K-Resilient Nash Equilibria in Multi-Player Concurrent Reachability Games

A Thesis Submitted
in Partial Fulfilment of the Requirements
for the Degree of

Master of Technology

by

Mohammed Rizwan Rawani
Roll No. : 11111030

under the Supervision of

Prof. Anil Seth

to the

Department of Computer Science and Engineering

Indian Institute of Technology Kanpur

June, 2014

CERTIFICATE

It is certified that the work contained in this thesis entitled "*K-Resilient Nash Equilibria in Multi-Player Concurrent Reachability Games*", by *Mohammed Rizwan Rawani (Roll No. 11111030)*, has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Anil Seth

June, 2014

Dr. Anil Seth

Professor
Department of Computer Science and Engineering
Indian Institute of Technology Kanpur
Kanpur - 208016

Abstract

Game theoretic concepts play a very important role in formal specification and verification of systems. Systems that involve multiple agents interacting with each other can be modelled as multi-player games. Game theoretic solution concepts can then be used to study the properties of such systems.

In a multi-player game, every player has a set of strategies to choose from. Players play from among the strategies available to them. A strategy profile is a combination of strategies, one for each player. A strategy profile decides the complete outcome of the game. For every outcome of the game, each player has an associated payoff. Players are rational and play to maximize their respective payoffs. Players may also be allowed to play mixed (randomized) strategies (where they assign probabilities to each of the available strategies and then randomly select a particular strategy). In such cases expected values of payoffs are used to characterize the game.

A Nash equilibrium is a strategy profile such that no player has an incentive on unilaterally deviating from its strategy. If mixed strategies are allowed, every game has at least one Nash equilibrium. The solution concept of Nash equilibrium is widely used to get stable configurations and strategy profiles of a multi-agent system. But, a Nash equilibrium is susceptible to deviations by more than one player. Players can form coalitions and deviate (to try increasing their payoffs) thus disturbing the stability of the system.

In this work, we consider non-deterministic multi-player concurrent reachability games and we study the solution concept of *k-resilient* Nash equilibrium in these games. A *k-resilient* Nash equilibrium is a strategy profile such that, if at most k players deviate from their respective strategies, none of the deviating player can benefit. The solution concept of *k-resilient* Nash equilibrium is resilient to deviations by upto at most k players, thus giving more stable configurations and strategy profiles. But a *k-resilient* Nash equilibrium may not exist in general.

In this work, we prove some properties associated with non-deterministic multi-player concurrent reachability games that characterize *k-resilient* Nash equilibria in these games when only pure strategies (non-randomized strategies) are allowed. These properties are then used to develop algorithms for checking existence of *k-resilient* Nash equilibrium in finite non-deterministic multi-player concurrent reachability games in untimed and timed settings when only pure strategies are allowed. Our algorithms are in 2-EXPTIME. The properties that we prove also contain all the necessary information to compute a *k-resilient* Nash equilibrium if it exists.

...dedicated to my parents...

Acknowledgements

I would like to express my sincere gratitude towards my thesis supervisor Prof. Anil Seth for his constant support and encouragement. He gave me complete freedom to work according to my interests and comfort. The meetings and discussions that I had with him gave a proper direction to my efforts.

I would also like to thank the faculty and staff of the Department of CSE for the beautiful academic environment that they have created here. I also thank the administrative staff of IIT Kanpur for the constant efforts they put to make everything run smoothly.

I am thankful to all my friends here at IIT Kanpur. I got to learn a lot with them and I got the company of some really nice people who made these two years memorable for me. In particular, I would like to thank Akhil, Ajay, Ankit, Ashu, Atanu, Hrishit, Manash, Mangat, Nitesh, Parveen, Prashant, Shahbaz, Sourabh and Vinay for all the learning and fun we had together. I would also like to thank Tejas for really looking after the Y11 batch like an elder brother. He has been a great friend and a great guide, someone we all look up to.

I thank the office staff and workers in my hostel, especially, the mess workers, the canteen workers, gardeners, sweepers and washer-men. These people have had a very important role in making my stay at IIT Kanpur really comfortable.

I would like to thank my cousin Dr. Parvez Memon and his family who live in Kanpur. I frequently visited them during my stay here and got a homely environment.

I am thankful to my best friends Nupoor, Shubham and Tanya, who for the last five years, have always been there for me.

Last, but not the least, I would like to thank my parents and my brother for their love, constant support and encouragement. Without their support and patience this work would not have been possible.

Mohammed Rizwan Rawani

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In today's world, we are surrounded by complex systems. We use automated devices for many tasks in our daily life. In many cases, failure to ensure the correct functioning of these devices can lead to heavy losses. For example, an ATM machine working incorrectly can cause heavy financial loss. Similarly, incorrect functioning of the flight control systems installed in an aeroplane can put a risk on the lives of people aboard. Therefore, it is very important to ensure that a system functions correctly and has the desired properties. While system testing prior to usage can help find and resolve errors, it cannot ensure that the system is error free. To ensure that the system is free of design errors, formal specification and verification techniques come in play.

Formal specification and verification techniques work by mathematically modelling the system and its specification and then mathematically verifying that the system model has the desired properties. There are many approaches to formal specification and verification. One such approach is to use game theoretic concepts. Systems that involve multiple agents interacting with each other can be modelled as multi-player games. Game theoretic solution concepts can then be used to study the properties of such systems.

1.1 Games, Strategies and Equilibria

In this section, we briefly describe the terminology used in game theory literature and required for this thesis. We formally define these terms in Chapter 2.

Games: A game consists of a finite number of players. Players play from among a number of available strategies. A combination of strategies, one for each player, decides the outcome of the game. Every outcome of the game has an associated payoff vector. A payoff vector is actually a tuple of real valued payoffs, one for each player. Players are rational and play to maximize their respective payoffs. Players may also be allowed to play mixed strategies (where they assign probabilities to each of the available strategies and then randomly select a particular strategy). In such cases expected values of payoffs are used to characterize the game.

A game can be represented using a matrix which lists the payoff vectors for every possible combination of pure strategies (non-randomized strategies) of players. Such a representation is called normal form representation of a game. For example, Table 1.1 shows the normal form representation of a two player rock-paper-scissors game. The strategies of first player (or row player) are listed in rows and the strategies of second player (or column player) are listed in columns. The resulting payoff vectors are written in corresponding cells. In a payoff vector, the first value is the payoff received by row player and the second value is the payoff received by column player.

Table 1.1: Rock-Paper-Scissors game in normal form.

	Rock	Paper	Scissors
Rock	0 , 0	-1 , 1	1 , -1
Paper	1 , -1	0 , 0	-1 , 1
Scissors	-1 , 1	1 , -1	0 , 0

Games played on graphs: Normal form representation of games is pretty exhaustive and is often inconvenient. If the number of players and available strategies is large, normal form representation is practically not possible. Various other succinct representations have been proposed. Games played on graphs is one such representation. Games played on graphs are very important and useful in formal modelling of systems and in formal specification and verification [14].

In this representation, a system is modelled as a state transition diagram. The states represent various configurations of the system. These games may be turn-based games or concurrent games. In turn-based games, in each state, only one player decides the next state. In concurrent games, in each state, every player chooses an action from a set of allowed actions. The combination of actions (one for each player) then determines the next state. The payoffs of the players are defined with respect to the runs (paths) in this graph. Players may have qualitative objectives (0-1 objectives) like reachability objectives, safety objectives, Büchi objectives etc. Players may also have quantitative objectives where a real valued payoff is associated to each run. This may be done by defining for each player, a preference ordering on the runs of the graph.

A strategy for a player is a mapping that associates every finite path in the graph to an action available to the player in the last state of that path. A strategy profile is a tuple of strategies, one for each player. A strategy profile determines the complete outcome of the game.

Nash equilibrium: A Nash equilibrium is a strategy profile such that no player has a benefit on unilaterally deviating from its strategy. If mixed strategies are allowed, every game has at least one Nash equilibrium [16]. In the rock-paper-scissors game shown in Table 1.1, there is a mixed strategy Nash equilibrium when both the players randomize uniformly between the three pure strategies available to them.

K-resilient Nash equilibrium: A *k-resilient* Nash equilibrium is a strategy profile such that, if at most k players deviate from their respective strategies, none of the deviating player can benefit. Such a strategy profile is *resilient* to deviations by upto at most k players [1, 2]. Thus, a Nash equilibrium is actually a *1-resilient* Nash equilibrium. A *k-resilient* Nash equilibrium may not exist in general [1, 2].

1.2 Our Contribution and Motivation

Contribution of this thesis: We consider non-deterministic multi-player concurrent reachability games as defined in [6, 7] and we prove some properties that characterize *k-resilient* Nash equilibria in these games when only pure strategies are allowed. We then use these properties to develop algorithms for checking existence of *k-resilient* Nash equilibrium in finite non-deterministic multi-player concurrent reachability games in untimed and timed settings when only pure strategies are allowed. For this purpose, we generalize the concept of *suspect players* [6–9] and *repellor sets* [6–8] to *k-suspect coalitions* and *k-repellor sets* respectively. The properties that we prove also contain all the necessary information to compute a *k-resilient* Nash equilibrium if it exists. Our algorithms, constructions and proofs are a natural generalization of those in [6, 7] used to characterize Nash equilibria in non-deterministic multi-player concurrent reachability games.

Motivation: Although, a Nash equilibrium can tolerate unilateral deviations and hence be considered a stable strategy profile (considering that the players are rational), it is susceptible to deviations by more than one player. Players can form coalitions and deviate in order to increase their payoffs. This may disturb the stability of the system. Therefore, it makes sense to study solution concepts that are resistant to deviations by coalitions of players also. Although, one might assume that a coalition will deviate only if all the members of the coalition have some incentive (as the players are rational) and some equilibrium concepts have been proposed considering only such deviations [4, 5, 15], a *k-resilient* Nash equilibrium is an even stronger notion as it also considers deviations where even only one player of the coalition can benefit. Abraham et al. [1] discuss several reasons to consider such deviations. There may be situations where only one player effectively controls the coalition. This can happen in a network if a player can hijack nodes in the network. A player may threaten other players or persuade them to deviate by promising side payments. This notion can also take care of situations where players make arbitrary moves. Abraham et al. [1, 2] consider that there would be some kind of communication among the deviating coalition (as they use mediators to implement resilient strategies).

We claim that deviations by multiple players are possible even without any prior communication. We describe this scenario considering the risk taking behaviour of the players. Consider a strategy profile which is a Nash equilibrium. No player has an incentive on unilaterally deviating from its respective strategy. So, if communication between players is not allowed, it seems that this would be a perfectly stable strategy profile (as the players are rational). However, in real world situations, players might exhibit risk taking behaviour. If a player is not happy with his payoff, he might want to take a risk by deviating in the hope that some other player(s) would also deviate (reasoning similarly), and hence he may be benefited. While taking such risks, he might take into account his loss if any, in case no one else deviates and his deviation is unilateral. He may be ready to bear this loss considering his dissatisfaction with his current payoff and the benefits he may gain if the risk turns out in his favour. If the Nash equilibrium that we considered is a weak Nash equilibrium, a player may take such risks without the fear of any loss in case no one else deviates and his deviation is unilateral. For example, in games with qualitative objectives such as reachability games, a player can have the payoff of either 0 or 1. In such games, if a player has a payoff of 0 in the Nash equilibrium selected, he may like to deviate and take a risk because he has nothing to lose. Thus, the risk taking ability can be a threat to the stability of the system. Now consider a strategy profile which is a *k-resilient* Nash equilibrium. In this case, if a player takes such a risk, it would be required that more than k players deviate for the risk to turn out in his favour. Higher the value of k, lower are the chances that the risk turns out in his favour. Therefore higher the resilience, lower is the risk taking ability of the players and hence more stable is the system.

1.3 Related Work

Games played on graphs have been extensively used in formal specification and verification techniques. Henzinger [14] reviews some important work in this area. There has also been a lot of work on Nash equilibria and other related solution concepts (like subgame-perfect equilibria and secure equilibria) in turn-based multi-player games. Grädel and Ummels [11] present a very nice survey of the work in this area. An important result is that every deterministic turn-based game with Borel winning conditions has a pure strategy Nash equilibrium [10].

Recently, there has been a focus on multi-player concurrent games. The existence problem of Nash equilibrium in finite non-deterministic multi-player concurrent reachability games is NP-complete [6, 7, 9] whereas it is decidable in polynomial time for games with Büchi objectives [8, 9]. For timed games with reachability objectives, existence problem of Nash equilibrium is EXPTIME-complete [6, 7, 9]. Similar results exist for games with safety objectives, co-Büchi objectives, Rabin objectives and parity objectives [9].

Various solution concepts have been proposed to take care of deviations by more than one player. Halpern [12, 13] gives a brief review of these solution concepts. In [4], a concept of strong equilibrium is proposed. A strong equilibrium is a strategy profile such that no coalition of players can deviate in such a way that all its members benefit. In [5], coalition-proof Nash equilibrium is proposed which is a strategy profile such that no coalition has a self enforcing deviation. A self enforcing deviation is a deviation by a coalition such that all its members benefit from the deviation and no subset of this coalition has a further self enforcing deviation possible. The argument for considering only self enforcing deviations is that among the deviations considered for strong equilibrium, only self enforcing deviations may actually be taken. One may argue that if a deviation by a coalition is not self enforcing, players in the coalition would not take the deviation (as a subset of the coalition could cheat on them). According to the definitions, every strong equilibrium is also a coalition-proof Nash equilibrium. In [15], coalition-proof correlated equilibrium is studied which is similar to coalition-proof Nash equilibrium but considers correlated strategies. All these solution concepts consider only those deviations where all the members of the deviating coalition can benefit from the deviation. This is due to the natural assumption that a player would not deviate with a coalition if he has no incentive in doing so. However, recently Abraham et al. [1, 2] proposed the solution concept of *k-resilient* Nash equilibrium which also considers the deviations in which even one member of the coalition can benefit. The reasons to consider such deviations as given in [1] have been discussed in the previous section. Abraham et al. [1, 2] prove that *k-resilient* Nash equilibria exist in rational secret sharing and multi-party computation games with mediators, with some bounds on the values of k and n (total number of players). They also give various bounds and conditions on when can the mediator be simulated using cheap talk. In all the above works, some kind of prior communication is assumed before the deviation. However, we feel that deviation by multiple players is also possible without any prior communication. This is due to the risk taking behaviour that players may exhibit as explained in the previous section.

1.4 Organization of the Thesis

In Chapter 2, we discuss non-deterministic multi-player concurrent reachability games and we prove some properties that characterize *k-resilient* Nash equilibria in these games. We then use these properties to develop the algorithm to check the existence of *k-resilient* Nash equilibrium in finite non-deterministic multi-player concurrent reachability games when only pure strategies are allowed. The properties that we prove also contain all the necessary information to compute a *k-resilient* Nash equilibrium if it exists. Chapter 3 discusses multi-player timed concurrent reachability games and *k-resilient* Nash equilibrium in these games. Chapter 4 gives a brief conclusion of the thesis and discusses some possible future extensions of the work in this thesis. In Appendix A, we give a brief review of the theory of timed automata from [3] required for proper understanding of our work in timed games.

Chapter 2

Multi-Player Concurrent Reachability Games

In this chapter, we first describe the notion of non-deterministic multi-player concurrent reachability games and define the solution concept of *k-resilient* Nash equilibrium in these games. We then develop mathematical tools (*k-suspect coalitions* and *k-repellor sets*) to characterize *k-resilient* Nash equilibria in these games when only pure strategies are allowed. We then prove some properties associated with *k-suspect coalitions* and *k-repellor sets*. These properties are then used to develop the algorithm to check the existence of *k-resilient* Nash equilibrium in finite non-deterministic multi-player concurrent reachability games when only pure strategies are allowed. The properties that we prove also contain all the necessary information to compute a *k-resilient* Nash equilibrium if it exists.

2.1 Preliminaries

In this section, we recall the definitions of some basic terms from [6, 7]. These terms are used throughout the thesis.

Definition 2.1.1. A transition system S is defined as a 2-tuple, $S = (States, Edg)$ where:

- $States$ is a possibly uncountable set of states.
- $Edg \subseteq States \times States$ is the set of transitions (edges).

A finite transition system is a transition system in which the set of states ($States$) is finite.

Given a transition system S, following terms are defined with respect to S:

1. A path π in S, is a non-empty sequence $(s_i)_{0 \leq i < n}$ (where $n \in \mathbb{N} \cup \{+\infty\}$) of states of S such that $(s_i, s_{i+1}) \in Edg$ for every $i < n - 1$.
2. The length of a path $\pi = (s_i)_{0 \leq i < n}$ is denoted by $|\pi|$. $|\pi| = n - 1$.
3. The set of finite paths (or *histories*) of S is denoted by $Hist_S$.
4. The set of infinite paths (or *plays*) of S is denoted by $Plays_S$.
5. The set of all the paths of S is denoted by $Path_S$. $Path_S = Hist_S \cup Plays_S$.
6. Given a path $\pi = (s_i)_{0 \leq i < n}$ and an integer $j < n$, the j^{th} prefix of π (denoted by $\pi_{\leq j}$) is the finite path $(s_i)_{0 \leq i < j+1}$.
7. If a path π is a history (a finite path), the last state of π is denoted by $last(\pi)$. $last(\pi) = s_{|\pi|}$.

2.2 Multi-Player Concurrent Reachability Games

In this section, we describe the notion of non-deterministic multi-player concurrent reachability games as defined in [6, 7]. We then describe the concept of strategies, strategy profiles and the solution concept of *k-resilient* Nash equilibrium in these games.

Definition 2.2.1. A non-deterministic multi-player concurrent reachability game G is defined as a 7-tuple, $G = (States, Edg, Agt, Act, Mov, Tab, \Omega)$ where:

- $(States, Edg)$ is a transition system.

- Agt is a finite set of players (agents).
- Act is a possibly uncountable set of actions.
- $Mov : States \times Agt \to 2^{Act} \setminus \{\emptyset\}$ is a mapping that indicates the actions available to a given player in a given state.
- $Tab : States \times Act^{Agt} \to 2^{Edg} \setminus \{\emptyset\}$ is a mapping that associates a given tuple of actions of players in a given state to the resulting set of transitions. It is required that if $(s', s'') \in Tab(s, (m_A)_{A \in Agt})$, then $s' = s$.
- $\Omega : Agt \to 2^{States}$ is a mapping that assigns to each agent, a set of states, which is the reachability objective of that agent (i.e., the agent wants to reach at least one of these states).

In a game G, from some state s, each player A selects one action m_A from its set $Mov(s, A)$ of allowed actions. The tuple of actions $(m_A)_{A \in Agt}$ thus formed is called a move and is written as m_{Agt}. This results in a set of transitions $Tab(s, (m_A)_{A \in Agt})$ (or simply $Tab(s, m_{Agt})$). One of these transitions is applied (non-deterministically) which gives the next state of the game. In this way, the game continues to form a path π in its underlying transition system.

We associate a payoff vector $v_{Agt} = (v_A)_{A \in Agt}$ to every path π in G. If a path π visits $\Omega(A)$ (i.e, at least one state in π belongs to the set $\Omega(A)$), then we let $v_A(\pi) = 1$, otherwise $v_A(\pi) = 0$ (where $v_A(\pi)$ is the payoff received by player A, when the path π is taken in G).

As in [6, 7], we use the notations $Hist_G$, $Play_G$ and $Path_G$ for the set of *histories*, *plays* and *paths* respectively, in the underlying transition system of G. We also write $Hist_G(s)$, $Play_G(s)$ and $Path_G(s)$ for respective subsets of *histories*, *plays* and *paths* starting in state s.

We now recall the definitions of *strategy* and *strategy profile* from [6, 7].

Definition 2.2.2. In a game G, a strategy for a player $A \in Agt$ is a mapping $\sigma_A : Hist_G \to Act$ such that for every $\pi \in Hist_G$, $\sigma_A(\pi) \in Mov(last(\pi), A)$.

Given a subset of agents (also called a *coalition*) $P \subseteq Agt$, a strategy σ_P for the coalition P is a tuple of strategies, one for each player in P ($\sigma_P = (\sigma_A)_{A \in P}$).

Definition 2.2.3. In a game G, a strategy profile σ_{Agt} is a tuple of strategies, one for each player in Agt (i.e., a strategy profile is a strategy for the coalition Agt). $\sigma_{Agt} = (\sigma_A)_{A \in Agt}$.

With respect to the sets of strategies, following notations are used in this thesis:

- $Strat_G^A$: Set of all strategies for a player $A \in Agt$ in a game G.
- $Strat_G^P$: Set of all strategies for a coalition $P \subseteq Agt$ in a game G.
- $Strat_G^{Agt}$: Set of all strategy profiles in a game G.

Remark 2.2.4. We only consider pure strategies (non-randomized strategies) in this thesis.

In a game G, for a coalition P and a strategy σ_P (for P), a path $\pi = (s_i)_{0 \leq i \leq |\pi|}$ is said to be compatible with the strategy σ_P, if, for every $j \leq |\pi| - 1$, there is a move $m_{Agt} = (m_A)_{A \in Agt}$ such that:

- $m_A \in Mov(s_j, A)$ for every $A \in Agt$.
- $m_A = \sigma_A(\pi_{\leq j})$ for every $A \in P$.
- $(s_j, s_{j+1}) \in Tab(s_j, m_{Agt})$

In a game G, the paths that are compatible with a strategy σ_P (of a coalition P) are also called *outcomes* of the strategy σ_P. We use the notation $Out_G(\sigma_P)$ to denote the set of outcomes of strategy σ_P. The set of finite outcomes of strategy σ_P is denoted by $Out_G^f(\sigma_P)$. The set of infinite outcomes of strategy σ_P is denoted by $Out_G^\infty(\sigma_P)$. $Out_G(\sigma_P) = Out_G^f(\sigma_P) \cup Out_G^\infty(\sigma_P)$. We write $Out_G(s, \sigma_P)$, $Out_G^f(s, \sigma_P)$ and $Out_G^\infty(s, \sigma_P)$ for the respective sets of outcomes, finite outcomes and infinite outcomes of strategy σ_P starting in state s. These notations are as used in [6, 7].

In a game G, given a move m_{Agt} and an action m'_B for some player B, we write $m_{Agt}[B \to m'_B]$ for the move n_{Agt} with $n_A = m_A$ when $A \neq B$ and $n_B = m'_B$.

In a game G, given a move m_{Agt} and an action tuple m'_P for a coalition P ($m'_P = (m'_A)_{A \in P}$), we write $m_{Agt}[P \rightarrow m'_P]$ for the move n_{Agt} with $n_A = m_A$ for every $A \notin P$ and $n_A = m'_A$ for every $A \in P$.

In a game G, given a strategy profile $\alpha_{Agt} \in Strat_G^{Agt}$ and a strategy α'_B for some player B ($\alpha'_B \in Strat_G^B$), we write $\alpha_{Agt}[B \rightarrow \alpha'_B]$ for the strategy profile β_{Agt} with $\beta_A = \alpha_A$ when $A \neq B$ and $\beta_B = \alpha'_B$.

In a game G, given a strategy profile $\alpha_{Agt} \in Strat_G^{Agt}$ and a strategy α'_P for a coalition P ($\alpha'_P \in Strat_G^P$), we write $\alpha_{Agt}[P \rightarrow \alpha'_P]$ for the strategy profile β_{Agt} with $\beta_A = \alpha_A$ for every $A \notin P$ and $\beta_A = \alpha'_A$ for every $A \in P$.

In a game G, let k be a non-negative integer such that $k \leq |Agt|$ (k is a non-negative integer less than or equal to the number of players in G). We use the notation 2_k^{Agt} to denote the set of all subsets of Agt of size at most k.

$$2_k^{Agt} = \{P \mid P \in 2^{Agt} \; and \; |P| \leq k\}$$

We now extend the definition of *pseudo-Nash equilibrium* given in [6, 7] to *k-resilient pseudo-Nash equilibrium*.

Definition 2.2.5. Given a non-deterministic concurrent reachability game G and a state s in G, a *k-resilient* pseudo-Nash equilibrium (for some $k \leq |Agt|$) in G from s is a pair (σ_{Agt}, π) where $\sigma_{Agt} \in Strat_G^{Agt}$ and $\pi \in Out_G(s, \sigma_{Agt})$, such that, for every $P \in 2_k^{Agt}$ and every $\sigma'_P \in Strat_G^P$, it holds:

$$\forall \pi' \in Out_G(s, \sigma_{Agt}[P \rightarrow \sigma'_P]).\ \forall B \in P.\ v_B(\pi') \leq v_B(\pi)$$

Such an outcome π is called a *k-optimal* play for the strategy profile σ_{Agt}.

The payoff vector of a *k-resilient* pseudo-Nash equilibrium (σ_{Agt}, π) is a tuple $(v_A(\pi))_{A \in Agt}$ where $v_A(\pi) = 1$ if π visits $\Omega(A)$ and $v_A(\pi) = 0$ otherwise.

In case of deterministic concurrent reachability games, there is only one outcome of a given strategy profile (π is uniquely determined by σ_{Agt}) and hence *k-resilient* pseudo-Nash equilibria coincide with real *k-resilient* Nash equilibria as defined in

[1, 2]: they are strategy profiles such that, if at most k players deviate from their respective strategies, none of the deviating player can benefit.

In case of non-deterministic concurrent reachability games, a strategy profile σ_{Agt} for a *k-resilient* pseudo-Nash equilibrium may give rise to several outcomes. The choice of playing the *k-optimal* play π is then made cooperatively by all the players. Once a strategy profile is fixed, non-determinism is resolved by all the players by choosing one of the possible outcomes in such a way that if at most k players deviate from their respective strategies or choice of outcome, none of the deviating player can benefit.

Example 2.2.6. Figure 2.1 shows a three-player concurrent reachability game. There are nine states in the game (labelled s_0 to s_8). The three players are A, B and C. The actions allowed to player A in state s_0 are a_1 and a_2. The actions allowed to player B in state s_0 are b_1 and b_2. The actions allowed to player C in state s_0 are c_1 and c_2. Edges in the game are labelled with a move (tuple of actions) of players. When players play a move (select their respective actions), the transition that corresponds to the played move is applied. Edges (transitions) starting in states s_1 to s_8 are self loops and are not labelled with any move. The transitions that are not labelled with any move can be applied for every possible tuple of actions (move) of players. Some states are also labelled with names of players in brackets. If a player's name appears in brackets within a state label, it means that the reachability set of that player contains that state. Observe that $\Omega(A) = \{s_3, s_6, s_7, s_8\}$, $\Omega(B) = \{s_1, s_3, s_4\}$ and $\Omega(C) = \{s_3, s_6, s_7, s_8\}$.

From state s_0, there are five Nash equilibria:

1. The strategy profile with the move (a_1, b_1, c_1) in state s_0 is a *1-resilient* Nash equilibrium (or simply a Nash equilibrium).

2. The strategy profile with the move (a_2, b_1, c_2) in state s_0 is a *1-resilient* Nash equilibrium (or simply a Nash equilibrium).

3. The strategy profile with the move (a_2, b_2, c_1) in state s_0 is a *1-resilient* Nash

equilibrium (or simply a Nash equilibrium).

4. The strategy profile with the move (a_2, b_2, c_2) in state s_0 is a *1-resilient* Nash equilibrium (or simply a Nash equilibrium).

5. The strategy profile with the move (a_1, b_2, c_1) in state s_0 is a *3-resilient* Nash equilibrium.

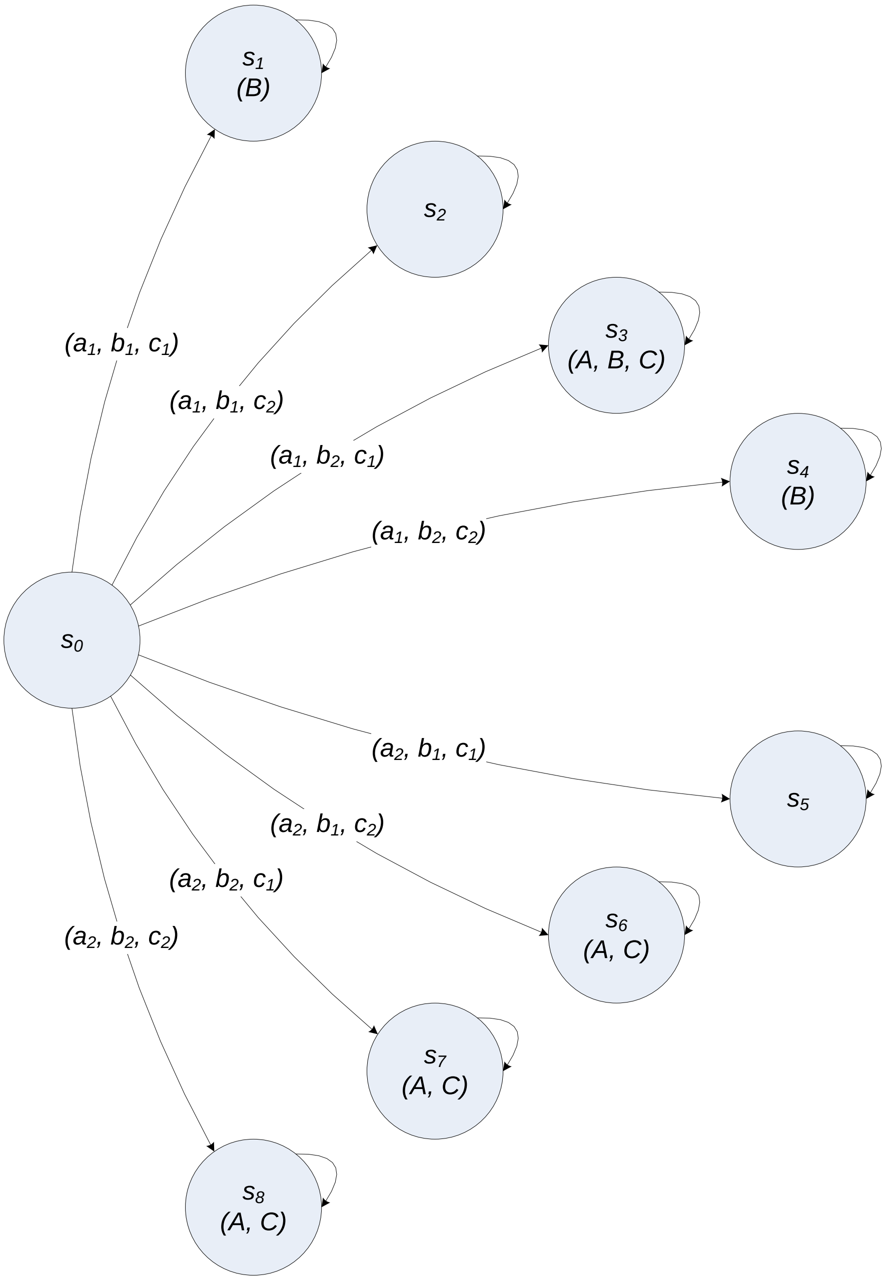

Figure 2.1: A 3-player concurrent reachability game.

Remark 2.2.7. Note that every *k-resilient* Nash equilibrium is also a *j-resilient* Nash equilibrium for every $j \leq k$.

Example 2.2.8. Figure 2.2 shows a two-player concurrent reachability game. There are three states in the game (labelled s_0 to s_2). The two players are A_1 and A_2. The actions allowed to both the players in state s_0 are a and b. Edges are labelled with moves of players. A transition is applied when one of its labelled move is played. Edges (transitions) starting in states s_1 and s_2 are self loops and are not labelled with any move. The transitions that are not labelled with any move can be applied for every possible tuple of actions (move) of players. Some states are also labelled with names of players in brackets. If a player's name appears in brackets within a state label, it means that the reachability set of that player contains that state. Observe that $\Omega(A_1) = \{s_1\}$ and $\Omega(A_2) = \{s_2\}$.

From state s_0, if both the players choose the same action, the game goes to state s_1. If they choose different actions, the game goes to state s_2. Observe that this game does not exhibit any pure strategy Nash equilibrium (There does not exist a *k-resilient* Nash equilibrium for any value of k).

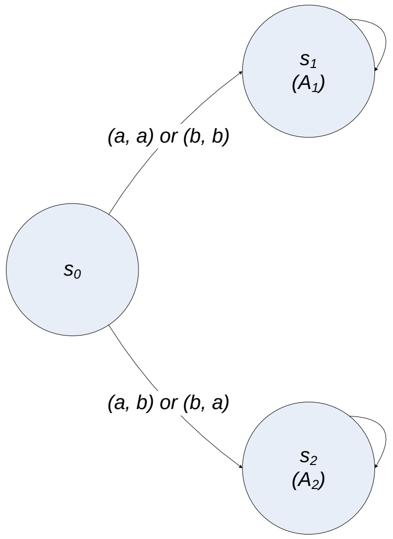

Figure 2.2: A 2-player concurrent reachability game.

Remark 2.2.9. A *k-resilient* Nash equilibrium may not necessarily exist in general.

2.3 Characterizing *K-Resilient* Nash Equilibria

In this section, we define the notion of *k-suspect coalitions* and *k-repellor sets* in non-deterministic multi-player concurrent reachability games. We then prove some properties associated with *k-suspect coalitions* and *k-repellor sets* that characterize *k-resilient* Nash equilibria in these games. These properties are then used to develop the algorithm to check the existence of *k-resilient* Nash equilibrium in finite non-deterministic multi-player concurrent reachability games. The properties that we prove also contain all the necessary information to compute a *k-resilient* Nash equilibrium if it exists.

We now extend the definition of *suspect players* given in [6–9] to *k-suspect coalitions.*

Definition 2.3.1. In a game G, for some $k \leq |Agt|$, the set of *k-suspect coalitions* for an edge $e = (s, s')$, given a move m_{Agt} (denoted as $k - Susp_G(e, m_{Agt})$) is defined as the set:

$$\begin{aligned} k - Susp_G(e, m_{Agt}) = \{P \in 2_k^{Agt} \mid \forall B \in P.\ \exists m'_B \in Mov(s, B) \\ s.t.\ e \in Tab(s, m_{Agt}[P \rightarrow m'_P]) \\ \text{where } m'_P = (m'_B)_{B \in P} \text{ is an action tuple for } P\} \end{aligned}$$

Intuitively, in a game G, a coalition P of size at most k is a *k-suspect coalition* for an edge e, given a move m_{Agt} (i.e., $P \in k - Susp_G(e, m_{Agt})$), whenever players of P can deviate from their corresponding actions in m_{Agt} (while other actions in m_{Agt} are unchanged) and can take edge e (i.e., players of P can deviate from their corresponding actions in m_{Agt} in such a way that edge e is one of the possible transitions).

Remark 2.3.2. For an edge $e = (s, s')$, given a move m_{Agt}, if $e \in Tab(s, m_{Agt})$ then $k - Susp_G(e, m_{Agt}) = 2_k^{Agt}$.

Definition 2.3.3. In a game G, for some $k \leq |Agt|$, the set of *k-suspect coali-*

tions for a finite path $\pi = (s_i)_{i \leq |\pi|}$, given a strategy profile σ_{Agt} (denoted as $k - Susp_G(\pi, \sigma_{Agt})$) is defined as the set:

$$k - Susp_G(\pi, \sigma_{Agt}) = \bigcap_{i < |\pi|} k - Susp_G((s_i, s_{i+1}), (\sigma_A(\pi_{\leq i}))_{A \in Agt})$$

Lemma 2.3.4. *In a game G, for some $k \leq |Agt|$, given $\sigma_{Agt} \in Strat_G^{Agt}$, $\pi \in Hist_G$ and a coalition P of size at most k ($P \in 2_k^{Agt}$), the following three propositions are equivalent:*

(a) $P \in k - Susp_G(\pi, \sigma_{Agt})$

(b) $\exists \sigma'_P \in Strat_G^P.\ \pi \in Out_G^f(\sigma_{Agt}[P \to \sigma'_P])$

(c) $\pi \in Out_G^f((\sigma_A)_{A \in Agt \setminus P})$

Proof. Propositions *(b)* and *(c)* are trivially equivalent. We prove here, the equivalence of propositions *(a)* and *(b)* which proves the lemma.

(a) $\Rightarrow$ *(b)*:

Let $\pi = (s_i)_{i \leq |\pi|}$ and $P \in k - Susp_G(\pi, \sigma_{Agt})$. By definition of $k - Susp_G(\pi, \sigma_{Agt})$, for every $i < |\pi|$, $P \in k - Susp_G((s_i, s_{i+1}), (\sigma_A(\pi_{\leq i}))_{A \in Agt})$. For every $i < |\pi|$, let $(\sigma_A(\pi_{\leq i}))_{A \in Agt}$ denote the move m_{Agt^i} ($m_{Agt^i} = (m_{A^i})_{A \in Agt}$). Now for every $i < |\pi|$, it holds that:

$$\forall B \in P.\ \exists m'_{B^i} \in Mov(s_i, B) \ s.t.\ (s_i, s_{i+1}) \in Tab(s_i, m_{Agt^i}[P \to m'_{P^i}])$$
$$\text{where } m'_{P^i} = (m'_{B^i})_{B \in P}$$

We now define a strategy σ'_P for the coalition P as follows. For every $i < |\pi|$ and for every $B \in P$, let $\sigma'_B(\pi_{\leq i}) = m'_{B^i}$ and let $\sigma'_P = (\sigma'_B)_{B \in P}$. For this σ'_P, we have:

$$\pi \in Out_G^f(\sigma_{Agt}[P \to \sigma'_P])$$

(b) $\Rightarrow$ *(a)*:

Let $\pi = (s_i)_{i \leq |\pi|}$ and let $\sigma'_P = (\sigma'_B)_{B \in P}$. For every $i < |\pi|$, let $m_{Agt^i} = (\sigma_A(\pi_{\leq i}))_{A \in Agt}$ and let $m'_{P^i} = (\sigma'_B(\pi_{\leq i}))_{B \in P}$.

From *(b)*, We have $\pi \in Out^f_G(\sigma_{Agt}[P \to \sigma'_P])$. Thus for every $i < |\pi|$, we have:

$$\begin{aligned} & (s_i, s_{i+1}) \in Tab(s_i, m_{Agt^i}[P \to m'_{P^i}]) \\ \Rightarrow \quad & P \in k - Susp_G((s_i, s_{i+1}), m_{Agt^i}) \\ \Rightarrow \quad & P \in k - Susp_G((s_i, s_{i+1}), (\sigma_A(\pi_{\leq i}))_{A \in Agt}) \end{aligned}$$

As this is true for every $i < |\pi|$, we have:

$$\begin{aligned} & P \in \bigcap_{i<|\pi|} k - Susp_G((s_i, s_{i+1}), (\sigma_A(\pi_{\leq i}))_{A \in Agt}) \\ \Rightarrow \quad & P \in k - Susp_G(\pi, \sigma_{Agt}) \end{aligned}$$

□

Intuitively, Lemma 2.3.4 states that in a game G, a coalition P of size at most k is a *k-suspect coalition* for a path π, given a strategy profile σ_{Agt} (i.e., $P \in k - Susp_G(\pi, \sigma_{Agt})$), whenever the coalition P has a strategy to enforce the path π under the strategies $(\sigma_A)_{A \in Agt \setminus P}$ of the players not in P (i.e., players of P can deviate from their corresponding strategies in σ_{Agt} in such a way that path π is one of the possible outcomes).

Remark 2.3.5. For a finite path π, given a strategy profile σ_{Agt}, if $\pi \in Out^f_G(\sigma_{Agt})$ then $k - Susp_G(\pi, \sigma_{Agt}) = 2^{Agt}_k$.

In a game G, for some $k \leq |Agt|$, consider the set of ordered pairs (P, X) where $P \subseteq Agt$ is a coalition and $X \subseteq 2^{Agt}_k$ (X is a subset of the set of all subsets of Agt of size at most k). Let us define a partial ordering on the set of ordered pairs (P, X) as follows:

$$(P', X') \leq (P, X) \textit{ iff } P' \subseteq P \textit{ and } X' \subseteq X$$

The relation '$\leq$' is trivially reflexive, transitive and anti-symmetric.

We now extend the definition of *repellor sets* given in [6–8] to *k-repellor sets*.

Definition 2.3.6. In a game G, for some $k \leq |Agt|$, given $P \subseteq Agt$ and $X \subseteq 2_k^{Agt}$, the *k-repellor set* of the ordered pair (P, X) (denoted as $k - Rep_G(P, X)$) is defined inductively as follows: as the base case, we let $k - Rep_G(\emptyset, X) = States$ for every $X \subseteq 2_k^{Agt}$. Then assuming that $k - Rep_G(P', X')$ has been defined for every $(P', X') \lneq (P, X)$, we let $k - Rep_G(P, X)$ be the largest set satisfying the following two conditions:

(a) $\forall B \in P.\ k - Rep_G(P, X) \cap \Omega(B) = \emptyset$

(b) $\forall s \in k - Rep_G(P, X).\ \exists m_{Agt} \in Act^{Agt}.\ \forall s' \in States.\ s' \in k - Rep_G(P', X')$ where $X' = k - Susp_G((s, s'), m_{Agt}) \cap X$ and $P' = P \bigcap \left(\bigcup_{W \in X'} W \right)$

Intuitively, in a game G, for some $k \leq |Agt|$, given $P \subseteq Agt$ and $X \subseteq 2_k^{Agt}$, $k - Rep_G(P, X)$ is the set of states from where players can cooperate to stay in $k - Rep_G(P, X)$, thus never satisfying the objectives of players in P, such that if any coalition $Q \in X$ deviates from its strategy and breaks the cooperation, it won't help fulfilling the objectives of players in $P \cap Q$ (i.e, if a coalition $Q \in X$ deviates and breaks the cooperation, the players of Q which are also in P can't be benefited). Intuitively, in $k - Rep_G(P, X)$, P is the set of players whose objectives must not be fulfilled by deviations from coalitions in X.

Lemma 2.3.7. *In a game G, for some $k \leq |Agt|$, given $P, Q \subseteq Agt$ and $X, Y \subseteq 2_k^{Agt}$. If $Q \subseteq P$ and $Y \subseteq X$, then $k - Rep_G(P, X) \subseteq k - Rep_G(Q, Y)$.*

Proof. We prove this lemma by induction on the ordered pair (P, X).

Base case: $k - Rep_G(\emptyset, \emptyset) = States$. Hence, base case is trivial.

Inductive step: For some (P, X), assume that the result holds for every $(P', X') \leq (P, X)$. We now need to prove that the result also holds for (P, X).

Now consider $Q \subseteq P$, and $Y \subseteq X$ and consider the set $k - Rep_G(P, X)$. We prove that $k - Rep_G(P, X) \subseteq k - Rep_G(Q, Y)$ by showing that every state $s \in k - Rep_G(P, X)$ satisfies both the conditions (a) and (b) of Definition 2.3.6 to be in $k - Rep_G(Q, Y)$.

(a) Every $B \in Q$ is also in P. Therefore, we have:

$$\forall s \in k - Rep_G(P, X).\ \forall B \in Q.\ s \notin \Omega(B)$$

(b) For every $s \in k - Rep_G(P, X)$, there exists a move m_{Agt}, such that every state $s' \in States$ satisfies:

$$s' \in k - Rep_G(P', X')$$

where $X' = k - Susp_G((s, s'), m_{Agt}) \cap X$ and $P' = P \bigcap \left(\bigcup_{W \in X'} W \right)$.

Here, $X' \subseteq X$ and $P' \subseteq P$. Therefore we have $(P', X') \leq (P, X)$. So, we can apply induction hypothesis on (P', X').

Let $X'' = k - Susp_G((s, s'), m_{Agt}) \cap Y$ and let $P'' = Q \bigcap \left(\bigcup_{W \in X''} W \right)$.

Clearly $X'' \subseteq X'$ and $P'' \subseteq P'$. Applying induction hypothesis on (P', X'), we get:

$$\begin{aligned} & k - Rep_G(P', X') \subseteq k - Rep_G(P'', X'') \\ \Rightarrow \qquad & s' \in k - Rep_G(P'', X'') \end{aligned}$$

where $X'' = k - Susp_G((s, s'), m_{Agt}) \cap Y$ and $P'' = Q \bigcap \left(\bigcup_{W \in X''} W \right)$.

We have shown that every state $s \in k - Rep_G(P, X)$ satisfies both the conditions (a) and (b) of Definition 2.3.6 to be in $k - Rep_G(Q, Y)$. Therefore $k - Rep_G(P, X) \subseteq k - Rep_G(Q, Y)$ □

We now extend the definition of *secure moves* given in [6, 7] to *k-secure moves.*

Definition 2.3.8. In a game G, for some $k \leq |Agt|$, given $P \subseteq Agt$, $X \subseteq 2_k^{Agt}$ and a state s, the set of *k-secure moves* for the tuple (s, P, X) (denoted as $k - Secure_G(s, P, X)$) is defined as:

$$k - Secure_G(s, P, X) = \left\{ m_{Agt} \in Act^{Agt} \;\middle|\; \forall s' \in States.\ s' \in k - Rep_G(P', X') \right.$$

$$\left. \text{where } X' = k - Susp_G((s, s'), m_{Agt}) \cap X \text{ and } P' = P \bigcap \left(\bigcup_{W \in X'} W \right) \right\}$$

Intuitively, $k - Secure_G(s, P, X)$ is the set of moves available from state $s \in k - Rep_G(P, X)$ for staying in $k - Rep_G(P, X)$ (hence the word *k-secure*).

A special case: In a game G, for some $k \leq |Agt|$, given $P \subseteq Agt$, $X \subseteq 2_k^{Agt}$ and a state s. If $P \bigcap \left(\bigcup_{W \in X} W \right) \neq P$ (i.e., there is at least one player in P which is not a member of any coalition in X), then playing a *k-secure move* from $k - Rep_G(P, X)$ may take us outside $k - Rep_G(P, X)$. This may happen as follows: if a *k-secure move* $m_{Agt} \in k - Secure_G(s, P, X)$ is played and a transition (s, s') is applied as a result, then according to condition (b) of Definition 2.3.6:

$$\begin{array}{ll} & s' \in k - Rep_G(P', X') \\ \text{where} & X' = k - Susp_G((s, s'), m_{Agt}) \cap X \\ \Rightarrow & X' = 2_k^{Agt} \cap X \\ \Rightarrow & X' = X \\ \text{and} & P' = P \bigcap \left(\bigcup_{W \in X'} W \right) \\ \Rightarrow & P' = P \bigcap \left(\bigcup_{W \in X} W \right) \\ \Rightarrow & P' \neq P \text{ (as } P' \subsetneq P) \end{array}$$

Therefore, $s' \in k - Rep_G(P', X)$ but s' is not necessarily in $k - Rep_G(P, X)$. From Lemma 2.3.7, we have $k - Rep_G(P, X) \subseteq k - Rep_G(P', X)$. Therefore, s' may or may not belong to $k - Rep_G(P, X)$. Thus, taking a *k-secure move* from $k - Rep_G(P, X)$ in such cases does not necessarily mean staying in $k - Rep_G(P, X)$. Although, this seems to violate our intuitive meaning of *k-secure moves*, this is not something undesirable. Its just a special case. This happens because there is at least one player $B \in P$ which is not a member of any coalition in X. Therefore if $s \in k - Rep_G(P, X)$, then $s \notin \Omega(B)$. But after a *k-secure move* m_{Agt} is played resulting in the transition (s, s'), B may then be allowed to fulfil its objective (s' may belong to $\Omega(B)$) according to the definition (as B is not a member of any coalition in X).

This special case does not affect our results because, in this thesis, we will only be considering the cases where $P \bigcap \left(\bigcup_{W \in X} W \right) = P$ (i.e., every player in P is a member of at least one coalition in X). In such cases taking a *k-secure move* from $k - Rep_G(P, X)$ results in staying in $k - Rep_G(P, X)$ as expected according to the intuitive meaning of *k-secure moves.*

Definition 2.3.9. In a game G, for some $k \leq |Agt|$, given $P \subseteq Agt$ and $X \subseteq 2_k^{Agt}$, we define the transition system $k - S_G(P, X) = (States, Edg')$ as follows: $(s, s') \in Edg'$ iff there exists some move $m_{Agt} \in k - Secure_G(s, P, X)$ such that $(s, s') \in Tab(s, m_{Agt})$. Note in particular that every $s \in k - Rep_G(P, X)$ has an outgoing transition in $k - S_G(P, X)$.

Lemma 2.3.10. *In a game G, for some $k \leq |Agt|$, given $s \in States$, $P \subseteq Agt$ and $X \subseteq 2_k^{Agt}$ such that $P \bigcap \left(\bigcup_{W \in X} W \right) = P$. Then $s \in k - Rep_G(P, X)$ if and only if there exists an infinite path π in $k - S_G(P, X)$ starting from s.*

Proof. ($\Rightarrow$) We construct $\pi = (s_i)_{i \geq 0}$ inductively starting with $s_0 = s$. Then assuming that $s_i \in k - Rep_G(P, X)$ for some i, we can define s_{i+1} to be such that (s_i, s_{i+1}) corresponds to a *k-secure move* $m_{Agt} \in k - Secure_G(s_i, P, X)$. Then we have:

$$\begin{array}{ll}
 & s_{i+1} \in k - Rep_G(P', X') \\
\text{where} & X' = k - Susp_G((s_i, s_{i+1}), m_{Agt}) \cap X \\
\Rightarrow & X' = 2_k^{Agt} \cap X \\
\Rightarrow & X' = X \\
\text{and} & P' = P \bigcap \left(\bigcup_{W \in X'} W \right) \\
\Rightarrow & P' = P \bigcap \left(\bigcup_{W \in X} W \right) \\
\Rightarrow & P' = P
\end{array}$$

Therefore $s_{i+1} \in k - Rep_G(P, X)$. This holds for every $i \geq 0$. Hence π is an infinite path in $k - S_G(P, X)$.

($\Leftarrow$) Writing $\pi = (s_i)_{i\geq 0}$, by construction of transition system $k - S_G(P, X)$, we have $(s_0, s_1) \in Tab(s_0, m_{Agt})$ for some $m_{Agt} \in k - Secure_G(s_0, P, X)$. Therefore $s_0 \in k - Rep_G(P, X)$ which means $s \in k - Rep_G(P, X)$. □

We now recall the following lemma from [7]. The proof of the following lemma is same as in [7].

Lemma 2.3.11. *In a game G, let $P \subseteq Agt$ be a coalition and let $\sigma_P \in Strat_G^P$ be a strategy for coalition P ($\sigma_P = (\sigma_A)_{A\in P}$). Let $s \in States$ and $\pi \in Out_G^f(s, \sigma_P)$ ending in some state s'. For any history τ starting in s', define $\sigma_A^{-\pi}(\tau) = \sigma_A(\pi \cdot \tau)$. Then*

$$\pi \cdot Out_G(s', (\sigma_A^{-\pi})_{A\in P}) \subseteq Out_G(s, \sigma_P)$$

Proof. Let $\pi' \in \pi \cdot Out_G(s', (\sigma_A^{-\pi})_{A\in P})$. Let us write $\pi = (s_i)_{i\leq|\pi|}$ and $\pi' = (s'_i)_{i\leq|\pi'|}$.

If $i < |\pi|$, then $(s'_i, s'_{i+1}) = (s_i, s_{i+1})$, so that $(s'_i, s'_{i+1}) \in Tab(s'_i, m_{Agt^i})$ for some move $m_{Agt^i} = (m_{A^i})_{A\in Agt}$ such that $m_{A^i} = \sigma_A(\pi_{\leq i})$ for every $A \in P$.

Otherwise, if $i \geq |\pi|$, let ρ be the path such that $\pi \cdot \rho = \pi'_{\leq i}$. Then $(s'_i, s'_{i+1}) \in Tab(s'_i, m'_{Agt^i})$ for some move $m'_{Agt^i} = (m'_{A^i})_{A\in Agt}$ such that $m'_{A^i} = \sigma_A^{-\pi}(\rho)$ for every $A \in P$. Therefore $m'_{A^i} = \sigma_A(\pi \cdot \rho) = \sigma_A(\pi'_{\leq i})$ for every $A \in P$.

Hence $\pi' \in Out_G(s, (\sigma_A)_{A\in P})$. Hence $\pi \cdot Out_G(s', (\sigma_A^{-\pi})_{A\in P}) \subseteq Out_G(s, \sigma_P)$. □

Lemma 2.3.12. *In a game G, for some $k \leq |Agt|$, given $s \in States$, $P \subseteq Agt$ and $X \subseteq 2_k^{Agt}$ such that $P \bigcap \left(\bigcup_{W\in X} W \right) = P$. Let $\pi \in Play_G(s)$ be an infinite play with initial state s. Then π is a path in $k - S_G(P, X)$ if and only if there exists $\sigma_{Agt} \in Strat_G^{Agt}$ such that $\pi \in Out_G(s, \sigma_{Agt})$ and for every $Q \in X$ and every $\sigma'_Q \in Strat_G^Q$, it holds that:*

$\forall \pi' \in Out_G(s, \sigma_{Agt}[Q \to \sigma'_Q])$. π' does not visit $\Omega(B)$ for every $B \in P \cap Q$.

Proof. ($\Rightarrow$) Assume that $\pi = (s_i)_{i\geq 0}$ is an infinite play in $k - S_G(P, X)$. Let us define the following partial functions:

(i) $c : States \times 2^{Agt} \times 2^{2_k^{Agt}} \to Act^{Agt}$: for every tuple (s, R, Y) such that $s \in k - Rep_G(R, Y)$, we let $c(s, R, Y) = m_{Agt}$ for some move $m_{Agt} \in k - Secure_G(s, R, Y)$.

(ii) $d : \mathbb{N} \rightarrow Act^{Agt}$: for every $i \in \mathbb{N}$, we let $d(i) = m_{Agt^i}$ for some move $m_{Agt^i} \in k - Secure_G(s_i, P, X)$ such that $(s_i, s_{i+1}) \in Tab(s_i, m_{Agt^i})$. This is well defined because π is a play in $k - S_G(P, X)$.

The strategy profile σ_{Agt} is defined as follows:

On prefixes of π, we let $\sigma_A(\pi_{\leq i}) = (d(i))(A)$ for every $i \in \mathbb{N}$, where $(d(i))(A)$ represents the action corresponding to the player A in the move $d(i)$.

For any $\pi' = (s'_i)_{i \leq |\pi'|}$ that is not a prefix of π, we let $\sigma_A(\pi') = (c(s'_{|\pi'|}, P', X'))(A)$ where $X' = k - Susp_G(\pi', \sigma_{Agt}) \cap X$, $P' = P \bigcap \left(\bigcup_{W \in X'} W \right)$ and $(c(s'_{|\pi'|}, P', X'))(A)$ represents the action corresponding to the player A in the move $c(s'_{|\pi'|}, P', X')$.

By construction, we have $\pi \in Out_G(s, \sigma_{Agt})$. Let Q be a coalition in X $(Q \in X)$ and let σ'_Q be a strategy for the coalition Q $(\sigma'_Q \in Strat^Q_G)$ such that $\sigma'_Q = (\sigma'_B)_{B \in Q}$. Let $\pi' \in Out_G(s, \sigma_{Agt}[Q \rightarrow \sigma'_Q])$. Assuming $\pi' = (s'_i)_{i \geq 0}$, we show by induction on i that:

$$s'_i \in k - Rep_G(P \cap Q, \{Q\}) \text{ for every } i \in \mathbb{N}$$

Base case: When $i = 0$, $s'_0 = s$. We know that π is a play in $k - S_G(P, X)$ starting from s. Therefore, from Lemma 2.3.10, we get that $s \in k - Rep_G(P, X)$ which implies $s'_0 \in k - Rep_G(P, X)$. As $P \cap Q \subseteq P$ and $\{Q\} \subseteq X$, from Lemma 2.3.7 we get $s'_0 \in k - Rep_G(P \cap Q, \{Q\})$. Hence base case is true.

Inductive step: Assume that for some $i \geq 0$, $s'_i \in k - Rep_G(P \cap Q, \{Q\})$. Let $(\sigma_A(\pi'_{\leq i}))_{A \in Agt} = m_{Agt^i}$. Let $(\sigma'_B(\pi'_{\leq i}))_{B \in Q} = m'_{Q^i}$. By construction of π', we have $(s'_i, s'_{i+1}) \in Tab(s'_i, m_{Agt^i}[Q \rightarrow m'_{Q^i}])$ which implies:

$$Q \in k - Susp_G((s'_i, s'_{i+1}), m_{Agt^i})$$

If $\pi'_{\leq i}$ is not a prefix of π, then the strategy profile σ_{Agt} is defined using c. Therefore, $(\sigma_A(\pi'_{\leq i}))_{A \in Agt} = m_{Agt^i} = c(s'_i, P', X')$ where $X' = k - Susp_G(\pi', \sigma_{Agt}) \cap X$ and $P' = P \bigcap \left(\bigcup_{W \in X'} W \right)$. By definition, $s'_i \in k - Rep_G(P', X')$ and $m_{Agt^i} \in$

$k - Secure_G(s'_i, P', X')$. From condition (b) of Definition 2.3.6, we get that:

$$s'_{i+1} \in k - Rep_G(P'', X'')$$

where $X'' = k - Susp_G((s'_i, s'_{i+1}), m_{Agt^i}) \cap X'$ and $P'' = P' \bigcap \left(\bigcup_{W \in X''} W \right)$

By construction of π', we know that $Q \in k - Susp_G(\pi', \sigma_{Agt})$. And by induction hypothesis, $s'_i \in k - Rep_G(P \cap Q, \{Q\})$. Therefore $\{Q\} \subseteq X'$ and $P \cap Q \subseteq P'$. Now as $Q \in k - Susp_G((s'_i, s'_{i+1}), m_{Agt^i})$, we get that $\{Q\} \subseteq X''$ and $P \cap Q \subseteq P''$. Therefore from Lemma 2.3.7, we conclude that $s'_{i+1} \in k - Rep_G(P \cap Q, \{Q\})$.

Now if $\pi'_{\leq i}$ is a prefix of π, then the strategy profile σ_{Agt} is defined using d. Therefore, $(\sigma_A(\pi'_{\leq i}))_{A \in Agt} = m_{Agt^i} = d(s'_i, P, X)$. By definition, $s'_i \in k - Rep_G(P, X)$ and $m_{Agt^i} \in k - Secure_G(s'_i, P, X)$. From condition (b) of Definition 2.3.6, we get that:

$$s'_{i+1} \in k - Rep_G(P'', X'')$$

where $X'' = k - Susp_G((s'_i, s'_{i+1}), m_{Agt^i}) \cap X$ and $P'' = P \bigcap \left(\bigcup_{W \in X''} W \right)$.

Now as $Q \in k - Susp_G((s'_i, s'_{i+1}), m_{Agt^i})$, we get that $\{Q\} \subseteq X''$ and $P \cap Q \subseteq P''$. Therefore from Lemma 2.3.7, we conclude that $s'_{i+1} \in k - Rep_G(P \cap Q, \{Q\})$.

This proves the inductive step. Hence π' does not visit $\Omega(B)$ for every $B \in P \cap Q$.

($\Leftarrow$) Let σ_{Agt} be a strategy profile such that $\pi \in Out_G(s, \sigma_{Agt})$ and for every $Q \in X$ and every $\sigma'_Q \in Strat^Q_G$, it holds that:

$\forall \pi' \in Out_G(s, \sigma_{Agt}[Q \to \sigma'_Q])$. π' does not visit $\Omega(B)$ for every $B \in P \cap Q$.

The proof for the implication is by induction on the ordered pair (P, X).

Base case: If $P = \emptyset$, then by Definition 2.3.6, $k - Rep_G(\emptyset, X) = States$ for every $X \subseteq 2^{Agt}_k$. Hence the result holds for the base case.

Inductive step: Let us assume that the implication is true for every $(P', X') \lneq (P, X)$ which satisfy the condition $P' \bigcap \left(\bigcup_{W \in X'} W \right) = P'$.

We define the set:

$S = \{s' \in States \mid \exists \rho \in Hist_G(s)$ with $last(\rho) = s'$ and $X \subseteq k - Susp_G(\rho, \sigma_{Agt})\}$

Let $s' \in S$ and $\rho \in Hist_G(s)$ be a history starting in s such that $last(\rho) = s'$

and $X \subseteq k - Susp_G(\rho, \sigma_{Agt})$. According to Lemma 2.3.4, any coalition $Q \in X$ has a strategy $\sigma'_Q \in Strat_G^Q$ such that $\rho \in Out_G^f(s, \sigma_{Agt}[Q \to \sigma'_Q])$. As a consequence ρ does not visit $\Omega(B)$ for every $B \in P \cap Q$, which means $s \notin \Omega(B)$ and $s' \notin \Omega(B)$ for every $B \in P \cap Q$. This shows that for every $Q \in X$, it holds: $S \cap \Omega(B) = \emptyset$ for every $B \in P \cap Q$.

Let $m_{Agt} = (\sigma_A(\rho))_{A \in Agt}$ and $s'' \in States$. Then:

If $X \subseteq k - Susp_G((s', s''), m_{Agt})$, then $s'' \in S$.

Otherwise $X \cap k - Susp_G((s', s''), m_{Agt}) \subsetneq X$. We write X' for this set and we let $P' = P \bigcap \left(\bigcup_{W \in X'} W \right)$. So $(P', X') \lneq (P, X)$ and we can apply induction hypothesis on (P', X'). Pick a coalition $Q \in X'$ and let $\rho' = \rho \cdot s''$. According to Lemma 2.3.11:

$$\rho' \cdot Out_G(s'', (\sigma_A^{-\rho'})_{A \in Agt \setminus Q}) \subseteq Out_G(s, \sigma_{Agt \setminus Q})$$

But, according to the given condition, any path in $Out_G(s, \sigma_{Agt \setminus Q})$ does not visit $\Omega(B)$ for every $B \in P \cap Q$. As $P' \subseteq P$, $\rho' \cdot Out_G(s'', (\sigma_A^{-\rho'})_{A \in Agt \setminus Q})$ does not visit $\Omega(B)$ for every $B \in P' \cap Q$. Applying induction hypothesis on (P', X') and using Lemma 2.3.10, we get $s'' \in k - Rep_G(P', X')$.

This proves that $S \subseteq k - Rep_G(P, X)$ (as any state $s' \in S$ satisfies both the conditions of Definition 2.3.6 to stay in $k - Rep_G(P, X)$) and the move $(\sigma_A(\rho))_{A \in Agt}$ belongs to $k - Secure_G(last(\rho), P, X)$. Since π is an outcome of σ_{Agt}, π involves playing *k-secure moves* to stay in $k - Rep_G(P, X)$. Hence π is a play in $k - S_G(P, X)$. □

Theorem 2.3.13. *Let G be a non-deterministic multi-player concurrent reachability game and let $s \in States$. There is a k-resilient pseudo-Nash equilibrium in G from s with a payoff vector $v_{Agt} = (v_A)_{A \in Agt}$ if and only if, letting $P = \{A \in Agt \mid v_A = 0\}$, there is an infinite path π in $k - S_G(P, 2_k^{Agt})$ which starts in s and visits $\Omega(A)$ for every $A \notin P$. Furthermore, π is the k-optimal play for this k-resilient pseudo-Nash equilibrium.*

Proof. ($\Rightarrow$) Let (σ_{Agt}, π) be a *k-resilient* pseudo-Nash equilibrium from s with payoff vector $v_{Agt} = (v_A)_{A \in Agt}$. This means σ_{Agt} is a strategy profile and $\pi \in Out_G(s, \sigma_{Agt})$.

Let $P = \{A \in Agt \mid v_A = 0\}$. Therefore π visits $\Omega(A)$ for every $A \notin P$.

Also, as (σ_{Agt}, π) is a *k-resilient* pseudo-Nash equilibrium, for every $Q \in 2_k^{Agt}$ and every $\sigma'_Q \in Strat_G^Q$, it holds that:

$$\forall \pi' \in Out_G(s, \sigma_{Agt}[Q \to \sigma'_Q]).\ \forall B \in Q.\ v_B(\pi') \leq v_B(\pi)$$

But $v_B(\pi) = 0$ for every $B \in P$ which means $v_B(\pi) = 0$ for every $B \in P \cap Q$. This implies $v_B(\pi') = 0$ for every $B \in P \cap Q$ or the path π' does not visit $\Omega(B)$ for every $B \in P \cap Q$.

Therefore, we have that, for every $Q \in 2_k^{Agt}$ and every $\sigma'_Q \in Start_G^Q$, it holds:

$\forall \pi' \in Out_G(s, \sigma_{Agt}[Q \to \sigma'_Q])$. π' does not visit $\Omega(B)$ for every $B \in P \cap Q$.

By Lemma 2.3.12, we get that π must be a path in $k - S_G(P, 2_k^{Agt})$.

($\Leftarrow$) Let π be an infinite path in $k - S_G(P, 2_k^{Agt})$ such that π visits $\Omega(A)$ for every $A \notin P$.

As π is a play in $k - S_G(P, 2_k^{Agt})$, according to Lemma 2.3.12 there is a strategy profile $\sigma_{Agt} \in Strat_G^{Agt}$ such that, $\pi \in Out_G(s, \sigma_{Agt})$ and for every $Q \in 2_k^{Agt}$ and every $\sigma'_Q \in Strat_G^Q$, it holds:

$\forall \pi' \in Out_G(s, \sigma_{Agt}[Q \to \sigma'_Q])$. π' does not visit $\Omega(B)$ for every $B \in P \cap Q$.

Therefore, we have that, for every $Q \in 2_k^{Agt}$ and every $\sigma'_Q \in Strat_G^Q$, it holds:

$$\forall \pi' \in Out_G(s, \sigma_{Agt}[Q \to \sigma'_Q]).\ \forall B \in Q.\ v_B(\pi') \leq v_B(\pi)$$

Hence (σ_{Agt}, π) is a *k-resilient* pseudo-Nash equilibrium. □

Theorem 2.3.13 gives a necessary and sufficient condition for existence of a *k-resilient* pseudo-Nash equilibrium in a game G. In case a *k-resilient* pseudo-Nash equilibrium exists, the *k-repellor* sets and the corresponding transition systems contain all the necessary information to compute the equilibrium. The following theorem gives a generic method to compute a *k-resilient* pseudo-Nash equilibrium if it exists.

Theorem 2.3.14. *In a game G, if π is an infinite path in $k-S_G(P, 2_k^{Agt})$ from a state s visiting $\Omega(A)$ for every $A \notin P$, then there is a k-resilient pseudo-Nash equilibrium (σ_{Agt}, π) where the strategy profile σ_{Agt} consists in playing k-secure moves in the transition system $k - S_G(P \cap P', 2_k^{Agt} \cap X')$ for some $P' \subseteq Agt$ and $X' \subseteq 2_k^{Agt}$ satisfying the condition $(P \cap P') \bigcap \left(\bigcup_{W \in 2_k^{Agt} \cap X'} W \right) = (P \cap P')$.*

Proof. For every $P \subseteq Agt$ and every $X \subseteq 2_k^{Agt}$ such that $P \bigcap \left(\bigcup_{W \in X} W \right) = P$ and for every $s \in k - Rep_G(P, X)$, there is a *k-secure move* $m_{Agt} \in k - Secure_G(s, P, X)$ such that every $(s, s') \in Tab(s, m_{Agt})$ is an edge of $k - S_G(P, X)$ (Lemma 2.3.10). For every $P \subseteq Agt$ and every $X \subseteq 2_k^{Agt}$ such that $P \bigcap \left(\bigcup_{W \in X} W \right) = P$, let us define memoryless strategies $\sigma_{Agt}^{P,X}$ which select such a *k-secure move* to stay in $k - Rep_G(P, X)$. For every $s \in k - Rep_G(P, X)$, every $\pi \in Out_G(s, \sigma_{Agt}^{P,X})$ is a path in $k - S_G(P, X)$. The idea is that as soon as any coalition $Q \in X$ deviates, we compute the sets $X' = X \cap k - Susp_G((s, s'), \sigma_{Agt}^{P,X})$ and $P' = P \bigcap \left(\bigcup_{W \in X'} W \right)$ and then we play strategies $\sigma_{Agt}^{P',X'}$ so that the players of the deviating coalition which are also in P can't benefit.

Let $\pi = (s_i)_{i \geq 0}$ be an infinite path in $k - S_G(P, 2_k^{Agt})$ that starts in state s and visits $\Omega(A)$ for every $A \notin P$. The strategy profile σ_{Agt} aims at playing π, and if any coalition $Q \in 2_k^{Agt}$ deviates, the strategies will be computed as mentioned earlier. Therefore, we define the strategy profile σ_{Agt} as follows:

For every history $\pi' = (s'_i)_{i \leq m}$ in the game, we distinguish between several cases:

1. If π' is a prefix of π ($s'_i = s_i$ if $i \leq m$), then we define $(\sigma_A(\pi'))_{A \in Agt}$ as a *k-secure move* with respect to $(P, 2_k^{Agt})$ such that $(s'_m, s_{m+1}) \in Tab(s'_m, (\sigma_A(\pi'))_{A \in Agt})$, so that the play continues in π.

2. If π' is a path in $k - S_G(P, 2_k^{Agt})$ but not a prefix of π, then we define $(\sigma_A(\pi'))_{A \in Agt}$ as a *k-secure move* with respect to $(P, 2_k^{Agt})$ from $last(\pi')$. This choice can be made memoryless and depends only on $last(\pi')$.

3. Otherwise decompose π' as $\pi_1 \cdot \pi_2 \cdot \pi_3 \cdots \pi_l$ such that there exist $P_l \subseteq P_{l-1} \cdots \subseteq$

$P_2 \subseteq P$ and $X_l \subseteq X_{l-1} \cdots \subseteq X_2 \subseteq 2_k^{Agt}$ with:

(a) π_1 is a path in $k - S_G(P, 2_k^{Agt})$.

(b) If $1 < j \leq l$, then π_j is a path in $k - S_G(P \cap P_j, 2_k^{Agt} \cap X_j)$.

(c) $X_2 = k - Susp_G((last(\pi_1), first(\pi_2)), (\sigma_A(\pi_1))_{A \in Agt})$

and

$P_2 = P \bigcap \left(\bigcup_{W \in X_2} W \right)$.

(d) If $1 < j < l$, then

$X_{j+1} = X_j \cap k - Susp_G((last(\pi_j), first(\pi_{j+1})), (\sigma_A^{P_j, X_j}(last(\pi_j)))_{A \in Agt})$

and

$P_{j+1} = P_j \bigcap \left(\bigcup_{W \in X_{j+1}} W \right)$.

And we define $\sigma_A(\pi')$ as $\sigma_A^{P_l, X_l}(last(\pi'))$.

We claim that (σ_{Agt}, π) is a *k-resilient* pseudo-Nash equilibrium. It is given that π visits $\Omega(A)$ for every $A \notin P$. If $v_{Agt} = (v_A)_{A \in Agt}$ is the payoff vector associated with π, then $v_A(\pi) = 1$ for every $A \notin P$ and $v_A(\pi) = 0$ for every $A \in P$. Now suppose a coalition $Q \in 2_k^{Agt}$ deviates and follows strategy σ'_Q. Then for every $\pi' \in Out_G(s, \sigma_{Agt}[Q \to \sigma'_Q])$, we can decompose π' as $\pi_1 \cdot \pi_2 \cdot \pi_3 \cdots \pi_l$ as described above. Also, as the coalition Q deviated from its strategy resulting in the path π', $Q \in k - Susp_G(\pi', \sigma_{Agt})$. Hence $Q \in X_j$ for every $1 \leq j \leq l$. This implies that for every $1 \leq j \leq l$, π_j does not visit $\Omega(B)$ for every $B \in P_j \cap Q$. Therefore the path π' does not visit $\Omega(B)$ for every $B \in P \cap Q$.

Therefore, we have that, for every $Q \in 2_k^{Agt}$ and every $\sigma'_Q \in Strat_G^Q$, it holds:

$$\forall \pi' \in Out_G(s, \sigma_{Agt}[Q \to \sigma'_Q]).\ \forall B \in Q.\ v_B(\pi') \leq v_B(\pi)$$

Hence (σ_{Agt}, π) is a *k-resilient* pseudo-Nash equilibrium. □

2.4 Application to Finite Games

In this section, we provide the algorithm to check the existence of *k-resilient* Nash equilibrium in finite non-deterministic multi-player concurrent reachability games when only pure strategies are allowed.

A finite non-deterministic multi-player concurrent reachability game is a game in which the underlying transition system $((States, Edg))$ is finite and the set of actions (Act) is also finite.

Given a finite game G, a state s in G and a positive integer $k \leq |Agt|$, we can apply Theorem 2.3.13 to develop an algorithm for checking existence of a *k-resilient* pseudo-Nash equilibrium in the game G from state s.

Algorithm 2.4.1 Existence of a *k-resilient* pseudo-Nash equilibrium in a game.

Input: A finite non-deterministic multi-player concurrent reachability game G, a state s in G and a positive integer $k \leq |Agt|$.

Output: A boolean value: **true** if there exists a *k-resilient* pseudo-Nash equilibrium in G from s; **false** otherwise.

```
 1: for every possible payoff vector $v_{Agt} = (v_A)_{A \in Agt}$ do
 2:    $P \leftarrow \{A \in Agt \mid v_A = 0\}$
 3:    Compute $k - Rep_G(P, 2^{Agt}_k)$
 4:    Construct $k - S_G(P, 2^{Agt}_k)$
 5:    for every play $\pi$ in $k - S_G(P, 2^{Agt}_k)$ starting from $s$ do
 6:       if $\pi$ visits $\Omega(A)$ for every $A \notin P$ then
 7:          return true
 8:       end if
 9:    end for
10: end for
11: return false
```

The correctness of Algorithm 2.4.1 follows from Theorem 2.3.13.

Time complexity analysis: In Algorithm 2.4.1, the loop of step 1 runs $2^{|Agt|}$ times (as the number of possible payoff vectors is $2^{|Agt|}$). In step 2, construction of the set P involves checking v_A for every $A \in Agt$. Hence step 2 takes $\mathrm{O}(|Agt|)$ time. Step 3 and step 4 involve computation of $k - Rep_G(P, 2^{Agt}_k)$ and construction of $k - S_G(P, 2^{Agt}_k)$ respectively. This can be achieved by filling a table $R(s', Q, Y)$ for every $s' \in States$, every $Q \subseteq Agt$ and every $Y \subseteq 2^{Agt}_k$ such that $R(s', Q, Y) = 1$ if $s' \in k - Rep_G(Q, Y)$ and $R(s', Q, Y) = 0$ otherwise. This table can be filled inductively. As the base step, we let $R(s', \emptyset, Y) = 1$ for every $s' \in States$ and every $Y \subseteq 2_k^{Agt}$ (by Definition 2.3.6). Then assuming that $R(s', Q', Y')$ has been computed for every $s' \in States$ and every $(Q', Y') \lneq (Q, Y)$, the entries $R(s', Q, Y)$ for every $s' \in States$ are filled as follows: we start with letting $R(s', Q, Y) = 1$ for every $s' \notin \bigcup\limits_{A \in Q} \Omega(A)$ and then iteratively set $R(s', Q, Y)$ to 0 for those states that do not satisfy condition (b) of Definition 2.3.6 to stay in $k - Rep_G(Q, Y)$. Checking condition (b) of Definition 2.3.6 is achieved by enumerating the set of moves m_{Agt}, computing the set of *k-suspect coalitions* for each state and checking whether the state belongs to the required *k-repellor set* (by computing corresponding X' and P' as in condition (b) of Definition 2.3.6). This also gives us the set of *k-secure moves* thus building $k - S_G(Q, Y)$. This process takes time, polynomial with respect to the number of states and the size of transition table. But it requires $k - Rep_G(Q', Y')$ to be computed for every $(Q', Y') \lneq (Q, Y)$. Therefore, in order to compute $k - Rep_G(P, 2_k^{Agt})$ and construct $k - S_G(P, 2_k^{Agt})$, we need to do this procedure for every $(P', X') \leq (P, 2_k^{Agt})$. In case $k << |Agt|$, $|2_k^{Agt}|$ can be approximated as $O(|Agt|^{k+1})$ and the number of subsets of 2_k^{Agt} is of the order of $2^{|Agt|^{k+1}}$. Therefore if $k << |Agt|$, then step 3 and step 4 together take time polynomial with respect to the number of states and size of transition table but $O(2^{|Agt|^{k+1}})$ time with respect to the number of agents and the input k. In the worst case scenario, k is of the order of $|Agt|$ and the number of subsets of 2_k^{Agt} is of the order of $2^{2^{|Agt|}}$. Therefore, in the worst case scenario, step 3 and step 4 together take time polynomial with respect to the number of states and size of transition table but doubly exponential with respect to the number of agents. The loop of step 5 runs exponential times with respect to the number of states and size of transition table in the worst case (as the number plays in $k - S_G(P, 2_k^{Agt})$ is exponential with respect to the number of states and size of transition table in the worst case). In step 6, checking that a path π visits $\Omega(A)$ for every $A \notin P$ can be achieved in polynomial time with respect to the number of states and number of agents. Overall, the algorithm takes exponential time with respect to the number of states and size of transition table but doubly exponential time with respect to the number of agents in the worst case (note that if $k << |Agt|$, then the algorithm takes $O(2^{|Agt|^{k+1}})$ time with respect to the number of agents and the input k which is better than $O(2^{2^{|Agt|}})$ if $k << |Agt|$). Therefore, the algorithm runs in time doubly exponential with respect to the size of input in the worst case. Hence the algorithm is in 2-EXPTIME.

In case a *k-resilient* pseudo-Nash equilibrium exists in a game, it can be computed using the generic method given in Theorem 2.3.14 and its proof.

Chapter 3

Timed Concurrent Reachability Games

In this chapter, we describe the notion of timed games and their importance in formal specification and verification of real time systems. We then describe the semantics of timed games with reachability objectives in terms of an infinite non-deterministic multi-player concurrent reachability game. We then define the notion of region games which are finite games derived from the classical concept of region based abstraction of timed games. We prove that the translation of a timed concurrent reachability game to its associated region game preserves a *k-resilient* Nash equilibrium (if one exists). Thus, existence of *k-resilient* Nash equilibrium in a timed concurrent reachability game can be checked by translating the timed game to its associated region game and then checking existence of *k-resilient* Nash equilibrium in the region game (using methods described in Chapter 2). For proper understanding of timed games and their associated region games, the reader should be familiar with the basic concepts of timed automata and their associated region automata. We describe the basic theory of timed automata (required for proper understanding of this chapter) in Appendix A.

3.1 Timed Games

For formal specification and verification of real time systems that involve timing constraints (such as flight control systems installed in an aeroplane), we need to include the timing constraints while modelling the system specification. If a system desires some properties that include timing constraints, we should be able to include the concept of ‘time’ in the system specification. In order to verify that the system has such desired properties (that include timing constraints), the system model must be able to capture such properties completely in its specification. For this purpose, Alur and Dill [3] define a concept of timed automata. Timed automata are a natural extension of finite automata in the sense that transitions in timed automata not just read a symbol (over an alphabet) but also take into account the ‘time’ at which a symbol is read. If a symbol represents an event, then transitions in timed automata depend not only on the occurrence of an event, but also on the time of occurrence of an event. Thus, timed automata are able to capture properties that involve timing constraints and hence are useful in formal specification and verification of real time systems. For more details on timed automata, we refer the reader to Appendix A.

Timed games are defined in a way similar to timed automata, but include necessary game theoretic concepts. Real time systems that involve multiple agents interacting with each other can be modelled as multi-player timed games. Game theoretic solution concepts can then be used to study the properties of such systems. In this thesis, we are interested in multi-player timed concurrent reachability games as defined in [6, 7].

We begin with defining the concept of clocks and clock valuations as defined in [3, 6, 7].

Definition 3.1.1. Consider a finite set of clocks χ. A valuation v over the finite set of clocks χ is an application $v : \chi \to \mathbb{R}_+$ that assigns to each clock $x \in \chi$, a positive real number that signifies the units of time since the clock x was last reset.

If v is a valuation over the set of clocks χ and $t \in \mathbb{R}_+$, then $v + t$ is the valuation that assigns to each $x \in \chi$, the value $v(x) + t$.

If v is a valuation over the set of clocks χ and $\varphi \subseteq \chi$, then $[\varphi \leftarrow 0]v$ is the valuation that assigns the value 0 to each $y \in \varphi$ and the value $v(x)$ to each $x \in \chi \backslash \varphi$.

We now recall the concept of clock constraints from [3, 6, 7].

Definition 3.1.2. A clock constraint over the set of clocks χ is a formula built on the grammar $\zeta(\chi) \ni g ::= x \backsim c \mid g \wedge g$, where x ranges over χ, $\backsim \in \{<, \leq, =, \geq, >\}$ and c is an integer.

A valuation v over a set of clocks χ satisfies a clock constraint g over χ if on assigning the values $v(x)$ to each $x \in \chi$, the clock constraint g evaluates to true. If a valuation v satisfies a clock constraint g, we write it as $v \models g$.

We now recall the definition of multi-player timed concurrent reachability games from [6, 7].

Definition 3.1.3. A multi-player timed concurrent reachability game G is defined as a 7-tuple, $G = (Loc, \chi, Inv, Trans, Agt, Owner, \Omega)$ where:

- Loc is a finite set of locations.

- χ is a finite set of clocks.

- $Inv : Loc \rightarrow \zeta(\chi)$ assigns an invariant to each location. If l is a location and $Inv(l) = g$, it means that while the game is at location l, the valuation of clocks in χ must satisfy the clock constraint g.

- $Trans \subseteq Loc \times \zeta(\chi) \times 2^{\chi} \times Loc$ is the set of transitions. If $\delta = (l, g, z, l')$ is a transition, it means that the transition δ is firable only if the clock constraint g evaluates to true. Further, on firing the transition δ, the game goes from location l to location l' and the clocks in the set z are reset to 0.

- Agt is a finite set of agents (or players).

- $Owner : Trans \rightarrow Agt$ assigns an agent to each transition. If $Owner(\delta) = A$, it means that only player A can fire the transition δ.

- $\Omega : Agt \to 2^{Loc}$ assigns to each agent, a set of locations which is the reachability objective of that agent (i.e., the agent wants to reach at least one of these locations).

A multi-player timed concurrent reachability game G is played as follows: a state of the game is an ordered pair (l, v) where $l \in Loc$ and v is a valuation over the set χ of clocks, provided that $v \models Inv(l)$. From each state (l, v) (starting from an initial state $s_0 = (l, \mathbf{0})$, where $\mathbf{0}$ is a valuation that assigns the value 0 to every clock $x \in \chi$ and is assumed to satisfy $Inv(l)$), every player A chooses a non-negative real number d and a transition $\delta = (l, g, z, l')$ with the intended meaning that the player A wants to delay for d time units and then fire the transition $\delta = (l, g, z, l')$. There are various natural restrictions on these choices:

- Spending d time units in l must be allowed, i.e., $v + d' \models Inv(l)$ for every $0 \leq d' \leq d$. As the invariants are convex, this is equivalent to having only $v + d \models Inv(l)$.

- Player A must be the owner of the transition $\delta = (l, g, z, l')$, i.e., $Owner(\delta) = A$.

- The transition $\delta = (l, g, z, l')$ is firable after d time units, i.e., $v + d \models g$.

- The invariant of l' must be satisfied when entering l', i.e., $[z \leftarrow 0](v + d) \models Inv(l')$.

If (and only if) there is no such possible choice for some player A, then A chooses a null action (denoted by $\perp$).

Given a tuple of choices m_{Agt} of all the players, with $m_A \in (\mathbb{R}_+ \times Trans) \cup \{\perp\}$, a player B such that $d_B = min\{d_A \mid A \in Agt \text{ and } m_A = (d_A, \delta_A)\}$ is selected non-deterministically, and the corresponding transition $\delta_B = (l, g_B, z_B, l')$ is applied leading to a new state $(l', [z_B \leftarrow 0](v + d_B))$.

3.2 Semantics of Timed Games

The semantics of a multi-player timed concurrent reachability game as described in the previous section can be expressed in terms of an infinite-state non-deterministic multi-player concurrent reachability game. We recall the semantics of multi-player timed concurrent reachability games from [6, 7]. With a multi-player timed concurrent reachability game $G = (Loc, \chi, Inv, Trans, Agt, Owner, \Omega)$, we can associate the infinite non-deterministic multi-player concurrent reachability game $G' = (States, Edg, Agt, Act, Mov, Tab, \Omega')$ such that:

- $States = \{(l, v) \mid l \in Loc,\ v : \chi \to \mathbb{R}_+ \text{ such that } v \models Inv(l)\}$.

- $s_0 = (l_0, \mathbf{0})$ is the initial state.

- The set of transitions $Trans$ in G give rise to the set of edges Edg in G' as follows: for every $d \in \mathbb{R}_+$, every $\delta = (l, g, z, l')$ in $Trans$ and every $(l, v) \in States$ such that $v + d \models Inv(l) \wedge g$ and $[z \leftarrow 0](v + d) \models Inv(l')$, there is an edge $((l, v), (l', [z \leftarrow 0](v + d)))$ in Edg.

- The set of actions is $Act = \{(d, \delta) \mid d \in \mathbb{R}_+,\ \delta \in Trans\} \cup \{\perp\}$.

- An action (d, δ) (where $\delta = (l, g, z, l')$) is allowed to player A in state (l, v) iff the following conditions hold:

  - $(l, v + d) \in States$ (this is the case when $v + d \models Inv(l)$).
  - $\delta = (l, g, z, l')$ is such that $Owner(\delta) = A$.
  - $v + d \models g$.
  - $[z \leftarrow 0](v + d) \models Inv(l')$

  Then $Mov((l, v), A)$ is the set of actions allowed to player A in state (l, v) when this set is non empty, and $Mov((l, v), A) = \{\perp\}$ otherwise.

- Given a state $(l, v) \in States$ and a tuple of actions $(m_A)_{A \in Agt}$ (a move m_{Agt})

allowed from this state, $Tab((l,v), m_{Agt})$ is defined as the set:

$$\Big\{((l,v),(l',v')) \;\Big|\; \exists B.\ d_B = min\{d_A \mid A \in Agt \text{ and } m_A = (d_A, \delta_A)\} \text{ and } \delta_B = (l, g_B, z_B, l') \text{ and } v' = [z_B \leftarrow 0](v + d_B)\Big\}$$

- For every $A \in Agt$, $\Omega'(A) = \{(l,v) \mid (l,v) \in States \text{ and } l \in \Omega(A)\}$.

Multi-player timed concurrent reachability games inherit the notions of path, history, play, strategy, strategy profile, outcome and *k-resilient* pseudo-Nash equilibrium via the semantics described above.

In this thesis, we consider only non-blocking multi-player timed concurrent reachability games. Non-blocking games are games in which for every state (l,v), at least one player has an allowed action:

$$\prod_{A \in Agt} Mov((l,v), A) \neq \{(\bot)_{A \in Agt}\}$$

3.3 Region Games

In this section, we recall the notion of region games from [6, 7]. Region games are finite games derived from the region based abstraction of timed games. This relies on the classical notion of *clock regions* and *region automaton* associated with a timed automaton [3]. For more details on *clock regions* and *region automaton*, we refer the reader to Appendix A.

We now recall the definition of region games from [6, 7].

Definition 3.3.1. Let $G = (Loc, \chi, Inv, Trans, Agt, Owner, \Omega)$ be a multi-player timed concurrent reachability game. We define a region game G_R associated to G as $G_R = (States_R, Edg_R, Agt, Act_R, Mov_R, Tab_R, \Omega_R)$ where:

- $States_R = \{(l,r) \in Loc \times \Re \mid r \models Inv(l)\}$ where $\Re$ is the set of clock regions.

- Edg_R is the set of transitions of the region automaton underlying G.

- $Act_R = \{(r, p, \delta) \mid r \in \Re,\ p \in \{1, 2, 3\} \text{ and } \delta \in Trans\} \cup \{\bot\}$.

- $Mov_R : States_R \times Agt \to 2^{Act_R} \setminus \{\emptyset\}$ is such that:

$$\begin{aligned} Mov_R((l, r), A) = \Big\{(r', p, \delta) \Bigm| & \ r' \in Succ(r),\ r' \models Inv(l), \\ & p \in \{1, 2, 3\} \text{ if } r' \text{ is time-elapsing, else } p = 1, \\ & \delta = (l, g, z, l') \in Trans \text{ is such that } r' \models g \\ & \text{and } [z \leftarrow 0]r' \models Inv(l') \text{ and } Owner(\delta) = A\Big\} \end{aligned}$$

  if it is non-empty and $Mov_R((l, r), A) = \{\bot\}$ otherwise. Roughly, the index p allows the players to say if they want to play first, second or later if their region is selected.

- $Tab_R : States_R \times Act_R^{Agt} \to 2^{Edg_R} \setminus \{\emptyset\}$ is such that for every $(l, r) \in States_R$ and every $m_{Agt} \in \prod_{A \in Agt} Mov_R((l, r), A)$, if we write r' for $min\{r_A \mid m_A = (r_A, p_A, \delta_A)\}$ and p' for $min\{p_A \mid m_A = (r', p_A, \delta_A)\}$, then

$$\begin{aligned} Tab_R((l, r), m_{Agt}) = \Big\{((l, r), (l_B, [z_B \leftarrow 0]r_B)) \Bigm| & \ m_B = (r_B, p_B, \delta_B) \\ & \text{with } r_B = r',\ p_B = p' \text{ and } \delta_B = (l, g_B, z_B, l_B)\Big\} \end{aligned}$$

- For every $A \in Agt$, $\Omega_R(A) = \{(l, r) \mid (l, r) \in States_R \text{ and } l \in \Omega(A)\}$

3.4 *K-Resilient* Nash Equilibria in Timed Games

In this section, we prove that the translation of a multi-player timed concurrent reachability game to its associated region game preserves a *k-resilient* Nash equilibrium (if one exists). Thus, existence of *k-resilient* Nash equilibrium in a multi-player timed concurrent reachability game can be checked by translating the timed game to its associated region game and then checking existence of *k-resilient* Nash equilibrium in the region game.

Lemma 3.4.1. *Consider two games G and G' involving the same set of agents (Agt) with reachability objectives given as Ω and Ω' respectively. For some $k \leq |Agt|$, assume that there exists a binary relation $\backsim_k$ between states of G and states of G' such that if $s \backsim_k s'$, then:*

- *For every $A \in Agt$, if $s' \in \Omega'(A)$ then $s \in \Omega(A)$.*
- *For every move m_{Agt} in G, there exists a move m'_{Agt} in G' such that:*
  - *For every t' in G', there is $t \backsim_k t'$ in G s.t. $k - Susp_{G'}((s', t'), m'_{Agt}) \subseteq k - Susp_G((s, t), m_{Agt})$.*
  - *For every $(s, t) \in Tab(s, m_{Agt})$, there is a $(s', t') \in Tab(s', m'_{Agt})$ s.t. $t \backsim_k t'$.*

Then for every $P \subseteq Agt$ and every $X \subseteq 2^{Agt}_k$ such that $P \bigcap \left(\bigcup_{W \in X} W \right) = P$ and for every s and s' such that $s \backsim_k s'$, it holds:

1. *If $s \in k - Rep_G(P, X)$, then $s' \in k - Rep_{G'}(P, X)$.*
2. *For every $(s, t) \in Edg_{k-Rep}$, there exists $(s', t') \in Edg'_{k-Rep}$ s.t. $t \backsim_k t'$, where Edg_{k-Rep} and Edg'_{k-Rep} are the set of edges in transition systems $k - S_G(P, X)$ and $k - S_{G'}(P, X)$ respectively.*

Proof. The proof is by induction on the ordered pair (P, X).

Base case: When $P = \emptyset$, $k - Rep_G(\emptyset, X) = States$ and $k - Rep_{G'}(\emptyset, X) = States'$ for every $X \subseteq 2^{Agt}_k$. Hence the results hold trivially.

Inductive step: For some (P, X), assume that the implication holds true for every $(P', X') \lneq (P, X)$ which satisfy the condition $P' \bigcap \left(\bigcup_{W \in X'} W \right) = P'$.

Let us define a set R as:

$$R = \{s' \in States' \mid \exists s \in k - Rep_G(P, X) \text{ and } s \backsim_k s'\}$$

Pick $s' \in R$ and a corresponding $s \in k - Rep_G(P, X)$ such that $s \backsim_k s'$. In particular, for every $A \in P$, we have $s \notin \Omega(A)$, hence $s' \notin \Omega'(A)$. Therefore $R \cap \Omega'(A) = \emptyset$ for every $A \in P$.

Also, since $s \in k - Rep_G(P, X)$, there exists m_{Agt} such that for every $t \in States$, we have $t \in k - Rep_G(P', X')$ where $X' = k - Susp_G((s,t), m_{Agt}) \cap X$ and $P' = P \bigcap \left(\bigcup\limits_{W \in X'} W \right)$. Since $s \backsim_k s'$, there exists m'_{Agt} such that for every $t' \in States'$, there exists $t \backsim_k t'$ such that $k - Susp_{G'}((s', t'), m'_{Agt}) \subseteq k - Susp_G((s,t), m_{Agt})$. Then:

If $X \subseteq k - Susp_G((s,t), m_{Agt})$, then $X' = X$ and $P' = P$ which gives $t \in k - Rep_G(P, X)$ and therefore $t' \in R$.

Otherwise $k - Susp_G((s,t), m_{Agt}) \cap X \subsetneq X$. As a consequence, we also have $k - Susp_{G'}((s', t'), m'_{Agt}) \cap X \subsetneq X$. We have $X' = k - Susp_G((s,t), m_{Agt}) \cap X$, $P' = P \bigcap \left(\bigcup\limits_{W \in X'} W \right)$ and $t \in k - Rep_G(P', X')$. Now since $X' \subsetneq X$ and $P' \subseteq P$, we have $(P', X') \lneq (P, X)$. Applying induction hypothesis on (P', X'), we get $t' \in k - Rep_{G'}(P', X')$. Let $k - Susp_{G'}((s', t'), m'_{Agt}) \cap X = X''$ and $P \bigcap \left(\bigcup\limits_{W \in X''} W \right) = P''$. Then we have $X'' \subseteq X'$ and $P'' \subseteq P'$. Therefore by Lemma 2.3.7, $t' \in k - Rep_{G'}(P'', X'')$ where $X'' = k - Susp_{G'}((s', t'), m'_{Agt}) \cap X$ and $P'' = P \bigcap \left(\bigcup\limits_{W \in X''} W \right)$.

This proves that $R \subseteq k - Rep_{G'}(P, X)$ (as any $s' \in R$ satisfies both the conditions of Definition 2.3.6 to stay in $k - Rep_{G'}(P, X)$). Therefore the first property of the implication follows.

Now we need to prove the second property of the implication. Let $s \backsim_k s'$ and $(s,t) \in Edg_{k-Rep}$: there exists $m_{Agt} \in k - Secure_G(s, P, X)$ such that $(s,t) \in Tab(s, m_{Agt})$. Since $m_{Agt} \in k - Secure_G(s, P, X)$, any state $u \in States$ satisfies $u \in k - Rep_G(P', X')$ where $X' = k - Susp_G((s,u), m_{Agt}) \cap X$ and $P' = P \bigcap \left(\bigcup\limits_{W \in X'} W \right)$.

Since $s \backsim_k s'$, there exists m'_{Agt} such that for every $u' \in States'$, there exists $u \backsim_k u'$ such that $k - Susp_{G'}((s', u'), m'_{Agt}) \subseteq k - Susp_G((s,u), m_{Agt})$. From the first property of implication, we get that $u' \in k - Rep_{G'}(P', X')$ (this is because $u \in k - Rep_G(P', X')$). Let $k - Susp_{G'}((s', u'), m'_{Agt}) \cap X = X''$ and $P \bigcap \left(\bigcup\limits_{W \in X''} W \right) = P''$. Then we have $X'' \subseteq X'$ and $P'' \subseteq P'$. Therefore by Lemma 2.3.7, $u' \in k - Rep_{G'}(P'', X'')$.

Therefore, we have that, m'_{Agt} is such that for every $u' \in States'$, it holds: $u' \in k - Rep_{G'}(P'', X'')$ where $X'' = k - Susp_{G'}((s', u'), m'_{Agt}) \cap X$ and $P'' =$

$P \bigcap \left(\bigcup_{W \in X''} W \right)$. This proves that m'_{Agt} is a *k-secure move*, i.e., $m'_{Agt} \in k - Secure_{G'}(s', P, X)$. For this m'_{Agt}, we also have $(s', t') \in Tab(s', m'_{Agt})$ for some t' such that $t \backsim_k t'$. Such an $(s', t') \in Edg'_{k-Rep}$. This proves the second property of the implication. □

Consider a multi-player timed concurrent reachability game G and its associated region game G_R. We shall prove that the timed game G and its associated region game G_R simulate each other in the sense of Lemma 3.4.1. Our proofs are a natural generalization of those in [6, 7].

As in [6, 7], we pick three partial functions $f_1, f_2, f_3 : \mathbb{R}_+^{\chi} \times \Re \rightarrow \mathbb{R}_+$ such that, for every valuation v and every clock region r, if there is some $t \in \mathbb{R}_+$ such that $v + t \in r$, then all three functions are defined at (v, r), and we require that $v + f_i(v, r) \in r$ for every $i \in \{1, 2, 3\}$, and that $f_1(v, r) < f_2(v, r) < f_3(v, r)$ if r is open and $f_1(v, r) = f_2(v, r) = f_3(v, r)$ otherwise. If no such t exists, then all three functions are undefined at (v, r).

***From timed game* G *to region game* G_R**: We show that for every $k \leq |Agt|$, there is a binary relation $\backsim_k$ (as defined in Lemma 3.4.1) between states of G and states of G_R such that $(l, v) \backsim_k (l, r)$ if r is the region containing v. By definition of Ω in G and Ω_R in G_R, the first condition for the relation $\backsim_k$ as given in Lemma 3.4.1 holds.

Now, for every state (l, v) and every move $m_{Agt} = (m_A)_{A \in Agt}$ in G, we define the move $(\lambda_A)_{A \in Agt} = \lambda((l, v), m_{Agt})$ in G_R as follows: let $d^1 = min\{d_A \mid A \in Agt\ s.t.\ m_A = (d_A, \delta_A)\}$ and $d^2 = min\{d_A \mid A \in Agt\ s.t.\ m_A = (d_A, \delta_A) \text{ with } d_A > d^1\}$. Then for every $A \in Agt$:

- If $m_A = (d^1, \delta)$, then we set $\lambda_A = (r, 1, \delta)$ where r is the region corresponding to valuation $v + d^1$.

- If $m_A = (d^2, \delta)$, then we set $\lambda_A = (r, p, \delta)$ where r is the region corresponding to valuation $v + d^2$, and $p = 2$ if r is time-elapsing and $p = 1$ otherwise.

- If $m_A = (d_A, \delta)$ with $d_A > d^2$, then we set $\lambda_A = (r, p, \delta)$ where r is the region corresponding to valuation $v + d_A$, and $p = 3$ if r is time-elapsing and $p = 1$ otherwise.

- If $m_A = \perp$, then $\lambda_A = \perp$.

The above definition of $\lambda_{Agt} = (\lambda_A)_{A \in Agt}$ is same as in [6, 7]. Clearly, if m_A is allowed to A in state (l, v) in G, then λ_A is allowed to A in state (l, r) in G_R where r is the region containing v, since it corresponds to the same transition played in the correct region.

We now recall the following lemma (and its proof) from [7].

Lemma 3.4.2. *For every $((l, v), (l', v')) \in Tab((l, v), m_{Agt})$, it holds that: $((l, r), (l', r')) \in Tab_R((l, r), \lambda_{Agt})$ where r and r' are the regions containing v and v' respectively.*

Proof. As $((l, v), (l', v')) \in Tab((l, v), m_{Agt})$, there is some B such that $m_B = (d_B, \delta_B)$ with $d_B = min\{d_A \mid A \in Agt \ s.t. \ m_A = (d_A, \delta_A)\}$ and $\delta_B = (l, g_B, z_B, l')$ such that $v' = [z_B \leftarrow 0](v + d_B)$. In this case, we have $\lambda_B = (r_B, 1, \delta_B)$ with $v + d_B \in r_B$ and $r_B = min\{r_A \mid A \in Agt$ and $\lambda_A = (r_A, p_A, \delta_A)\}$. Therefore $((l, r), (l', r')) \in Tab_R((l, r), \lambda_{Agt})$ with $r' = [z_B \leftarrow 0]r_B$ so that r' contains $[z_B \leftarrow 0](v + d_B)$. □

Lemma 3.4.3. *For every region r', there is a valuation $v' \in r'$, such that for every $k \leq |Agt|$, it holds that:*

$$k - Susp_{G_R}(((l, r), (l', r')), \lambda_{Agt}) \subseteq k - Susp_G(((l, v), (l', v')), m_{Agt})$$

Proof. Let $P \in k - Susp_{G_R}(((l, r), (l', r')), \lambda_{Agt})$ and let λ'_P be an action tuple for coalition P ($\lambda'_P = (\lambda'_B)_{B \in P}$ with $\lambda'_B = (r'_B, p'_B, \delta'_B)$ for every $B \in P$) such that $((l, r), (l', r')) \in Tab_R((l, r), \lambda_{Agt}[P \to \lambda'_P])$.

We have to prove that $P \in k - Susp_G(((l, v), (l', v')), m_{Agt})$ for some $v' \in r'$. For this, we need to prove that there exists $v' \in r'$ and an action tuple m'_P for coalition P such that $((l, v), (l', v')) \in Tab((l, v), m_{Agt}[P \to m'_P])$. We first define m'_P. Let $m'_P = (m'_B)_{B \in P}$. For every $B \in P$, we define $m'_B = (d'_B, \delta'_B)$ as follows:

- If $r'_B < min(\{r_A \mid A \notin P\} \cup \{r'_A \mid A \in P \text{ and } A \neq B\})$, then $m'_B = (f_1(v, r'_B), \delta'_B)$.
- If $r'_B = min(\{r_A \mid A \notin P\} \cup \{r'_A \mid A \in P \text{ and } A \neq B\})$, then:
  - If $p'_B < min(\{p_A \mid A \notin P \text{ and } r_A = r'_B\} \cup \{p'_A \mid A \in P \text{ and } A \neq B \text{ and } r'_A = r'_B\})$, then $m'_B = (d'_B, \delta'_B)$ such that $v + d'_B \in r'_B$ and $d'_B < min(\{d_A \mid A \notin P \text{ and } r_A = r'_B\} \cup \{d'_A \mid A \in P \text{ and } A \neq B \text{ and } r'_A = r'_B\})$.
  - If $p'_B = min(\{p_A \mid A \notin P \text{ and } r_A = r'_B\} \cup \{p'_A \mid A \in P \text{ and } A \neq B \text{ and } r'_A = r'_B\})$, then $m'_B = (d'_B, \delta'_B)$ such that $v + d'_B \in r'_B$ and $d'_B = min(\{d_A \mid A \notin P \text{ and } r_A = r'_B\} \cup \{d'_A \mid A \in P \text{ and } A \neq B \text{ and } r'_A = r'_B\})$.
  - If $p'_B > min(\{p_A \mid A \notin P \text{ and } r_A = r'_B\} \cup \{p'_A \mid A \in P \text{ and } A \neq B \text{ and } r'_A = r'_B\})$, then $m'_B = (d'_B, \delta'_B)$ such that $v + d'_B \in r'_B$ and $d'_B > min(\{d_A \mid A \notin P \text{ and } r_A = r'_B\} \cup \{d'_A \mid A \in P \text{ and } A \neq B \text{ and } r'_A = r'_B\})$.
- Otherwise $m'_B = (f_{p'_B}(v, r'_B), \delta'_B)$

It remains to prove that $((l, v), (l', v')) \in Tab((l, v), m_{Agt}[P \to m'_P])$ for some $v' \in r'$. Since $((l, r), (l', r')) \in Tab_R((l, r), \lambda_{Agt}[P \to \lambda'_P])$, we have to prove that any player who plays minimal region and index in $\lambda_{Agt}[P \to \lambda'_P]$ also plays shortest delay in $m_{Agt}[P \to m'_P]$. Let the player who plays minimal region and index in $\lambda_{Agt}[P \to \lambda'_P]$ be C.

From our definition of action tuple m'_P, the following equivalence holds for every $B \in P$:

$$d'_B = min\Big(\{d_A \mid A \in Agt \backslash P \; s.t. \; m_A = (d_A, \delta_A)\} \cup \{d'_A \mid A \in P \; s.t. \; m'_A = (d'_A, \delta'_A)\}\Big)$$

$$\Leftrightarrow$$

$$r'_B = min\Big(\{r_A \mid A \in Agt \backslash P \; s.t. \; \lambda_A = (r_A, p_A, \delta_A)\} \cup \{r'_A \mid A \in P \; s.t. \; \lambda'_A = (r'_A, p'_A, \delta'_A)\}\Big)$$

and

$$p'_B = min\Big(\{p_A \mid A \in Agt \backslash P \; s.t. \; \lambda_A = (r'_B, p_A, \delta_A)\} \cup \{p'_A \mid A \in P \; s.t. \; \lambda'_A = (r'_B, p'_A, \delta'_A)\}\Big)$$

Now, if some $C \in P$ plays minimal region and index in $\lambda_{Agt}[P \to \lambda'_P]$, then from the above equivalence, C also plays shortest delay in $m_{Agt}[P \to m'_P]$ and the result follows.

And, if some $C \notin P$ plays minimal region and index in $\lambda_{Agt}[P \to \lambda'_P]$, then by construction of λ_{Agt}, $p_C = 1$ and $d_C = d^1$. Moreover, for every $B \in P$, r'_B is either r_C or a later region. In both cases, d'_B is larger than or equal to d^1. Therefore C also plays shortest delay in $m_{Agt}[P \to m'_P]$ and the result follows. □

From Lemma 3.4.2 and Lemma 3.4.3, the second condition for the relation $\backsim_k$ (as given in Lemma 3.4.1) between states of G and states of G_R holds. Hence we have shown that, for every $k \leq |Agt|$, there is a binary relation $\backsim_k$ (as defined in Lemma 3.4.1) between states of G and states of G_R such that $(l, v) \backsim_k (l, r)$ if r is the region containing v.

From region game G_R to timed game G: We now show that for every $k \leq |Agt|$, there is a binary relation $\backsim_k$ (as defined in Lemma 3.4.1) between states of G_R and states of G such that $(l, r) \backsim_k (l, v)$ if v is contained in the region r. By definition of Ω in G and Ω_R in G_R, the first condition for the relation $\backsim_k$ as given in Lemma 3.4.1 holds.

Now, consider an action tuple (a move) $\alpha_{Agt} = (\alpha_A)_{A \in Agt}$ from state (l, r) in G_R. From state (l, v) in G (s.t. v is contained in r), we define the action tuple $(\mu_A)_{A \in Agt} = (\mu((l, v), (\alpha_A)))_{A \in Agt}$ in G as follows:

- If $\alpha_A = \perp$, then $\mu((l, v), (\alpha_A)) = \perp$.

- If $\alpha_A = (r_A, p_A, \delta_A)$, then we let $\mu((l, v), (\alpha_A)) = (f_{p_A}(v, r_A), \delta_A)$.

The above definition of $\mu_{Agt} = (\mu_A)_{A \in Agt}$ is same as in [6, 7]. Again, if α_A is allowed to player A in state (l, r) in G_R, then μ_A is allowed to A in state (l, v) in G, since it corresponds to playing the same transition in the same region.

We now recall the following lemma (and its proof) from [7].

Lemma 3.4.4. *For any transition $((l, r), (l', r')) \in Tab_R((l, r), \alpha_{Agt})$, there exists a transition $((l, v), (l', v')) \in Tab((l, v), \mu_{Agt})$ where v and v' are contained in regions*

r and r' respectively.

Proof. Since $((l,r),(l',r')) \in Tab_R((l,r),\alpha_{Agt})$, there is a player B such that $r_B = min\{r_A \mid A \in Agt\ s.t.\ \alpha_A = (r_A, p_A, \delta_A)\}$ and $p_B = min\{p_A \mid A \in Agt\ s.t.\ \alpha_A = (r_B, p_A, \delta_A)\}$ and $\delta_B = (l, g_B, z_B, l')$ and $r' = [z_B \leftarrow 0]r_B$. Therefore, $f_{p_B}(v, r_B) = min\{f_{p_A}(v, r_A) \mid A \in Agt\ s.t.\ \alpha_A = (r_A, p_A, \delta_A)\}$ and $v' = [z_B \leftarrow 0](v + f_{p_B}(v, r_B))$ is such that $(l', v') \in Tab((l,v),\mu_{Agt})$ with $v' \in r'$. □

Lemma 3.4.5. *For any v', writing r' for the region containing v', we have that, for every $k \leq |Agt|$:*

$$k - Susp_G(((l,v),(l',v')),\mu_{Agt}) \subseteq k - Susp_{G_R}(((l,r),(l',r')),\alpha_{Agt})$$

Proof. Let $P \in k - Susp_G(((l,v),(l',v')),\mu_{Agt})$ and let μ'_P be an action tuple for coalition P ($\mu'_P = (\mu'_B)_{B \in P}$ with $\mu'_B = (d'_B, \delta'_B)$ for every $B \in P$) such that $((l,v),(l',v')) \in Tab((l,v),\mu_{Agt}[P \to \mu'_P])$.

We have to prove that $P \in k - Susp_{G_R}(((l,r),(l',r')),\alpha_{Agt})$. For this, we need to prove that there exists an action tuple α'_P for coalition P such that $((l,r),(l',r')) \in Tab((l,r),\alpha_{Agt}[P \to \alpha'_P])$. Let us define a value d^B_{min} as:

$$d^B_{min} = min\Big(\{d_A \mid A \in Agt \backslash P\ s.t.\ \mu_A = (d_A, \delta_A)\} \cup \{d'_A \mid A \in P\ s.t.\ \mu'_A = (d'_A, \delta'_A)\}\Big)$$

Now we define $\alpha'_P = (\alpha'_B)_{B \in P}$ as follows: for every $B \in P$, we let $\alpha'_B = (r'_B, p'_B, \delta'_B)$ where r'_B is the region associated to $v + d'_B$ and $p'_B = 1$ if r'_B is not time-elapsing or $d'_B = d^B_{min}$ and $p'_B = 3$ otherwise.

It remains to prove that $((l,r),(l',r')) \in Tab((l,r),\alpha_{Agt}[P \to \alpha'_P])$. Since $((l,v),(l',v')) \in Tab((l,v),\mu_{Agt}[P \to \mu'_P])$, we need to prove that the player(s) playing minimal delay in $\mu_{Agt}[P \to \mu'_P]$ also play(s) minimal region and index in $\alpha_{Agt}[P \to \alpha'_P]$. Let the player playing minimal delay in $\mu_{Agt}[P \to \mu'_P]$ be C.

If $C \in P$, it is the case that $d'_C \leq d^1$, then if C plays a region not containing $v+d^1$, then $d'_C < d^1$ and C is the only player proposing minimal delay in $\mu_{Agt}[P \to \mu'_P]$

and minimal region and index in $\alpha_{Agt}[P \to \alpha'_P]$ and the result follows. If C plays in a region containing $v + d^1$, then again any player (including C) proposing minimal delay in $\mu_{Agt}[P \to \mu'_P]$ also proposes minimal region and index in $\alpha_{Agt}[P \to \alpha'_P]$ and the result follows.

If $C \notin P$, then also any player proposing minimal delay in $\mu_{Agt}[P \to \mu'_P]$ also proposes minimal region and index in $\alpha_{Agt}[P \to \alpha'_P]$ and the result follows. □

From Lemma 3.4.4 and Lemma 3.4.5, the second condition for the relation $\backsim_k$ (as given in Lemma 3.4.1) between states of G_R and states of G holds. Hence we have shown that, for every $k \leq |Agt|$, there is a binary relation $\backsim_k$ (as defined in Lemma 3.4.1) between states of G_R and states of G such that $(l, r) \backsim_k (l, v)$ if v is contained in the region r.

Hence, from Lemmas 3.4.1, 3.4.2, 3.4.3, 3.4.4 and 3.4.5, we have proved that for every $k \leq |Agt|$, there is a binary relation $\backsim_k$ (as defined in Lemma 3.4.1), such that the the relation $\backsim_k$ exists between states of G and states of G_R and the relation $\backsim_k$ also exists between states of G_R and states of G (i.e., it is the case that for every state (l, v) in G and every state (l, r) in G_R, if r is the region containing v then $(l, v) \backsim_k (l, r)$ and $(l, r) \backsim_k (l, v)$). Hence the timed game G and its associated region game G_R simulate each other in the sense of Lemma 3.4.1 (via relation $\backsim_k$ for every $k \leq |Agt|$).

Theorem 3.4.6. *Let G be a multi-player timed concurrent reachability game and let G_R be its associated region game. Then for every $k \leq |Agt|$, there is a k-resilient pseudo-Nash equilibrium in G from $(s, \boldsymbol{0})$ if and only if there is a k-resilient pseudo-Nash equilibrium in G_R from $(s, [\boldsymbol{0}])$ where $[\boldsymbol{0}]$ is the region associated to valuation $\boldsymbol{0}$. Furthermore, this equivalence is constructive.*

Proof. We have proved earlier that the timed game G and its associated region game G_R simulate each other in the sense of Lemma 3.4.1 via relation $\backsim_k$ for every $k \leq |Agt|$. In particular, we have proved earlier that for every $k \leq |Agt|$, there is a binary relation $\backsim_k$ (as defined in Lemma 3.4.1) such that for every state (l, v) in G

and every state (l, r) in G_R, if r is the region containing v then $(l, v) \backsim_k (l, r)$ and $(l, r) \backsim_k (l, v)$. Now from Lemma 3.4.1, it can be inferred that for every $P \subseteq Agt$ and every $X \subseteq 2_k^{Agt}$ such that $P \bigcap \left(\bigcup_{W \in X} W \right) = P$ and for every state (l, v) in G and corresponding state (l, r) in G_R such that r is the region containing v, the following two results hold:

1. $(l, v) \in k - Rep_G(P, X) \Leftrightarrow (l, r) \in k - Rep_{G_R}(P, X)$.

2. $((l, v), (l', v')) \in Edg_{k-Rep}^G \Leftrightarrow ((l, r), (l', r')) \in Edg_{k-Rep}^{G_R}$ where r' is the region containing v' and Edg_{k-Rep}^G and $Edg_{k-Rep}^{G_R}$ are the set of edges in the transition systems $k - S_G(P, X)$ and $k - S_{G_R}(P, X)$ respectively.

It follows that for every $P \subseteq Agt$, there is a path ρ in $k - S_G(P, 2_k^{Agt})$ if and only if there is a corresponding path ρ' in $k - S_{G_R}(P, 2_k^{Agt})$ which visits exactly the same regions visited by ρ.

Therefore, there is an infinite path ρ in $k - S_G(P, 2_k^{Agt})$ from $(s, \mathbf{0})$ which visits $\Omega(A)$ for every $A \notin P$, if and only if, there is an infinite path ρ' in $k - S_{G_R}(P, 2_k^{Agt})$ from $(s, [\mathbf{0}])$ which visits $\Omega_R(A)$ for every $A \notin P$.

From Theorem 2.3.13, we conclude that there is a *k-resilient* pseudo-Nash equilibrium in G from $(s, \mathbf{0})$ if and only if there is a *k-resilient* pseudo-Nash equilibrium in G_R from $(s, [\mathbf{0}])$. This equivalence is constructive because given the strategy profile for one game, the strategy profile for the other game can be computed using the translations λ (which maps moves in G to equivalent moves in G_R) or μ (which maps moves in G_R to equivalent moves in G) as the case may be. This computation is possible because for every history in G, there is an equivalent projection of that history in G_R which visits exactly the same regions. □

By Theorem 3.4.6, we can infer that the translation of a multi-player timed concurrent reachability game to its associated region game preserves a *k-resilient* pseudo-Nash equilibrium (if one exists). Hence, in order to check the existence of *k-resilient* pseudo-Nash equilibrium in a multi-player timed concurrent reachability game, we can translate the timed game to its associated region game and then check the existence of *k-resilient* pseudo-Nash equilibrium in the region game.

As the region games in this case are finite non-deterministic multi-player concurrent reachability games, we can apply Algorithm 2.4.1 to check the existence of *k-resilient* pseudo-Nash equilibrium in the region games. Hence, in order to check the existence of *k-resilient* pseudo-Nash equilibrium in a multi-player timed concurrent reachability game, we can first translate the timed game to its associated region game and then we can apply Algorithm 2.4.1 to check the existence of *k-resilient* pseudo-Nash equilibrium in the region game and report its output.

The translation of a multi-player timed concurrent reachability game to its associated region game results in an exponential blow-up in the number of states (w.r.t the number of locations and clocks in the timed game) and an exponential blow-up in the size of transition table. Recall that Algorithm 2.4.1 runs in time exponential with respect to the number of states and the size of transition table but doubly exponential with respect to the number of agents. Although, the translation of a multi-player timed concurrent reachability game to its associated region game results in an exponential blow-up in the number of states and the size of transition table, the number of agents is unchanged (as a timed game and its associated region game have the same set of agents). Therefore, even after the exponential blow-up in the number of states and the size of transition table, the algorithm runs in time doubly exponential with respect to the number of states, size of transition table and the number of agents. Hence the algorithm is doubly exponential with respect to the size of input and is still in 2-EXPTIME.

In case a *k-resilient* pseudo-Nash equilibrium exists, it can be computed in the region game using the generic method described in Theorem 2.3.14 and its proof. The *k-resilient* pseudo-Nash equilibrium in the corresponding timed game can then be computed using the translation μ (which maps moves in the region game to equivalent moves in the timed game). Recall form Theorem 3.4.6 that the equivalence of existence of *k-resilient* pseudo-Nash equilibrium in a timed game and its associated region game is constructive. Hence this computation is possible as for every history in the timed game, there is an equivalent projection of that history in the region game which visits exactly the same regions.

Chapter 4

Conclusion and Future Work

In this thesis, we considered non-deterministic multi-player concurrent reachability games in untimed and timed settings and we studied the solution concept of *k-resilient* Nash equilibrium in these games. We proved some properties that characterize *k-resilient* Nash equilibria in these games when only pure strategies are allowed. These properties were then used to develop algorithms for checking existence of *k-resilient* Nash equilibrium in finite non-deterministic multi-player concurrent reachability games and in multi-player timed concurrent reachability games when only pure strategies are allowed. Our algorithms run in time, doubly exponential with respect to the size of input and hence are in 2-EXPTIME. Further, the properties that we proved also contain all the necessary information to compute a *k-resilient* Nash equilibrium if it exists and hence we gave a generic method to compute a *k-resilient* Nash equilibrium if it exists.

In particular, we generalized the concept of *suspect players* [6–9] and *repellor sets* [6–8] to *k-suspect coalitions* and *k-repellor sets* respectively. We also extended the notion of *secure moves* [6, 7] to *k-secure moves.* We defined the notion of transition systems $k - S_G(P, X)$ where $P \subseteq Agt$ and $X \subseteq 2^{Agt}_k$ (here Agt is the set of players (agents) in the game G and 2^{Agt}_k is set of all subsets of agents of size at most k) which are a generalization of the transition systems $S_G(P)$ as defined in [6, 7] but extended in a natural way to suit our setting. We proved various properties concerning *k-suspect coalitions*, *k-repellor sets*, *k-secure moves* and the transition systems $k - S_G(P, X)$ that characterize *k-resilient* Nash equilibria in non-deterministic multi-player concurrent reachability games.

The main result that we proved is that there exists a *k-resilient* pseudo-Nash equilibrium in a non-deterministic multi-payer concurrent reachability game G from a state s with a payoff vector v_{Agt}, if and only if, letting P be the set of players whose payoffs are 0 in the payoff vector v_{Agt}, there is an infinite path π in the transition system $k - S_G(P, 2_k^{Agt})$ which starts in s and visits $\Omega(A)$ for every A not in P (here $\Omega(A)$ is a subset of states which is the reachability objective of player A). This result gave us a necessary and sufficient condition for existence of *k-resilient* pseudo-Nash equilibrium in a non-deterministic multi-player concurrent reachability game and was used to develop the algorithm for checking existence of *k-resilient* pseudo-Nash equilibrium in finite non-deterministic multi-player concurrent reachability games when only pure strategies are allowed. We also gave a generic method to compute a *k-resilient* pseudo-Nash equilibrium if it exists as the various properties that we proved contain all the necessary information for computing a *k-resilient* pseudo-Nash equilibrium if it exists.

In order to decide the existence of *k-resilient* pseudo-Nash equilibrium in multi-player timed concurrent reachability games, we used the classical region based abstraction and constructed the region game. The construction of the region game is same as in [6, 7]. We proved that the translation of a timed game to its associated region game preserves a *k-resilient* pseudo-Nash equilibrium (if any) and hence the existence of *k-resilient* pseudo-Nash equilibrium in a timed game can be decided by checking the existence of *k-resilient* pseudo-Nash equilibrium in the corresponding region game. As a region game is a finite non-deterministic multi-player concurrent reachability game, we can apply the algorithm that we developed for checking existence of *k-resilient* pseudo-Nash equilibrium in these games. Further, the translation proposed (from a timed game to its associated region game) can also be used to compute a *k-resilient* pseudo-Nash equilibrium if it exists.

Our algorithms, constructions and proofs are a natural generalization of those in

[6, 7] used to characterize Nash equilibria in non-deterministic multi-player concurrent reachability games. Our algorithms take time, doubly exponential with respect to the size of input and hence are in 2-EXPTIME.

Some possible future extensions of this work are:

1. We have not been able to prove the optimality of our algorithms. It may be interesting to check if the algorithms can be made better using non-determinism or if 2-EXPTIME (complexity class of proposed algorithms) is actually the lower bound for the problems.

2. We have considered games where only pure strategies are allowed. The work may be extended to games where players can have mixed strategies.

3. We have considered games with reachability objectives only. The work may be extended to study *k-resilient* Nash equilibria in games with other qualitative objectives like safety objectives, Büchi objectives, co-Büchi objectives, Rabin objectives and parity objectives.

Appendix A

The Theory of Timed Automata

In this appendix, we briefly describe the theory of timed automata required for the proper understanding of our work in timed games (discussed in Chapter 3). The contents of this appendix are taken from [3].

For formal specification and verification of real time systems that involve timing constraints (such as flight control systems installed in an aeroplane), we need to include the timing constraints while modelling the system specification. If a system desires some properties that include timing constraints, we should be able to include the concept of 'time' in the system specification. In order to verify that the system has such desired properties (that include timing constraints), the system model must be able to capture such properties completely in its specification. For this purpose, Alur and Dill [3] define a concept of timed automata. Timed automata are a natural extension of finite automata in the sense that transitions in timed automata not just read a symbol (over an alphabet) but also take into account the 'time' at which a symbol is read. If a symbol represents an event, then transitions in timed automata depend not only on the occurrence of an event, but also on the time of occurrence of an event. Thus, timed automata are able to capture properties that involve timing constraints and hence are useful in formal specification and verification of real time systems.

A.1 Timed Automata

We begin by recalling the the definition of time sequences from [3].

Definition A.1.1. A time sequence $\tau = \tau_1\tau_2\ldots$ is an infinite sequence of time values $\tau_i \in \mathbb{R}_+$ satisfying the following properties:

1. *Monotonicity:* τ increases strictly monotonically, i.e., $\tau_i < \tau_{i+1}$ for every $i \geq 1$.

2. *Progress:* For every $t \in \mathbb{R}_+$, there is some $i \geq 1$ such that $\tau_i > t$.

We now recall the definition of timed words from [3].

Definition A.1.2. A timed word over an alphabet Σ is a pair (σ, τ) where $\sigma = \sigma_1\sigma_2\ldots$ is an infinite word over Σ and $\tau = \tau_1\tau_2\ldots$ is a time sequence. A timed language over Σ is a set of timed words over Σ.

If a timed word (σ, τ) is viewed as an input to an automaton, it presents the symbol σ_i at time τ_i. If each symbol σ_i is interpreted to denote the occurrence of an event, then the corresponding component τ_i is interpreted as the time of occurrence of σ_i. Sometimes it may be required to allow multiple events to occur simultaneously, i.e., it may be required to allow the same time value to be associated with multiple consecutive events in the sequence. This may be achieved by using a slightly different definition of timed words requiring a time sequence to increase only monotonically (i.e., $\tau_i \leq \tau_{i+1}$ for every $i \geq 1$) instead of strictly monotonically. Alur and Dill [3] prove that all their results hold in this alternative model also.

Transition tables with timing constraints: When an automaton makes a state transition, the choice of the next state depends on the input symbol read. In case of timed transition tables, we want this choice to also depend upon the time at which the input symbol is read relative to the times of previously read symbols. For this purpose, Alur and Dill [3] associate a finite set of real valued clocks to timed transition tables. A clock can be reset to zero simultaneously with any transition. At any instant, the reading of a clock equals the time elapsed since the last time it was reset. With each transition, we can associate a clock constraint and require that the transition may be taken only if the current values of the clocks satisfy this constraint.

We now recall the concept clock valuations as defined in [3, 6, 7].

Definition A.1.3. Consider a finite set of clocks χ. A valuation v over the finite set of clocks χ is an application $v : \chi \to \mathbb{R}_+$ that assigns to each clock $x \in \chi$, a positive real number that signifies the units of time since the clock x was last reset.

If v is a valuation over the set of clocks χ and $t \in \mathbb{R}_+$, then $v + t$ is the valuation that assigns to each $x \in \chi$, the value $v(x) + t$. Also, $t \cdot v$ is the valuation that assigns to each $x \in \chi$, the value $t \cdot v(x)$.

If v is a valuation over the set of clocks χ and $\varphi \subseteq \chi$, then $[\varphi \leftarrow t]v$ is the valuation that assigns the value t to each $y \in \varphi$ and the value $v(x)$ to each $x \in \chi \setminus \varphi$.

We now recall the concept of clock constraints from [3, 6, 7].

Definition A.1.4. A clock constraint over the set of clocks χ is a formula built on the grammar $\zeta(\chi) \ni g ::= x \backsim c \mid g \wedge g$, where x ranges over χ, $\backsim \in \{<, \leq, =, \geq, >\}$ and c is an integer.

A valuation v over a set of clocks χ satisfies a clock constraint g over χ if on assigning the values $v(x)$ to each $x \in \chi$, the clock constraint g evaluates to true. If a valuation v satisfies a clock constraint g, we write it as $v \models g$.

We now recall the definition of timed transition tables from [3].

Definition A.1.5. A timed transition table T is a 5-tuple, $T = (\Sigma, S, S_0, \chi, E)$ where:

- Σ is a finite alphabet.
- S is a finite set of states.
- $S_0 \subseteq S$ is a set of start states.
- χ is a finite set of clocks.

- $E \subseteq S \times S \times \Sigma \times 2^{\chi} \times \zeta(\chi)$ gives the set of transitions. An edge (s, s', a, z, g) represents a transition from state s to state s' on input symbol a. The set $z \subseteq \chi$ gives the clocks to be reset with this transition, and g is a clock constraint over χ that must be true while this transition is taken.

Given a timed word (σ, τ), the timed transition table T starts in one of its start states at time 0 with all its clocks initialized to 0. As time advances, the values of all the clocks change reflecting the elapsed time. At time τ_i, T changes from state s to state s' using some transition of the form (s, s', σ_i, z, g) reading input σ_i, if the current valuation of the clocks satisfies g. With this transition the clocks in z are reset to 0 and thus start counting time with respect to the time of occurrence of this transition.

We now recall the definition of runs in timed transition tables from [3].

Definition A.1.6. A run r, denoted by $(\overline{s}, \overline{v})$, of a timed transition table $T = (\Sigma, S, S_0, \chi, E)$ over a timed word (σ, τ) is an infinite sequence of the form:

$$r : (s_0, v_0) \xrightarrow[\tau_1]{\sigma_1} (s_1, v_1) \xrightarrow[\tau_2]{\sigma_2} (s_2, v_2) \xrightarrow[\tau_3]{\sigma_3} \cdots$$

with $s_i \in S$ and $v_i \in \mathbb{R}_+^{\chi}$ is a valuation over χ for every $i \geq 0$, satisfying the following requirements:

- *Initiation:* $s_0 \in S_0$ and $v_0(x) = 0$ for every clock $x \in \chi$.

- *Consecution:* For every $i \geq 1$, there is an edge E of the form $(s_{i-1}, s_i, \sigma_i, z_i, g_i)$ such that $(v_{i-1} + \tau_i - \tau_{i-1})$ satisfies g_i and v_i equals $[z_i \leftarrow 0](v_{i-1} + \tau_i - \tau_{i-1})$.

For a run r, the set $inf(r)$ consists of those states $s \in S$ such that $s = s_i$ for infinitely many $i \geq 0$ (i.e., the set $inf(r)$ consists of those states that occur infinitely often in the run r).

We now recall the definition of timed Büchi automata from [3].

Definition A.1.7. A timed Büchi automaton is a 6-tuple, $M = (\Sigma, S, S_0, \chi, E, F)$ where $(\Sigma, S, S_0, \chi, E)$ is a timed transition table and $F \subseteq S$ is a set of accepting states.

A run $r = (\overline{s}, \overline{v})$ of a timed Büchi automaton over a timed word (σ, τ) is an accepting run iff $inf(r) \cap F \neq \emptyset$.

For a timed Büchi automaton M, the language $L(M)$ of timed words it accepts is defined to be the set $\{(\sigma, \tau) \mid M \text{ has an accepting run over } (\sigma, \tau)\}$.

We now recall the definition of timed regular languages from [3].

Definition A.1.8. A timed language L is a timed regular language iff $L = L(M)$ for some timed Büchi automaton M.

We now recall the following theorem from [3].

Theorem A.1.9. *The class of timed regular languages is closed under finite union and intersection.*

We now recall the definition of timed Muller automata from [3].

Definition A.1.10. A timed Muller automaton is a 6-tuple, $M = (\Sigma, S, S_0, \chi, E, \mathbb{F})$ where $(\Sigma, S, S_0, \chi, E)$ is a timed transition table and $\mathbb{F} \subseteq 2^S$ specifies an acceptance family.

A run $r = (\overline{s}, \overline{v})$ of a timed Muller automaton over a timed word (σ, τ) is an accepting run iff $inf(r) \in \mathbb{F}$.

For a timed Muller automaton M, the language $L(M)$ of timed words it accepts is defined to be the set $\{(\sigma, \tau) \mid M \text{ has an accepting run over } (\sigma, \tau)\}$.

We now recall the following theorem from [3].

Theorem A.1.11. *A timed language is accepted by some timed Büchi automaton iff it is accepted by some timed Muller automaton.*

A.2 Clock Regions

At every point in time, the future behaviour of a timed transition table is determined by its state and the current valuation of its clocks. This motivates the following definition of extended states in timed transition tables [3].

Definition A.2.1. For a timed transition table $T = (\Sigma, S, S_0, \chi, E)$, an extended state is a pair (s, v) where $s \in S$ and v is valuation over the set of clocks χ.

The number of such extended states is uncountable. But if two extended states with the same T-state agree on the integral parts of all clock values and also agree on the ordering of the fractional parts of all clock values, then the runs starting from the two extended states are very similar. This is because integral parts of the clock values decide whether or not a particular clock constraint is met, whereas the ordering of fractional parts decides which clock will change its integral part first. The integral parts of clock values can get arbitrarily large. But if a clock x is never compared with a constant greater than c, then its actual value once it exceeds c, is of no consequence in deciding the allowed paths.

Now we formalize this notion as in [3]. For every $t \in \mathbb{R}_+$, let $\lfloor t \rfloor$ denote the integral part of t and let $fract(t)$ denote the fractional part of t, i.e., $t = \lfloor t \rfloor + fract(t)$. Assume that every clock in χ appears in some clock constraint.

We now recall the definition of the equivalence relation $\backsim$ over the set of all the clock valuations over χ from [3].

Definition A.2.2. Let $T = (\Sigma, S, S_0, \chi, E)$ be a timed transition table. For every $x \in \chi$, let c_x be the largest integer c such that $(x \leq c)$ or $(x \geq c)$ is a sub-formula of some clock constraint appearing in E. The equivalence relation $\backsim$ is defined over the set of all clock valuations over χ; for clock valuations v and v', $v \backsim v'$ iff the following conditions hold:

1. For every $x \in \chi$, either $\lfloor v(x) \rfloor$ and $\lfloor v'(x) \rfloor$ are same, or both $v(x)$ and $v'(x)$ are greater than c_x.

2. For every $x,\ y \in \chi$ with $v(x) \leq c_x$ and $v(y) \leq c_y$, $fract(v(x)) \leq fract(v(y))$ iff $fract(v'(x)) \leq fract(v'(y))$.

3. For every $x \in \chi$ with $v(x) \leq c_x$, $fract(v(x)) = 0$ iff $fract(v'(x)) = 0$.

A clock region for T is an equivalence class of clock valuations induced by $\backsim$. As in [3], we use the notation $[v]$ to denote the clock region to which v belongs.

Each clock region can be uniquely characterized by a finite set of clock constraints it satisfies.

Note that there are only finite number of regions. Also note that for a clock constraint g of T, if $v \backsim v'$, then v satisfies g iff v' satisfies g. We say that a clock region α satisfies a clock constraint g iff every $v \in \alpha$ satisfies g. Each region can be represented by specifying:

1. For every clock $x \in \chi$, one clock constraint from the set:

$$\{x = c \mid c = 0, 1, \ldots, c_x\} \cup \{c - 1 < x < c \mid c = 1, \ldots, c_x\} \cup \{x > c_x\}$$

2. For every pair of clocks x and y such that $c - 1 < x < c$ and $d - 1 < y < d$ appear in (1) above for some c and d, whether $fract(x)$ is less than, equal to, or greater than $fract(y)$.

We now recall the following lemma from [3].

Lemma A.2.3. *The number of clock regions for a timed transition table T is finite and is bounded exponentially with respect to the number of clocks and the length of clock constraints of T.*

A.3 Region Automata

Consider a timed transition table $T = (\Sigma, S, S_0, \chi, E)$. We are interested in constructing a (untimed) transition table whose paths mimic the runs of T in a certain way. As in [3], we denote this desired transition table by $R(T)$, the region automaton of T. A state of $R(T)$ records the state of the timed transition table T and the equivalence class of the current clock valuation. The state of $R(T)$ is of the form (s, α) with $s \in S$ and α is a clock region. The intended interpretation is that whenever the extended state of T is (s, v), the state of $R(T)$ is $(s, [v])$. The region automaton starts in some state $(s_o, [v_0])$ where s_0 is a start state of T and the clock valuation v_0 assigns 0 to every clock. The transition relation of $R(T)$ is defined so that the intended simulation is obeyed. It has an edge from (s, α) to (s', α') labelled with a iff T in state s with clock valuation $v \in \alpha$ can make a transition on a to the extended state (s', v') for some $v' \in \alpha'$. The edge relation can be conveniently defined using a time-successor relation over the clock regions. We now recall the definition of time-successor relation over the set of clock regions from [3].

Definition A.3.1. A clock region α' is a time-successor of a clock region α iff for every $v \in \alpha$, there exists $t \in \mathbb{R}_+$ such that $v + t \in \alpha'$.

Intuitively, the time-successors of a clock region α are the clock regions that will be visited by a clock valuation $v \in \alpha$ as time progresses. The set of all the time-successors of a clock region α is denoted by $Succ(\alpha)$. If α' is a time-successor of α, we write it as $\alpha' \in Succ(\alpha)$.

Let $T = (\Sigma, S, S_0, \chi, E)$ be a timed transition table. For every $x \in \chi$, let c_x be the largest integer c such that $(x \leq c)$ or $(x \geq c)$ is a sub-formula of some clock constraint appearing in E. Given a clock region α, we now recall the method of computing the set of all the time-successors of α ($Succ(\alpha)$) form [3]. First observe that the time-successor relation is a transitive relation. Now consider different cases:

1. First suppose that α satisfies the constraint $(x > c_x)$ for every clock $x \in \chi$. In this case, the only time successor of α is α itself.

2. Now suppose that the set χ_0 of clocks x such that α satisfies the constraint $(x = c)$ for some $c \leq c_x$ is non-empty. In this case, as time progresses, the fractional parts of the clocks in χ_0 become non-zero, and the clock region changes immediately. In this case the time-successors of α are same as the time-successors of clock region β specified as below:

    (a) For $x \in \chi_0$, if α satisfies $(x = c_x)$ then β satisfies $(x > c_x)$, otherwise if α satisfies $(x = c)$ for $c < c_x$, then β satisfies $(c < x < c + 1)$. For $x \notin \chi_0$, the constraint in β is same as that in α.

    (b) For clocks x and y such that $x < c_x$ and $y < c_y$ hold in α, the ordering relationship in β between their fractional parts is same as that in α.

3. If both the above cases do not apply, then let χ_0 be the set of clocks x for which α does not satisfy $(x > c_x)$ and which have the maximal fractional part (i.e., for all clocks y for which α does not satisfy $(y > c_y)$, $fract(y) \leq fract(x)$ is a constraint of α). In this case, as the time progresses, the clocks in χ_0 assume integer values. Let β be the clock region specified by:

   (a) For $x \in \chi_0$, if α satisfies $(c - 1 < x < c)$ then β satisfies $(x = c)$. For $x \notin \chi_0$, the constraint in β is same as that in α.

   (b) For clocks x and y such that $(c - 1 < x < c)$ and $(d - 1 < y < d)$ appear in (a) above, the ordering relationship in β between their fractional parts is same as that in α.

   In this case, the time-successors of α include α, β and all the time-successors of β.

A region α is time elapsing if it is a time-successor of itself otherwise it is non-time elapsing, i.e., if $\alpha \in Succ(\alpha)$ then α is time elapsing otherwise α is non-time elapsing.

We now recall the definition of region automata from [3].

Definition A.3.2. For a timed transition table $T = (\Sigma, S, S_0, \chi, E)$, the corresponding region automaton $R(T)$ is a transition table over the alphabet Σ.

- The states of $R(T)$ are of the form (s, α) where $s \in S$ and α is a clock region.
- The initial states are of the form $(s_0, [v_0])$ where $s_0 \in S_0$ and $v_0(x) = 0$ for every $x \in \chi$.
- $R(T)$ has an edge $((s, \alpha), (s', \alpha'), a)$ iff there is an edge $(s, s', a, z, g) \in E$ and a region α'' such that:
  - α'' is a time-successor of α.
  - α'' satisfies g.
  - $\alpha' = [z \leftarrow 0]\alpha''$

We now recall the concept of projection of a run r in T from [3].

Definition A.3.3. For a run $r = (\overline{s}, \overline{v})$ of T over the timed word (σ, τ) of the form:

$$r : (s_0, v_0) \xrightarrow[\tau_1]{\sigma_1} (s_1, v_1) \xrightarrow[\tau_2]{\sigma_2} (s_2, v_2) \xrightarrow[\tau_3]{\sigma_3} \cdots$$

define its projection $[r] = (\overline{s}, [\overline{v}])$ to be the sequence:

$$[r] : (s_0, [v_0]) \xrightarrow{\sigma_1} (s_1, [v_1]) \xrightarrow{\sigma_2} (s_2, [v_2]) \xrightarrow{\sigma_3} \cdots$$

From the definition of the edge relation for $R(T)$, it follows that $[r]$ is a run of $R(T)$ over σ. Since time progresses without bound along r, every clock $x \in \chi$ is either reset infinitely often or (from a certain time onwards), it increases without bound. Hence for every $x \in \chi$, for infinitely many $i \geq 0$, $[v_i]$ satisfies $[(x = 0) \vee (x > c_x)]$. We now recall the definition of progressive runs from [3].

Definition A.3.4. A run $r = (\overline{s}, \overline{\alpha})$ of the region automaton $R(T)$ of the form:

$$r : (s_0, \alpha_0) \xrightarrow{\sigma_1} (s_1, \alpha_1) \xrightarrow{\sigma_2} (s_2, \alpha_2) \xrightarrow{\sigma_3} \cdots$$

is progressive iff for every clock $x \in \chi$, there are infinitely many $i \geq 0$ such that α_i satisfies $[(x = 0) \vee (x > c_x)]$

Thus, for a run r of T over (σ, τ), $[r]$ is a progressive run of $R(T)$ over σ.

We now recall the following lemma from [3].

Lemma A.3.5. *If r is a progressive run of $R(T)$ over σ, then there exists a time sequence τ and a run r' of T over (σ, τ) such that r equals $[r']$.*

Given a timed language L, we define the language $Untime(L)$ (as in [3]) as the set of words:

$$Untime(L) = \{\sigma \mid (\sigma, \tau) \in L \text{ for some time sequence } \tau\}$$

For a timed automaton M, its region automaton can be used to recognize $Untime(L(M))$. We now recall the following theorem from [3]. It is stated for timed Büchi automata, but it also holds for timed Muller automata.

Theorem A.3.6. *Given a timed Büchi automaton $M = (\Sigma, S, S_0, \chi, E, F)$ where $T = (\Sigma, S, S_0, \chi, E)$ is a timed transition table and $F \subseteq S$ is a set of accepting states, there exists a Büchi automaton over Σ which accepts $Untime(L(M))$ and the transition table of this Büchi automaton is $R(T)$.*

In this appendix, we have reviewed the basic theory of timed automata from [3] which is required for proper understanding of our work in timed games (discussed in Chapter 3).

Bibliography

[1] Ittai Abraham, Danny Dolev, Rica Gonen, and Joe Halpern. Distributed computing meets game theory: robust mechanisms for rational secret sharing and multiparty computation. In *Proceedings of the 25th Annual ACM Symposium on Principles of Distributed Computing (PODC'06)*, pages 53–62. ACM, 2006.

[2] Ittai Abraham, Danny Dolev, and Joseph Y. Halpern. Lower bounds on implementing robust and resilient mediators. In *Proceedings of the 5th Theory of Cryptography Conference*, volume 4948 of *Lecture Notes in Computer Science*, pages 302–319. Springer, 2008.

[3] Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183 – 235, 1994.

[4] Robert J. Aumann. Acceptable points in general cooperative n-person games. *Contributions to the Theory of Games IV, Annals of Mathematical Studies 40*, pages 287–324, 1959.

[5] B. Douglas Bernheim, Bezalel Peleg, and Michael D. Whinston. Coalition-proof Nash equilibria i. concepts. *Journal of Economic Theory*, 42(1):1 – 12, 1987.

[6] Patricia Bouyer, Romain Brenguier, and Nicolas Markey. Nash equilibria for reachability objectives in multi-player timed games. In *Proceedings of the 21st International Conference on Concurrency Theory (CONCUR'10)*, volume 6269 of *Lecture Notes in Computer Science*, pages 192–206. Springer, 2010.

[7] Patricia Bouyer, Romain Brenguier, and Nicolas Markey. Nash equilibria for reachability objectives in multi-player timed games. Research Report LSV-10-12, ENS Cachan, France, 2010.

[8] Patricia Bouyer, Romain Brenguier, Nicolas Markey, and Michael Ummels. Nash equilibria in concurrent games with Büchi objectives. In *Proceedings of the 31st Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'11)*, volume 13 of *Leibniz International Proceedings in Informatics*, pages 375–386. Leibniz-Zentrum für Informatik, 2011.

[9] Romain Brenguier. *Nash Equilibria in Concurrent Games - Application to Timed Games.* PhD thesis, Laboratoire Spécification et Vérification, ENS Cachan, France, 2012.

[10] Krishnendu Chatterjee, Rupak Majumdar, and Marcin Jurdziski. On Nash equilibria in stochastic games. In *Proceedings of the 13th EACSL Annual Conference on Computer Science Logic (CSL'04)*, volume 3210 of *Lecture Notes in Computer Science*, pages 26–40. Springer, 2004.

[11] Erich Grädel and Michael Ummels. Solution concepts and algorithms for infinite multiplayer games. In *New Perspectives on Games and Interaction*, volume 4 of *Texts in Logic and Games*, pages 151–178. Amsterdam University Press, 2008.

[12] Joseph Y. Halpern. Beyond Nash equilibrium: Solution concepts for the 21st century. In *Proceedings of the 27th ACM Symposium on Principles of Distributed Computing (PODC'08)*, pages 1–10. ACM, 2008.

[13] Joseph Y. Halpern. Beyond Nash equilibrium: Solution concepts for the 21st century. In *Lectures in Game Theory for Computer Scientists*, pages 264–289. Cambridge University Press, 2011.

[14] Thomas A. Henzinger. Games in system design and verification. In *Proceedings of the 10th Conference on Theoretical Aspects of Rationality and Knowledge (TARK'05)*, pages 1–4. National University of Singapore, 2005.

[15] D. Moreno and J. Wooders. Coalition-proof equilibrium. *Games and Economic Behavior*, 17(1):80–112, 1996.

[16] John F. Nash. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences of the USA*, 36(1):48–49, 1950.