dot{\hat{\delta}}_i$ is also square-integrable. Since both $\hat{\delta}_i$ and $\dot{\hat{\delta}}_i$ are square-integrable, $\hat{\delta}_i$ is asymptotically stable [44]. □

The following theorem establishes the result in Lemma 4.3 for the case when the disturbance $\wp$ is a linear operator.

**Theorem 4.4.** Let $\wp$ be a finite-gain linear time-invariant operator $\mathcal{H}$ with transfer matrix $H(s)$. If for all $\omega$

$$(d_i + g_i)\ \ H(j\omega) R_i^{-1} + R_i^{-1} H^*(j\omega) - R_i^{-1} \ \ + \frac{1}{\gamma^2} \sum_{j=1}^{N} a_{ij} R_j^{-1} \geq 0 \qquad (4.27)$$

and if $[A, \sqrt{Q_i}\,]$ is detectable, then the system (4.25) is asymptotically stable.

*Proof*. Expressing $\wp$ as a linear operator $\mathcal{H}$ and using Parseval's theorem, we get

$$\begin{aligned}
&\left\langle u_i, [(d_i + g_i)(2\wp - I) R_i^{-1} + (1/\gamma^2) \textstyle\sum_{j=1}^{N} a_{ij} R_j^{-1}] u_i \right\rangle \\
&= \left\langle u_i, [(d_i + g_i)(2\mathcal{H} - I) R_i^{-1} + (1/\gamma^2) \textstyle\sum_{j=1}^{N} a_{ij} R_j^{-1}] u_i \right\rangle \\
&= \frac{1}{2\pi} \int_{-\infty}^{\infty} \mathcal{U}_i^*(j\omega) [(d_i + g_i)(H(j\omega) R_i^{-1} + R_i^{-1} H^*(j\omega) - R_i^{-1}) + (1/\gamma^2) \textstyle\sum_{j=1}^{N} a_{ij} R_j^{-1}] \mathcal{U}(j\omega) d\omega \\
&\geq 0
\end{aligned}$$

where the last inequality holds by the assumption in (4.27). The proof is completed by Lemma 4.3. □

From Theorem 4.4 we derive our main robustness results. The following corollary holds for a particular selection of matrices $R_i$ and disturbance $\wp$.

**Corollary 4.1.** Let $R_i = \mathrm{diag}_k\{r_{i,k}\}$ for positive scalars $r_{i,k}$, and let the disturbance $\wp$ be such that

$$\wp u_i = \begin{bmatrix} \wp_1 u_{i,1} \\ \vdots \\ \wp_m u_{i,m} \end{bmatrix} \tag{4.28}$$

If each of the perturbations $\wp_k$ is linear time-invariant with proper transfer function $H_k(s)$, $\mathrm{Re}\{s_j\} < 0$ for all poles $s_j$ of $H_k(s)$, and

$$\mathrm{Re}\{H_k(j\omega)\} \geq \frac{1}{2}\left[1 - \frac{1}{(d_i + g_i)\gamma^2} \sum_{j=1}^{N} a_{ij} \frac{r_{i,k}}{r_{j,k}}\right] \tag{4.29}$$

for all $\omega$, then the system (4.25) is asymptotically stable.

*Proof*. $\mathrm{Re}\{s_j\} < 0$ assures that $\wp$ has finite gain. Take $H(s) = \mathrm{diag}_k\{H_k(s)\}$. Now,

$$\begin{aligned}
&(d_i + g_i)r_{i,k}^{-1}(H_k(j\omega) + H_k^*(j\omega) - 1) + \frac{1}{\gamma^2}\sum_{j=1}^{N} a_{ij} r_{j,k}^{-1} \\
&\qquad = 2(d_i + g_i)(r_{i,k}^{-1}\,\mathrm{Re}\{H_k(j\omega)\} - r_{i,k}^{-1}) + \frac{1}{\gamma^2}\sum_{j=1}^{N} a_{ij} r_{j,k}^{-1} \\
&\qquad \geq 2(d_i + g_i)r_{i,k}^{-1}\left(\frac{1}{2} - \frac{1}{2\gamma^2}\frac{1}{d_i + g_i}\sum_{j=1}^{N} a_{ij}\frac{r_{i,k}}{r_{j,k}}\right) - (d_i + g_i)r_{i,k}^{-1} + \frac{1}{\gamma^2}\sum_{j=1}^{N} a_{ij} r_{j,k}^{-1} \\
&\qquad = 0
\end{aligned}$$

As the conditions of Theorem 4.4 are satisfied, (4.25) is asymptotically stable. □

We can finally determine the phase and gain margins of minmax strategies by means of the following result, which follows from Corollary 4.1.

**Corollary 4.2.** Let the conditions of Corollary 4.1 hold. A phase shift $\phi_i$, with $|\phi_i| \leq 60^\circ + \theta$, where $\theta = \arccos(0.25(c + \sqrt{12 - 3c^2}))$ and $c = 1 - (d_i + g_i)^{-1}\gamma^{-2}\sum_{j=1}^{N} a_{ij} r_{i,k} r_{j,k}^{-1}$, in the respective feedback loops of each of the controls $u_i$ will leave an asymptotically stable system. Moreover, inserting a gain of $\alpha_k$ such that

$$\alpha_k \geq \frac{1}{2}\left[1 - \frac{1}{(d_i + g_i)\gamma^2}\sum_{j=1}^{N} a_{ij}\frac{r_{i,k}}{r_{j,k}}\right] \tag{4.30}$$

in the feedback loop of the controllers $u_{i,k}$, leaves the system asymptotically stable.

*Proof*. Expressing the complex number $H_k(j\omega)$ in polar form, it is clear from Corollary 4.1 that the condition for stability is

$$\cos\phi_k(\omega) \geq \frac{1}{2} - \frac{1}{2}(d_i + g_i)^{-1}\gamma^{-2}\sum_{j=1}^{N} a_{ij}\,\frac{r_{i,k}}{r_{j,k}}$$

or

$$\left|\phi_k(\omega)\right| \leq \arccos\left(\frac{1}{2} - \frac{1}{2}(d_i + g_i)^{-1}\gamma^{-2}\sum_{j=1}^{N} a_{ij}\,\frac{r_{i,k}}{r_{j,k}}\right)$$

Using trigonometric identities, we can express this result as

$$\left|\phi_k(\omega)\right| \leq \arccos\left(\frac{1}{2}\right) + \arccos\left(\frac{c + \sqrt{12 - 3c^2}}{4}\right)$$

with the constant $c$ defined in the corollary statement. This proves the minimum phase margin of minmax policies.

Furthermore, it is clear from Corollary 4.1 that if $H_k(j\omega)$ represents a scalar gain $\alpha_k$, then (4.30) guarantees stability of the system. □

**Remark 4.3.** Corollary 4.2 shows that the minmax strategies have infinite gain margin, gain reduction tolerance of more than 50% and a phase margin of more than 60°. The amount of additional phase delay and the amount of additional gain reduction tolerance depend on the selection of matrices $R_i$, parameter $\gamma$ and the graph topology. This is an improved result from the LQR robustness properties, which are known to have infinite gain margin, 50% gain reduction tolerance and 60° of phase margin.

## 4.5. Off-Policy Learning for Minmax Strategies

In this section, an off-policy RL algorithm is proposed to determine the solutions of the Riccati equations (4.11) and obtain the control policies (4.9) that solve the minmax strategies problem. This method is designed such that the agents learn their optimal policies using only data measured from their environment, without any knowledge of the system dynamics (2.2).

To design the off-policy RL algorithm, define the variables $u_i^k$ and $\upsilon_j^k$ as auxiliary control policies and express the system dynamics (2.5) as

$$\dot{\delta}_i = A\delta_i + (d_i + g_i)Bu_i^k - \sum_{j=1}^{N} a_{ij} B\upsilon_j^k + (d_i + g_i)B(u_i - u_i^k) - \sum_{j=1}^{N} a_{ij} B(u_j - \upsilon_j^k). \quad (4.31)$$

Here, the variables $u_i^k$ and $\upsilon_j^k$ are the policies to be updated. Notice here that the input $u_j$ corresponds to the actual policy employed by agent $j$, while $\upsilon_j^k$ is agent $i$'s estimation of the worst-case neighbor policy.

Let $V_i^k = \delta_i^T P_i^k \delta_i$ represent the value function $V_i$ at the $k$th iteration of our algorithm, and notice that the expression

$$V_i^k(\delta_i(t+T)) - V_i^k(\delta_i(t)) = \int_t^{t+T} \dot{V}_i^k(\delta_i) d\tau = 2\int_t^{t+T} \delta_i^T P_i^k \dot{\delta}_i d\tau \quad (4.32)$$

holds. Using the dynamics (4.31) in (4.32), we obtain

$$\begin{aligned} \frac{1}{2} V_i^k(\delta_i(t+T)) - \frac{1}{2} V_i^k(\delta_i(t)) &= \int_t^{t+T} \delta_i^T P_i^k \left[ A\delta_i + (d_i + g_i)Bu_i^k - \sum_{j=1}^{N} a_{ij} B\upsilon_j^k \right] d\tau \\ &+ (d_i + g_i)\int_t^{t+T} \delta_i^T P_i^k B(u_i - u_i^k) d\tau - \int_t^{t+T} \delta_i^T P_i^k \sum_{j=1}^{N} a_{ij} B(u_j - \upsilon_j^k) d\tau \end{aligned} \quad (4.33)$$

From the Hamiltonian (4.8), we can obtain the $k$th iteration Bellman equation as

$$\begin{aligned} 0 = \delta_i^T Q_i \delta_i + (d_i + g_i) u_i^{kT} R_i u_i^k - \gamma^2 \sum_{j=1}^{N} a_{ij} \upsilon_j^{kT} R_j \upsilon_j^k \\ + 2\delta_i^T P_i^k \left( A\delta_i + (d_i + g_i)Bu_i^k - \sum_{j=1}^{N} a_{ij} B\upsilon_j^k \right). \end{aligned} \quad (4.34)$$

Using (4.34) in (4.33) yields

$$\begin{aligned} V_i^k(\delta_i(t+T)) - V_i^k(\delta_i(t)) &= \int_t^{t+T} \left[ \delta_i^T Q_i \delta_i + (d_i + g_i) u_i^{kT} R_i u_i^k - \gamma^2 \sum_{j=1}^{N} a_{ij} \upsilon_j^{kT} R_j \upsilon_j^k \right] d\tau \\ &- 2(d_i + g_i)\int_t^{t+T} u_i^{k+1,T} R_i (u_i - u_i^k) d\tau + 2\int_t^{t+T} \sum_{j=1}^{N} a_{ij} \upsilon_j^{k+1,T} R_j (u_j - \upsilon_j^k) d\tau \end{aligned} \quad (4.35)$$

where we have also used the fact that the control policies $u_i^{k+1}$ and $\upsilon_j^{k+1}$ at iteration $k+1$ are defined as

$$u_i^{k+1} = -R_i^{-1} B^T P_i^k \delta_i \tag{4.36}$$

and

$$\upsilon_j^{k+1} = -\frac{1}{\gamma^2} R_j^{-1} B^T P_i^k \delta_i \tag{4.37}$$

respectively. Lemma 4.4 shows the equivalence between (4.35) and (4.34).

**Lemma 4.4.** The solution $V_i^k = \delta_i^T P_i^k \delta_i$ of (4.35) is equivalent to the solution of the Bellman equation (4.34) in the sense that matrix $P_i^k$ makes (4.34) hold.

*Proof*. Using (4.32), we can express (4.35) as

$$\int_t^{t+T} \left( \dot{V}_i^k(\delta_i) + 2(d_i + g_i) u_i^{k+1,T} R_i (u_i - u_i^k) - 2\sum_{j=1}^{N} a_{ij} \upsilon_j^{k+1,T} R_j (u_j - \upsilon_j^k) \right) d\tau$$
$$= -\int_t^{t+T} \left[ \delta_i^T Q_i \delta_i + (d_i + g_i) u_i^{kT} R_i u_i^k - \gamma^2 \sum_{j=1}^{N} a_{ij} \upsilon_j^{kT} R_j \upsilon_j^k \right] d\tau.$$

Let $\dot{V}_i^k = \nabla V_i^k \dot{\delta}_i$ and use (4.31), (4.36) and (4.37) to get

$$\int_t^{t+T} \nabla V_i^{k+1} \left[ A\delta_i + (d_i + g_i) B u_i^k - \sum_{j=1}^{N} a_{ij} B \upsilon_j^k \right] d\tau$$
$$= -\int_t^{t+T} \left[ \delta_i^T Q_i \delta_i + (d_i + g_i) u_i^{kT} R_i u_i^k - \gamma^2 \sum_{j=1}^{N} a_{ij} \upsilon_j^{kT} R_j \upsilon_j^k \right] d\tau.$$

which clearly holds if and only if $V_i^k = \delta_i^T P_i^k \delta_i$ satisfies (4.34). □

Now, the off-policy RL algorithm consists in solving (4.35) for $V_i^k$, $u_i^{k+1}$ and $\upsilon_j^{k+1}$ simultaneously. Finding such solutions does not require any knowledge about the system dynamics (2.2). A useful method to solve (4.35) using measured data along the system trajectories is described in [37].

Algorithm 4.1 presents the iterative procedure for each agent $i$ to determine its minmax policy (4.9).

```
Algorithm 4.1 (Off-Policy RL for Minmax Strategies)
    1: Select initial stabilizing control policies $u_i^0$ and $v_j^0$ for all $j \in \mathcal{N}_i$.
    2: Apply a fixed policy $u_i \neq u_i^k$ and collect the required system information at $M$ different sampling intervals.
    3: Use the information collected in Step 2 to solve the Bellman equation (4.35) for $V_i^k$, $u_i^{k+1}$ and $v_j^{k+1}$ for all $j \in \mathcal{N}_i$.
    4: Go to Step 2. On convergence, stop.
```

The following theorem shows the convergence properties of Algorithm 4.1.

**Theorem 4.5.** Algorithm 4.1 converges to the policies (4.9) - (4.10), where matrix $P_i$ solves the AREs (4.11).

*Proof*. Follows directly from Lemma 4.4 and the proof of convergence of the iterative procedure consisting of solving the Bellman equation (4.34) and updating the policies (4.36) - (4.37) presented in [55]. □

## 4.6. Simulation Results

A numerical example is here presented to test the validity of our theoretical results. Consider a set of 5 agents and one leader connected in a communication graph as shown in Fig. 4.2. If $j \in \mathcal{N}_i$, let $a_{ij} = 1$. Each agent is taken with linear dynamics given by (2.2), where

$$A = \begin{bmatrix} 1 & 2 \\ -2 & -1 \end{bmatrix}, \qquad B = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}.$$

The minmax performance indices of the agents are defined by (4.5) with $Q_1 = Q_3 = 2I$, $Q_4 = I$ and $Q_2 = Q_5 = 3I$, where $I$ is the identity matrix. Let all agents use the same values for $R = 2I$ and $\gamma = 2$.

Algorithm 4.1 is used to learn the solution of the Riccati equations (4.11). The resulting matrices $P_i$ are shown below.

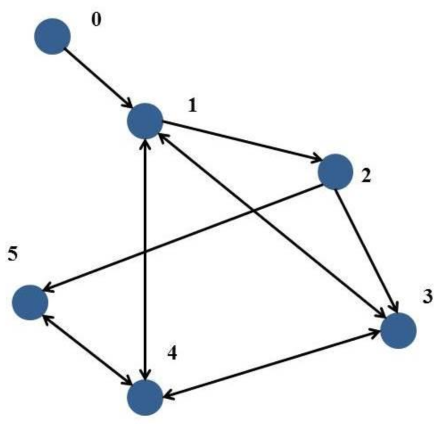

Fig. 4.2. Graph topology for simulation of minmax strategies.

$$P_1 = \begin{bmatrix} 0.5055 & 0.0314 \\ 0.0314 & 0.3537 \end{bmatrix}, \qquad P_2 = \begin{bmatrix} 1.1928 & 0.1314 \\ 0.1314 & 0.7597 \end{bmatrix},$$

$$P_3 = \begin{bmatrix} 0.6756 & 0.0612 \\ 0.0612 & 0.4441 \end{bmatrix}, \qquad P_4 = \begin{bmatrix} 0.4991 & 0.0707 \\ 0.0707 & 0.3068 \end{bmatrix},$$

$$P_5 = \begin{bmatrix} 0.8049 & 0.0542 \\ 0.0542 & 0.5558 \end{bmatrix}.$$

The minmax control policies are now given by (4.9) with the appropriate matrix $P_i$ for each agent. Using these policies, the agents successfully achieve synchronization with trajectories shown in Fig. 4.3. Figure 4.4 shows the trajectories of the agents also along a time axis.

## 4.7. Conclusion

Minmax strategies were designed and analyzed as an alternative solution concept for differential graphical games. The assumption made by each agent about the worst intentions of its neighbors yields robust control policies. Such policies are always distributed in the sense that the agents use only local information obtained from the graph topology.

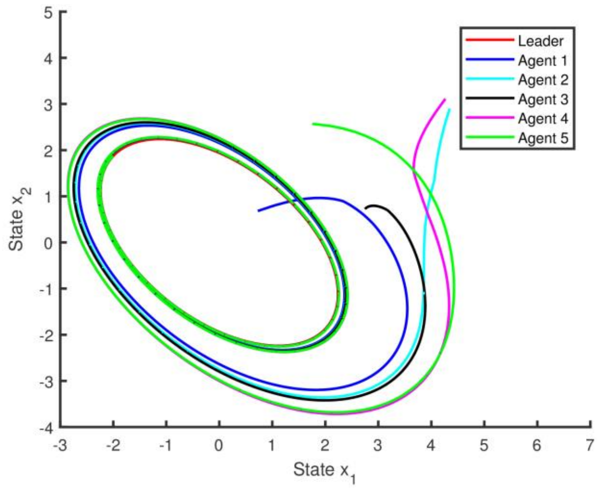

Fig. 4.3. State trajectories with minmax policies.

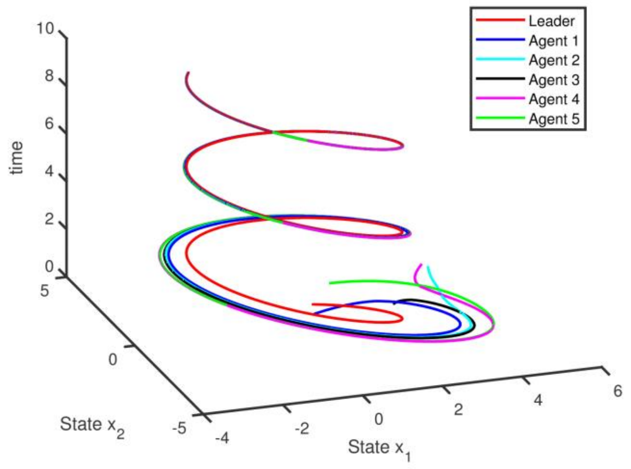

Fig. 4.4. Synchronization in time with minmax policies.

The proposed off-policy RL algorithm is a practical method to determine the minmax strategies of the agents without any knowledge about the system dynamics; moreover, this algorithm allows to compute the worst-case neighbor policy $\upsilon_j$ even when this is not the control policy used by the neighbors.

Despite its attractive features, the robustness properties of minmax may be too conservative for certain applications. For this reason, research about differential graphical games is continued by considering different solution concepts that still allow solutions using distributed input policies as described in subsequent chapters.

Chapter 5

# MULTIAGENT PURSUIT-EVASION GAMES: FINITE-TIME CAPTURE AND ASYMPTOTIC BEHAVIORS

## 5.1. Introduction

In the previous chapter, the minmax strategies problem was solved for multiagent graphical games. This chapter presents pursuit-evasion games as a specific set of applications on which minmax strategies is implemented.

Pursuit-evasion games are one of the most interesting and widely studied interactions in multi-agent systems. Their applications include aircraft control and missile guidance in military implementations, as well as in civilian applications such as collision avoidance designs in intelligent transportation systems, wireless sensors networks and sport game strategies. Animal behavior in hunting scenarios can also be studied using differential game analysis. Thus, the agents in pursuit-evasion games can be autonomous mobile robots, unmanned air vehicles, spacecraft, wireless sensors or living organisms.

The single-pursuer single-evader game has been studied in detail since R. Isaac's development of strategic policies for both players [4]. The single-pursuer single-evader case is a zero-sum game that can be solved using an extension of the well-known Bellman equations, known as the Hamilton-Jacobi-Isaacs (HJI) equations [5]. A closed-form solution of the game was obtained by Bryson and Ho, [56]. Extensions of these results have been obtained for the case of two pursuers versus one evader [57], and for the multiple-pursuer single-evader case [41], [58]-[61]. In [59], a pursuit-evasion game between aircraft is solved by differentiation of a particular value function. The discrete-time multiple-pursuer single-evader game is solved in [60]. In [7] a thorough survey about the studies on multiple-pursuer single-evader games is presented.

In recent years, modern interacting multiagent systems have motivated the study of general multiple-pursuer multiple-evader games. In [61], the pursuit-evasion game with two multi-agent teams was analyzed with a distributed hybrid approach using time potential fields. In [62] conditions to guarantee capture or evasion are determined by defining level-set functions as objectives for the players. Suboptimal strategies are studied in [63] by decomposing the game into multiple two-player games. The possibility of the players being connected in a communication graph is considered in [64], where the pursuit-evasion game is formulated as a connectivity control problem.

In the pursuit-evasion game, performance functions are used to represent the interest of the pursuers in reaching the evaders' position and the interest of the evaders in avoiding capture. These individual performance indices are then used to define Nash equilibrium and other solution concepts for the game [56], [1].

As it has been described in this dissertation, the mathematical techniques to analyze the behavior of multiagent systems with limited sensing capabilities have been developed in the field of graph theory. The most studied interaction for multi-agent systems control using graph theory is consensus [65]; some results in this chapter can be seen as a generalization of the consensus-seeking protocol. Multiagent systems controllers rely on the implementation of distributed algorithms on which each agent computes its feedback controller using only the state information it can perceive from its neighbors [13]. Stability analysis of multi-agent systems in graphs has been studied in [53], [66].

The general multiagent pursuit evasion (MPE) games present some interesting challenges in designing coordinated strategies among all the pursuers to accomplish the goal of capturing the evaders. This requires the development of strategies that combine the tools in both differential game theory and cooperative control theory to solve the distributed pursuit-evasion game. Cooperation among agents has been well studied [67], [68], and their extensions when considering a well-defined minimax performance index was recently developed using a differential game approach [20].

This chapter is organized as follows. The different graphs and variables that are used throughout the chapter are defined in Section 5.2. MPE games on directed graphs are formulated in Section 5.3, where Nash equilibrium and minmax strategies are studied as possible solutions of the games. In Section 5.4, an algorithm for target selection is proposed, and conditions for finite-time capture are stated. Asymptotic behaviors are analyzed in Section 5.5. Finally, the proposed control policies are tested via simulation studies in Section 5.6.

## 5.2.Definition of Graphs and System Variables

This section presents the basic notions and definitions required to analyze the MPE game on graphs. Although these definitions are based on the notation presented in Chapter 2, they differ by adding useful features to the variables used in this chapter.

### *5.2.1. Three graphs for MPE game interactions*

The distributed MPE game involves players with limited sensing range, that is, each agent is able to measure only the position of its closest neighbors. In this subsection, three different graphs are proposed to analyze the MPE game problem.

Define the graph $\mathcal{G}_p = (V_p, E_p)$ with the $N$ pursuers as the set of nodes $V_p = \{v_{p1}, \ldots, v_{pN}\}$ and a set of edges $E_p \subseteq V_p \times V_p$. Let $a_{ik}$ be the connectivity weights of graph $\mathcal{G}_p$, with $a_{ik} = 1$ if $(v_{pk}, v_{pi}) \in E_p$ and $a_{ik} = 0$ otherwise. Define the in-degree of pursuer $i$ as $d_i^p = \sum_{k=1}^N a_{ik}$ and the in-degree matrix of the graph $D_p = \mathrm{diag}\{d_i^p\}$. The weighted adjacency matrix is $\mathcal{A}_p = [a_{ik}]$. Finally, define the graph Laplacian matrix $L_p = D_p - \mathcal{A}_p$.

Similarly, the graph $\mathcal{G}_e = (V_e, E_e)$ represents the interactions among the $M$ evaders as the nodes $V_e = \{v_{e1}, \ldots, v_{eM}\}$. The graph weights are $b_{jl}$, with $b_{jl} = 1$ if $(v_{el}, v_{ej}) \in E_e$, and $b_{jl} = 0$ otherwise. The in-degree of evader $j$ is $d_j^e = \sum_{l=1}^M b_{jl}$. Define the matrices $D_e = \mathrm{diag}\{d_j^e\}$,

$\mathcal{A}_e = [b_{jl}]$ and $L_e = D_e - \mathcal{A}_e$ .

Finally, define the bipartite graph $\mathcal{G}_{pe} = (V_{pe}, E_{pe})$ that includes every player in the game, with the set of $N$ pursuers as one partition and the set of $M$ evaders as the other. Graph $\mathcal{G}_{pe}$ captures the information exchange among the agents in different partitions. Specifically, edge weights $c_{ij}$ represent the knowledge of pursuer $i$ about the position of evader $j$, with $c_{ij} = 1$ if pursuer $i$ can measure evader $j$ and $c_{ij} = 0$ otherwise. Similarly, $e_{ji}$ stands for the knowledge of evader $j$ about the position of pursuer $i$. The in-degree of pursuer $i$ in graph $\mathcal{G}_{pe}$ is defined as $d_i^{pe} = \sum_{j=1}^{M} c_{ij}$ , and the in-degree of evader $j$ is $d_j^{ep} = \sum_{i=1}^{N} e_{ji}$ . Define the pursuer-partition matrices $D_{pe} = \text{diag}\{d_i^{pe}\}$ and $\mathcal{A}_{pe} = [c_{ij}]$, and the evader-partition matrices $D_{ep} = \text{diag}\{d_j^{ep}\}$ and $\mathcal{A}_{ep} = [e_{ji}]$ .

### *5.2.2. Local errors and dynamics*

Consider a team of $N$ dynamical agents, regarded as the pursuers, each with linear dynamics

$$\dot{x}_i = Ax_i + Bu_i \tag{5.1}$$

$i = 1, \dots, N$ , where $x_i \in \mathbb{R}^n$ is the position of pursuer $i$ in the $n$-dimensional space of the game. Consider also a set of $M$ agents as the evader team, with dynamics

$$\dot{y}_j = Ay_j + Bv_j \tag{5.2}$$

$j = 1, \dots, M$ , where $y_j \in \mathbb{R}^n$ is the position of the $j$-th evader.

Similar to Chapter 2, the local error variable $\delta_i$ is defined as the position difference of pursuer $i$ with respect to its neighbors,

$$\delta_i = \sum_{k=1}^{N} a_{ik}(x_i - x_k) + \sum_{j=1}^{M} c_{ij}(x_i - y_j). \tag{5.3}$$

The error dynamics can be found from (5.3), using the system dynamics (5.1) - (5.2), as

$$\dot{\delta}_i = A\delta_i + (d_i^p + d_i^{pe})Bu_i - \sum_{k=1}^{N} a_{ik}Bu_k - \sum_{j=1}^{M} c_{ij}Bv_j. \tag{5.4}$$

Similarly, we define the local position error for evader $j$ as

$$\varepsilon_j = \kappa\sum_{l=1}^{M} b_{jl}(y_j - y_l) - \sum_{i=1}^{N} e_{ji}(y_j - x_i) \tag{5.5}$$

where $\kappa > 0$ is a scalar gain, and the local error dynamics are

$$\dot{\varepsilon}_j = A\varepsilon_j + (\kappa d_j^e - d_j^{ep})Bv_j - \kappa\sum_{l=1}^{M} b_{jl}Bv_l + \sum_{i=1}^{N} e_{ji}Bu_i. \tag{5.6}$$

The *center of gravity* of the agents is now defined to facilitate the analysis developed in this chapter. In particular, the center of gravity for pursuer $i$, $\bar{x}_{-i}$, is defined as the weighted average of the positions of pursuer $i$*'s neighbors*,

$$\bar{x}_{-i} = \frac{1}{d_i^p + d_i^{pe}}\left(\sum_{k=1}^{N} a_{ik}x_k + \sum_{j=1}^{M} c_{ij}y_j\right). \tag{5.7}$$

Then we can determine the center of gravity dynamics as

$$\begin{aligned}\dot{\bar{x}}_{-i} &= \frac{1}{d_i^p + d_i^{pe}}\left[\sum_{k=1}^{N} a_{ik}(Ax_k + Bu_k) + \sum_{j=1}^{M} c_{ij}(Ay_j + Bv_j)\right] \\ &= A\bar{x}_{-i} + B\bar{u}_{-i},\end{aligned}$$

where

$$\bar{u}_{-i} = \frac{1}{d_i^p + d_i^{pe}}\left(\sum_{k=1}^{N} a_{ik}u_k + \sum_{j=1}^{M} c_{ij}v_j\right). \tag{5.8}$$

We can now represent the local position error of pursuer $i$ (5.3), and its dynamics (5.4), in terms of the center of gravity as follows

$$\delta_i = (d_i^p + d_i^{pe})(x_i - \bar{x}_{-i}), \tag{5.9}$$

$$\dot{\delta}_i = A\delta_i + (d_i^p + d_i^{pe})Bu_i - (d_i^p + d_i^{pe})B\bar{u}_{-i}. \tag{5.10}$$

Similarly, the pseudo-center of gravity for evader $j$, $\bar{y}_{-j}$, is defined based on the neighbors of evader $j$ as

$$\bar{y}_{-j} = \frac{1}{kd_j^e - d_j^{ep}} \left( \kappa \sum_{l=1}^{M} b_{jl} y_l - \sum_{i=1}^{N} e_{ji} x_i \right) \tag{5.11}$$

with dynamics

$$\dot{\bar{y}}_{-j} = \frac{1}{\kappa d_j^e + d_j^{ep}} \left[ \sum_{l=1}^{M} b_{jl} (Ay_l + Bv_l) - \sum_{i=1}^{N} e_{ji} (Ax_i + Bu_i) \right]$$

$$= A\bar{y}_{-j} + B\bar{v}_{-j}$$

where

$$\bar{v}_{-j} = \frac{1}{\kappa d_j^e - d_j^{ep}} \left( \sum_{l=1}^{M} b_{jl} v_l - \sum_{i=1}^{N} e_{ji} u_i \right). \tag{5.12}$$

Using this definition, the local position error of evader $j$ (5.5) and its dynamics (5.6) can be written as

$$\varepsilon_j = (\kappa d_j^e - d_j^{ep})(y_j - \bar{y}_{-j}), \tag{5.13}$$

$$\dot{\varepsilon}_j = A\varepsilon_j + (\kappa d_j^e + d_j^{ep}) Bv_j - (\kappa d_j^e + d_j^{ep}) B\bar{v}_{-j}. \tag{5.14}$$

## 5.3.Formulation and Solutions for Multiagent Pursuit-Evasion Games

This section presents the formal definition of multiagent pursuit-evasion games. Conditions for asymptotic capture are analyzed. New definitions for Nash equilibrium and Minmax strategies as main solution concepts for the MPE games are also introduced.

### *5.3.1. Definitions for MPE games on graphs*

Consider the following formulation for MPE games. Let the evaders have the objective of maximizing their distances from their neighboring pursuers. Moreover, the evaders also desire to maintain their group cohesion, that is, to stay close to their teammates. The justification for this objective is that, in many practical applications, the agents of a team may want to perform their tasks without losing contact with each other. Similarly, the pursuers desire to collectively intercept as many evaders as possible while remaining close to their teammates. In Section 5.4, we explore a different scenario where the agents do not have the group cohesion objective.

The goals of each pursuer can be represented by means of a scalar function $J_{pi}(\delta_i, u_i)$, regarded as the performance index of the game for pursuer $i$. The performance index of pursuer $i$ can now be defined as

$$J_{pi} = \int_0^\infty \left[ \delta_i^T Q_{pi}(\delta_i)\delta_i + u_i^T R_p(\delta_i) u_i \right] dt \tag{5.15}$$

where $Q_{pi}(\delta_i) = Q_{pi}^T(\delta_i) > 0$ and $R_p(\delta_i) = R_p^T(\delta_i) > 0$.

Pursuer $i$ is thus concerned with the minimization of $J_{pi}$. From (5.9), the dependence of $J_{pi}$ on the local position errors $\delta_i$ can be seen as the goal of pursuer $i$ to minimize its distance with respect to the center of gravity of all its neighbors.

On the other side, evader $j$ desires to minimize its cost represented by the performance index $J_{ej}(\varepsilon_j, v_j)$. Considering its goals of fleeing from the pursuers while remaining close to the evaders in its neighborhood, the performance index for evader $j$ can be defined as

$$J_{ej} = \int_0^\infty \left[ \varepsilon_j^T Q_{ej}(\varepsilon_j)\varepsilon_j + v_j^T R_e(\varepsilon_j) v_j \right] dt \tag{5.16}$$

where $Q_{ej}(\varepsilon_j) = Q_{ej}^T(\varepsilon_j) > 0$, $R_e(\varepsilon_j) = R_e^T(\varepsilon_j) > 0$ and $\varepsilon_j$ is expressed in (5.13). Notice that the pseudo-center of gravity (5.11), used in (5.13), considers the opposite sign of the relative position of the pursuers such that minimization of the errors $\varepsilon_j$ implies escaping from the pursuers, as desired.

The scalar $\kappa$ in (5.5) can now be seen as the priority of the evaders to stay close to each other, against their drive to escape from the pursuers. The value of $\kappa$ can be selected according to the objectives of the game.

Using these definitions, the MPE differential games on communication graphs is defined as follows.

**Definition 5.1** (**Multi-agent pursuit-evasion game).** Define the MPE game as

$$\begin{aligned} V_{pi}(\delta_i) &= \min_{u_i} J_{pi}(\delta_i, u_i) \\ V_{ej}(\varepsilon_j) &= \min_{v_j} J_{ej}(\varepsilon_j, v_j) \end{aligned} \tag{5.17}$$

where $V_{pi}$ and $V_{ej}$ are the values of the game for pursuer $i$ and for evader $j$, respectively.

Pursuers and evaders must now determine the control policies $u_i$ and $v_j$, respectively, that solve the MPE game (5.17).

To state the following definition, notice that the performance index of an agent depends not only on its own behavior, but also on the behavior of its neighbors. Let $u_{-i}$ be the set of control policies of the pursuers neighbors of pursuer $i$, and $v_{-i}$ represent the policies of all the evaders neighbors of $i$. Thus, although we have defined functions $J_{pi}(\delta_i, u_i)$ and $J_{ej}(\varepsilon_j, v_j)$ in terms of the local variables, we can represent explicitly their dependence on neighbor policies as $J_{pi}(\delta_i, u_i) = J_{pi}(\delta_i, u_i, u_{-i}, v_{-i})$ for the pursuers, and $J_{ej}(\varepsilon_j, v_j) = J_{ej}(\varepsilon_j, v_j, u_{-j}, v_{-j})$ for the evaders. Using these expressions, Nash equilibrium for MPE games is defined as follows.

**Definition 5.2. (Nash equilibrium in MPE games).** Control policies $u_i$, $i = 1, \ldots, N$, and $v_j$, $j = 1, \ldots, M$, form a Nash equilibrium if the inequalities

$$\begin{aligned} J_{pi}\left(\delta_i, u_i^*, u_{-i}^*, v_{-i}^*\right) &\le J_{pi}\left(\delta_i, u_i, u_{-i}^*, v_{-i}^*\right), \\ J_{ej}\left(\varepsilon_j, v_j^*, u_{-j}^*, v_{-j}^*\right) &\le J_{ej}\left(\varepsilon_j, v_j, u_{-j}^*, v_{-j}^*\right), \end{aligned}$$

hold for all agents in the game.

In Nash equilibrium, every agent uses its optimal policy against the optimal policy of its neighbors. In the following subsection, Nash equilibrium is studied as the most important solution concept for the MPE games (5.17).

*5.3.2. Nash equilibrium solution for MPE games*

Given neighbor policies $u_k$ and $v_j$, the optimal control policy of pursuer $i$ can be obtained by means of the $i$th Hamiltonian function defined as

$$\begin{aligned} H_{pi} &= \delta_i^T Q_{pi}\delta_i + u_i^T R_p u_i + \dot{V}_{pi}(\delta_i) \\ &= \delta_i^T Q_{pi}\delta_i + u_i^T R_p u_i + \nabla V_{pi}^T(\delta_i)\left( A\delta_i + (d_i^p + d_i^{pe})Bu_i - \sum_{k=1}^{N} a_{ik} Bu_k - \sum_{j=1}^{M} c_{ij} Bv_j \right) \end{aligned}$$

where $V_{pi}(\delta_i)$ is the value function of the game for pursuer $i$, and the dynamics (5.4) has been used. The optimal policy of pursuer $i$ minimizes $H_{pi}$, and is found to be

$$u_i^* = -\frac{1}{2}(d_i^p + d_i^{pe})R_p^{-1}(\delta_i)B^T \nabla V_{pi}(\delta_i) \qquad (5.18)$$

Using the same procedure, define the Hamiltonian function for evader $j$ as

$$\begin{aligned} H_{ej} &= \varepsilon_j^T Q_{ej}\varepsilon_j + v_j^T R_e v_j + \dot{V}_{ej}(\varepsilon_j) \\ &= \varepsilon_j^T Q_{ej}\varepsilon_j + v_j^T R_e v_j + \nabla V_{ej}^T(\varepsilon_j)\left( A\varepsilon_j + (\kappa d_j^e - d_j^{ep})Bv_j - \kappa\sum_{l=1}^{M} b_{jl} Bv_l + \sum_{i=1}^{N} e_{ji} Bu_i \right) \end{aligned}$$

and the optimal policy for evader $j$ is given by

$$v_j^* = -\frac{1}{2}(\kappa d_j^e - d_j^{ep})R_e^{-1}(\varepsilon_j)B^T \nabla V_{ej}(\varepsilon_j) \qquad (5.19)$$

The functions $V_{pi}$ and $V_{ej}$ are obtained as the solutions of the coupled HJI equations of the game,

$$\delta_i^T Q_{pi}\delta_i + u_i^{*T} R_p u_i^* + \nabla V_{pi}^T\left( A\delta_i + (d_i^p + d_i^{pe})Bu_i^* - \sum_{k=1}^{N} a_{ik} Bu_k^* - \sum_{j=1}^{M} c_{ij} Bv_j^* \right) = 0 \qquad (5.20)$$

and

$$\varepsilon_j^T Q_{ej}\varepsilon_j + v_j^{*T} R_e v_j^* + \nabla V_{ej}^T\left( A\varepsilon_j + (\kappa d_j^e - d_j^{ep})Bv_j^* - \kappa\sum_{l=1}^{M} b_{jl} Bv_l^* + \sum_{i=1}^{N} e_{ji} Bu_i^* \right) = 0 \qquad (5.21)$$

for $i = 1,\ldots,N$, $j = 1,\ldots,M$. Substituting control policies (5.18) and (5.19) in (5.20) and (5.21), we obtain

$$\nabla V_{pi}^T A_{pi}^{cl} + \delta_i^T Q_{pi}\delta_i + \frac{1}{2}\sum_{k=1}^{N} a_{ik}(d_k^p + d_k^{pe})\nabla V_{pi}^T B R_p^{-1} B^T \nabla V_{pk} + \frac{1}{2}\sum_{j=1}^{M} c_{ij}(\kappa d_j^e - d_j^{ep})\nabla V_{pi}^T B R_e^{-1} B^T \nabla V_{ej} = 0 \qquad (5.22)$$

and

$$\nabla V_{ej}^T A_{ej}^{cl} + \varepsilon_j^T Q_{ej}\varepsilon_j + \frac{\kappa}{2}\sum_{l=1}^{M} b_{jl}(\kappa d_l^e - d_l^{ep})\nabla V_{ej}^T B R_e^{-1} B^T \nabla V_{el} - \frac{1}{2}\sum_{i=1}^{N} e_{ji}(d_i^p + d_i^{pe})\nabla V_{ej}^T B R_p^{-1} B^T \nabla V_{pi} = 0 \qquad (5.23)$$

respectively, where the closed loop matrices are

$$A_{pi}^{cl} = A\delta_i - \frac{1}{4}(d_i^p + d_i^{pe})^2 B R_p^{-1} B^T \nabla V_{pi}$$

and

$$A_{ej}^{cl} = A\varepsilon_j - \frac{1}{4}(\kappa d_j^e - d_j^{ep})^2 B R_e^{-1} B^T \nabla V_{ej}$$

The following theorem states the conditions in the MPE game to achieve Nash equilibrium as defined in Definition 5.2.

***Theorem 5.1.*** Let the pursuers with dynamics (5.1) and the evaders with dynamics (5.2) be connected in a communication graph topology with local error dynamics (5.4) and (5.6). Let (5.18) and (5.19) be the control policies for pursuers *i* and evaders *j*, respectively, where the functions $V_{pi}$ and $V_{ej}$ are the solutions of the HJI equations (5.22) - (5.23), such that $V_{pi}(0) = V_{ej}(0) = 0$. Then, capture occurs in the MPE game (5.17) in the sense that dynamics (5.4) are stable. Moreover, game (5.17) is in Nash equilibrium, the value of the game for pursuer $i$ is given by $V_{pi}(\delta_i(0))$, and the value of the game for evader $j$ is $V_{ej}(\varepsilon_j(0))$.

*Proof*. To prove capture, select the function $V_{pi}(\delta_i(t))$ as a Lyapunov function candidate. Its derivative is given by

$$\begin{aligned}\dot{V}_{pi} &= \nabla V_{pi}^T \dot{\delta}_i \\ &= \nabla V_{pi}^T \left( A\delta_i + (d_i^p + d_i^{pe}) B u_i^* - \sum_{k=1}^{N} a_{ik} B u_k^* - \sum_{j=1}^{M} c_{ij} B v_j^* \right) \\ &= -\delta_i^T Q_{pi} \delta_i - u_i^{*T} R_p u_i^*\end{aligned}$$

because equation (5.20) holds. Thus, we conclude that dynamics (5.4) is stable and every pursuer captures its neighbors.

To prove Nash equilibrium, notice that we can write the performance index (5.15) as

$$J_{pi} = \int_0^\infty \left[ \delta_i^T Q_{pi}(\delta_i)\delta_i + u_i^T R_p(\delta_i) u_i \right] dt + V_{pi}(\delta_i(0)) + \int_0^\infty \dot{V}_{pi}(\delta_i(t)) dt$$

because $V_{pi}(\delta_i(\infty)) = V_{pi}(0) = 0$. For convenience we omit the explicit dependence of matrices $Q_{pi}$ and $R_p$ on the local errors $\delta_i$ in the remaining procedure. Notice that $\dot{V}_{pi}(\delta_i) = \nabla V_{pi}^T \dot{\delta}_i$ for any trajectory of $\delta_i(t)$. Thus, using (5.4) we get

$$\begin{aligned}J_{pi} = &\int_0^\infty \left[ \delta_i^T Q_{pi} \delta_i + u_i^T R_p u_i \right] dt + V_{pi}(\delta_i(0)) \\ &+ \int_0^\infty \nabla V_{pi}^T \left( A\delta_i + (d_i^p + d_i^{pe}) B u_i - \sum_{k=1}^{N} a_{ik} B u_k - \sum_{j=1}^{M} c_{ij} B v_j \right) dt\end{aligned}$$

Completing the squares, and using the fact that $(d_i^p + d_i^{pe}) \nabla V_{pi}^T B u_i = -2u_i^{*T} R_p u_i$ with $u_i^*$ in (5.18), yields

$$\begin{aligned}J_{pi} &= \int_0^\infty \left[ \delta_i^T Q_{pi} \delta_i + u_i^T R_p u_i + u_i^{*T} R_p u_i^* - u_i^{*T} R_p u_i^* \right] dt \\ &\quad + V_{pi}(\delta_i(0)) + \int_0^\infty \nabla V_{pi}^T \left( A\delta_i - \sum_{k=1}^{N} a_{ik} B u_k - \sum_{j=1}^{M} c_{ij} B v_j \right) dt - 2\int_0^\infty u_i^{*T} R_p u_i \, dt \\ &= \int_0^\infty \Bigg[ \delta_i^T Q_{pi} \delta_i + (u_i - u_i^*)^T R_p (u_i - u_i^*) - u_i^{*T} R_p u_i^* \\ &\quad + \nabla V_{pi}^T \left( A\delta_i - \sum_{k=1}^{N} a_{ik} B u_k - \sum_{j=1}^{M} c_{ij} B v_j \right) \Bigg] dt + V_{pi}(\delta_i(0))\end{aligned}$$

As $-u_i^{*T} R_p u_i^* = -2u_i^{*T} R_p u_i^* + u_i^{*T} R_p u_i^* = (d_i^p + d_i^{pe}) \nabla V_{pi}^T B u_i^* + \; u_i^{*T} R_p u_i^*$, we finally get

$$J_{pi} = \int_0^\infty \left[ \delta_i^T Q_{pi} \delta_i + u_i^{*T} R_p u_i^* + \nabla V_{pi}^T \left( A\delta_i + (d_i^p + d_i^{pe}) B u_i^* - \sum_{k=1}^{N} a_{ik} B u_k - \sum_{j=1}^{M} c_{ij} B v_j \right) \right] dt$$
$$+ \int_0^\infty (u_i - u_i^*)^T R_p (u_i - u_i^*) dt + V_{pi}(\delta_i(0))$$

As equation (5.20) holds, the first integral in this expression is equal to zero for neighbor policies $u_k = u_k^*$ and $v_j = v_j^*$. The remaining expression,

$$J_{pi} = \int_0^\infty (u_i - u_i^*)^T R_p (u_i - u_i^*) dt + V_{pi}(\delta_i(0))$$

shows that control policy $u_i^*$ minimizes the performance function of pursuer $i$ against neighbor policies $u_k^*$ and $v_j^*$, and its value of the game is $V_{pi}(\delta_i(0))$.

The same procedure can be performed to show that

$$J_{ej} = \int_0^\infty (v_j - v_j^*)^T R_e (v_j - v_j^*) dt + V_{ej}(\varepsilon_j(0))$$

for evader $j$ and, therefore, control policy $v_j^*$ minimizes $J_{ej}$. As these conditions hold for all agents in the game, Nash equilibrium is achieved. □

**Remark 5.1.** Notice that if there exist matrices $P_{pi}$ and $P_{ej}$, for all $i$ and $j$ in the game, such that the value functions $V_{pi}$ and $V_{ej}$ have the form

$$V_{pi}(\delta_i) = \delta_i^T P_{pi} \delta_i \tag{5.24}$$

and

$$V_{ej}(\varepsilon_j) = \varepsilon_j^T P_{ej} \varepsilon_j \tag{5.25}$$

then the control policies (5.18) and (5.19) take the form

$$u_i^* = -(d_i^p + d_i^{pe}) R_p^{-1}(\delta_i) B^T P_{pi} \delta_i \tag{5.26}$$

and

$$v_j^* = -(\kappa d_j^e - d_j^{ep}) R_e^{-1}(\varepsilon_j) B^T P_{ej} \varepsilon_j \tag{5.27}$$

respectively. In (5.26) and (5.27) the distributed property of the control policies is clear.

*5.3.3. Minmax strategies in MPE games*

It has been explained in previous chapters that, in the general case, there may not exist a set of functions $V_{pi}(\delta_i)$ and $V_{ej}(\varepsilon_j)$ that solve the HJI equations (5.22) - (5.23) to provide distributed control policies as in (5.26) - (5.27). In this subsection we use the minmax strategies developed in Chapter 4. The following definition states the concept of minmax strategy for MPE games.

***Definition 5.3 (Minmax strategies for MPE games).*** In an MPE game, the minmax strategy of pursuer $i$ is given by

$$u_i^* = \arg\min_{u_i} \max_{u_{-i}, v_{-i}} J_{pi} \tag{5.28}$$

and the minmax strategy for evader $j$ is

$$v_j^* = \arg\min_{v_j} \max_{u_{-j}, v_{-j}} J_{ej}. \tag{5.29}$$

To determine the minmax strategy for pursuer $i$, we can redefine the performance index (5.15) and formulate a zero-sum game between agent $i$ and a virtual target. Considering the center of gravity (5.7) as the target of agent $i$, we can define the performance index

$$J_{pi} = \int_0^\infty \left[ \delta_i^T Q_{pi} \delta_i + u_i^T R_p u_i - \bar{u}_{-i}^T R_{-i} \bar{u}_{-i} \right] dt \tag{5.30}$$

In order to define a meaningful weighting matrix $R_{-i}$ in (5.30), consider the expressions (5.8) and (5.18), and select

$$R_{-i}^{-1} = \frac{1}{d_i^p + d_i^{pe}} \left( \sum_{k=1}^{N} a_{ik} R_p^{-1} + \sum_{j=1}^{M} c_{ij} R_e^{-1} \right) \tag{5.31}$$

The solution of the zero-sum game is now determined by (5.18), where the value function $V_{pi}$ is the solution of the HJI equation

$$\nabla V_{pi}^T A \delta_i + \delta_i^T Q_{pi} \delta_i - (d_i^p + d_i^{pe}) \nabla V_{pi}^T B R_p^{-1} B^T \nabla V_{pi} + (d_i^p + d_i^{pe}) \nabla V_{pi}^T B R_{-i}^{-1} B^T \nabla V_{pi} = 0 \tag{5.32}$$

If the value function has a quadratic form as in (5.24), then the control policy is expressed as in (5.26) and matrix $P_{pi}$ is the solution of the Riccati equation

$$Q_{pi} + P_{pi}A + A^T P_{pi} - (d_i^p + d_i^{pe})^2 P_{pi} B (R_p^{-1} - R_{-i}^{-1}) B^T P_{pi} = 0 \tag{5.33}$$

Similarly, define the zero-sum game

$$J_{ej} = \int_0^\infty \left[ \varepsilon_j^T Q_{ej} \varepsilon_j + v_j^T R_e v_j - \bar{v}_{-j}^T R_{-j} \bar{v}_{-j} \right] dt \tag{5.34}$$

with

$$R_{-j}^{-1} = \frac{1}{d_j^e + d_j^{ep}} \left( \sum_{l=1}^{M} b_{jl} R_e^{-1} + \sum_{i=1}^{N} e_{ji} R_p^{-1} \right). \tag{5.35}$$

Assuming a quadratic value function (5.25), the minmax strategy for evader $j$ is the control policy (5.27) where matrix $P_{ej}$ solves the ARE

$$Q_{ej} + P_{ej}A + A^T P_{ej} - (\kappa d_j^e - d_j^{ep})^2 P_{ej} B (R_e^{-1} - R_{-j}^{-1}) B^T P_{ej} = 0 \tag{5.36}$$

The following theorem formalizes these results.

**Theorem 5.2.** Let the agents with dynamics (5.1) and (5.2) use control policies (5.26) and (5.27), respectively. Moreover, let matrices $P_{pi}$ and $P_{ej}$ be the solutions of the Riccati equations (5.33) and (5.36). Then, policy (5.26) is the minmax strategy of pursuer $i$ as defined in (5.28), and policy (5.27) is the minmax strategy of evader $j$ as in (5.29).

*Proof*. The Hamiltonian function associated with the performance index (5.30) is $H_{pi} = \delta_i^T Q_{pi} \delta_i + u_i^T R_p u_i - \bar{u}_{-i}^T R_{-i} \bar{u}_{-i} + \dot{V}_{pi}$. For a quadratic value function (5.24), the optimal control policy for the pursuer is (5.26) and for its target is $\bar{v}_{-i}^* = -(d_i^p + d_i^{pe}) R_{-i}^{-1} B^T P_{pi} \delta_i$. Substituting these control policies in $H_{pi}$ and equating to zero, we obtain the ARE (5.33). Following a similar procedure as in the proof of Theorem 5.1, and considering dynamics (5.10), we can complete the squares and write the performance index (5.30) as

$$\begin{aligned} J_{pi} &= \int_0^\infty \left[ \delta_i^T Q_{pi} \delta_i + u_i^T R_p u_i - \bar{u}_{-i}^T R_{-i} \bar{u}_{-i} \right] dt + V_{pi}(\delta_i(0)) \\ &\quad + \int_0^\infty \nabla V_{pi}^T \left( A\delta_i + (d_i^p + d_i^{pe}) B u_i - (d_i^p + d_i^{pe}) B \bar{u}_{-i} \right) dt \\ &= \int_0^\infty \left[ (u_i - u_i^*)^T R_p (u_i - u_i^*) - (\bar{u}_{-i} - \bar{u}_{-i}^*)^T R_{-i} (\bar{u}_{-i} - \bar{u}_{-i}^*) \right] dt + V_{pi}(\delta_i(0)) \end{aligned}$$

Therefore, (5.26) with $P_{pi}$ as in (5.33) is the minmax strategy of pursuer $i$. The same procedure can be used to prove that control policy (5.27) with $P_{ej}$ as the solution of equation (5.36) is the minmax strategy for evader $j$. □

**Remark 5.2.** Minmax strategies provide distributed control policies for the agents as long as there exist positive definite solutions $P_{pi}$ and $P_{ej}$ for the equations (5.33) and (5.36), respectively. Riccati equations of this form are known to have positive definite solutions if $(A,B)$ is stabilizable, $(A,\sqrt{Q_{pi}})$ and $(A,\sqrt{Q_{ej}})$ are observable, and $R_p^{-1} - R_{-i}^{-1} > 0$ and $R_e^{-1} - R_{-j}^{-1} > 0$.

## 5.4.Target Selection and Finite-Time Capture

In the previous section, the evaders had the objective of maintaining their group cohesion while escaping from the pursuers. In this section, we modify the formulation of the game to obtain different behaviors among the agents. Let the evaders have the objective of increasing their distances with respect to each other, to force the pursuers to separate as well. Now, each pursuer must select a single target among the evaders in its neighborhood. Moreover, the control strategies of the agents are designed to achieve capture in finite-time.

### *5.4.1. Target selection by the pursuers*

Consider a game on which each pursuer can initially observe several evaders according to the communication graph topology. In most real-world applications, it can be impractical to expect each pursuer to chase many evaders simultaneously. Instead, each pursuer can select only one target among the evaders, disregarding the position of all other agents. If each pursuer targets a different evader, their objective of capturing as many evaders as possible is fulfilled.

In this section, we assume that the numbers of pursuers and evaders are the same, i.e.,

$N = M$. If there were more evaders, some of them would be able to escape unchallenged and therefore are not part of this analysis. If there were more pursuers, some of them would need to chase already-targeted evaders and their participation would be trivial.

Algorithm 5.1 presents a procedure for pursuer $i$ to select a target among the evaders. Initially, the graph weights for pursuer $i$ are set as $a_{ik} \geq 0$ and $c_{ij} \geq 0$. Algorithm 5.1 relies on setting $a_{ik} = 0$ for all pursuers *k*, and $c_{ij} = 0$ for all evaders but one. This one evader becomes the target of pursuer $i$.

In the ideal scenario, every pursuer targets its closest evader. If two pursuers have the same closest evader, additional criteria is needed to decide which pursuer will change its target. We propose to change the target of the pursuer with shortest distance to its second-closest evader. Thus, long distances between a pursuer and its target are avoided. Algorithm 5.1 generalizes this idea using an iterative procedure that eliminates the longest distances between the agents, until each pursuer is left with only one target.

Pursuer $i$ is assumed to be unaware of the initial topologies in graphs $\mathcal{G}_p$ and $\mathcal{G}_{pe}$ except for the links that connect it to its neighbors. Thus, to perform Algorithm 5.1 pursuer $i$ must assume that its neighbors have the same state information as itself. Pursuer $i$ can define *virtual* (or assumed) connectivity weights for its neighbors as $\bar{a}_{kh} = 1$ if $k, h \in \mathcal{N}_i$ for pursuers $k$ and $h$, and $\bar{c}_{kj} = 1$ if $k, j \in \mathcal{N}_i$ for pursuer $k$ and evader $j$. In the following procedure, all agents are pursuer $i$'s neighbors.

Define $\rho_{ij} > 0$ as the distance between pursuer $i$ and a neighboring evader $j$, i.e., $\rho_{ij} = c_{ij} \left\| x_i - y_j \right\|$. Define also $\rho_{kj} = \bar{c}_{kj} \left\| x_k - y_j \right\|$ for all pursuers and evaders $k, j \in \mathcal{N}_i$. Let $\rho_{\max,i}$ be the longest distance between any pursuer $k \in i \cup \mathcal{N}_i$ and its possible targets, that is, $\rho_{\max,i} = \max\limits_{k,j}(\rho_{kj})$ for $k \in i \cup \mathcal{N}_i$, $j \in \mathcal{N}_i$. With these definitions Algorithm 5.1 can now be used for pursuer $i$ to select its target.

**Algorithm 5.1 (Target selection by the pursuers)**

```
1:  If $c_{ij} = 1$ for only one evader $j$, stop.
2:  Determine the pursuer $k$ and evader $j$ such that $\rho_{\max,i} = \rho_{kj}$,
    and set $\bar{c}_{kj} = 0$.
3:  If for some pursuer $k$, $\bar{c}_{kl} = 1$ for only one evader $l$, set $\bar{c}_{hl} = 0$
    for all pursuers $h \neq k$ and $c_{il} = 0$ if $i \neq k$.
4:  Go to Step 1.
```

In Algorithm 5.1, pursuer $i$ assigns targets to its teammates in order to discard those evaders for itself in benefit of the collective goals of the team. Notice that these virtual assignments may not be correspond to the actual selection of targets of the other pursuers. Step 2 in Algorithm 5.1 indicates that, as long as there are other options, the longest distance between a pursuer and an evader must be avoided. Step 3 expresses that once a pursuer targets an evader, the other pursuers discard that evader as a possible objective. The following theorem shows that if every agent in the game is each other neighbors, then target selection by Algorithm 5.1 minimizes the longest distance between a pursuer and its target.

***Theorem 5.3.*** Consider the MPE game where every pursuer selects its target using Algorithm 5.1. If the graphs $\mathcal{G}_p$ and $\mathcal{G}_{pe}$ are complete, then every pursuer selects a different target. Moreover, this selection of targets minimizes the longest distance between a pursuer and its target.

*Proof.* If the graph topologies are complete, then all agents possess the same state information and each pursuer selects precisely the same target that was assigned to it by all of its neighbors. By the construction of Algorithm 5.1, each pursuer selects a unique target.

Consider now the pursuer $i$ that travels the longest distance to capture its target. Let $i$ change its selection of target such that its distance to travel is reduced. Pursuer $i$ can only have discarded this closer target in Algorithm 5.1 because it was the only option for another pursuer $k$ to chase. Pursuer $k$ had selected its target either because its other options had the longest distances $\rho_{\max}$, or because other pursuers had previously selected those evaders. This consecutive change of targets must eventually lead to a pursuer left to chase an evader with a travel distance longer than the original longest distance produced by Algorithm 5.1. □

Using Algorithm 5.1, every pursuer modifies its perception of the environment such that its local error measurement focuses only on its target, i.e., if pursuer $i$ targets evader $j$, then $\delta_i = c_{ij}(x_i - y_j)$. Notice that the target selection procedure is also useful if the evaders change their objectives in the game and decide to separate from their teammates as well as from the pursuers.

Another practical consideration in MPE games consists in designing the agents to use a sustained control effort throughout the game, which allows achieving finite-time capture as studied in the following subsection.

*5.4.2. Finite-time intercept*

In practical applications, pursuers and evaders are expected to use their maximum effort to achieve their goals. In this case, a pursuer is able to intercept its target in finite-time because its velocity is not decreased when approaching the target.

To generate this behavior, let the $i$th pursuer use Algorithm 5.1 to target only one evader, and use the control policy (5.26) with matrix $R_p(\delta_i)$ selected as

$$R_p(\delta_i) = \|\delta_i\| r_p I \tag{5.37}$$

as long as $\delta_i \neq 0$, where $r_p$ is a positive scalar and $I$ is the identity matrix. Similarly, evader $j$ uses the policy (5.27) with $R_e(\varepsilon_j)$ as

$$R_e(\varepsilon_j) = (d_j^e + d_j^{ep})\|\varepsilon_j\| r_e I \tag{5.38}$$

where $r_e$ is a positive scalar and $d_j^e$ and $d_j^{ep}$ are defined in Section 5.2.

The following theorem shows that the selection of matrices (5.37) and (5.38) produce finite-time interception if the pursuer is allowed to use a greater control effort than the evader, and if the Lyapunov equation

$$P_{pi}A + A^T P_{pi} = -Q_s, \tag{5.39}$$

with $Q_s \geq 0$, holds for matrix $P_{pi}$. Notice that (5.39) is solvable if the real parts of the eigenvalues of matrix *A* are non-positive. Condition (5.39) is further studied in subsection 5.4.3.

**Theorem 5.4.** Consider an MPE game (5.17) where pursuer $i$ and evader $j$ have dynamics (5.1) and (5.2), respectively, with a marginally stable system matrix *A*. Pursuer $i$ selects evader $j$ as its only target. Let the control policies be (5.26) and (5.27), with weight matrices (5.37) and (5.38). Let the gain matrix $P_{pi} = P'^{T}_{pi} P'_{pi}$ be such that the Lyapunov equation (5.39) holds. Then, finite-time capture occurs if

$$r_p^{-1}\left\|P_{pi}\right\| \geq r_e^{-1}\left\|P_{ej}\right\|. \tag{5.40}$$

*Proof.* Define the candidate Lyapunov function $V_L = (1/2)\left\|P'_{pi}\delta_i\right\|^2 = (1/2)\delta_i^T P_{pi}\delta_i$. Thus,

$$\begin{aligned}\dot{V}_L &= \frac{1}{2}\delta_i^T(P_{pi}A + A^T P_{pi})\delta_i - r_p^{-1}\left\|\delta_i\right\|^{-1}\delta_i^T P_{pi}BB^T P_{pi}\delta_i - r_e^{-1}\left\|\varepsilon_j\right\|^{-1}\delta_i^T P_{pi}BB^T P_{ej}\varepsilon_j \\ &\leq -r_p^{-1}\left\|\delta_i\right\|^{-1}\delta_i^T P_{pi}BB^T P_{pi}\delta_i - r_e^{-1}\left\|\varepsilon_j\right\|^{-1}\delta_i^T P_{pi}BB^T P_{ej}\varepsilon_j\end{aligned}$$

because (5.39) holds. As $\delta_i^T P_{pi}BB^T P_{pi}\delta_i$ is a positive scalar, it is equal to its norm. Now,

$$\begin{aligned}\dot{V}_L &\leq -r_p^{-1}\left\|\delta_i\right\|^{-1}\left\|\delta_i^T P_{pi}BB^T P_{pi}\delta_i\right\| + r_e^{-1}\left\|\varepsilon_j\right\|^{-1}\left\|\delta_i^T P_{pi}BB^T P_{ej}\varepsilon_j\right\| \\ &\leq -r_p^{-1}\left\|\delta_i\right\|^{-1}\left\|P'_{pi}\delta_i\right\|\left\|P'_{pi}BB^T\right\|\left\|P_{pi}\right\|\left\|\delta_i\right\| + r_e^{-1}\left\|\varepsilon_j\right\|^{-1}\left\|P'_{pi}\delta_i\right\|\left\|P'_{pi}BB^T\right\|\left\|P_{ej}\right\|\left\|\varepsilon_j\right\| \\ &= -\beta_i\left\|P'_{pi}\delta_i\right\| = -\sqrt{2}\beta_i\sqrt{V_L}\end{aligned}$$

where

$$\beta_i = \left(r_p^{-1}\left\|P_{pi}\right\| - r_e^{-1}\left\|P_{ej}\right\|\right)\left\|P'_{pi}BB^T\right\|$$

Clearly, if (5.40) holds, then $\dot{V}_L < 0$ and capture occurs. Furthermore, we can solve the differential equation $\dot{V}_L V_L^{-1/2} = -\sqrt{2}\beta_i$ to obtain $V_L^{1/2}(t) = \left(\sqrt{2}/2\right)\beta_i t + V_L^{1/2}(0)$. This shows that, if capture occurs, then $V_L$ is equal to zero for a finite time *t*. □

### *5.4.3. Inverse optimal control for finite-time capture*

If matrix $P_{pi}$ is selected first such that the Lyapunov equation (5.39) holds, then it is not directly obtained as the solution of the ARE equation (5.33). An optimality result can still be obtained if the performance index (5.30) is selected accordingly by means of inverse optimal control. Theorem 5.5 shows an inverse optimal result by selecting the matrix $Q_{pi}$ in (5.30) as

$$Q_{pi} = Q_s + P_{pi} B R_p^{-1} B^T P_{pi} - P_{pi} B R_e^{-1} B^T P_{pi}. \tag{5.41}$$

**Theorem 5.5.** Let the pursuer $i$ with dynamics (5.1) use control policy (5.26), where matrix $P_{pi}$ is selected such that (5.39) holds for a given matrix $Q_s \geq 0$. Then, pursuer $i$ uses its minmax strategy with respect to the performance index (5.30) with matrix $Q_{pi}$ as in (5.41).

*Proof*. Substitute the matrix $Q_{pi}$ in (5.41) into the ARE (5.33) to obtain

$$Q_s + P_{pi} B R_p^{-1} B^T P_{pi} - P_{pi} B R_e^{-1} B^T P_{pi} + P_{pi} A + A^T P_{pi} - P_{pi} B R_p^{-1} B^T P_{pi} + P_{pi} B R_e^{-1} B^T P_{pi} = 0$$

and note that the equality holds because $P_{pi} A + A^T P_{pi} = -Q_s$. The proof is now completed as in the proof of Theorem 5.2. $\square$

Although target selection and finite-time capture are studied for their broad range of practical implementations, interesting behaviors arise in the MPE games when the agents employ asymptotic strategies. These new scenarios are analyzed in the following section.

## 5.5. Extensions and Asymptotic Behaviors

This section is concerned with the analysis of asymptotic capture in MPE games. Rendezvous and containment control are studied as particular cases of this scenario.

### *5.5.1. Rendezvous or asymptotic capture*

In this section, let the pursuers maintain their local errors as defined in (5.3), without the use of the target selection algorithm. Furthermore, let the matrices $R_p(\delta_i)$ and $R_e(\varepsilon_j)$ be constant throughout time, such that $R_p(\delta_i) = R_p$ and $R_e(\varepsilon_j) = R_e$. In the asymptotic version of the MPE games, capture occurs when all pursuers reach the position of all evaders.

Few definitions are required for the analysis performed in this section. Using the graph matrices defined in Section 5.2.1, define the generalized Laplacian matrix *L* as

$$L = \begin{bmatrix} L_p + D_{pe} & -\mathcal{A}_{pe} \\ \mathcal{A}_{ep} & \kappa L_e - D_{ep} \end{bmatrix} \tag{5.42}$$

Define also the matrix *K* as the block matrix

$$K = \begin{bmatrix} \operatorname{diag}(K_{pi}) & 0 \\ 0 & \operatorname{diag}(K_{ej}) \end{bmatrix} \tag{5.43}$$

where $K_{pi} = (d_i^p + d_i^{pe}) R_p^{-1} B^T P_{pi}$ for $i = 1, \ldots, N$, and $K_{ej} = -(\kappa d_j^e - d_j^{ep}) R_e^{-1} B^T P_{ej}$ for $j = 1, \ldots, M$.

Theorem 5.6 states the conditions for capture relating the stability properties of the game with the three communication graph topologies employed in (5.42).

**Theorem 5.6**. Consider the MPE game (5.17) with system dynamics (5.1) - (5.2) and performance indices (5.15) - (5.16). Define matrix *K* as in (5.43). If there exist matrices $P_{pi}$ and $P_{ej}$ such that the value functions (5.24) - (5.25) satisfy the AREs (5.22) - (5.23) and the agents use control policies (5.26) - (5.27), then the eigenvalues of the matrix $\left[(I \otimes A) - (L \otimes B)K\right] \in \mathbb{R}^{n(N+M)}$ have all negative real parts, i. e.,

$$\operatorname{Re}\{\lambda_k\left((I \otimes A) - (L \otimes B)K\right)\} < 0 \tag{5.44}$$

for $k = 1, \ldots, n(M+N)$.

*Proof*. Define the vectors $\delta = \left[\delta_1^T, \cdots, \delta_N^T, \varepsilon_1^T, \cdots, \varepsilon_M^T\right]^T$ and $u = \left[u_1^T, \cdots, u_N^T, v_1^T, \cdots, v_M^T\right]^T$. Using the local error dynamics (5.4) and (5.6), we can write

$$\dot{\delta} = (I \otimes A)\delta + (L \otimes B)u \tag{5.45}$$

Control policies (5.26) and (5.27) can be expressed as $u_i = -K_{pi}\delta_i$ and $v_j = -K_{ej}\varepsilon_j$, respectively, where matrices $K_{pi}$ and $K_{ej}$ are defined as for (5.43). Now we can write

$$u = -K\delta. \tag{5.46}$$

Substitution of (5.46) in (5.45) yields the global closed-loop dynamics

$$\dot{\delta} = \left[(I \otimes A) - (L \otimes B)K\right]\delta. \tag{5.47}$$

Theorem 1 shows that if matrices $P_{pi}$ and $P_{ej}$ satisfy the equations (5.22) - (5.23) then the control policies (5.26) and (5.27) make the pursuers achieve capture. This is equivalent to state that the system (5.47) is stable, and thus the condition (5.44) holds. □

The following behaviors are corollaries for Theorem 5.2.

*5.5.2. $\mathcal{L}_2$ gain bound*

The $\mathcal{L}_2$ gain bound in MPE games refers to the problem of determining a feedback policy $u_i(\delta_i)$ for pursuer $i$ such that when $\delta_i(0) = 0$ and for all neighbor policies $u_k \in \mathcal{L}_2[0,\infty)$ and $v_j \in \mathcal{L}_2[0,\infty)$, the inequality

$$\int_0^T (\delta_i^T Q_{pi}\delta_i + u_i^T R_p u_i)dt \le r_{-i}\int_0^T \left\|\bar{u}_{-i}\right\| dt \tag{5.48}$$

where $\bar{u}_{-i}$ is defined in (5.8), holds for a scalar $r_{-i} > 0$.

***Corollary 5.1.*** Let the conditions of Theorem 5.2 hold and let matrix $R_{-i}$ in (5.30) be $R_{-i} = r_{-i}I$. Then the $\mathcal{L}_2$ gain of pursuer $i$ is bounded above by the $\mathcal{L}_2$ gain of its neighbors according to the inequality (5.48).

*Proof*. If $R_{-i} = r_{-i}I$, then the Hamiltonian function for pursuer $i$ can be written as

$$H_{pi}(u_i, \bar{v}_{-i}) = \delta_i^T Q_{pi}\delta_i + u_i^T R_p u_i - r_{-i}\bar{u}_{-i}^T\bar{u}_{-i} + \dot{V}_{pi}(\delta_i). \tag{5.49}$$

Let $V_{pi}$ be the solution to the HJI equation $H_{pi}(u_i^*, \bar{u}_{-i}^*) = 0$. Considering the policies (5.26) and $\bar{u}_{-i}^* = -(d_i^p + d_i^{pe}) R_{-i}^{-1} B^T P_{pi} \delta_i$ complete the squares in (5.49) to obtain

$$\begin{aligned} H_{pi} &= \delta_i^T Q_{pi} \delta_i + u_i^{*T} R_p u_i^* - \bar{u}_{-i}^{*T} R_{-i} \bar{u}_{-i}^* + \dot{V}_{pi} + (u_i - u_i^*)^T R_p (u_i - u_i^*) - (\bar{u}_{-i} - \bar{u}_{-i}^*)^T R_{-i} (\bar{u}_{-i} - \bar{u}_{-i}^*) \\ &= (u_i - u_i^*)^T R_p (u_i - u_i^*) - (\bar{u}_{-i} - \bar{u}_{-i}^*)^T R_{-i} (\bar{u}_{-i} - \bar{u}_{-i}^*) \end{aligned}$$

Select $u_i = u_i^*$ to obtain $H_{pi} = -(\bar{u}_{-i} - \bar{u}_{-i}^*)^T R_{-i} (\bar{u}_{-i} - \bar{u}_{-i}^*) \le 0$, which from (5.49) implies

$$\delta_i^T Q_{pi} \delta_i + u_i^T R_p u_i - r_{-i} \bar{u}_{-i}^T \bar{u}_{-i} + \dot{V}_{pi}(\delta_i) \le 0. \tag{5.50}$$

Integrating the inequality (5.50), we get

$$\int_0^T (\delta_i^T Q_{pi} \delta_i + u_i^T R_p u_i - r_{-i} \|\bar{u}_{-i}\|) dt + V_{pi}(\delta_i(T)) - V_{pi}(\delta_i(0)) \le 0.$$

Now, $\delta_i(0) = 0$ implies $V_{pi}(\delta(0)) = 0$. As $V_{pi}(\delta_i(T)) > 0$ for all $\delta_i(T)$, inequality (5.48) is directly obtained. □

*5.5.3. Containment control*

In this subsection the multi-agent pursuit-evasion behaviors are related to the containment control problem [46]. Define the global vector of pursuers' positions as $x = [x_1^T, \ldots, x_N^T]^T$ and the global position vector of the evaders as $y = [y_1^T, \ldots, y_M^T]^T$. Corollary 5.2 shows that, for the special case of static evaders and taking the system matrix $A = 0$, the solution of the MPE games recovers the containment control results.

**Corollary 5.2.** Let the conditions of Theorem 5.2 hold, with system matrix $A = 0$ in (5.1) and (5.2). Select matrix $R_e = r_e I$ in performance index (5.34) and consider the matrix $K_{pi}$ defined as for (5.43). Let the graph topologies be such that there is a directed path from at least one evader to each pursuer. In the limit $r_e \to \infty$, the convex hull of the evaders' positions, according to the expression

$$x = [(L_p + D_{pe})^{-1} \mathcal{A}_{pe} \otimes I] y. \tag{5.51}$$

is an equilibrium set for the pursuers dynamics. Furthermore, if the matrix

$$(I \otimes B)\,\mathrm{diag}(K_{pi})\left[(L_p + D_{pe}) \otimes I\right] \qquad (5.52)$$

is nonsingular, then (5.51) is the only stable equilibrium set for the pursuers dynamics.

*Proof*. Notice that control policies (5.27) with $r_e \to \infty$ produce static evaders. Using control policies $u_i = -K_{pi}\delta_i$, the global pursuer dynamics is given by

$$\dot{x} = -(I \otimes B)\,\mathrm{diag}(K_{pi})\left[(L_p + D_{pe}) \otimes I\right]x + (I \otimes B)\,\mathrm{diag}(K_{pi})\left[\mathcal{A}_{pe} \otimes I\right]y \qquad (5.53)$$

Substituting (5.51) in (5.53), and using the properties of the Kronecker product [54], we get $\dot{x} = 0$. Therefore, the points in (5.51) are equilibrium points for the pursuers. It is easy to prove that, if there is a directed path from at least one evader to each pursuer, all eigenvalues of $L_p + D_{pe}$ are positive, all the elements of $(L_p + D_{pe})^{-1}\mathcal{A}_{pe}$ are nonnegative, and matrix $(L_p + D_{pe})^{-1}\mathcal{A}_{pe}$ has all row sums equal to one [69]. This shows that (5.51) is the convex hull of the evaders' positions.

Now, let $\dot{x} = 0$ in (5.53). The resulting linear equation has (5.51) as its unique solution if matrix (5.52) is nonsingular. □

**Remark 5.3.** From the definition of $K_{pi}$, notice that matrix (5.52) is nonsingular for the special case of single integrator dynamics, that is, if $B = I$.

Corollary 5.3 shows a similar result, allowing the evaders to be moving in a formation with a constant velocity.

***Corollary 5.3.*** Let the conditions in Corollary 5.2 hold, and let all the evaders move with constant velocities, $\dot{y}_j = v$, $j = 1, \ldots, M$. Control inputs

$$u_i = -(d_i^p + d_i^{pe})R_p^{-1}(\delta_i)B^T P_{pi}\delta_i + v \qquad (5.54)$$

make the pursuers converge to the convex hull of the evaders according to the expression (5.51) if matrix (5.52) is nonsingular.

*Proof*. Using control policies (5.54), the global pursuer dynamics is

$$\begin{aligned}\dot{x} = -(I \otimes B)\,\mathrm{diag}(K_{pi})\left[(L_p + D_{pe}) \otimes I\right]x + (\underline{1}_N \otimes v) \\ + (I \otimes B)\,\mathrm{diag}(K_{pi})\left[\mathcal{A}_{pe} \otimes I\right]y\end{aligned} \tag{5.55}$$

where $\underline{1}_N \in \mathbb{R}^{N\times 1}$ is a vector of ones. The derivative of (5.55) is given by

$$\ddot{x} = -(I \otimes B)\,\mathrm{diag}(K_{pi})\left[(L_p + D_{pe}) \otimes I\right]\dot{x} + (I \otimes B)\,\mathrm{diag}(K_{pi})\left[\mathcal{A}_{pe} \otimes I\right](\underline{1}_M \otimes v)$$

where $\underline{1}_M \in \mathbb{R}^{M\times 1}$. If $\dot{x} = \left[(L_p + D_{pe})^{-1}\mathcal{A}_{pe} \otimes I\right](\underline{1}_M \otimes v)$, then we get $\ddot{x} = 0$, and this result is unique if matrix (5.52) is nonsingular. Furthermore, by properties of the Kronecker product and of the graph matrices $L_p$, $D_{pe}$ and $\mathcal{A}_{pe}$, $\left[(L_p + D_{pe})^{-1}\mathcal{A}_{pe} \otimes I\right](\underline{1}_M \otimes v) = \underline{1}_N \otimes v$. Substitute the derivative $\dot{x} = \underline{1}_N \otimes v$ in (5.55) to obtain

$$0 = -(I \otimes B)\,\mathrm{diag}(K_{pi})\left[(L_p + D_{pe}) \otimes I\right]x + (I \otimes B)\,\mathrm{diag}(K_{pi})\left[\mathcal{A}_{pe} \otimes I\right]y.$$

Again, (5.51) is the unique solution of this equation. □

## 5.6. Simulation Results

Numerical simulations for finite-time intercept, asymptotic capture and containment control are presented below.

### *5.6.1. Finite-time capture*

Consider an MPE game with three pursuers and three evaders in $\mathbb{R}^2$ with single integrator dynamics, i.e., $A = 0$ and $B = I$ in (5.1) - (5.2), connected in a communication graph such that

$$L_p = \begin{bmatrix} 1 & -1 & 0 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}, \qquad L_e = \begin{bmatrix} 2 & -1 & -1 \\ 0 & 1 & -1 \\ -1 & -1 & 2 \end{bmatrix},$$
$$\mathcal{A}_{pe} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}, \qquad \mathcal{A}_{ep} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}.$$

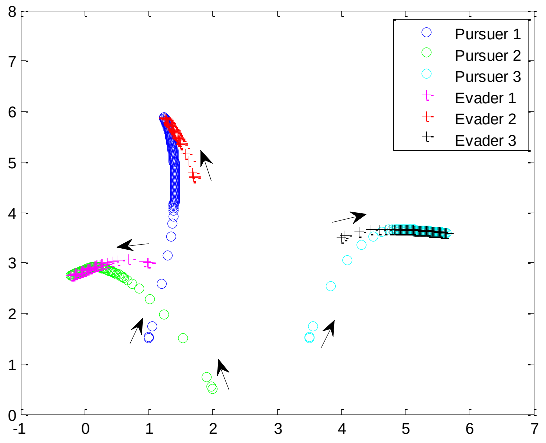

Fig. 5.1. MPE game with finite-time capture.

A simulation of this system is performed, using performance indices (5.15) and (5.16) with $Q = 5I$ , and input weighting matrices as in (5.37) and (5.38) with $r_p = 1$ and $r_e = 3$. Effectiveness of control policies (5.26) and (5.27) is tested. Furthermore, let the evaders try to separate from each other, besides escaping from the pursuers, to force the pursuers to separate as well. The pursuers use Algorithm 5.1 to select a target among the evaders. The result of this game is shown in Fig. 5.1. Each pursuer achieves capture of its target.

### *5.6.2. Asymptotic capture*

Asymptotic minmax strategies are simulated for the same systems above. The evaders are now interested in maintaining their team cohesion and the pursuers do not select an individual target. To show the effect of the value of $\kappa$ in the behavior of the agents, two cases are considered. First, a value of $\kappa = 3$ is taken. The result of this simulation is shown in Figure 5.2, which shows that capture occurs, verifying the theoretical results. Then, a value of $\kappa = 1.2$ is used. This smaller value represents a lower priority of the evaders to remain together. Figure 5.3 displays this result, where the evaders do not converge to a single point in the state space.

### *5.6.3. Containment control*

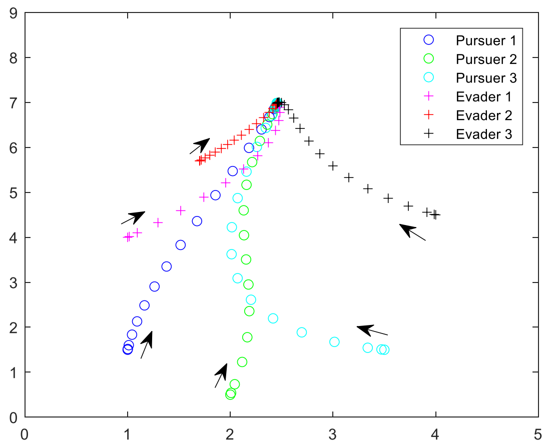

Fig. 5.2. Asymptotic behavior in a MPE game. A large κ makes the evaders attract each other.

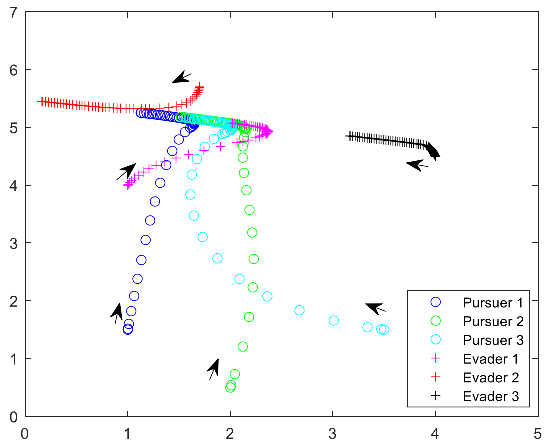

Fig. 5.3. Asymptotic behavior in a MPE game with small κ. The evaders have a low priority to remain together.

Using the same parameters as in the asymptotic capture simulation above, let the evaders remain static throughout the game.

The behavior of the pursuers under these circumstances is displayed in Figure 5.4. The pursuers can be seen converging to the convex hull of the positions of the evaders, as stated in Corollary 5.2.

Finally, let the evaders move in a formation with constant speeds, and let the pursuers use the control inputs (5.54). Take the vector $\nu = [2,\ 1]^T$ as the constant velocity of the evaders.

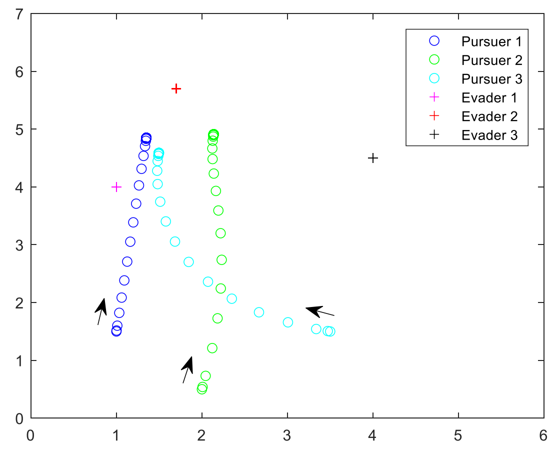

Fig. 5.4. Containment behavior of the agents. The evaders are static and the pursuers converge to their convex hull.

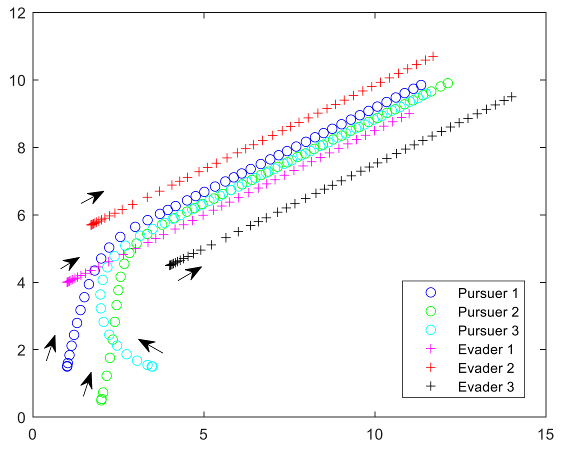

Fig. 5.5. Containment behavior with moving evaders. The evaders move in a formation with constant speed.

Figure 5.5 shows that the pursuers reach the convex hull of the positions of the evaders and maintain their positions thereafter.

## 5.7. Conclusion

Performance of the players in MPE games on communication graphs was studied for performance indices that can include both cooperative and adversarial objectives among the agents. Nash equilibrium is guaranteed if the solutions of the coupled HJI equations exist. Otherwise, it is still possible to design the control policies of the agents to obtain a minimum guaranteed performance value, using minmax strategies. Conditions for capture were found to depend on the structure of the graph topologies and on different control design parameters, like the difference between the relative speeds of the teams or the inclination of the evaders of either staying together or escaping from the pursuers. Different emergent behaviors arise from changes in the goals of the players. Simulation plots show the differences between these behaviors, verifying the obtained results.

# Chapter 6

# FURTHER SOLUTION CONCEPTS FOR DIFFERENTIAL GRAPHICAL GAMES

## 6.1. Introduction

In Chapter 4, minmax strategies were proposed to address the inconveniences of Nash equilibrium in graphical games. Different from the Nash solution, minmax provides optimal distributed control policies to the agents under mild conditions related to the system dynamics and the performance indices of the game. Minmax strategies were proven to have strong robustness properties, improving the well-known gain and phase margin results of the single-agent LQR. This is an expected characteristic of minmax strategies due to its parallel formulation with the $H_\infty$ control problem.

Minmax strategies, however, may not provide the ideal policies for different applications of graphical games. The assumption that the neighbors will perform their worst-case behavior may be too conservative. Consider, for example, the application of a platoon of autonomous vehicles in a straight-line formation. In this case, each agent desires to keep a safe distance with respect to the vehicle in front. A direct application of minmax strategies may imply that the worst-case behavior of the front vehicle is to drive backwards, reducing the safety distance. This assumption leads to an absurd behavior that would yield inadequate control policies in the vehicles trying to avoid a collision.

The minmax formulation has both advantages and inconveniences that make it suitable for many but not all applications. Other solution concepts can also be proposed to solve differential graphical games and provide different characteristics to the control policies. In this chapter we define and solve three alternative formulations of graphical games: minmax regret, graph-weight projection strategies and local-error projection strategies.

Minmax regret was already defined for normal-form games in Chapter 2. It consists of defining the regret of an agent as the difference between the payoff obtained after playing a particular strategy and the payoff that would have been received if the agent had played optimally. This formulation seems naturally suitable for graphical games, because the limited knowledge of the agents prevent them from playing an optimal strategy against each other as in Nash equilibrium. If the optimal policy not achievable, the concept of regret becomes relevant. In minmax regret strategies, an agent simply prepares itself to avoid the worst possible regret. The differential games version of this strategy is obtained by finding an integral expression for the regret of the agent, and solving a minmax problem with it.

The other two methods presented in this chapter enter the category we call *projection strategies*. Projection is a term used in psychology to describe the natural mechanism by means of which a person assumes that the people around him possess the same behavioral characteristics as himself. This is the reason why an aggressive person is also usually on the defensive, expecting the same behavior from his peers.

In projection strategies, the agents make the assumption that some aspect of their personal knowledge is shared by their neighbors. Two cases are here considered. First, we define the *graph-weight projection* strategies, on which an agent assumes that its neighbors are all the agents in the game and that they can also observe each other. In this sense, they project their knowledge about the graph information onto others. Under this formulation, the agents solve a nonzero-sum game with their neighbors, expecting them to do the same.

The second projection strategy is regarded as *local-error projection*. In this case, agent $i$ acknowledges that its neighbor, agent $j$, has neighbors of its own, but assumes that those neighbors have the opposite relative position with respect to $j$ than all the neighbors of $i$ with respect to $i$ itself. In other words, agent $i$ assumes that the values of the local errors for both agents have the same magnitude and opposite sign. Using this assumption, an agent can solve a game formulated as for Nash equilibrium, although such equilibrium is not reached.

The rest of this chapter is organized as follows. Section 6.2 formulates the minmax regret graphical game by extending the normal-form game definitions in Chapter 2. Section 6.3 introduces the graph-weight projection strategies, and Section 6.4 solves the local-error projection game. In all this cases, the resulting policies of the agents are analyzed for their robustness characteristics. Section 6.5 presents numerical simulations of the proposed solutions to test their applicability in graphical games. The conclusions of the chapter are finally presented in Section 6.6.

## 6.2. Minmax Regret for Differential Graphical Games

The normal-form version of minmax regret was presented in Section 2.3. The essence of this solution can be intuitively explained as follows. Against every policy of agent $j$, there is an optimal policy for agent $i$. If agent $i$ does not play that optimal action, it will have a regret, which is defined as the difference between the optimal payoff and the suboptimal reward obtained. Furthermore, for every action of agent $i$, there is a particular action of agent $j$ that makes $i$'s regret as large as possible. The minmax regret policy of an agent is then defined as the action it has to play to minimize its worst possible regret.

In this section, we formulate the minmax regret strategies for differential games in communication graphs. Again, the objective is to obtain an expression for the regret of an agent, and determine the policies of its neighbors that could maximize such regret. Agent $i$ has only to act optimally against those policies. Consider the system dynamics and local errors defined in (2.2) - (2.5). The regret of an agent is formally defined as follows.

**Definition 6.1 (Regret).** Let $u_i^*$ be the best response of agent $i$ against given strategies $u_{-i}$ of its neighbors. The regret of agent $i$ for playing any other policy $u_i$ is given by

$$\mathcal{R}_i(u_i, u_{-i}) = J_i(u_i, u_{-i}) - J_i(u_i^*, u_{-i}) \tag{6.1}$$

□

In Definition 6.1, we use the fact that agent $i$ will pay a higher cost for playing a non-optimal strategy to define the regret as the difference of that cost with respect to the optimal value of $J_i$.

To determine an analytical expression for the regret (6.1), consider a general cost function with quadratic control term for agent $i$ as

$$J_i = \int_0^\infty r_i(\delta_i, u_{-i}) + u_i^T R_i u_i \; dt \tag{6.2}$$

It has been proven ([5], [20] and proof of Theorem 4.1 in this thesis) that for optimal neighbor strategies, the final cost of an agent is given by

$$J_i = V_i(\delta_i(0)) + \int_0^\infty (u_i - u_i^*)^T R_i (u_i - u_i^*) dt$$

where $V_i(\delta_i)$ is the value function of agent $i$. This implies that if agent $i$ uses its optimal policy, i.e., $u_i = u_i^*$, then the optimal cost is

$$J_i^* = V_i(\delta_i(0)). \tag{6.3}$$

Assume, on the contrary, that agent $i$ uses a suboptimal strategy, deviating from the optimal control such that

$$u_i = u_i^* + \Delta u_i \tag{6.4}$$

then, the cost paid after the game is

$$\begin{aligned} J_i &= V_i(\delta_i(0)) + \int_0^\infty (u_i^* + \Delta u_i - u_i^*)^T R_i (u_i^* + \Delta u_i - u_i^*) dt \\ &= V_i(\delta_i(0)) + \int_0^\infty \Delta u_i^T R_i \Delta u_i dt \end{aligned} \tag{6.5}$$

Thus, the regret (6.1) can be obtained as the subtraction of (6.3) from (6.5), which yields,

$$\begin{aligned}\mathcal{R}_i &= J_i - J_i^* \\ &= V_i(\delta_i(0)) + \int_0^\infty \Delta u_i^T R_i \Delta u_i dt - V_i(\delta_i(0)) \\ &= \int_0^\infty \Delta u_i^T R_i \Delta u_i dt.\end{aligned} \tag{6.6}$$

Integral (6.6) is the regret of agent $i$ for playing the policy (6.4). Notice that if the optimal inputs of the agents are linear state-feedback policies as $u_i^* = -K_i^* \delta_i$, and agent $i$ uses the linear policy

$$u_i = -(K_i^* + \Delta K_i)\delta_i \tag{6.7}$$

then the regret (6.6) takes the form

$$\mathcal{R}_i = \int_0^\infty \delta_i^T \Delta K_i^T R_i \Delta K_i \delta_i dt. \tag{6.8}$$

Expression (6.8) is used here to define the minmax regret strategy for agent $i$.

Ideally, to define minmax regret strategies we would consider first the worst possible regret for each admissible control policy $u_i$. Then we would select the minmax regret policy $u_i^*$ as the control input that leads to the smallest of all such regrets. This approach is useless, however, because the worst possible regret for any policy $u_i$ is always infinite in magnitude. To show this, consider any policy of the form $u_i = -K_i \delta_i$. Substituting in the dynamics (2.5), we get

$$\begin{aligned}\dot{\delta}_i &= A\delta_i - (d_i + g_i)BK_i\delta_i - \sum_{j=1}^N a_{ij} B u_j \\ &= \left[A - (d_i + g_i)BK_i\right]\delta_i - \sum_{j=1}^N a_{ij} B u_j\end{aligned} \tag{6.9}$$

As the system (6.9) is stabilizable by assumption, we can always select the control policies $u_j$ to arbitrarily relocate its poles. This can be accomplished by selecting $u_j = -K_j \delta_i$, such that

$$\dot{\delta}_i = \left[A - (d_i + g_i)BK_i + \sum_{j=1}^N a_{ij} BK_j\right]\delta_i. \tag{6.10}$$

Because we are interested only in admissible policies, we assume the poles of the system (6.10) will be left with negative real parts. However, these poles can be left arbitrarily close to zero, decreasing the rate of convergence of $\delta_i$ and, therefore, increasing the value of the integral (6.8). In the limit, the worst regret is found to happen when the neighbors $u_j$ make the system (6.9) marginally stable, which leads to an infinite regret.

An adequate formulation of minmax regret strategies for differential games requires us to modify the normal-form games definition using the regret in (6.8) as follows.

**Definition 6.2 (Minmax regret for graphical games).** The minmax regret strategy of agent $i$ in a differential graphical game is given by

$$u_i^* = \arg\min_{u_i}\left[\max_{u_j}\int_0^\infty\left(\delta_i^T \Delta K_i^T R_i \Delta K_i \delta_i + (d_i + g_i)u_i^T R_i u_i - \sum_{j=1}^N a_{ij} u_j^T R_j u_j\right)dt\right] \quad (6.11)$$

where $u_i^* = -K_i^* \delta_i$, $u_j^* = -K_j^* \delta_i$, $\Delta K_i = K_i - K_i^*$ and $K_i$ is the control feedback that makes all the controllable poles of the matrix $A-(d_i+g_i)BK_i + \sum_{j=1}^N a_{ij}BK_j^*$ equal to zero. □

According to Definition 6.2, the objective of agent $i$ is to minimize the regret of using the worst possible deviation from its optimal policy $u_i^*$ against the neighbor policies that maximize such regret. Using a similar reasoning as before, the worst possible deviation from the optimal policy of $i$ is taken such that the final closed loop system (6.10) is marginally stable.

By now, it is clear to us that the control policies $u_i^*$ and $\upsilon_j^*$ that solve the problem (6.11) have the form

$$u_i^* = -R_i^{-1}B^T P_i \delta_i \quad (6.12)$$

and

$$\upsilon_j^* = -R_j^{-1}B^T P_i \delta_i \quad (6.13)$$

Substituting $K_j^* = R_j^{-1}B^T P_i$ in the system dynamics (6.10) we obtain

$$\dot{\delta}_i = \left[ A + \sum_{j=1}^{N} a_{ij} B R_j^{-1} B^T P_i - (d_i + g_i) B K_i \right] \delta_i \,. \tag{6.14}$$

The matrix $K_i$ that makes system marginally stable is now given by

$$K_i = \bar{K}_i + \frac{1}{d_i + g_i} \sum_{j=1}^{N} a_{ij} R_j^{-1} B^T P_i \tag{6.15}$$

where matrix $\bar{K}_i$ locates all the controllable poles of the system matrix $A - (d_i + g_i) B \bar{K}_i$ over the imaginary axis. The following lemma formalizes this result.

**Lemma 6.1.** Consider a multiagent system with dynamics (2.5) and let all the neighbors of agent $i$ use the control policies (6.13) for a given matrix $P_i$. Then, the policy $u_i = -K_i \delta_i$ with $K_i$ as in (6.15) makes the closed-loop system (6.14) marginally stable.

*Proof*. Follows trivially from substituting (6.15) in (6.14). □

The following theorem shows that the control input (6.12) corresponds to the minmax regret policy of agent $i$ as defined in Definition 6.2 if matrix $P_i$ solves the Riccati equation

$$\begin{aligned} &\bar{K}_i^T R_i \bar{K}_i + P_i \;\; A + B\bar{R}_i R_i \bar{K}_i \;\; + \;\; A + B\bar{R}_i R_i \bar{K}_i \;\; ^T P_i \\ &\qquad - P_i B \left[ (d_i + g_i) R_i^{-1} - \sum_{j=1}^{N} a_{ij} R_j^{-1} - \bar{R}_i R_i \bar{R}_i \right] B^T P_i = 0 \end{aligned} \tag{6.16}$$

with $\bar{R}_i = \dfrac{1}{d_i + g_i} \displaystyle\sum_{j=1}^{N} a_{ij} R_j^{-1} - R_i^{-1}$.

**Theorem 6.1.** Consider a multiagent system with dynamics (2.5). The minmax regret problem (6.11) is solved when agent $i$ uses the policy (6.12), where matrix $P_i$ solves the algebraic Riccati equation (6.16).

*Proof*. Similar to the solution of minmax strategies in Section 4, the minmax problem (6.11) is solved by the policies (6.12) - (6.13) with matrix $P_i$ as the solution of the ARE

$$\Delta K_i^T R_i \Delta K_i + P_i A + A^T P_i - (d_i + g_i) P_i B R_i^{-1} B^T P_i + \sum_{j=1}^{N} a_{ij} P_i B R_j^{-1} B^T P_i = 0 \tag{6.17}$$

From (6.12), $K_i^* = R_i^{-1}B^TP_i$, and by Lemma 6.1, $K_i$ is given by (6.15). Then, $\Delta K_i$ computed as $\Delta K_i = K_i - K_i^* = \bar{K}_i + \bar{R}_i B^T P_i$, with $\bar{K}_i$ defined for (6.15) and $\bar{R}_i$ defined for (6.16). Substituting this result in (6.17) and performing algebraic operations, (6.16) results. □

The formulation of the minmax regret game in (6.11) has an evident parallel to the minmax strategies and, therefore, we can obtain a direct result about the robustness properties of the control policies (6.12). This result is stated in the following theorem. The proof is omitted for being identical to the robustness analysis procedure in Chapter 4.

**Theorem 6.2.** Consider the perturbed system (4.25) with the control policy (6.12) where $P_i$ solves the ARE (6.16), and let the conditions of Corollary 4.1 hold. Then, a phase shift of $\left|\phi_i\right| \le 60^\circ + \theta$, where $\theta = \arccos(0.25(c + \sqrt{12 - 3c^2}))$ and $c = 1 - (d_i + g_i)^{-1}\gamma^{-2}\sum_{j=1}^{N} a_{ij} r_{i,k} r_{j,k}^{-1}$, in the respective feedback loops of each of the controls $u_i$ will leave an asymptotically stable system. Moreover, inserting a gain of $\alpha_k$ as in (4.30) in the feedback loop of the controllers $u_{i,k}$, leaves the system asymptotically stable. □

The next two sections present and analyze the solution concepts regarded as projection strategies.

## 6.3. Projection Strategies: Graph-Weights Projection

Both the minmax policy and the minmax regret policy prepare an agent to face some type of worst-case adversity. In contrast, Section 6.3 and 6.4 propose two different strategies as alternative solution concepts on which the assumptions made by an agent are of a cooperative nature. In this section, graph-weights projection strategies are introduced to solve graphical games. The following definition states the assumptions made by each agent $i$ about the information available to its neighbors.

**Definition 6.3 (Graph-weights projection).** Let $\hat{a}^i_{jk}$ be the assumed graph weight that agent $i$ assigns to its neighbor $j \in \mathcal{N}_i$ about a third agent $k$. In particular, define $\hat{a}^i_{jk} = a_{ik}$ for $k \neq i, j$, $\hat{a}^i_{ji} = a_{ij}$ and $\hat{a}^i_{jj} = 0$. Similarly, let the assumed pinning gain of $j$ be $\hat{g}^i_j = g_i$. □

**Remark 6.1.** Definition 6.3 describes the assumptions that an agent $i$ makes about its neighbors in order to compute an optimal control policy. In the general case, these values do not correspond to the actual values of the graph weights $a_{jk}$ and $g_j$. □

The interpretation of Definition 6.3 is for agent $i$ to assume that agent $j$ can see exactly the same agents and leader as itself, with the same weight in the links.

The local error variable of agent $j$, $j \in \mathcal{N}_i$, as assumed by agent $i$, is given by

$$\hat{\delta}^i_j = \sum_{k=1}^{N} \hat{a}^i_{jk}(x_j - x_k) + \hat{g}^i_j(x_j - x_0). \tag{6.18}$$

Operating from (6.18) and using Definition 6.3, we can obtain a relation between $\hat{\delta}^i_j$ and $\delta_i$ as

$$\begin{aligned}
\hat{\delta}^i_j &= \sum_{k=1}^{N} a_{ik}(x_j - x_k) + a_{ij}(x_j - x_i) + g_i(x_j - x_0) + (d_i + g_i)x_i - (d_i + g_i)x_i \\
&= \delta_i + (d_i + g_i)x_j + a_{ij}(x_j - x_i) - (d_i + g_i)x_i \\
&= \delta_i + (d_i + g_i + a_{ij})(x_j - x_i).
\end{aligned}$$

Therefore, if agent $j$ had the graph information as assumed by agent $i$, it would be able to compute the value of $\delta_i$ as

$$\delta_i = \hat{\delta}^i_j - (d_i + g_i + a_{ij})(x_j - x_i). \tag{6.19}$$

Let agent $i$ use these information to define a nonzero-sum game between its neighbors and itself. This is achieved by considering the error dynamics (2.5) as the variable of concern by agent $i$ and its neighbors, where the neighbors obtain the information $\delta_i$ from (6.19).

Notice that we can also express the dynamics (2.5) as

$$\dot{\delta}_i = A\delta_i + \sum_{j \in i \cup \mathcal{N}_i} \hat{B}^i_j u_j \tag{6.20}$$

where $\hat{B}^i_j = -a_{ij}B$ for $j \neq i$, and $\hat{B}^i_i = (d_i + g_i)B$. The form of the dynamics (6.20) is commonly used to represent multiagent systems for nonzero-sum games [5]. Definition 6.4 below formalizes the game solved in this section.

**Definition 6.4 (Graph-weights projection graphical game).** Let the objective of agent $i$ be the minimization of the cost function

$$u_i^* = \arg\min_{u_i} J_i \triangleq \arg\min_{u_i} \int_0^\infty \left( \delta_i^T Q_i \delta_i + (d_i + g_i) u_i^T R_{ii} u_i + \sum_{j \in \mathcal{N}_i} a_{ij} u_j^T R_{ij} u_j \right) dt \quad (6.21)$$

when its neighbors $j \in \mathcal{N}_i$ use the policies

$$\upsilon_j^* = \arg\min_{u_j} \hat{J}_j^i \triangleq \arg\min_{u_j} \int_0^\infty \left( \delta_i^T Q_j \delta_i + (d_i + g_i) u_i^T R_{ji} u_i + \sum_{k \in \mathcal{N}_i} a_{ik} u_k^T R_{jk} u_k \right) dt. \quad (6.22)$$

□

In Theorem 6.3, it is proven that the solution of the game (6.21) - (6.22) is given by the optimal policy

$$u_i^* = -R_{ii}^{-1} B^T P_i \delta_i \quad (6.23)$$

where matrix $P_i$ solves a set of $N_i + 1$ coupled Riccati equations. One of the AREs is given by

$$\begin{aligned} Q_i + P_i A + A^T P_i - (d_i + g_i) P_i B R_{ii}^{-1} B^T P_i + \sum_{j \in \mathcal{N}_i} a_{ij} \hat{P}_j^i B R_{jj}^{-1} R_{ij} R_{jj}^{-1} B^T \hat{P}_j^i \\ - \sum_{j \in \mathcal{N}_i} a_{ij} P_i B R_{jj}^{-1} B^T \hat{P}_j^i - \sum_{j \in \mathcal{N}_i} a_{ij} \hat{P}_j^i B R_{jj}^{-1} B^T P_i = 0 \end{aligned} \quad (6.24)$$

and the other $N_i$ AREs have the form

$$\begin{aligned} Q_j + \hat{P}_j^i A + A^T \hat{P}_j^i + (d_i + g_i) P_i B R_{ii}^{-1} R_{ji} R_{ii}^{-1} B^T P_i + \sum_{k \in \mathcal{N}_i} a_{ik} \hat{P}_k^i B R_{kk}^{-1} R_{jk} R_{kk}^{-1} B^T \hat{P}_k^i \\ - (d_i + g_i) \hat{P}_j^i B R_{ii}^{-1} B^T P_i - (d_i + g_i) P_i B R_{ii}^{-1} B^T \hat{P}_j^i - \sum_{k \in \mathcal{N}_i} a_{ik} \hat{P}_j^i B R_{kk}^{-1} B^T \hat{P}_k^i \\ - \sum_{k \in \mathcal{N}_i} a_{ik} \hat{P}_k^i B R_{kk}^{-1} B^T \hat{P}_j^i = 0 \end{aligned} \quad (6.25)$$

for all $j \in \mathcal{N}_i$. The notation $\hat{P}_j^i$ expresses the solutions of (6.24) - (6.25) under the assumptions of agent $i$.

**Theorem 6.3.** Consider a multiagent system with dynamics (6.20). Then, the control policy (6.23), where $P_i$ solves the set of $N_i + 1$ coupled Riccati equations (6.24) - (6.25), solves the graph-weight projection game defined by (6.21) - (6.22).

*Proof*. The associated Hamiltonian for the optimization problem (6.21) is

$$H_i = \delta_i^T Q_i \delta_i + (d_i + g_i) u_i^T R_{ii} u_i + \sum_{j \in \mathcal{N}_i} a_{ij} u_j^T R_{ij} u_j + 2\delta_i^T P_i \left( A\delta_i + \sum_{j \in i \cup \mathcal{N}_i} \hat{B}_j^i u_j \right) \qquad (6.26)$$

and the optimal policy for agent $i$ is obtained from $H_i$ using the stationary condition $\frac{\partial H_i}{\partial u_i} = 0$, which yields $u_i^* = -\frac{1}{d_i + g_i} R_{ii}^{-1} \hat{B}_i^{i,T} P_i \delta_i$. Using the definition $\hat{B}_i^i = (d_i + g_i)B$ from (6.20), we get (6.23). Similarly, the assumed behavior of the neighbors of $i$ is given by the problem (6.22) with associated Hamiltonian

$$\hat{H}_j^i = \delta_i^T Q_j \delta_i + (d_i + g_i) u_i^T R_{ji} u_i + \sum_{k \in \mathcal{N}_i} a_{ik} u_k^T R_{jk} u_k + 2\delta_i^T \hat{P}_j^i \left( A\delta_i + \sum_{k \in i \cup \mathcal{N}_i} \hat{B}_k^i u_k \right) \qquad (6.27)$$

from which, using the values $\hat{B}_j^i = -a_{ij} B$ for $j \in \mathcal{N}_i$, we get the neighbor policies

$$\upsilon_j^* = R_{jj}^{-1} B^T \hat{P}_j^i \delta_i \qquad (6.28)$$

Substituting the policies (6.23) and (6.28) in (6.26) and equating to zero, we obtain the HJ equations

$$\delta_i^T \Bigg[ Q_i + P_i A + A^T P_i - (d_i + g_i) P_i B R_{ii}^{-1} B^T P_i + \sum_{j \in \mathcal{N}_i} a_{ij} \hat{P}_j^i B R_{jj}^{-1} R_{ij} R_{jj}^{-1} B^T \hat{P}_j^i$$
$$+ \sum_{j \in \mathcal{N}_i} a_{ij} P_i B R_{jj}^{-1} B^T \hat{P}_j^i + \sum_{j \in \mathcal{N}_i} a_{ij} \hat{P}_j^i B R_{jj}^{-1} B^T P_i \Bigg] \delta_i = 0$$

that are solved for all $\delta_i$ if the Riccati equation (6.24) is satisfied by $P_i$ and $\hat{P}_j^i$, $j \in \mathcal{N}_i$. Similarly, each of the $N_i$ neighbors of $i$ define their own corresponding HJ equations from

(6.27), which yield the AREs (6.25). The proof of optimality of the control policies (6.23) and (6.28) follows similarly as in the proof of Theorem 4.1 in this dissertation. □

**Remark 6.2.** The graph weight projection strategies provide cooperative control policies for the agents, at the cost of an increase in the computational complexity required to compute the solution of the $N_i + 1$ AREs (6.24) - (6.25).

The formulation of the graph-weight projection game does not allow to obtain the gain and phase margin of the control policies (6.23) as performed in Chapter 4 for minmax strategies. However, we can still obtain some results about the robustness properties of the graph-weight projection strategies.

Consider the perturbed version of the system (2.5) as

$$\dot{\hat{\delta}}_i = A\hat{\delta}_i + (d_i + g_i)B(\wp_i u_i) - \sum_{j=1}^{N} a_{ij} B(\wp_j u_j) \tag{6.29}$$

The robustness properties of the control policies (6.23) in the system (6.29) are now presented.

**Theorem 6.4.** If the perturbations $\wp_j$, $j \in i \cup \mathcal{N}_i$, of the system (6.29) are such that

$$\begin{aligned}\Big\langle \delta_i, [(d_i + g_i)P_i B(2\wp_i - I)R_{ii}^{-1}B^T P_i + \textstyle\sum_{j\in\mathcal{N}_i} a_{ij} P_i B(\wp_j - I)R_{jj}^{-1}B^T \hat{P}_j^i \\ + \textstyle\sum_{j\in\mathcal{N}_i} a_{ij}\hat{P}_j^i B R_{jj}^{-1}(\wp_j - I)B^T P_i]\delta_i \Big\rangle \ge 0\end{aligned} \tag{6.30}$$

for all $\delta_i$, and if, additionally, $A, (Q_i + \sum_{j\in\mathcal{N}_i} a_{ij}\hat{P}_j^i B R_{jj}^{-1} R_{ij} R_{jj}^{-1} B^T \hat{P}_j^i)^{1/2}$ is detectable, then $\delta_i$ is asymptotically stable.

*Proof*. Using the control policies $u_i^*$ and $v_j^*$ in (6.23) and (6.28), and the ARE (6.24) in the perturbed system (6.29), then for every $\tau$,

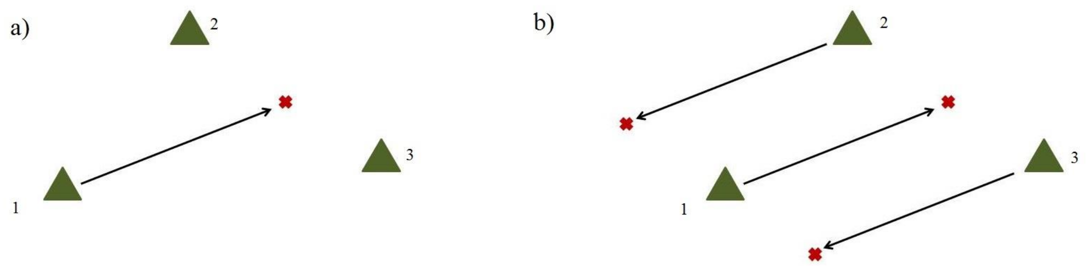

Fig. 6.1. a) The vector points from agent 1 to the center of gravity of its neighbors. b) Agent 1 assumes all its neighbors have their center of gravity in the opposite relative position as itself.

$$\begin{aligned}\delta_i^T(0)P_i\delta_i(0) &= \hat{\delta}_i^T(\tau)P_i\hat{\delta}_i(\tau) - \int_0^\tau \frac{d}{dt}\hat{\delta}_i^T(t)P_i\hat{\delta}_i(t)dt \\ &= -2\left\langle \hat{\delta}_i, P_i(A-(d_i+g_i)B\wp_i R_{ii}^{-1}B^TP_i - \sum\nolimits_{j\in\mathcal{N}_i} a_{ij}B\wp_j R_{jj}^{-1}B^T\hat{P}_j^i)\hat{\delta}_i \right\rangle \\ &= \left\langle \hat{\delta}_i, (P_iA + A^TP_i + 2(d_i+g_i)P_iB\wp_i R_{ii}^{-1}B^TP_i + 2\sum\nolimits_{j\in\mathcal{N}_i} a_{ij}P_iB\wp_j R_{jj}^{-1}B^T\hat{P}_j^i)\hat{\delta}_i \right\rangle \\ &= \Big\langle \hat{\delta}_i, (Q_i + \sum\nolimits_{j\in\mathcal{N}_i} a_{ij}\hat{P}_j^iBR_{jj}^{-1}R_{ij}R_{jj}^{-1}B^T\hat{P}_j^i + (d_i+g_i)P_iB(2\wp_i - I)R_{ii}^{-1}B^TP_i \\ &\qquad + \sum\nolimits_{j\in\mathcal{N}_i} a_{ij}P_iB(\wp_j - I)R_{jj}^{-1}B^T\hat{P}_j^i + \sum\nolimits_{j\in\mathcal{N}_i} a_{ij}\hat{P}_j^iBR_{jj}^{-1}(\wp_j - I)^TB^TP_i)\hat{\delta}_i \Big\rangle\end{aligned}$$

where $\hat{\delta}_{i\tau}$ is the truncation of $\hat{\delta}_i$ as in (2.14). Similar to the proof of Lemma 4.3, the inequality $\delta_i^T(0)P_i\delta_i(0) - \left\langle \hat{\delta}_i, (Q_i + \sum\nolimits_{j\in\mathcal{N}_i} a_{ij}\hat{P}_j^iBR_{jj}^{-1}R_{ij}R_{jj}^{-1}B^T\hat{P}_j^i)\hat{\delta}_i \right\rangle$ holds for the condition (6.30). The rest of the proof follows as in Lemma 4.3. □

The following section presents our last proposed solution concept for differential graphical games.

## 6.4. Projection Strategies: Local-Error Projection

We can define a different solution concept when all agents in the game are aware of the graph topology that links them together, even if each agent is unable to measure the states of any agent but its neighbors. In this case, an agent can make assumptions about the position of its neighbors' neighbors, and use this information to determine a solution to the corresponding

HJ equations. This idea motivates the local-error projection strategy, on which agent $i$ projects its private information, $\delta_i$, onto its neighbors. In particular, let agent $i$ assume $\delta_j = -\delta_i$.

This assumption has a clear geometrical interpretation. To describe it, define the center of gravity of the neighbors of agent $i$ as

$$\overline{x}_{-i} = \frac{1}{d_i + g_i}\left(\sum_{j=1}^{N} a_{ij}x_j + g_i x_0\right) \tag{6.31}$$

Using (6.31), we can express the local error variable of agent $i$ as

$$\begin{aligned}\delta_i &= \sum_{j=1}^{N} a_{ij}(x_i - x_j) + g_i(x_i - x_0)\\ &= (d_i + g_i)x_i - \sum_{j=1}^{N} a_{ij}x_j - g_i x_0\\ &= (d_i + g_i)x_i - (d_i + g_i)\overline{x}_i\\ &= (d_i + g_i)(x_i - \overline{x}_i)\end{aligned}$$

This expression shows that $\delta_i$ is a vector that points from the state $x_i$ to the center of gravity of $i$'s neighbors, $\overline{x}_i$, weighted by the factor $d_i + g_i$. The objective of agent $i$ of minimizing $\delta_i$ is equivalent to state that agent $i$ has the center of gravity $\overline{x}_{-i}$ as actual target. Now, the assumption $\delta_j = -\delta_i$ implies that agent $j$ and its target have the opposite relative position with respect to each other as the position of agent $i$ and its own target, as shown in Fig. 6.1.

The formulation of the local-error projection strategies is considerably simpler than the formulation in Section 6.3, and is formalized in the following definition.

**Definition 6.5 (Local-error projection graphical game).** Let each agent $i$ have as objective to determine the control policy $u_i^*$ that solves the minimization problem

$$u_i^* = \arg\min_{u_i} J_i \triangleq \arg\min_{u_i} \int_0^\infty \left(\delta_i^T Q_i \delta_i + (d_i + g_i)u_i^T R_i u_i + \sum_{j=1}^{N} a_{ij} u_j^T R_j u_j\right) dt \tag{6.32}$$

while making the assumption $\delta_j = -\delta_i$. □

Theorem 6.5 shows that the control policy

$$u_i^* = -R_i^{-1} B^T P_i \delta_i \tag{6.33}$$

where $P_i$ is the solution of the set of coupled AREs

$$\begin{aligned} Q_i + P_i A + A^T P_i - (d_i + g_i) P_i B R_i^{-1} B^T P_i + \sum_{j=1}^{N} a_{ij} P_j B R_j^{-1} B^T P_j \\ - \sum_{j=1}^{N} a_{ij} P_i B R_j^{-1} B^T P_j - \sum_{j=1}^{N} a_{ij} P_j B R_j^{-1} B^T P_i = 0 \end{aligned} \tag{6.34}$$

for $i = 1, \ldots, N$, solves the game (6.32).

**Theorem 6.5.** Consider a multiagent system with dynamics (2.5) and let agent $i$ make the assumption $\delta_j = -\delta_i$. Then, the control policy (6.33), where $P_i$ is the solution of the set of $N$ coupled Riccati equations (6.34), solves the local-error projection graphical game (6.32).

*Proof*. The control policy (6.33) is obtained by using the stationary condition from the Hamiltonian of problem (6.32) given by

$$H_i = \delta_i^T Q_i \delta_i + (d_i + g_i) u_i^T R_i u_i + \sum_{j=1}^{N} a_{ij} u_j^T R_j u_j + 2\delta_i^T P_i \left( A\delta_i + (d_i + g_i) B u_i - \sum_{j=1}^{N} a_{ij} B u_j \right).$$

By defining the Hamiltonian for agent $j$ as

$$H_j = \delta_j^T Q_j \delta_j + (d_j + g_j) u_j^T R_j u_j + \sum_{k=1}^{N} a_{jk} u_k^T R_k u_k + 2\delta_j^T P_j \left( A\delta_j + (d_j + g_j) B u_j - \sum_{k=1}^{N} a_{jk} B u_k \right)$$

we can determine that its optimal control policy is given by

$$u_j^* = -R_j^{-1} B^T P_j \delta_j .$$

By means of the assumption $\delta_j = -\delta_i$, agent $i$ considers that the policy of its neighbors is then given by

$$\upsilon_j^* = R_j^{-1} B^T P_j \delta_i \tag{6.35}$$

Using (6.33) and (6.35) in $H_i$, we obtain the HJ equations

$$\delta_i^T \Bigg[ Q_i + P_i A + A^T P_i - (d_i + g_i) P_i B R_i^{-1} B^T P_i + \sum_{j=1}^{N} a_{ij} P_j B R_j^{-1} B^T P_j \\ - \sum_{j=1}^{N} a_{ij} P_i B R_j^{-1} B^T P_j - \sum_{j=1}^{N} a_{ij} P_j B R_j^{-1} B^T P_i \Bigg] \delta_i = 0$$

which imply the AREs (6.34). Optimality of the control policy (6.33) is finally proven using a similar procedure as in the proof of Theorem 4.1 □

**Remark 6.3.** Notice that, in local-error projection games, each agent must solve a set of up to $N$ coupled algebraic Riccati equations of the form (6.34). This is because each ARE uses neighbor information from each agent; thus, for agent $i$ to solve the ARE of its neighbor $j$, it must also solve $j$'s neighbors equations. For the same reason, all agents must have enough information about the graph topology expressed by the weights $a_{ij}$ and $g_i$ required in the AREs.

The robustness analysis of the local-error projection strategies is performed similarly as in the previous section. Consider the perturbed system

$$\dot{\hat{\delta}}_i = A\hat{\delta}_i + (d_i + g_i) B \wp_i u_i - \sum_{j=1}^{N} a_{ij} B \wp_j u_j. \tag{6.36}$$

**Theorem 6.6.** If the perturbations $\wp_j$, $j \in i \cup \mathcal{N}_i$, of the system (6.36) are such that

$$\Big\langle \hat{\delta}_i, [(d_i + g_i) P_i B (2\wp_i - I) R_i^{-1} B^T P_i + \sum\nolimits_{j\in\mathcal{N}_i} a_{ij} P_i B (\wp_j - I) R_j^{-1} B^T P_j \\ + \sum\nolimits_{j\in\mathcal{N}_i} a_{ij} P_j B R_j^{-1} (\wp_j - I)^T B^T P_j ) \hat{\delta}_i \Big\rangle \geq 0 \tag{6.37}$$

*Proof*. Follows similarly as in the proof of Theorem 6.4. □

The comparison between the robustness properties of these solution concepts is left for Section 6.6. First, we present the simulation results of the methods presented in this chapter.

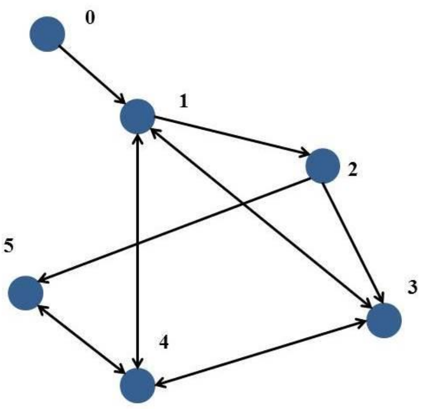

Fig. 6.2. Graph topology for simulation of graphical games strategies.

## 6.5. Simulation Results

A numerical example is here presented to test the validity of the solutions presented above. Consider a set of 5 agents and one leader connected in a communication graph as shown in Fig. 4.2. If $j \in \mathcal{N}_i$, let $a_{ij} = 1$. Each agent is taken with linear dynamics with

$$A = \begin{bmatrix} 1 & 2 \\ -2 & -1 \end{bmatrix}, \qquad B = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}.$$

For all cases, the performance indices of the agents are defined using the matrices $Q_i = 3I$ and $R_i = 2I$, for all agents $i$. Minmax regret requires to determine the matrix $\bar{K}_i$ that makes the closed-loop system marginally stable. This matrix is found to be

$$\bar{K}_i = \begin{bmatrix} 0.1667 & 0.3333 \\ -0.3333 & -0.1667 \end{bmatrix}$$

for agents $i = 1,3,4,5$. For agent $2$, the result is

$$\bar{K}_2 = \begin{bmatrix} 0.1111 & 0.2222 \\ -0.2222 & -0.1111 \end{bmatrix}.$$

The matrices $P_i$ that solve the Riccati equations are given by

$$P_1 = P_3 = P_4 = P_5 = \begin{bmatrix} 0.1089 & 0.0544 \\ 0.0544 & 0.1089 \end{bmatrix}, \qquad P_2 = \begin{bmatrix} 0.1029 & 0.0514 \\ 0.0514 & 0.1029 \end{bmatrix}.$$

The state trajectories of this simulation are shown in Fig. 6.2 and Fig. 6.3.

Graph-weight projection strategies are simulated using the same values of $R_{ij} = 2I$ for all agents $i$, $j$. The corresponding set of Riccati equations (6.24) - (6.25) is solved for each agent. The solutions of the AREs are now given as

$$P_i = \begin{bmatrix} 0.4635 & 0.0140 \\ 0.0140 & 0.36 \end{bmatrix}$$

for all agents $i$. The state trajectories in this case are shown in Fig. 6.4 and 6.5.

Finally, the local-error projection strategies are solved. The same values of $Q_i = 3I$ and $R_{ij} = 2I$ are used for all agents $i$, $j$. The simulation results are shown in Fig. 6.6 and Fig. 6.7. For the selection of performance indices in these examples, the solutions of the coupled AREs are identical to the graph-weight projection case above, given by

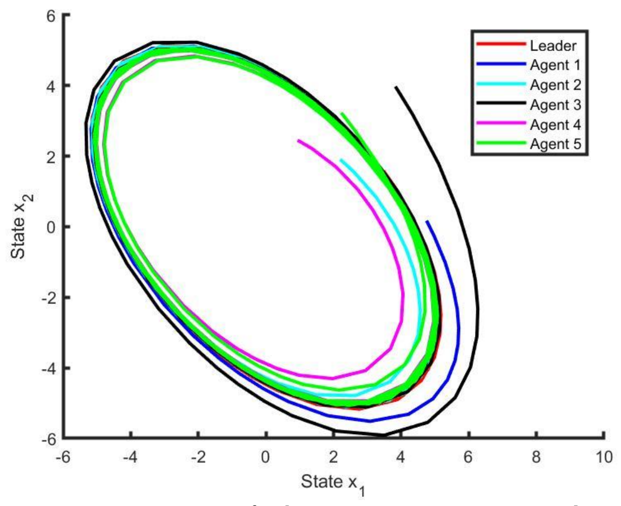

Fig. 6.3. State trajectories for the agents using minmax regret policies.

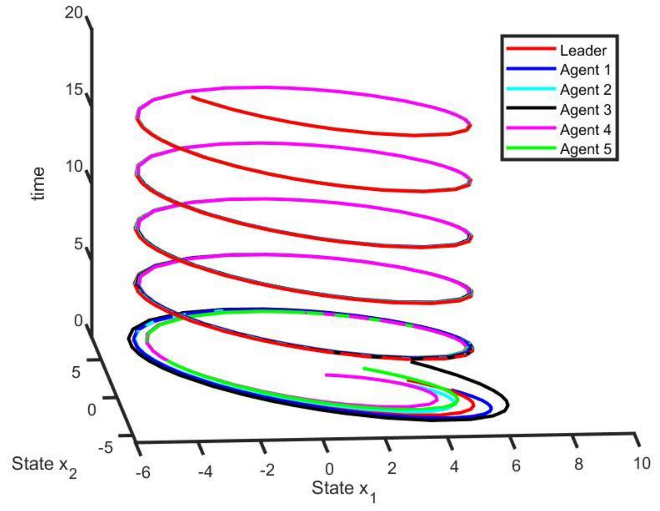

Fig. 6.4. Evolution of the state trajectories with respect to time for minmax regret policies.

$$P_i = \begin{bmatrix} 0.4635 & 0.0140 \\ 0.0140 & 0.3600 \end{bmatrix}.$$

## 6.6. Conclusions

Three alternative solutions for differential graphical games were proposed in this chapter. Each of these solution concepts presents different characteristics that make it suitable for a particular set of graphical games problems. Depending on the assumptions made by the

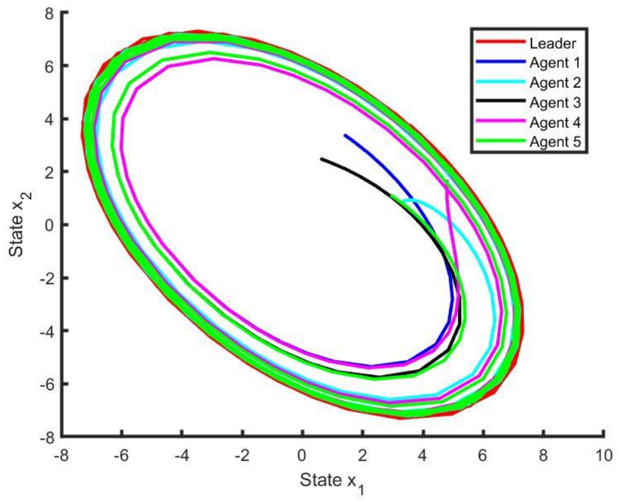

Fig. 6.5. State trajectories for the agents using graph-weight projection strategies.

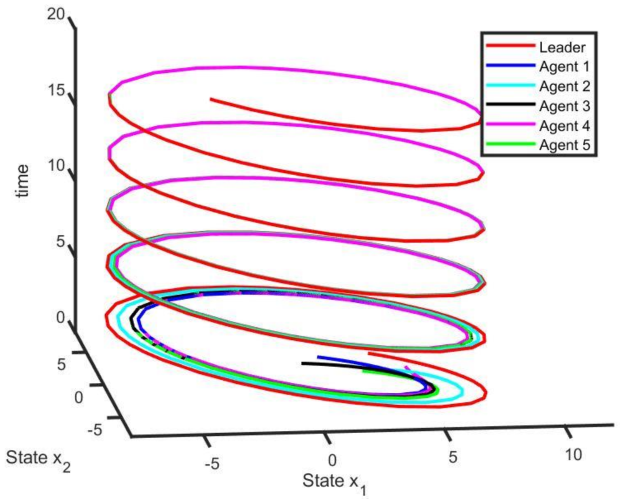

Fig. 6.6. Evolution of the state trajectories with respect to time for graph-weight projection strategies.

agents, the computational complexity, the number of equations required to be solved and the robustness properties of the control policies may vary.

The minmax regret policy is obtained by solving a single, decoupled Riccati equation as in the minmax strategies case. The conditions for such solution to exist, however, are less evident. Furthermore, additional computation steps are employed like, for example, the calculation of the matrix $\bar{K}_i$ that makes the system marginally stable. The main advantage of

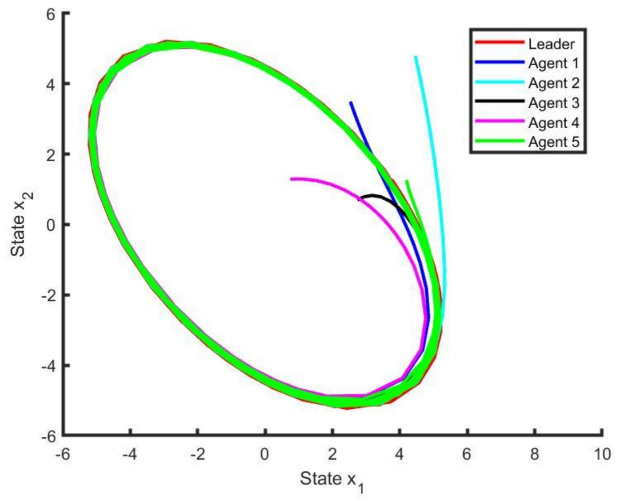

Fig. 6.7. State trajectories for the agents using local-error projection strategies.

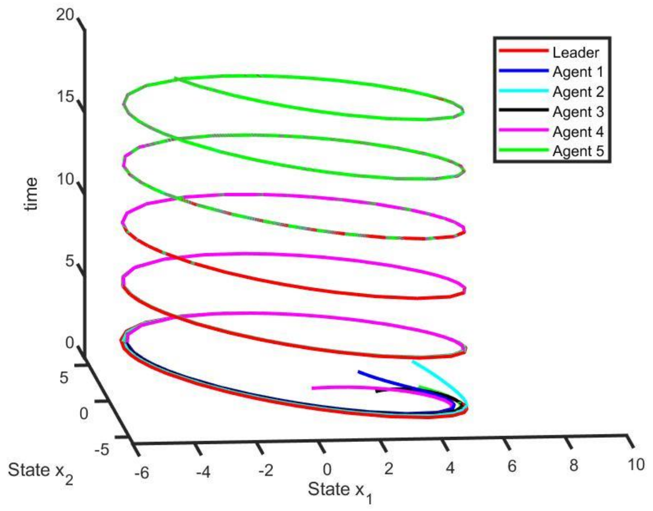

Fig. 6.8. Evolution of the state trajectories with respect to time for local-error projection strategies.

minmax regret is provided by its formulation. After all, it may be more natural for an agent to have the objective of minimizing its regret than to assume the worst behavior of its neighbors.

Graph-weight projection strategies provide control policies on which the agents make cooperative assumptions. The main consequence of this scenario is the fact that the agents must solve a set of coupled AREs, on which they employ the assumed collaborative policies of the neighbors. The number of Riccati equations that must be simultaneously solved is limited by the total number of neighbors of the agent. Furthermore, the agents do not need any knowledge of the graph topology, except the edges that link them to their neighbors.

Finally, the local-error projection strategies, were presented. To design these strategies, all agents need a complete knowledge of the graph topology. Then, they need to solve a set of coupled AREs with as many equations as the total number of agents in the game. Comparing the conditions (6.30) and (6.37), we notice the similarities overcome any minor difference between them. The simulation results, indeed, show that the solutions for graph-weight projection and local-error projection strategies may coincide under the appropriate selection of weighting matrices in the performance indices of the agents.

# Chapter 7

# CONCLUSION AND FUTURE WORK

This dissertation presented the formulation and solutions of different graphical games for dynamical systems. The most important solution concept in game theory, Nash equilibrium, is in general not attainable in graphical games due to the limited information that the players receive through the graph topology. To overcome this lack of knowledge, it is argued in this work that the agents must make assumptions about their neighbors' objectives, or about the information the neighbors have available.

The first game formulated, Bayesian games, addresses the problem of games between agents that are uncertain about each other's intentions. Each agent is designed to have beliefs about its surroundings, to update those beliefs in accordance to the evidence it obtains from its environment, and to perform its optimal behavior dependent on its beliefs. It is important to notice that, although the Bayesian games formulation was described specifically for a Bayes-Nash equilibrium result, the same belief structure can be used for any of the new solution concepts proposed in this research.

Four solution concepts were then introduced as alternatives for Nash equilibrium. These solutions can be immediately separated in two main categories: minmax solutions and projection solutions. The minmax solutions include the minmax strategies and the minmax regret strategies. In both cases, an agent makes the assumption that its neighbors will perform their worst-case policies to negatively affect its performance. The optimal policies for these strategies are provided by the solution of a single, decoupled Riccati equation. The conservative assumption that the neighbors will attempt to maximize a cost function generate robust policies in the agents.

The projection solutions include the graph-weight projection strategies and the local-error projection strategies. These solutions are characterized by the fact that an agent assumes that its neighbors and itself share the same information of some sort. Both strategies use more general formulations of the graphical game, and require the solution of many coupled Riccati equations. The graph-weight projection solution formulates a game including only its neighbors, which require solving the same number of AREs as the number of neighbors the agent has. The local-error projection strategy requires an increased number of AREs to solve. Each of these strategies is better for certain applications, depending on the total number of agents and the expected number of neighbors per agent.

The following are some of the directions for continuation of this work.

1. Design different belief-update algorithms Bayesian games to reduce the computational complexity of the Bayes rule, or to increase its efficiency in the uncertain environments provided by a graph topology.
2. Extend the proposed results for multiagent systems with nonlinear dynamics.
3. Formulate non-synchronization games in graphs to increase the scope of applicability of the proposed methods. Richer dynamics can be allowed for the agents that do not involve following the same target.
4. Consider practical implementations of the solution concepts here designed. The increasing interest in swarms of UAVs or in autonomous vehicles, for example, provide the opportunity to test the reliability of these results.

# Appendix

# DYNAMIC MULTIOBJECTIVE CONTROL USING REINFORCEMENT LEARNING

## A.1. Introduction

In Chapter 3 of this dissertation, it was shown that the solution of a Bayesian game can be determined as the solution of a multiobjective optimization problem. Similarly, many engineering problems require describing the goals of a system by means of two or more performance indices, rather than the single cost function employed in classical optimal control. Using several performance indices provides more flexibility to represent the expected behavior of the system in ways that are difficult to express otherwise. Examples of these applications can be found in [74]. The study of multiobjective optimization control is therefore a natural extension of the usual analysis in the current literature [75].

Multiobjective optimal control has been studied in [75]-[77] where the concept of Pareto domination is employed to compare the desirability of two vector functions. Using this notion, the control input is designed such that improving an objective function unilaterally implies making another worse [50], [78]. Most of the papers in the literature deal with static multiobjective optimization. In this appendix, we present a practical method of policy iteration to solve the multiobjective optimization problem for nonlinear dynamical continuous-time systems.

Several methods exist to compute a Pareto optimum. These include numerical methods embedded in software packages, as well as analytic procedures such as the weighted sum, or scalarization, technique [50]. The weighted sum method consists of combining the different cost indices into a single scalar function by computing their convex sum. This is a practical method in many applications, but it presents many technical drawbacks. These include the presence of unreachable Pareto optimal results, a strong, non-intuitive dependence on the selected sum weights, and a large computational burden as the number of desired solutions increases [50]. For these reasons, this appendix presents a general approach for multiobjective control that can be applied with any of the existing multiobjective optimization methods.

Reinforcement learning is a set of artificial intelligence methods that has had an increasing success in the last decade for providing a system with the ability to improve its performance as it gains experience while attempting to achieve its goals [70]-[72], [5]. In the last few years, reinforcement learning approaches have been adopted in control theory where the performance of a dynamical system is measured by means of a scalar function that represents the cost spent by the system along time. Reinforcement learning techniques, properly defined for control of dynamical systems, are described in [73].

The main motivation of this Appendix is to design a control strategy that allows to solve optimization problems that cannot be expressed by a single cost function. When the objectives of the system are conflicting with each other, a tradeoff must be achieved. The controller is based on reinforcement learning methods to avoid the difficult task of solving the Hamilton-Jacobi-Bellman equation [5] and to relax the need of full knowledge of the dynamic model of the system.

The appendix is organized as follows. Basic definitions for multiobjective optimization and for the multiobjective optimal control problem are described in Section A.2. Section A.3 shows the basic transformations employed to obtain an iterative suboptimal control sequence. In Section A.4, a policy iteration algorithm to solve the multiobjective optimization problem is designed, with considerations to allow its implementation in practical applications. Section A.5 studies the linear systems case. Finally, Section A.6 concludes with a numerical example.

## A.2. Multiobjective Performance of a Dynamical System

Multiobjective optimization deals with the problem of minimizing two or more objective functions simultaneously [78]. In mathematical terms, this problem is expressed as

$$\min_{x \in X} V(x) \tag{A.1}$$

where $x \in \mathbb{R}^n$ is selected inside a feasible set $X$ and $V : \mathbb{R}^n \to \mathbb{R}^M$ is a vector with $M$ elements, $V(x) = [V_1(x), \ldots, V_M(x)]^T$, with $V_i(x)$, $i = 1, \ldots, M$, the functions to be minimized. In the general case, there does not exist a solution $x$ that achieves the minimization of all functions $V_i(x)$ simultaneously, and the concepts of Pareto domination and Pareto optimality must be introduced.

**Definition A.1 (Pareto domination).** A vector $W \in \mathbb{R}^M$ is said to Pareto dominate vector $V \in \mathbb{R}^M$ if $W_j \leq V_j$ for all $j = 1, \ldots, M$, and $W_j < V_j$ for at least one $j$, where $V_j$ and $W_j$ are the $j$-th entries of vectors $V$ and $W$, respectively. Notation $W \leq V$ for vectors $W \in \mathbb{R}^M$ and $V \in \mathbb{R}^M$, indicate that $W$ is not Pareto dominated by $V$, i.e., either $V = W$ or there is at least one entry $j$ such that $V_j > W_j$. Notation $W \preceq V$ means that $W_j \leq V_j$ for all $j = 1, \ldots, M$. □

Employing these definitions, the concept of Pareto optimality can be stated as follows.

**Definition A.2 (Pareto optimality).** A solution $x^*$ of problem (A.1) is said to be Pareto optimal if $V(x^*) \leq V(x)$ for all $x \in X$. □

The outcome $V(x^*)$ of a Pareto optimal solution $x^*$ is also said to be Pareto optimal. In general, a multiobjective optimization problem has multiple Pareto optimal outcomes, and the set of all Pareto optimal outcomes for a given problem is regarded as the Pareto front. Here, we represent the Pareto front as $\mathcal{V}^\pi$, such that $V(x^*) \in \mathcal{V}^\pi$.

Consider now a dynamical system with general nonlinear dynamics

$$\dot{x} = f(x,u) \tag{A.2}$$

where $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^m$ are the state vector and the control input of the system, respectively, and $f$ is a continuously differentiable function. In the multiobjective optimization problem, the performance of system (A.2) is evaluated with respect to $M$ different performance indices

$$J_j(x(0),u) = \int_0^\infty r_j(x(0),u)dt \tag{A.3}$$

$j = 1, \ldots, M$, where each $r_j$ is a scalar, positive definite, continuously differentiable function. The feedback control function $u(x)$ is said to be *admissible* if it is continuous, stabilizes the dynamics (A.2) and makes $J_j(x,u(x))$ finite for all $j=1,\dots,M$. The class of functions satisfying these properties is denoted as $U^0$. Define the vector $J$ as $J=\left[J_1,\dots,J_M\right]^T$. It is our interest to find a function $u(x)\in U^0$ such that vector $J$ is minimized in the Pareto sense.

For a fixed control policy $u(x)$, define the value functions

$$V_j(x(t))=\int_t^\infty r_j(x(\tau),u)d\tau \tag{A.4}$$

for $j=1,\dots,M$. Furthermore, let $V=\left[V_1,\dots,V_M\right]^T$. A differential equivalent to the value function (A.4) is given by the Bellman equations

$$0=r_j(x,u)+\nabla V_j^T f(x,u)\triangleq H_j(x,\nabla V_j,u) \tag{A.5}$$

where $\nabla V_j$ is the gradient of $V_j$, and $H_j(x,\nabla V_j,u)$ is the $j$th Hamiltonian function of the system. Note that the orbital derivative of $V_j(x)$ is given by

$$\dot V_j(x)=\nabla V_j\dot x=-r_j(x,u). \tag{A.6}$$

Define also

$$V^*(x(0))\triangleq\inf_{u\in U^0}J(x(0),u) \tag{A.7}$$

where, in general, $V^*$ is not unique, and $V^*\in\mathcal{V}_J^\pi$ with $\mathcal{V}_J^\pi$ the Pareto front of vector $J$.

Define the Pareto optimal vector $H^*$ as

$$H^*(x,\nabla V)=\min_{u\in U^0}H(x,\nabla V,u) \tag{A.8}$$

where $H=\left[H_1,\dots,H_M\right]^T$ and $\nabla V=\left[\nabla V_1,\dots,\nabla V_M\right]^T$. $H^*$ is Pareto optimal in the sense that, for each state vector $x$ and vector $V$, $H^*(x,\nabla V)\le H(x,\nabla V,u)$ for every control policy $u\in U^0$.

In general, it is possible to select different control inputs, $u^{*1}$ and $u^{*2}$, such that $H(x,\nabla V,u^{*1})$ and $H(x,\nabla V,u^{*2})$ are both Pareto optimal. For this reason, we make the following assumption, useful for the analysis of the next section.

**Assumption A.1.** If there exists $u^*$ such that $H^*(x,\nabla V)=0$ for all $x$ and a given function $V(x)$, then select $u = u^*$.

Assumption A.1 is a restriction on the procedure employed to find a Pareto optimal vector in (A.8), and states that if the vector of zeros is one of the possible Pareto optimal results for $H^*$, then control policy $u$ must be selected accordingly. The consequences of Assumption A.1 are studied in Lemmas A.4 and A.5 in the following section.

## A.3. Multiobjective Suboptimal Control Sequences

This section defines and analyzes transformations to design suboptimal control policies in an iterative manner. This is an extension for multiobjective optimization of the results in [71].

Considering the multiobjective optimal control problem described in Section A.2, define $\mathcal{V}$ as the set of all continuously differentiable functions $V : \mathbb{R}^n \to \mathbb{R}^M$ such that $V(0) = 0$. Define also $\mathcal{V}^0$ as the subset of $\mathcal{V}$ such that $u(x, \nabla V) \in U^0$, i.e., the feedback control policies based on the vector function $V$ are admissible. Define the transformations $T_1$, $T_2$ and $T$ as follows.

**Definition A.4 (Function transformations)**.

1. Define the function $T_1 : \mathcal{V}^0 \to U^0$ as $T_1(V) = T_1(V_1, \ldots, V_M) = u$ where $V = [V_1, \ldots, V_M]^T$ and $u = u(x, \nabla V)$.

2. Define the function $T_2 : U^0 \to \mathcal{V}$ as $T_2(u) = V$, where $V \in \mathbb{R}^M$ and $V_j(x) = J_j(x, u)$, $j = 1, \ldots, M$.

3. Define the composite mapping $T : \mathcal{V}^0 \to \mathcal{V}$ as $T(V_1, \ldots, V_M) = T_2(T_1(V_1, \ldots, V_M)) = J(x, u)$, where $u = u(x, \nabla V)$.

□

Our objective now is to use these transformations to design a control sequence that converges to an optimal policy $u^*(x) \in U^0$. We begin our analysis by studying some of the properties of the vector functions defined in Section A.2, as well as those of transformations $T_1$, $T_2$ and $T$.

Lemma A.1 allows to compare two vector functions, $V$ and $W$, with entries $V_j$ and $W_j$ as in (A.4), when the respective Hamiltonian functions are known.

**Lemma A.1**. If $H(x,\nabla V,u)\le H(x,\nabla W,u)$ for given vector functions $V$, $W$, and control $u$, then $W\le V$.

*Proof*. By Definition A.2, we have that either $H(x,\nabla V,u)=H(x,\nabla W,u)$ or there exists an entry $j$ such that $H_j(x,\nabla V,u)<H_j(x,\nabla W,u)$. Assume the latter case and consider this same entry $j$. By definition of $H_j$ in (A.5), we have $r_j(x,u)+\nabla V_j^T f(x,u)<r_j(x,u)+\nabla W_j^T f(x,u)$, which implies $\nabla V_j^T f(x,u)<\nabla W_j^T f(x,u)$; that is, $\dot{V}_j<\dot{W}_j$. Integrating the inequality along the same motions yields $W_j<V_j$ and, therefore, $W\le V$. □

Lemma A.2 and Theorem A.1 relate the Pareto optimality of vector $V$ with the Pareto optimality of vector $H$ in (A.8).

**Lemma A.2**. Consider a control policy $u^*$ such that (A.8) holds. If $V_j^*(x)$, $j=1,\ldots,M$, solves the Bellman equation (A.5) for $u^*$, then $V^*(x)$ is Pareto optimal.

*Proof.* Consider a control policy $\bar{u}$ such that $H(x,\nabla V,\bar{u})\ne H(x,\nabla V,u^*)$. By Pareto optimality of $H_j(x,\nabla V_j,u^*)$, there exists an entry $j$ such that

$$H_j(x,\nabla V_j,\bar{u})>H_j(x,\nabla V_j,u^*). \tag{A.9}$$

For this entry $j$, let $V_j^*$ solve the Bellman equation for $u^*$ and $\bar{V}_j$ solve the Bellman equation for $\bar{u}$. Then, notice that $H_j(x,\nabla V_j^*,\bar{u})>H_j(x,\nabla V_j^*,u^*)=H_j(x,\nabla\bar{V}_j,\bar{u})=0$. Now, by Lemma A.1, $H_j(x,\nabla V_j^*,\bar{u})>H_j(x,\nabla\bar{V}_j,\bar{u})$ implies $\bar{V}_j>V_j^*$. □

**Theorem A.1**. Let the control policy $u^*$ be such that (A.8) holds, and $V^*$ such that $V_j^*$ solves the $j$ th Bellman equation (A.5) for $u^*$, for every entry $j=1,\ldots,M$. Then, $V^*\in\mathcal{V}_j^{\pi}$ with

$\mathcal{V}_J^\pi$ the Pareto front of $J$ as defined in (A.7).

*Proof.* As $u^*$ makes $H^*$ Pareto optimal, then, by Lemma A.2, $V^*$ is also Pareto optimal. Now, for all entries of vector $J$, we have

$$\begin{aligned} J_j &= \int_0^\infty r_j(x,u)dt \\ &= \int_0^\infty H_j(x,\nabla V_j^*,u)dt - V_j^*(x(\infty)) + V_j^*(x(0)) \end{aligned}$$

As $u^* \in U^0$, then $V(x(\infty)) = 0$. Therefore, $J_j = V_j^*(x(0))$ for all entries $j$, and Pareto optimality of $V^*$ implies Pareto optimality of $J$. This is $V^* \in \mathcal{V}_J^\pi$ as in (A.7). □

The proof of Theorem A.1 shows that $V^* = J$ when $V^*$ solves the Bellman equation for $u^*$. Lemma A.3 and Theorem A.2 show that solving the Bellman equation, regardless of the control function $u$, is a sufficient and necessary condition for a vector $V$ to satisfy the equality $V = T_2(u) = J$.

**Lemma A.3**. $V = T_2(u)$ if and only if $V_j$ satisfies $H_j(x,\nabla V,u) = 0$, for $j = 1,\ldots,M$.

*Proof.* If $H_j(x,\nabla V,u) = 0$, then $\dot{V}_j = \nabla V^T f(x,u) \;\; = H_j(x,\nabla V,u) - r_j(x,u) = -r_j(x,u) = \dot{J}_j$, and integrating both sides of the equality along the same motions for all entries of the vector, yields $V = J = T_2(u)$. Conversely, if $V = J$, then $\dot{V}_j = \dot{J}_j = -r_j(x,u)$, which implies $H_j = 0$. □

**Theorem A.2**. Let $V \in \mathcal{V}^0$ and $W \in \mathcal{V}$. Now, $W = T(V)$ if and only if $H(x,\nabla W,u(x,\nabla V)) = 0$.

*Proof*. Follows directly from Lemma A.3 and Definition A.4. □

Clearly, if $V$ is such that $H(x,\nabla V,u) = 0$, then $H^*(x,\nabla V) \le 0$, which means that the vector of zeros does not Pareto dominates $H^*$. However, this does not necessarily imply that all the elements of $H^*$ are nonpositive. Lemma A.4 solves this inconvenience.

**Lemma A.4**. Let Assumption A.1 hold. Then, $H_j^*(x,\nabla V) \le 0$ for all entries of $H^*$.

*Proof*. By Assumption A.1, if the vector of zeros is Pareto optimal, then $H_j^* = 0$ for all entries $j$. If $H=0$ is not Pareto optimal, then by definition of Pareto optimality we have $H_j^*(x,\nabla V)\leq 0$ for all $j$ with at least one strict inequality. □

As studied below, Lemma A.4 allows guaranteeing that all the entries of a vector are at least as small as the entries of another ($V \preceq W$) when an iterative algorithm is employed.

The following theorem shows the recursion required later in this section to design a suboptimal control sequence.

**Theorem A.3**. Let $V\in\mathcal{V}^0$ and $\bar{V}=T(V)$, and let Assumption A.1 hold. Then, $H^*(x,\nabla V)\preceq 0$ implies $V^*\leq\bar{V}\preceq V$, with $V^*$ Pareto optimal.

*Proof*. Take $u=u^*(x,\nabla V)$. By Assumption A.1 and Lemma A.4, $H_j^*(x,\nabla V)\leq 0$ for every $j=1,\ldots,M$. Then, we can express $\dot{V}_j=H_j^*(x,\nabla V)-r_j(x,u)\leq -r_j(x,u)=\dot{J}_j$.

As $\bar{V}_j=T(V)=J_j$ implies $\dot{\bar{V}}_j=\dot{J}_j$, then $\dot{V}_j\leq\dot{\bar{V}}_j$. Integrating the inequality we get $\bar{V}_j\leq V_j$ for all entries $j$. □

In the single objective optimization problem, it is clear that an iterative repetition of the operation in Theorem A.3 leads the function vector $\bar{V}$ to the unique optimal value function $V^*$. In the multiobjective optimization case, Assumption A.1 is required to prevent leaping among different Pareto optima at each iteration, as proven in Lemma A.5 and Theorem A.4.

**Lemma A.5**. Let $V^*$ be Pareto optimal and let Assumption A.1 hold. If $W^*$ is any other Pareto optimal value function such that $V^*\neq W^*$, then $W^*\neq T(V^*)$.

*Proof.* Assume $W=T(V^*)$. If Assumption A.1 holds, by Lemma A.4 we have $H_j^*(x,\nabla V)\leq 0$ for all entries $j$. By Theorem A.3, we have $W_j\leq V_j^*$ for all $j$. As $W_j^*>V_j^*$ for some $j$, for any other Pareto optimal vector $W^*$, then $W^*$ cannot be reached. □

**Theorem A.4**. If a Pareto optimal solution $V^*\in\mathcal{V}^0$ exists, then $V^*=T(V^*)$. Conversely, $V=T(V)$ implies $V=V^*$.

*Proof*. Consider two Pareto optimal vectors $V^*$ and $W^*$. By Theorem A.3, if $\bar{V} = T(V^*)$, then $W^* \leq \bar{V} \preceq V^*$; by Lemma A.5 and definition of Pareto optimality, $\bar{V} \preceq V$ implies $\bar{V} = V^*$. Conversely, if $V = T(V)$, by Theorem A.2 we have $H^*(x, \nabla V) = 0$ and $V$ solves the Bellman equations (A.5); by Lemma A.2, $V = V^*$. □

We finally formalize the idea of using the result in Theorem A.3 to build a sequence of successive approximations that converge to a Pareto optimal solution $V^*$ $V^*$.

**Theorem A.5**. Take $V^0 \in \mathcal{V}^0$ and $V^{k+1} = T(V^k)$. Then $V^* \leq V^{k+1} \preceq \cdots \preceq V^0$ for a Pareto optimal solution $V^*$.

*Proof.* The proof follows inductively from Theorem A.3, noting that the current estimate of the optimal value function at step $k$ is $V^k = T_2(u^k)$ and taking the control policy at step $k+1$ based on $V^k$, i.e., $u^{k+1} = u(x, \nabla V^k) = T_1(V^k)$. Convergence to a single Pareto optimal result is provided by Theorem A.4. □

## A.4. Integral Reinforcement Learning Algorithm for Multiobjective Suboptimal Control

In this section, we use the analysis of Section A.3 to design an integral reinforcement learning (IRL) algorithm using the structure of policy iteration [70], [73], [79], that is shown to converge to a Pareto optimal solution of vector $V$, then used to generate the optimal policy $u(x, \nabla V^*)$. Here, it is assumed that the state values of system (A.2) are known, even if part of its mathematical model is uncertain.

In [79], an IRL algorithm that converges to the solution $V^*$ of the Bellman equation for a single performance index was developed. This section presents the integral reinforcement learning in multiobjective optimization form.

Notice that the $j$ th value function (A.4) can be expressed as

$$V_j(x(t)) = \int_t^{t+T} r_j(x(\tau),u)d\tau + V_j(x(t+T)) \qquad \text{(A.10)}$$

for any time interval $T > 0$. Given the functions $V_j(x)$ and $r_j(x,u)$, equation (A.10) does not require knowledge about the system dynamics (A.2). Lemma A.6 shows that the solution $V_j(x)$ of (A.10) is the value function (A.4) that solves equation (A.5).

**Lemma A.6**. Assume the control policy $u(x)$ stabilizes the system dynamics (A.2). Then, the solution $V_j(x)$ of equation (A.10) is equivalent to the solution of the Bellman equation (A.5).

*Proof.* If equation (A.5) holds for $V_j$, then $\dot{V}_j = \nabla V_j^T f(x,u) = -r_j(x,u)$. Integrating both sides of the equation, we get

$$\int_t^{t+T} r_j(x,u)d\tau = -\int_t^{t+T} \dot{V}_j(x(\tau))d\tau = -V_j(x(t+T)) + V_j(x(t))$$

which is the same equation as (A.10). □

The following algorithm presents the multiobjective optimal controller by reinforcement learning. The policy evaluation step consists of solving Equation (A.10). This corresponds to the transformation $T_2$ in Definition A.4. The policy improvement step is based on Equation (A.8), and corresponds to the transformation $T_1$. Convergence of Algorithm A.1 is proven in Theorem A.6.

**Algorithm A.1 (IRL for Multiobjective Optimization)**

```
1: Select an admissible control policy $u^0$.
2: Solve for $V^k$ from the set of equations
   $V_j^k(x(t)) = \int_t^{t+T} r_j(x(\tau),u)d\tau + V_j^k(x(t+T))$        (A.11)
3: Update the control policy as
   $u^{k+1} = \arg\min_u H(x, \nabla V^k, u)$        (A.12)
4: Go to Step 2. On convergence, stop.
```

**Theorem A.6**. Assume there exists an admissible control input $u$ for system (A.2). Perform Algorithm A.1 such that Assumption A.1 holds in step 3. Then, Algorithm A.1 converges to a Pareto optimal solution $V^*$. Moreover, the control policy $u(x,\nabla V^*)$ optimizes the performance index vector $J$.

*Proof.* From equation (A.12) and Assumption A.1, we have that $H_j(x,\nabla V_j^k,u^{k+1}) \le 0$ for all $j=1,\ldots,M$. As function $V_j^{k+1}$ solves equation (A.11), then by Lemma A.6 $H_j(x,\nabla V_j^{k+1},u^{k+1})=0$. From both results we get $H_j(x,\nabla V_j^k,u^{k+1}) \le H_j(x,\nabla V_j^{k+1},u^{k+1})$. By Lemma A.1, this implies $V_j^{k+1} \le V_j^k$ and vector $V^{k+1}$ is not Pareto dominated by $V^k$. By Theorems A.5, these properties hold for every iteration until a Pareto optimal vector $V^*$ is obtained.

By Theorem A.1, if (A.11) holds for $V^*$, then $V^*=J^*$. Thus, $V^*$ guarantees a Pareto optimal performance of the system. □

**Remark A.1**. Equation (A.11) avoids the use of the system dynamics (A.2) in the policy evaluation step of the algorithm, and equation (A.12) requires only partial knowledge of the mathematical model of the system [79].

In the following section it is shown how to use partial knowledge of a linear system with a particular Pareto optimization solver in Algorithm A.1.

## A.5. Multiobjective Linear Quadratic Regulator

Consider a system with linear dynamics

$$\dot{x} = Ax + Bu. \tag{A.13}$$

The performance of the system is measured using $M$ different performance indices with quadratic terms, given by

$$J_j = \int_0^\infty \left(x^T Q_j x + u^T R_j u\right)dt \tag{A.14}$$

$j = 1, \ldots, M$, where $Q_j > 0$ and $R_j > 0$ are symmetric matrices. Express each of the $M$ value functions in quadratic form as

$$V_j = x^T P_j x \tag{A.15}$$

$j = 1, \ldots, M$, with $P_j = P_j^T > 0$.

In order to apply the multiobjective IRL algorithm, express the functions (A.15) in the form (A.10); that is,

$$x^T(t)P_j x(t) = \int_t^{t+T} \left( x^T Q_j x + u^T R_j u \right) d\tau + x^T(t+T)P_j x(t+T). \tag{A.16}$$

Solving this equation becomes an easier task if we employ the Kronecker product to express the term $x^T P_j x$ as $x^T(t)P_j x(t) = \mathrm{vec}(P_j)^T (x \otimes x)$, where $\mathrm{vec}(P_j)$ is the column vector obtained by stacking the columns of $P_j$. Moreover, as matrix $P_j$ is symmetric and the expression $x \otimes x$ includes all possible products of the entries of $x$, each of the vectors $\mathrm{vec}(P_j)$ and $x \otimes x$ include repeated terms. Represent these vectors after removing all the redundant terms as $\bar{p}_j$ and $\bar{x}$, respectively, which consist of $n(n+1)/2$ components. Now, we can write

$$x^T P_j x = \bar{p}_j^T \bar{x} \tag{A.17}$$

Using the expression (A.17), we rewrite equation (A.16) as

$$\bar{p}_j^T (\bar{x}(t) - \bar{x}(t+T)) = \int_t^{t+T} \left( x^T Q_j x + u^T R_j u \right) d\tau \tag{A.18}$$

and the goal is to find the values of $\bar{p}_j$ that satisfy (A.18) given the measurements $x(t)$ and $x(t+T)$, and the employed control input $u$. This objective can be achieved using recursive least squares after collecting several samples of equation (A.18) [73].

The Hamiltonian functions for this system are

$$H_j = x^T Q_j x + u^T R_j u + 2x^T P_j \left( Ax + Bu \right). \tag{A.19}$$

The optimal control policy $u^*$ for system (A.13) is the input $u = -Kx$ that makes the vector

$H = \left[H_1, \cdots, H_M\right]^T$ Pareto optimal.

Several methods can be used to determine $u^*$. Here, we propose a general procedure that allows this problem to be solved by any multiobjective optimization software package.

Substitute the policy $u = -Kx$ in each of the Hamiltonian functions (A.19), to obtain

$$H_j = x^T Q_j x + x^T K^T R_j K x + x^T P_j \left(A - BK\right) x + x^T \left(A - BK\right)^T P_j x \tag{A.20}$$

It is well known that the minimization of each individual $H_j$ with respect to $K$ is achieved using the optimal gain matrix $K^* = R_j^{-1} B^T P_j$. However, this optimization problem can be characterized differently to be programed in a multi-objective optimization solver. Theorem A.7 shows that minimizing (A.20) by means of matrix $K$ is equivalent to minimize the sum of the eigenvalues of the matrix $K^T R_j K - P_j BK - K^T B^T P_j$. To simplify the notation, define the variables

$$S_j = Q_j + K^T R_j K + P_j \left(A - BK\right) + \left(A - BK\right)^T P_j \tag{A.21}$$

and

$$S'_j = K^T R_j K - P_j BK - K^T B^T P_j. \tag{A.22}$$

The $i$th eigenvalue of a matrix $S$ is denoted as $\lambda_i(S)$.

**Theorem A.7**. Let $H_j = x^T S_j x$, where $S_j$ is the symmetric matrix (A.21). Then, solving the minimization problem

$$K^* = \arg\min_K H_j$$

is equivalent to solving the eigenvalue minimization problem

$$K^* = \arg\min_K \sum_{i=1}^{n} \lambda_i (S'_j),$$

with $S'_j$ as in (A.22).

*Proof*. Take the optimal matrix $K^*$ such that $H_j^* = x^T S_j^* x$, with $S_j^* = Q_j + K^{*T} R_j K^* + P_j(A - BK^*) + (A - BK^*)^T P_j$, is minimal; this means $x^T S_j^* x \le x^T S_j x$ for $S_j$ in (A.21) using any matrix $K$. Now we can write $x^T(S_j - S_j^*)x \ge 0$ and, therefore, $S_j - S_j^*$ is a positive semidefinite matrix. Note that for the matrices (A.21) and (A.22), we have $S_j - S_j^* = S'_j - S'^*_j$. As all the eigenvalues of $S'_j - S'^*_j$ are nonnegative, and the trace of a matrix is equal to the sum of its eigenvalues, then $\mathrm{tr}(S'_j - S'^*_j) \ge 0$, which implies $\mathrm{tr}(S'_j) \ge \mathrm{tr}(S'^*_j)$. We conclude that matrix $K^*$ generates the matrix $S'^*_j$ with minimal sum of its eigenvalues. □

By Theorem A.7, minimization of the Hamiltonian vector $H$ can be achieved by finding the gain matrix $K^*$ such that, for given matrices $P_j$, $j = 1, \dots, M$ we have

$$K^* = \arg\min_K \begin{bmatrix} \sum_{i=1}^{n} \lambda_i \left( K^T R_1 K - P_1 BK - K^T B^T P_1 \right) \\ \vdots \\ \sum_{i=1}^{n} \lambda_i \left( K^T R_M K - P_M BK - K^T B^T P_M \right) \end{bmatrix} \tag{A.23}$$

**Remark A.2**. Problem (A.23) is expressed without knowledge of matrix $A$ of the system dynamics (A.13).

Algorithm A.2 expresses the policy iteration procedure presented in Algorithm A.1, modified for the linear systems case.

**Algorithm A.2 (IRL for Linear Multiobjective Optimization)**

```
1: Select an admissible control policy $u^0 = K^0 x$.
2: Solve the set of equations (A.11) for $V^k$.
3: Solve the multiobjective optimization problem (A.23) and
   update the control policies as $u^{k+1} = K^{k+1} x$.
4: Go to Step 2. On convergence, stop.
```

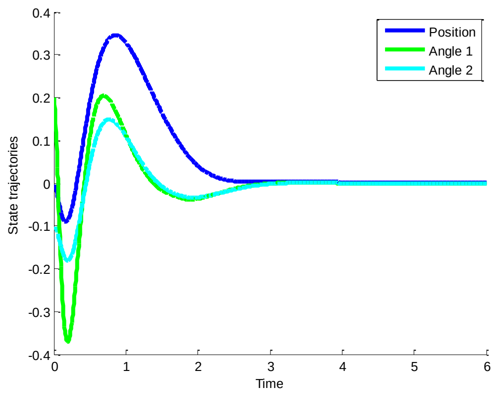

Fig. A.1. State trajectories of a linear system with multiobjective optimization.

## A.6. Simulation Results

Algorithm 2 is now employed to achieve stabilization of the linearized double inverted pendulum in a cart [86], [87], represented by the dynamic equations (A.13), where

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 86.69 & -21.61 & 0 & 0 & 0 \\ 0 & -40.31 & 39.45 & 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 6.64 \\ 0.08 \end{bmatrix},$$

state $x_1$ is the position of the cart, $x_2$ and $x_3$ are the angles of both pendulums, and the remaining states are the velocities. As performance objectives, *i*) regulation of all states is required and *ii*) the values of $x_2$ and $x_3$ must be as close to each other as possible. The performance indices (A.14) can now be defined as

$$Q_1 = \begin{bmatrix} 200 & 0 & 0 & 0 & 0 & 0 \\ 0 & 200 & 0 & 0 & 0 & 0 \\ 0 & 0 & 200 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad Q_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

and $R_1 = R_2 = 1$. The sample time per iteration is $T = 0.05$.

The Matlab function for multiobjective optimization *fgoalattain* is employed in to determine the feedback control matrix $K$ at each iteration. *fgoalattain* allows to generate different points in the Pareto front of the problem. The state trajectories for $x_1$, $x_2$ and $x_3$ after implementation of Algorithm A.2 are shown in Figure A.1. All states are shown to be stabilized by the controller. The final gain matrix $K$ is $K = \left[11.90 \quad 110.67 \quad -165.9 \quad 13.30 \quad 4.20 \quad -26.32\right]$.

## REFERENCES

[1] Y. Shoham and K. Leyton-Brown, *Multiagent systems. Algorithmic, Game-Theoretic and Logical Foundations.* New York, NY: Cambridge University Press, 2008.

[2] J. Nash, “Equilibrium points in n-person games,” *Proceedings of the National Academy of Scienes USA,* vol. 36, pp. 48–49, 1950.

[3] J. Nash, “Non-cooperative games,” *Annals of Mathematics,* vol. 54, pp. 286–295, 1951.

[4] R. Isaacs, *Differential Games. A Mathematical Theory with Applications to Warfare and Pursuit, Control and Optimization*. New York, USA: John Wiley & Sons, 1965.

[5] F. L. Lewis, D. Vrabie and V. L. Syrmos, *Optimal Control,* 3rd ed., New Jersey: John Wiley & Sons, inc., 2012.

[6] M. Johnson, S. Bhasin and W. E. Dixon, “Nonlinear two-player zero-sum game approximate solution using a policy iteration algorithm,” *in Proceedings of the IEEE Conference on Decision and Control,* pp. 142-147, 2011.

[7] S. S. Kumkov, S. Le Menec and V. S. Patsko, “Zero-sum pursuit-evasion differential games with many objects: survey of publications,” *Dynamic Games and Applications*, vol. 7, no. 4, pp. 609-633, 2017.

[8] M. Abu-Khalaf, F. L. Lewis and J. Huang, “Neurodynamic programming and zero-sum games for constrained control systems,” *IEEE Transactions on Neural Networks,* vol. 19, no. 7, pp. 1243–1252, 2008.

[9] H. Li, D. Liu and D. Wang, “Integral reinforcement learning for linear continuous-time zero-sum games with completely unknown dynamics,” *IEEE Transactions on Automation Science and Engineering*, vol. 11, No. 3, pp. 706-714, 2014.

[10] R. Song, F. L. Lewis and Q. Wei, “Off-policy integral reinforcement learning method to solve nonlinear continuous-time multiplayer nonzero-sum games,” *IEEE Transactions on Neural Networks and Learning Systems,* vol. 28, no. 3, pp. 704–713, 2017.

[11] R. Kamalapurkar, T. Dinh, P. Walters and W. E. Dixon, “Approximate optimal cooperative decentralized control for consensus in a topological network of agents with uncertain nonlinear dynamics,” *in Proceedings of the American Control Conference,* pp. 1322-1327, 2013.

[12] Y. Hong, J. Hu and L. Gao, “Tracking control for multi-agent consensus with an active leader and variable topology,” *Automatica,* vol. 47, pp. 1177–1182, 2006.

[13] F. L. Lewis, H. Zhang, K. Hengster-Movric and A. Das, *Cooperative control of multi-agent systems. Optimal and adaptive design approaches*. London: Springer-Verlag, 2014.

[14] X. Li, X. Wang and G. Chen, “Pinning a complex dynamical network to its equilibrium,” *IEEE Transactions on Circuits and Systems I. Regular papers,* vol. 51, no. 10, pp. 2074–2087, 2004.

[15] R. Olfati-Saber, J. A. Fax and R. M. Murray, “Consensus and cooperation in networked multi-agent systems,” *Proceedings of the IEEE*, vol. 95, No. 1, pp. 215–233, 2007.

[16] Z. Qu, *Cooperative control of dynamical systems: applications to autonomous vehicles.* Springer-Verlag, New York, USA, 2009.

[17] W. Ren, R. Beard and E Atkins, “A survey of consensus problems in multi-agent coordination,” *In Proceedings of the American Control Conference,* pp. 1859–1864, 2005.

[18] W. Ren, K. Moore and Y. Chen, “High-order and model reference consensus algorithms in cooperative control of multivehicle systems,” *Journal of Dynamic Systems, Measurement and Control,* vol. 129, no. 5, pp. 678–688, 2007.

[19] H. Zhang, T. Feng, G. H. Yang and H. Liang, “Distributed cooperative optimal control for multiagent systems on directed graphs: An inverse optimal approach,” *IEEE Transactions on Cybernetics,* vol. 45, no. 7, pp. 1315–1326, 2015.

[20] K. G. Vamvoudakis, F. L. Lewis and G. R. Hudas, “Multiagent differential graphical games: Online adaptive learning solution for synchronization with optimality,” *Automatica*, vol. 48, pp. 1598-1611, 2012.

[21] F. A. Yaghmaie, F. L. Lewis and R. Su, “Output regulation of heterogeneous linear multi-agent systems with differential graphical game,” *International Journal of Robust and Nonlinear Control,* vol. 26, pp. 2256–2278, 2016.

[22] H. Cao, E. Ertin and A. Arora, “Minmax equilibrium of networked differential games,” *ACM Transactions on Autonomous and Adaptive Systems,* vol. 3, no. 4, pp. 1–21, 2008.

[23] Z. Qu and M. A. Simaan, “A design of distributed game strategies for networked agents,” *IFAC Proceeding Volumes,* vol. 42, no. 20, pp. 270–275, 2009.

[24] J. C. Harsanyi, “Games with incomplete information played by Bayesian players, I-III,” *Management Science Theory*, vol. 14, No. 3, pp. 159–182, 1967.

[25] J. Deb and E. Kalai, “Stability in large Bayesian games with heterogeneous players,” *Journal of Economic Theory*, vol. 157, pp. 1041-1055, 2015.

[26] G. Carmona and K. Podczeck, “Ex-post stability of Bayes-Nash equilibria of large games,” *Games and Economic Behavior*, vol. 74, pp. 418-430, 2012.

[27] E. Cartwright and M. Wooders, “On purification of equilibrium in Bayesian games and expost Nash equilibrium,” *International Journal of Game Theory*, vol. 38, pp. 127-136, 2009.

[28] T. Basar and P. Bernhard, $H_\infty$ *-Optimal Control and Related Minmax Design Problems.* Birhauser, Boston, MA, 1995.

[29] G. Zames, “Feedback and optimal sensitivity: Model reference transformations, multiplicative seminorms and approximate inverses,” *In Proceedings of the 17th Allerton Conference,* pp. 744–752, 1979.

[30] H. Kwakernaak, “Robust control and $H_\infty$ -optimization, tutorial paper,” *Automatica,* vol. 29, no. 2, pp. 255–273, 1993.

[31] Z. Li, Z. Duan and G. Chen, “On $H_\infty$ and $H_2$ performance regions of multi-agent systems,” *Automatica,* vol. 47, pp. 797–803, 2011.

[32] J. C. Doyle, K. Glover and P. P. Khargonekar, “State-space solutions to standard $H_2$ and $H_\infty$ control problems,” *IEEE Transactions on Automatic Control,* vol. 34, no. 8, pp. 831–847, 1998.

[33] M. I. Abouheaf and M. S. Mahmoud, “Online policy iteration solution for dynamic graphical games,” presented at the 13th International Multi-Conference on Systems, Signals & Devices, Leipzig, Germany, 2016, pp. 787–797.

[34] R. Kamalapurkar, J. R. Klotz, P. Walters and W. E. Dixon, “Model-based reinforcement learning in differential graphical games,” *IEEE Transactions on Control of Network Systems,* vol. 5, no. 1, pp. 423–433, 2018.

[35] A. Al-Tamimi, F. L. Lewis and M. Abu-Khalaf, “Model-free Q-learning designs for linear discrete-time zero-sum games with application to $H_\infty$ control,” *Automatica,* vol. 43, pp. 473–481, 2007.

[36] Y. Jiang and Z. P. Jiang, “Computational adaptive optimal control for continuous-time linear systems with completely unknown dynamics,” *Automatica,* vol. 48, pp. 2699–2704, 2012.

[37] H. Modares, F. L. Lewis and Z. P. Jiang, “$H_\infty$ tracking control of completely unknown continuous-time systems via off-policy reinforcement learning,” *IEEE Transactions on Neural Networks and Learning Systems,* vol. 26, no. 10, pp. 2550–2562, 2015.

[38] B. Luo, T. Huang, H. N. Wu and X. Yang, “Data-driven $H_\infty$ control for nonlinear distributed parameter systems,” *IEEE Transactions on Neural Newtorks and Learning Systems,* vol. 26, no. 11, pp. 2949–2961, 2015.

[39] J. von Newman and O. Morgenstern, *Theory of games and economic behavior.* Princeton University Press, Princeton, NJ, 1944.

[40] P. Kumar and J. Van Schuppen, “On Nash equilibrium solutions in stochastic dynamic games,” *IEEE Transactions on Automatic Control*, vol. 25, No. 6, pp. 1146-1149, 1980.

[41] W. Lin, Z. Qu and M. A. Simaan, “Nash strategies for pursuit-evasion differential games involving limited observations,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 51, No. 2, pp. 1347-1356, 2015.

[42] P. S. Sastry, V.V. Phansalkar and M. A. L. Thathachar, “Decentralized learning of Nash equilibria in multi-person stochastic games with incomplete information,” *IEEE Transactions on Systems, Man and Cybernetycs*, vol. 24, No. 5, pp. 769-777, 1994.

[43] H. K. Khalil, *Nonlinear Systems.* Prentice Hall, Upper Saddle, NJ, 2nd ed., 1996.

[44] M. G. Safonov and M. Athans, “Gain and phase margin for multiloop LQG regulators,” *IEEE Transactions on Automatic Control,* vol. 22, no. 2, pp. 173–179, 1977.

[45] A. Isidori, *Nonlinear Control Systems*. Springer-Verlag, London, UK, 3rd ed., 1995.

[46] Z. Li, Z. Duan, G. Chen and L. Huang, “Consensus of multiagent systems and synchronization of complex networks: A unified viewpoint,” *IEEE Trans. Circuits and Systems*, vol. 57, No. 1, pp. 213–224, 2010.

[47] A. Jadbabaie, P. Molavi, A. Sandroni and A. Tahbaz-Salehi, “Non-Bayesian social learning,” *Games and Economic Behavior*, vol. 76, pp. 210-225, 2012.

[48] Q. Zhu, H. Tebine and T. Basar, “Heterogeneus learning in zero-sum stochastic games with incomplete information,” presented at the 49th IEEE Conference on Decision and Control, Atlanta, USA, 2010.

[49] P. M. Djuric and Y. Wang, “Distributed Bayesian learning in multiagent systems: Improving our understanding of its capabilities and limitations,” *IEEE Signal Processing Magazine*, vol. 29, No. 2, pp. 65-76, 2012.

[50] M. Caramia and P. Dell’Olmo, “Multi-objective optimization,” in *Multi-objective Management in Freight Logistics. Increasing capacity, service level and safety with optimization algorithms*. London: Springer-Verlag, 2008.

[51] R. Song, W. Xiao and H. Zhang, “Multi-objective optimal control for a class of unknown nonlinear systems based on finite-approximation-error ADP algorithm,” *Neurocomputing*, vol. 119, pp. 212–221, 2013.

[52] T. M. Mitchell, *Machine Learning.* McGraw-Hill, 1997.

[53] H. Zhang, F. L. Lewis and Z. Qu, “Lyapunov, adaptive and optimal design techniques for cooperative systems on directed communication graphs,” *IEEE Transactions on Industrial Electronics,* vol. 59, no. 7, pp. 3026–3041, 2012.

[54] J. W. Brewer, “Kronecker products and matrix calculus in system theory,” *IEEE Transactions on Circuits and Systems,* vol. 25, no. 9, pp. 772–781, 1978.

[55] H. N. Wu and B. Luo, “Neural network based online simultaneous policy update algorithm for solving the HJI equation in nonlinear $H_\infty$ control,” *IEEE Transactions on Neural Networks and Liearning Systems,* vol. 23, no. 12, pp. 1884–1895, 2012.

[56] A. Bryson and Y.-C. Ho, *Applied optimal control*. New York, USA: Taylor & Francis Group, 1975.

[57] M. A. Foley and W. E. Schmitendorf, “A class of differential games with two pursuers versus one evader,” *IEEE Trans. on Automatic Control*, vol. 19, no. 3, pp. 239-243, 2003.

[58] E. Bakolas and P. Tsiotras, “Relay pursuit of a maneuvering target using dynamic Voronoi diagrams,” *Automatica*, vol. 49, no. 9, pp. 2213-2220, 2012.

[59] J. S. Jang and C. J. Tomlin, “Control strategies in multi-player pursuit and evasion game,” presented at the AIAA Guidance, Navigation and Control Conference and Exhibit, San Francisco, USA, 2005.

[60] S. D. Bopardikar and F. Bullo, “On discrete-time pursuit-evasion games with sensing limitations,” *IEEE Trans. on Robotics*, vol. 24, no. 6, pp. 1429-1439, 2008.

[61] M. M. Zavlanos and G. J. Pappas, “Distributed hybrid control for multiple-pursuer multiple-evader games,” *in Proceedings of the 10th international conference on Hybrid Systems: computation and control*, pp. 787-789, 2007.

[62] D. M. Stipanovic, A. Melikyan and N. Hovakimyan, “Guaranteed strategies for nonlinear multi-player pursuit-evasion games,” *International Game Theory Review*, vol. 12, no. 1, pp. 1-17, 2010.

[63] J. Ge, L. Tang, J. Reimann and G. Vachtsevanos, “Suboptimal approaches to multiplayer pursuit-evasion differential games,” presented at the AIAA Guidance, Navigation and Control Conference and Exhibit, Keystone, CO, USA, 2006.

[64] D. Li and J. B. Cruz, “Graph-based strategies for multi-player pursuit evasion games,” *in Proceedings of the 46th IEEE Conference on Decision and Control*, pp. 4063-4068, 2007.

[65] W. Ren, R. W. Beard and E. M. Atkins, “Information consensus in multivehicle cooperative control,” *IEEE Control Systems*, vol. 27, no. 2, pp. 71-82, 2007.

[66] H. Zhang, Z. Li, Z. Qu and F. L. Lewis, “On constructing Lyapunov functions for multi-agent systems,” *Automatica*, vol. 58, pp. 39-42, 2015.

[67] Y. Cao, W. Yu, W. Ren and G. Chen, “An overview of recent progress in the study of distributed multi-agent coordination,” *IEEE Trans. on Industrial Informatics*, vol. 9, no. 1, pp. 427-438, 2013.

[68] J. A. Fax and R. M. Murray, “Information flow and cooperative control of vehicle formations,” *IEEE Trans. on Automatic Control*, vol. 49, no. 9, pp. 1465-1476, 2004.

[69] Z. Li, Z. Duan, W. Ren and G. Feng, “Containment control of linear multi-agent systems with multiple leaders of bounded inputs using distributed continuous controllers,” *International Journal of Robust and Nonlinear Control*, vol. 25, pp. 2101-2121, 2014.

[70] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An introduction*, The MIT Press, Cambridge, MA, 1998.

[71] R. J. Leake and R. W. Liu, “Construction of suboptimal control sequences,” J. SIAM Control, vol. 5, No. 1, pp. 54-63, 1967.

[72] D. P. Bertsekas, “Dynamic programming and suboptimal control: a survey from ADP to MPC,” European Journal of Control, vol. 11, pp. 310-334, 2005.

[73] K. G. Vamvoudakis, H. Modares, B. Kiumarsi and F. L. Lewis, “Game theory-based control system algorithms with real-time reinforcement learning,” *IEEE Control Systems Magazine,* pp. 33-52, 2017.

[74] G. P. Liu, J. B. Yang and J. F. Whidborne, *Multiobjective Optimisation and Control.* Research Studies Press, 2003.

[75] A. Gambier and E. Badreddin, “Multi-objective optimal control: An overview,” presented at the *IEEE Int. Conf. on Control Applications,* Oct. 1-3, 2007.

[76] F. Logist, S. Sager, C. Kirches and J. F. Van Impe, “Efficient multiple objective optimal control of dynamic systems using integer controls,” *Journal of Process Control,* vol. 20, pp. 810-822, 2010.

[77] A. Kumar and A. Vladimirsky, “An efficient method for multiobjective optimal control and optimal control subject to integral constraints,” *Journal of Computational Mathematics,* vol. 28, No. 4, pp. 517-551, 2010.

[78] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University press, New York, 2004.

[79] D. Vrabie, O. Pastravanu, M. Abu-Khalaf and F. L. Lewis, “Adaptive optimal control for continuous-time linear systems based on policy iteration,” *Automatica*, vol. 45, pp. 477-484, 2009.

[80] D. Vrabie, K. G. Vamvoudakis and F. L. Lewis, *Optimal Adaptive Control and Differential Games by Reinforcement Learning Principles*, The Institution of Engineering and Technology, London, UK, 2013.

[81] D. Liu, Q. Wei, D. Wang, X. Yang and H. Li, *Adaptive Dynamic Programming with Applications in Optimal Control*, Springer International Publishing, 2017.

[82] F.-Y. Wang, H. Zhang and D. Liu, “Adaptive dynamic programming: An introduction,” *IEEE Computational Intelligence Magazine*, pp. 39-47, 2009.

[83] R. Kamalapurkar, L. Andrews, P. Walters and W. E. Dixon, “Model-based reinforcement learning for infinite-horizon approximate optimal tracking,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, No. 3, pp. 753-758, 2017.

[84] T. Bian, Y. Jiang and Z.-P. Jiang, “Adaptive dynamic programming and optimal control of nonlinear nonaffine systems,” *Automatica*, vol. 50, pp. 2624-2632, 2014.

[85] Q. Yang and S. Jagannathan, “Reinforcement learning controller design for affine nonlinear discrete-time systems using online approximators,” *IEEE Trans. on Systems, Man and Cybernetics-Part B: Cybernetics*, vol. 42, no. 2, pp. 377-390, 2012.

[86] Q.-R. Li, W.-H. Tao, N. Sun, C.-Y. Zhang and L.-H. Yao, “Stabilization control of double inverted pendulum system,” presented at the *3rd Int. Conf. on Innovative Computing Information and Control,* Jun. 18-20, 2008.

[87] J.-L. Zhang and W. Zhang, “LQR self-adjusting based control for the planar double inverted pendulum,” *Physics Procedia,* vol. 24, Part C, pp. 1669-1676, 2012.

[88] Y. Song, Y. Wang and C. Wen, “Adaptive fault-tolerant PI tracking control with guaranteed transient and steady-state performance,” *IEEE Trans. on Automatic Control*, vol. 62, no. 1, pp. 481-487, 2017.

[89] Y. Song, X. Huang and C. Wen, “Tracking control for a class of unknown nonsquare MIMO nonaffine systems: A deep-rooted information based robust adaptive approach,” *IEEE Trans. on Automatic Control*, vol. 61, no. 10, pp. 3227-3233, 2016.

[90] H. Liu, G. Xie and L. Wang, “Necessary and sufficient conditions for containment control of networked multi-agents systems,” *Automatica*, vol. 48, pp. 1415-1422, 2012.

[91] W. Li, “A dynamics perspective of pursuit-evasion: Capturing and escaping when the pursuer runs faster than the agile evader,” *IEEE Trans. on Automatic Control*, vol. 62, no. 1, 2017.

[92] T. Basar and G. J. Olsder, *Dynamic Noncooperative Game Theory.* SIAM, Philadelphia, PA, 2 edition, 1999.

[93] K. Doya, “Reinforcement learning in continuous time and space,” *Neural Computing*, vol. 2, no. 1, pp. 219–245, 2000.

[94] B. Luo, H. Wu, and T. Huang, “Off-policy reinforcement learning for control design,” *IEEE Transactions on Cybernetics*, vol. 45, no. 1, pp. 65-76, 2015.

[95] B. M. Chen, *Robust and* $H_\infty$ *Control.* Springer-Verlag, London, UK, 2000.

[96] L. J. Savage, *The Foundations of Statistics.* Dover Press, Mineola, NY, 2 edition, 1972.

[97] G. Loomes and R. Sugden, “Regret theory: an alternative theory of rational choice under uncertainty,” *Economic Journal,* vol. 92, pp. 805–824, 1982.

[98] D. E. Bell, “Regret in decision making under uncertainty,” *Operations Research,* vol. 30, pp. 961–981, 1982.

[99] F. L. Lewis and D. Vrabie, “Reinforcement learning and adaptive dynamic programming for feedback control,” *IEEE Circuits and Systems Magazine,* pp. 32–50, 2009.

[100] J. Li, H. Modarez, T. Chai, F. L. Lewis and L. Xie, “Off-policy reinforcement learning for synchronization in multiagent graphical games,” *IEEE Transactions on Neural Networks and Learning Systems,* vol. 28, no. 10, pp. 2434–2445, 2017.

[101] H. Kwakernaak and R. Sivan, *Linear Optimal Control Systems.* John Wiley & Sons, New York, USA, 1972.

[102] H. Zhang, F. L. Lewis and A. Das, "Optimal design for synchronization of cooperative systems: state feedback, observer and output feedback," *IEEE Transactions on Automatic Control,* vol. 56, no. 8, pp. 1948–1952, 2011.

[103] G. P. Papavassilopoulos and J. B. Cruz, Jr., "On the existence of solutions to coupled matrix Riccati differential equations in linear quadratic Nash games," *IEEE Transactions on Automatic Control,* vol. 24, pp. 127–129, 1979.

## Biographical Information

Victor G. Lopez received the B.S. degree from the Universidad Autonoma de Campeche, Mexico, in 2010 and the M.S. degree from the Research and Advanced Studies Center (CINVESTAV), Mexico, in 2013. He is currently a Ph.D. student at the University of Texas at Arlington, TX, USA. Victor was a Lecturer at the Western Technologic Institute of Superior Studies (ITESO) in Guadalajara, Mexico, in 2015. His research interests include cyber-physical systems, game theory, distributed control, reinforcement learning and robust control.