A COMPUTATIONAL STUDY OF STRATEGY SWITCHING IN LARGE GAMES

by

Soumya Paul
The Institute of Mathematical Sciences
Chennai – 600113

A thesis submitted to the board of studies of Mathematical Sciences in partial fulfilment of the requirements for the award of

Doctor of Philosophy

of

HOMI BHABHA NATIONAL INSTITUTE

Homi Bhabha National Institute

Recommendations of the Viva Voce Board

As members of the Viva Voce Board, we recommend that the dissertation prepared by Soumya Paul entitled "A COMPUTATIONAL STUDY OF STRATEGY SWITCHING IN LARGE GAMES" may be accepted as fulfilling the dissertation requirement for the Degree of Doctor of Philosophy.

_____ **Date :**

Chairman -

_____ **Date :**

Convener -

_____ **Date :**

Member 1 -

_____ **Date :**

Member 2 -

Final approval and acceptance of this dissertation is contingent upon the candidate's submission of the final copies of the dissertation to HBNI.

I hereby certify that I have read this dissertation prepared under my direction and recommend that it may be accepted as fulfilling the dissertation requirement.

_____ **Date :**

Guide -

Declaration

I hereby declare that the investigation presented in this thesis has been carried out by me. The work is original and has not been submitted earlier as a whole or in part of a degree/diploma at this or any other Institution/University.

SOUMYA

Abstract

The aim of this work is to study large games: games with a large number of players and/or with a large temporal, spatial or logical structure, using techniques from automata theory, logic and game theory. We argue that the traditional way of analysing games is not only unwieldy but also cannot be properly motivated for such games. As a lot of uncertainties are present in such a game, players usually do not strategise in a single go but do so dynamically as they observe the outcomes of the game. They compose simple strategies to build more and more complex ones. They also employ several heuristic strategies. Switching between strategies forms a central part in such compositions. We formally study the notion of strategy switching in games.

We start off by introducing strategy switching in a few classical games: extensive form, parity and Muller and explore the running time complexity of the algorithms that compute the winning strategy in these games. Switching strategies naturally comes with a cost and we explore repeated strategic form games where the players incur a cost on every strategy switch they make. We then give a logical foundation to strategy switching by players. We introduce a simple logic in which the composition of strategies by players can be described and formally studied. We study the eventual behaviour of games where the strategies of the players are specified in this logic. Dynamic strategising by players may lead to some actions being sparingly played by them. These actions then become too expensive to maintain in the arena and hence get removed. This in turn forces the players to change their strategies. The process is mutually recursive and leads to interesting dynamic phenomena. We study games which change dynamically and intrinsically as the play progresses and show that the eventual behaviour can be decided.

We then study imitation as a heuristic strategy in games. We look at games where the players are divided into 'imitators' and 'optimisers'. The optimisers play optimal strategies whereas the imitators imitate one or more optimisers. We show that we can compute how worse-off the imitators are by playing imitative stratagies instead of optimising. Finally, we look

at games where the players are arranged in neighbourhoods. The players play strategic form games within their own neighbourhoods but can view the outcomes of certain other neighbourhoods and strategise accordingly. Players are also allowed to switch neighbourhoods between rounds. We show that when the strategies of the players are specified in a simple logic, which we introduce, we can predict the eventual outcome of such games. Moreover, when the strategies of the players are unknown, these games can be characterised in terms of potentials.

Acknowledgements

First and foremost I thank my advisor, Jam. From the time I joined the institute and was scared of even facing him, till the time we became friends, he has always been a guide in the truest sense of the term. I do not remember ever having any intense technical discussion with him; he always gave me a lot of ideas - the big picture. Everytime I would come back after meeting him, I would think back puzzled at what we actually talked about and if I understood anything at all of what he said. But somehow, results were obtained and papers were indeed written. My Ph.D years seemed to pass by like a breeze. He never pressurised me about anything; he was always calm and composed. I consider myself lucky to have had an advisor like him and shall always be indebted to him.

Secondly, I would like to thank my colleague and co-author of some of my papers, Sunil. He is a guy with a tremendous amount of patience and dedication. During our long discussions I could notice that he was always able to grasp any question I would ask him, however badly framed by me and even when I myself didn't have any idea of what I was asking!

I thank my mom for being so patient with me and always having her faith in me. She never asked me any question regarding my career or academics. She has always given me full freedom to pursue whatever I've wanted to pursue. The only question she asked and still asks is, "When will you come home?" I thank my other family members including my sister and my brother-in-law for always being a source of encouragement.

I thank my friends in MatScience: Sundar, for never refusing me a game of tennis, Alok, for always wanting to go out for a drink, Ajay, for making me jog every evening, Madhushree, for buying me stuff from CeeDeeYes, Yadu, for being a kind office-mate and keeping the bottle of water always full... and ofcourse Rahul, for being a constant source of entertainment.

I am indebted to my other friends: Abu for always encouraging me and lending me an ear whenever I needed one, Ritu for doing the same, Sanghamitra for thinking I am a genius, how much ever I would try to convince her otherwise, and never accepting anything less than perfect from

my part and Maina for rebuking me whenever I seemed like digressing. I thank Aparna and Saswati for being what they are.

Finally, I thank my Institute for providing me the opportunity and the atmosphere to carry out my work.

I remember dad who was ever so generous, ever so encouraging and ever so eager to see us receive the best of education, to see us happy and successful and for loving us more than his life. I wish he could see this day.

SOUMYA

.

To dad...

Contents

List of Figures

Chapter 1

Games, logic and automata

Game Theory is the study of situations where people with conflicting or collective interests make decisions to achieve certain outcomes. A game is traditionally a collection of strategies (how to play) for the players of the game with each player having preferences over various outcomes that are generated by tuples of strategies. One of the main aims of game theoretic analysis is to predict what outcomes would result if people play according to certain 'rationality' assumptions. In other words, given a game situation, we would like to predict 'equilibrium' or 'stable' play.

One of the many definitions of equilibrium play requires that players should not have an incentive to deviate from such a play. Towards this end, John Nash [Nas50] in his path breaking work showed that every finite game has an equilibrium, which is now known as the Nash equilibrium, from which no player has an incentive to unilaterally deviate. An equilibrium may be viewed as a 'solution concept' of a game.

Many attempts have been made at providing logical foundations to the various equilibria and other solution concepts. The fact that every finite game has a Nash equilibrium is quite encouraging. But why should players play for such an equilibrium? What is the reasoning required on their part to play a Nash equilibrium? Aumann and Dreze [AD05] point out that game theory started out by trying to develop a prescriptive theory for rational agents right from the seminal work of von Neumann and Morgenstern [vNM44] who envisaged game theory as constituting advice for players in game situations, so that strategies may be synthesised accordingly. Such a prescriptive theory must account for the beliefs and expectations each player has about the strategies of the other players.

John Harsanyi [Har77] in his book *Rational Behaviour and Bargaining*

Equilibrium in Games and Social Situations, developed a beautiful theory of 'rationality' and how 'rational' players actually behave. He argued that rational players play rationally and also believe that other players are rational. This means that they know that others know that they are rational, they know that others know that they know that the others are rational and so on. Such a hierarchy of knowledge in its infinite level is called the common knowledge of rationality. The existence of Nash equilibrium in a game in general requires the assumption that the players are rational and it is common knowledge that they are rational. Thus playing an equilibrium strategy requires players to reason based on how other players reason.

Such mutually recursive reasoning by the players can be justified when the games are small, in that, there are a small number of players or the structure of the games is small. However such justification rapidly breaks down when the games become larger and larger. We call a game large if it has one or both of the following features:

- A large number of players.
- A large temporal or spatial structure. That is, the game continues for a long duration, possibly unbounded, or the players are scattered across a large geographical or logical expanse. An example of a game satisfying the latter criterion is when A learns about X from her neighbour B who in turn learns about her from her own neighbour C and so on. Thus although A may not be geographically distant from X, she is logically so.

In such large games, various forms of uncertainties arise: players in one part of the game may be uncertain about the outcomes in some other part of the game, they may be uncertain about the strategies of the other players, even the number of players playing the game. We claim that such uncertainty can arise even in perfect information games of the classical sense. In such a situation, the traditional style of equilibrium analysis of games lack proper motivation. If the players do not even know how many others or who they are playing against, how are they to reason mutually recursively to attain equilibrium play?

Uncertainty in games has been handled traditionally with the concept of information sets. An information set of a player is a collection of game histories between which a player cannot distinguish with her observations so far in the game. Information sets model very well situations where players do not have complete information about the other players' moves or strategies. However they, in general, do not deal with uncertainty of the kind that arise

in large games as described above. The definition of information sets can no doubt be tweaked to take into account such uncertainties. For example, if a player A is uncertain about the number of other players playing the game, at every position, her information set would contain elements that correspond to there being 2, 3, ... players in the game. But then, such a model, among others, has the immediate drawback of being infinite. Hence, we feel that modelling such games using the notion information sets is unwieldy and problematic.

Another important aspect to consider in real-life game-playing situations is that the players have computational limitations (resources, memory etc.). They cannot in general carry out the complicated reasoning to play an equilibrium strategy. Moreover, the larger a game is, more is the complication of the equilibrium strategies. Hence, such a player does not usually strategise for the entire game right at the beginning, but does so incrementally. After playing the game for a while, she observes the strategies that the other players employed and the outcomes they received in the process. Depending on this and her own outcomes, the player might then switch to a different strategy in anticipation of countering the other players' strategies and also to attain better outcomes. The other players, on their part, of course observe this fact and hence they may themselves switch their own strategies. Hence strategical reasoning by players in games is naturally mutually recursive which results in their switching between 'less complicated' (atomic) strategies to build (usually) 'more complicated' strategies. Thus switching strategies constitutes a central aspect of strategic reasoning in games in general and in large games in particular.

Strategies are traditionally defined to be functions from the set of histories of a game to the set of possible actions. In this sense, since combining strategies by switching between them again results in such a function, the resulting object is again a strategy. But this new strategy, instead of being any arbitrary function, has well defined structure. Moreover, when the rules of composition are given in some natural logical form, the structure of these combined strategies is regular. Hence the long term effects of such strategies on the game can be analysed and their outcomes predicted.

In this thesis, we start out by introducing the notion of strategy-switches in traditional games: finite extensive form games, games on graphs as well as repeated strategic form games. We observe that there are many interesting questions one can ask about these games when the number of strategies used to play the games and/or the number of switches between these strategies is considered as a resource.

We first study the traditional two-player finite extensive form games as

well as games on graphs, with parity and Muller conditions where the strategies of players are restricted to a finite subset S of the set of all strategies. We ask the following questions:

- Given a finite set of strategies S, is it possible for a player to play a winning strategy by just switching between the strategies in S?
- If so what is the minimum number of strategies / strategy-switches required from the set S?

We give algorithms for these questions and analyse their running time complexity.

Such questions are relevant in situations when playing a strategy involves a cost, maybe the cost of setting up the infrastructure required to play the strategy. We look at repeated strategic form games where the players incur a cost if they switch their strategies between two successive rounds. Under such a situation, would a player switch her strategy? Yes, if switching her strategy fetches her enough dividends in the long run to nullify these costs. However, a player who has been playing a certain strategy for a long time may settle down to a kind of an inertia or an unwillingness to switch her strategy even though she knows that doing so might fetch her better outcomes. For instance, although I know that changing my present house (with the leaking faucet and the creaking doors) to a different (newer) one would be far more comfortable for me, the very thought of shifting all the luggage and the furniture and disrupting my day-to-day schedule deters me from doing so. Games where switching strategies involve such costs have been studied in the literature (see for instance [LW97, LW09, Cha90] and the references therein). We re-prove some of the results in our setting to gain insight into what analytical changes these costs bring about.

We then turn our attention to large games. In this setting, as we observed already, there are various departures from the traditional setting of games: players may be uncertain about the number of players playing the game, the outcomes of the different rounds of the game and so on. In such games, it is natural to consider players with a bounded amount of resources: memory, computational power, knowhow, expertise, experience etc. Moreover, in such games the traditional way of studying equilibria can not only be unwieldy but also cannot be properly motivated. Hence, our analysis of such games is different from the way games have been traditionally analysed; we look at such games from an orthogonal point of view. More precisely, instead of studying the existence of winning/optimal strategies, we study

games where the strategies of the players are pre-specified (logically, algorithmically or by finite state automata). The question we ask of our model is the classical question of game theory: the prediction of stable behaviour in the limit. But what do we mean by stability in the context we consider?

In the case of players with bounded resources who are uncertain about the game, it is the short term changes that are under the players' control. They switch between different strategies based on the outcomes of the play they observe. We are interested in questions concerning the eventual dynamics of the game:

- Which strategies survive eventually?
- What is the eventual outcome?
- Is stability with respect to switching attained?
- How worse-off are players employing heuristic strategies rather than playing best responses?

We study the above questions in a model where the strategies of the players have a compositional structure and are specified in terms of a simple syntax modelled on temporal and dynamic logics. We look at strategy-switching both from a logical and an algorithmic perspective.

In social situations, a natural consequence of switching strategies by players is that the game form itself changes intrinsically. The actions that are played by a small number of players may become too costly for the society to sustain and hence may cease to be available. This results in a change in the game form and we are interested in predicting which game forms finally emerge and remain stable given that the players play according to their strategy specifications and the society restricts the actions based on certain rules.

In this context, the converse question is also quite relevant and interesting: which actions of the players should the society restrict and how should it restrict them so that certain social goals are eventually achieved while minimising the cost? We address this question both in the case when the players are maximisers and when they play according to heuristics and show that in both cases it is enough for the society to keep track of a finite amount of information to generate the required action-restriction rules.

Our focus then shifts to the study of the rationale behind the strategy-switching by the players. As we already mentioned, in large games, resource-bounded players do not strategise for the entire game but do so dynamically revising their strategies based on certain heuristics. An important heuristic

to consider is that of imitation. Players tend to imitate other players with better expertise and knowhow and of whom they know have performed well in the game so far. As Dutta and Prasad [DP04] put it, "It is human to imitate," it is good for resource-bounded players to imitate as well! In this context we study games where there are two types of players, 'imitators' and 'optimisers', and where each type is specified by a finite automaton. We study the eventual outcomes in such games and show that one can give a reasonable answer as to how worse-off the players are playing imitative strategies rather than best responses. Imitation in games has been widely studied, for instance in [Ban92, EF95, Sch98, LP07, DP04]. We, however, take a different (automata theoretic) approach to analyse the effect of imitation in large games.

For games involving a large number of players, we consider a model where the players are arranged in certain neighbourhood structures. Such a structure is given in the form of a (finite) graph where the vertices of the graph represent the players and the edges represent the visibility relation of the players. Similar models have been considered in the literature by various authors. Young [PY00, PY93] considers models where the interaction structure of the players is represented by a finite undirected graph and studies how innovations spread through society by observation and interaction. Kearns et al [KLS01a, KLS01b, EGG06, EGG07] analyse games where the payoff of players depend only on her own action and the actions of her adjacent players as given by the graph structure. They study the computational aspects and show the existence of Nash equilibria in such games.

In our setting a neighbourhood is a maximal clique in the neighbourhood graph. Although the payoffs of the players are dependent only on the actions of the other players in the same neighbourhood, their strategising might depend on the actions and outcomes of all the players within their visibility range, and in particular players outside their own neighbourhood. Such a setting automatically provides a player with a rationale for switching strategies. If she can observe a player in a different neighbourhood who is performing better by playing a different strategy, she might switch to this strategy and apply it in her own neighbourhood in the hope of doing better. A player might even quit her own neighbourhood and join another one expecting her strategy to fetch her better dividends in this new neighbourhood.

In this context we study games where the players play simple imitative strategies and look at which actions and neighbourhood structures eventually arise and remain stable. We also characterise stable structures by relating them to potential games a la Monderer and Shapley [MS96]. Imitation

dynamics in games have been studied, for instance, in [ARV06, AFBH08], where they study asymptotic time complexities of the convergence or non-convergence to Nash equilibrium in congestion games when the players play imitative strategies. However, their models are probabilistic and lack the neighbourhood structure that we consider.

In statistical physics, when dealing with large systems one uses averages or other aggregate measures and uses such measures to reason about the systems. In our case we work with anonymous utilities when the number of players are large. Such games have been called 'anonymous games' in the literature [Blo99, Blo00, DP07, BFH09]: anonymous because the utility of a player playing a certain strategy depends only on the number of other players playing the same strategy rather than the identity of the players. Such utilities can be thought of as a kind of an aggregate measure providing us with a handle on the system and simplifying the analysis.

When the number of players is large, the analysis is also simplified if we can obtain results on a smaller 'derived' system and then lift these results to the whole system. The concept of 'types' comes to our rescue in this regard. In large games, albeit there are a large number of players, but the number of different ways these players play is not so many. Each such way of play is called a 'type' and a player playing in such a manner is said to be of that type. For example, an 'optimiser' is a type and so also is an 'imitator' and so on. To this end, we show how and when the analyses of games with a large number of players can be carried out with a comparatively smaller number of player 'types' and still the results obtained can be applied to the entire game.

1.1 Organisation of the thesis

We present the thesis in two parts. Though each part can be considered more or less independent of the other, the central theme in both the parts is strategy switching by players. In Part-I we introduce strategy switching in games and in Part-II we mainly look at the structure of such strategy switches. We heavily use techniques from automata theory in both these parts and it can in a way be seen as bridging the gap between the two.

Part-I: Introducing strategy switching in games

Part-I consists of chapters 2, 3 and 4. In Chapter 2 we consider what happens when in a game the players are restricted to use strategies only from a fixed finite set S. We are interested in knowing if it is possible for a player to play

winning / optimal strategies just by switching between strategies of S and if so how. We look at finite extensive form games, parity and Muller games in this context and study the computational complexity of these questions.

In Chapter 3 we study the case when switching strategies comes with a cost. We look at infinitely repeated strategic form games. A player incurs a fixed cost c if she switches her strategy from any round t to round $t+1$. Such games with switching costs have been studied in the literature. We re-prove some of the results for the sake of completeness.

In Chapter 4 we look at the eventual dynamics of concurrent-move games when the strategies of the players are explicitly specified. We introduce a logical syntax for the specification of the strategies of the players for this purpose. After that, we introduce the notion of probabilistic switching, which naturally generates mixed strategies. We again analyse the eventual outcome of the game when the players switch strategies probabilistically.

Part-II: Structure of strategy switching

Part-II consists of chapters 5, 6 and 7. In Chapter 5 we look at games that change intrinsically based on the actions / strategies played by the players. There is an implicit player - the society, who maintains the available actions of the players and incurs certain costs in making them available. If and when it feels that an action a is being played by a small number of players and/or it becomes too expensive for it to maintain the action a, it removes a from the set of available actions. This results in a change in the game and the players have to strategise afresh taking this change into account. The restrictions of the society are again specified using a logical syntax. We are interested in which game forms eventually arise and remain stable thereafter.

We then study the converse question: which actions of the players should the society restrict and how should it restrict them so that the social cost is minimised in the eventuality? We address this question both in the case when the players are maximisers and when they play according to strategy specifications.

In Chapter 6 we look at imitation as a heuristic strategy for players. We consider n-player games where some of the players are optimisers (play best response strategies) and the rest are imitators. The players have preferences over the strongly connected components in the arena. We justify and consider the case where the strategies of both the imitators and the optimisers are given in terms of finite automata. We analyse how worse-off the players are playing imitative strategies rather than best-responses.

In Chapter 7 we look at games where the players are arranged in neighbourhoods. The neighbourhood structure is given by a graph G which we call the neighbourhood graph. The vertices of the graph correspond to the players and the cliques in G are the neighbourhoods and the edges in G correspond to the visibility structure of the players. Although the payoffs of the players are affected only by the moves of the players in her own neighbourhoods, their strategising can depend on what they can view of their own and also other neighbourhoods. We consider two cases, one where the players stick to their own neighbourhoods throughout the course of the game and the other where players can also switch neighbourhoods. We study games with both of the above neighbourhood structures where the players play simple imitative strategies. We also study general games with such neighbourhood structures.

Finally, in Chapter 8 we conclude with future directions and possible extensions.

1.2 Background

In this section we develop the necessary preliminaries for the rest of the thesis.

1.2.1 Strategic form game

A strategic form or normal form game consists of a set $N = \{1, 2, \ldots, n\}$ of players and a collection of moves (strategies) A_i for each player i. The game is played in a single round in which each player i chooses a move a_i from A_i. This is done simultaneously and a player i does not know the move chosen by any other player j before she makes her choice. This defines a tuple $\mathbf{a} = (a_1, a_2, \ldots, a_n) \in \mathbf{A} = \prod_{i \in N} A_i$. Each player i has a preference $\sqsubseteq_i$ over the various tuples in $\mathbf{A}$. When this preference is a total order, we can represent it in the form of a payoff function $p_i : \mathbf{A} \to \mathbb{N}$[1]. Thus a strategic form game between n players is an n-dimensional matrix where each dimension of the matrix corresponds to a player i, the indices of the dimension correspond to the moves available to the player i and the entries of the matrix correspond to the payoff tuples of the players with respect to the indices.

[1] As we consider only ordinal preferences as opposed to expected utilities, such a representation is without loss of generalisation

1.2.2 Extensive form game arena

Throughout this exposition, we shall investigate the long-run dynamics of games, viz., eventual behaviour of players, eventual outcomes etc. Local or small temporal perturbations should not affect the global outcome or the eventual dynamics of the games. Hence, in such a setting, it is natural to assume that the play is 'unbounded', or in other words continues for an infinite amount of time. This has the advantage that small perturbations and changes, 'losses' and 'profits', mistakes and risks etc. are amortised away and need not affect eventual outcomes very much. Thus, although we shall look at finite games, our focus shall mostly be on infinite duration games.

In the above spirit, we define an extensive form arena to be a (finite or infinite) tree $\mathcal{T} = (T, E)$ where T is the set of vertices and E is the set of edges. The root of the tree $t_0 \in \mathcal{T}$ is the initial vertex of the arena. We call the pair $(\mathcal{T}, t_0)$ an initialised arena. For a vertex $t \in T$, we let $E(t)$ denote the set of outgoing edges of t and tE denote the set of children of t. That is, $E(t) = \{(t, t') \in E\}$ and $tE = \{t' \mid (t, t') \in E\}$. $N = \{1, 2, \ldots, n\}$ is the set of players and A_i is the set of actions of player $i \in N$. Let $\mathbf{A} = \prod_{i \in N} A_i$ and $A = \bigcup_{i \in N} A_i$. For every player i, there is a function $\Gamma_i : T \to 2^{A_i}$ such that $\Gamma_i(t)$ gives the set of actions that are available to the player i at the vertex t.

The edges of the arena $\mathcal{T}$ are labelled with tuples from $\mathbf{A}$. For simplicity, we assume that for every vertex $t \in T$ and for every tuple of actions $\mathbf{a} \in \prod_{i \in N} \Gamma_i(t)$, there is an edge (t, t') that is labelled with $\mathbf{a}$. A game is played on this arena as follows. Initially a token is placed at the root t_0 of $\mathcal{T}$. Whenever the token is at some vertex t, every player i chooses an action a_i from her set of available actions $\Gamma_i(t)$. This defines a tuple $\mathbf{a} = (a_1, a_2, \ldots, a_n) \in \mathbf{A}$. The token then moves along the outgoing edge of t labelled with $\mathbf{a}$ to the corresponding child. This process defines a branch of the tree $\mathcal{T}$ which is called a play. We denote the set of plays by P. A finite play, which is a prefix of a branch of $\mathcal{T}$, is called a history. We denote the set of all histories by H.

Example 1.1 *Consider an extensive form arena where there are two players 1 and 2 and at each position player 1 has two actions $\{a, b\}$ and player 2 has two actions $\{c, d\}$. This arena looks like Figure 1.1. We have simply mentioned the choices that are available at each node and have avoided writing down the entire sequence so as not to clutter the figure. For instance, the leftmost grandchild of the root node is actually aa, cc and not a, c. However, this is clear from the context.*

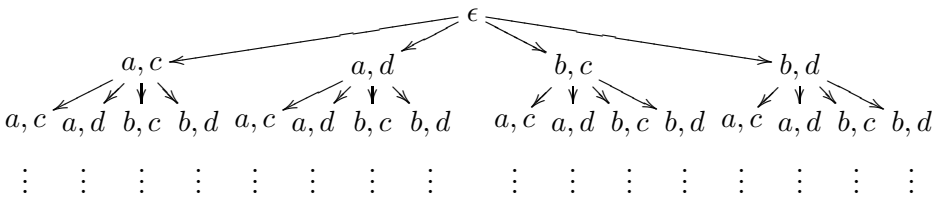

Figure 1.1: Extensive form game tree

Turn-based arena

A turn-based arena is a special case of the above arenas where the vertex set T is partitioned into $T = T_1 \cup T_2 \cup \ldots \cup T_n$ with the restriction that for any player i and any vertex $t \in T_i$, $\Gamma_j(t)$ is a singleton for all $j \neq i$. Thus, a turn-based arena is one where the players take turns in making moves in that at a vertex $t \in T_i$ it is the player i's choice that determines the next vertex; the other players do not have any control over it. We generally write a play $\rho = t_0 \overset{\mathbf{a}_1}{\to} t_1 \overset{\mathbf{a}_2}{\to} \ldots$ as $\rho = t_0 \overset{a_1}{\to} t_1 \overset{a_2}{\to} \ldots$ where $a_1 = \mathbf{a}_1(i)$ such that $t_0 \in T_i$, $a_2 = \mathbf{a}_2(j)$ such that $t_1 \in T_j$ and so on. Since at a vertex t, only one player has a non-trivial choice, such a representation is without loss of generality.

1.2.3 Graph arena

We wish to do algorithmic analysis on the extensive form arenas. For this we require that these arenas be presented to us in a finite fashion (when they are infinite). A natural way to present a game arena is by a set of rules, stating which moves are possible at what position and what will be the resulting position. See [KS10] for a treatment along these lines. We however take a more direct approach and let our extensive form arenas be unfoldings of finite directed graphs.

A graph arena is thus a directed graph $\mathcal{A} = (V, E)$ where V is the set of vertices and E is the set of edges. As in the case of extensive form arenas, for a vertex $v \in V$, we denote by vE the set of all the neighbours of v and by $E(v)$ the set of all outgoing edges of v. For simplicity we assume that $\mathcal{A}$ has no dead-ends, that is, $vE \neq \emptyset$ for all $v \in V$. As before, the set of players is $N = \{1, 2, \ldots, n\}$. An initialised arena $(\mathcal{A}, v_0)$ is one where an vertex $v_0 \in V$ is designated as an initial vertex. A subarena of $\mathcal{A}$ is a subgraph of $\mathcal{A}$ with no dead-ends.

Every player i has a finite set of actions A_i. Let $\mathbf{A} = \prod_{i \in N} A_i$ and $A = \bigcup_{i \in N} A_i$. For every player i, there is a function $\Gamma_i : V \to 2^{A_i}$ such that $\Gamma_i(v)$ gives the set of actions that are available to the player i at the vertex v. The edges of the arena $\mathcal{A}$ are labelled with tuples from $\mathbf{A}$. We once more assume that for every vertex $v \in V$ and for every tuple of actions $\mathbf{a} \in \prod_{i \in N} \Gamma_i(v)$, there is an edge (t, t') that is labelled with $\mathbf{a}$. A game is played on this arena as follows. Initially a token is placed at the initial vertex v_0 of V. Whenever the token is at some vertex v, every player i chooses an action a_i from her set of available actions $\Gamma_i(v)$. This defines a tuple $\mathbf{a} = (a_1, a_2, \ldots, a_n) \in \mathbf{A}$. The token then moves along the outgoing edge of v labelled with $\mathbf{a}$ to the corresponding neighbour. This process defines a path in $\mathcal{A}$ which is called a play. We denote the set of plays by P. A finite play is called a history. We denote the set of all histories by H.

Since we have assumed that our arenas have no dead-ends, all plays in such an arena are always infinite.

Turn-based arena

A turn-based arena is again a special case of the above arenas where the vertex set V is partitioned into $V = V_1 \cup V_2 \cup \ldots \cup V_n$ with the restriction that for any player i and any vertex $v \in V_i$, $\Gamma_j(v)$ is a singleton for all $j \neq i$. Once more, without loss of generality, we write a play $\rho = v_0 \xrightarrow{\mathbf{a}_1} v_1 \xrightarrow{\mathbf{a}_2} \ldots$ as $\rho = v_0 \xrightarrow{a_1} v_1 \xrightarrow{a_2} \ldots$ where $a_1 = \mathbf{a}_1(i)$ such that $v_0 \in V_i$, $a_2 = \mathbf{a}_2(j)$ such that $v_1 \in V_j$ and so on.

We would like to talk about various properties of a game position in logical terms. For this purpose we have a set of propositions $\mathcal{P}$ and a valuation function

$$val : V \to 2^{\mathcal{P}}$$

which gives the truth of these propositions at the vertices of the arena. These propositions can stand for facts of the form "player i received the highest payoff in round t", "player i played action a in round t" etc. Intuitively, these propositions are observations about the game based on which the players strategise.

1.2.4 Some notations

For a directed graph $G = (V, E)$ where the edges are labelled with elements from a set X, if (v, v') is an edge in E labelled with x then we often denote it by $v \xrightarrow{x} v'$. For a finite path $\rho = v_0 \xrightarrow{x_1} v_1 \xrightarrow{x_2} \ldots \xrightarrow{x_k} v_k$ in G we let $last(\rho) = v_k$. For a finite or infinite path $\rho = v_0 \xrightarrow{x_1} v_1 \xrightarrow{x_2} \ldots$, we let

$u(\rho) = x_1x_2\ldots$. For a finite sequence $u = x_1x_2\ldots x_k \in X^*$ and given a vertex $v \in V$, we let $\rho(v,u)$ be the path $v \xrightarrow{x_1} v_1 \xrightarrow{x_2} v_2 \xrightarrow{x_3} \ldots \xrightarrow{x_k} v_k$ in G. For a set X and for a sequence $u = x_0x_1\ldots x_k \in X^*$, we similarly let $last(u) = x_k$. For a sequence $u = x_1x_2\ldots \in X^*$, we let $u(i)$ be the ith element of u, u_i be the length i prefix of u, and u^i be the postfix of u starting at the ith element, that is, $u^i = x_ix_{i+1}\ldots$, where $i < |u|$, the length of the sequence u. For $x \in X$ and a sequence $u \in X^*$, we let $|u|_x$ be the number of occurrence's of x in u. We let $occ(u)$ be the set of elements that occur at least once in u, that is, $occ(u) = \{x \mid \exists i,\ u(i) = x\}$. The above notations generalise naturally to the set of infinite sequences over X, X^ω. For $u \in X^\omega$, we let $inf(u)$ be the set of elements that occur infinitely often in u, that is, $inf(u) = \{x \mid \forall i \exists j > i,\ u(j) = x\}$. For a tuple $\mathbf{a} \in \mathbf{A}$, we let $\mathbf{a}(i)$ be the ith component of $\mathbf{a}$.

1.2.5 Unfolding

The tree-unfolding or just the unfolding of an initialised graph arena $(\mathcal{A}, v_0)$ where $\mathcal{A} = (V, E)$ is a tree $\mathcal{T}_\mathcal{A} = (T, E)$ such that $T = V \times A^*$ and is defined inductively as follows:

- $t_0 = (v_0, \epsilon) \in T$ is the root of $\mathcal{T}_\mathcal{A}$.
- For every $t \in T$, the children of t are derived as follows. If $t = (v, \mathbf{u})$ then for every $\mathbf{a} \in \mathbf{A}$, $(v', \mathbf{ua})$ is a child of t where $v' \in vE$ and the edge (v, v') is labelled with $\mathbf{a}$ in $\mathcal{A}$.

$\mathcal{T}_\mathcal{A}$ is the extensive form game arena corresponding to the graph arena $\mathcal{A}$. The valuation of the propositions $\mathcal{P}$ on V is lifted naturally to T as: for every $t \in T$, $val(t) = val(v)$ where $t = (v, \mathbf{u})$. For a path $\rho = t_1 \xrightarrow{\mathbf{a}_1} t_2 \xrightarrow{\mathbf{a}_2} \ldots \xrightarrow{\mathbf{a}_{k-1}} t_k = (v_1, u_1) \xrightarrow{\mathbf{a}_1} (v_2, u_2) \xrightarrow{\mathbf{a}_2} \ldots \xrightarrow{\mathbf{a}_{k-1}} (v_k, u_k)$ in the unfolding $\mathcal{T}_\mathcal{A}$, we let $\pi(\rho)$ denote the projection of ρ to its first component, that is, $\pi(\rho) = v_1 \xrightarrow{\mathbf{a}_1} v_2 \xrightarrow{\mathbf{a}_2} \ldots \xrightarrow{\mathbf{a}_{k-1}} v_k$.

Example 1.2 *Let $N = \{1, 2\}$ be the players, $A_1 = \{a, b\}$ be the action set of player 1 and $A_2 = \{c, d\}$ be the action set of player 2. Consider the arena $\mathcal{A}$ shown in Figure 1.2 consisting of the two positions v_0 and v_1 where v_0 is the initial position. The tree unfolding of the above arena is the tree depicted in Example 1.1.*

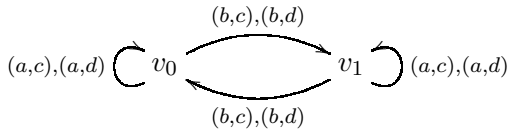

Figure 1.2: A game arena presented as a finite graph

1.2.6 Winning condition

An n-player game typically consists of an arena $\mathcal{A}$ and winning conditions $\phi_1, \phi_2, \ldots, \phi_n$. The winning conditions are subsets of the set of all plays in the arena, that is, $\phi_i \subseteq P$ for $1 \leq i \leq n$. The objective of every player i is to play so that the play is in her winning set ϕ_i. Such games are also called win-lose games because of the obvious reason that a player can either win or lose. However, note that the winning sets may not be disjoint, that is, it may be that $\phi_i \cap \phi_j \neq \emptyset$ for some $i \neq j$. If the winning sets form a partition of the set of plays P then the game is called zero-sum. Two-player zero-sum games have a single winning set ϕ with the convention that player 1 wins a play ρ in the arena $\mathcal{A}$ if and only if $\rho \in \phi$. Otherwise it is losing for her and winning for player 2. Various types of winning conditions have been studied in the literature for the case of two-player zero-sum games.

Borel condition: ϕ is a Borel set in the Cantor topology on the set of plays in the arena $\mathcal{A}$.

ω-regular condition: this is a general sub-class of the Borel winning condition and can be presented in different ways. ω-regular winning conditions naturally arise as sets specified by various specification languages. Three commonly used ω-regular conditions can be described as follows:

Reachability condition: Let $R \subseteq V,\ R \neq \emptyset$ be the reachability set. A play ρ in the arena $\mathcal{A}$ is said to be winning, $\rho \in \phi$, if and only if

$$\exists i \geq 0,\ \rho(i) \in R.$$

Let $C \subsetneq \mathbb{N}$ be a finite set of colours also called priorities. Let χ be a function that assigns a unique priority to each of the vertices in V:

$$\chi : V \to C$$

χ is lifted to sequences in V^* or V^ω as: for $u = v_1 v_2 \ldots$, $\chi(u) = \chi(v_1)\chi(v_2)\ldots$. χ can also be lifted to plays in $\mathcal{A}$ as: let $\rho = v_0 \xrightarrow{a_1} v_1 \xrightarrow{a_2} v_2 \xrightarrow{a_3} \ldots$ be a play in $\mathcal{A}$. Then $\chi(\rho) = \chi(v_0 v_1 v_2 \ldots)$.

Muller condition: Let $\mathcal{F} \subseteq 2^C$ be a family of subsets of C called the Muller sets. A play ρ in the arena $\mathcal{A}$ is said to be winning if and only if

$$inf(\chi(\rho)) \in \mathcal{F}.$$

Parity condition: A play ρ in the arena $\mathcal{A}$ is said to be winning, $\rho \in \phi$, if and only if

$$\min\{inf(\chi(\rho))\} \text{ is even}$$

Mean-payoff condition: is also a sub-class of the Borel condition. Here the vertices of the arena are labelled with weights or rewards. That is, there is a function $r : V \to \mathbb{Q}$ which associates a rational number with every vertex of the arena. A play ρ is winning, $\rho \in \phi$, if and only if the following quantity

$$\liminf_{k\to\infty} \frac{1}{k} \sum_{i=0}^{k} r(\rho(i))$$

is greater than a certain threshold, usually 0.

Another variation of n player games studied in the literature is where every player i, instead of having a winning set ϕ_i, has a preference relation $\sqsubseteq_i$ over the various subsets of C (the Muller sets). One of the aims of the players is to play in such a way that no player has an incentive to unilaterally deviate from such a play ρ. Player i then receives a payoff that is proportional to her preference of the play ρ in terms of her preference relation $\sqsubseteq_i$. Such a plan is called a Nash equilibrium and is defined in the next section.

1.2.7 Strategy

A strategy of a player tells her how to play the game. Formally, a strategy s_i of player i is a function

$$s_i : H \to A_i$$

A play ρ in the arena $\mathcal{A}$ is said to conform to a strategy s_i if for all $k \geq 0$ and all length k and $k+1$ prefixes ρ_k and ρ_{k+1} respectively of ρ, $last(\rho_k) \xrightarrow{\mathbf{a}} last(\rho_{k+1})$ implies $\mathbf{a}(i) = s_i(\rho_k)$.

Strategies can also be randomised. In that case, the strategy of a player at a particular history does not prescribe her one action but a probability distribution over the set of available actions. Such strategies are called mixed strategies. Formally, let $\Delta(A_i)$ be the set of probability distributions over the set A_i of actions of player i. Then, a mixed strategy s_i of player i is a function

$$s_i : H \to \Delta(A_i)$$

A non-randomised strategy s_i of a player i is then a special case of a mixed strategy where for every history ρ, $s_i(\rho)$ has positive support on exactly one action of A_i. These strategies are also called pure strategies. Henceforth, we shall use the word 'strategy' to denote pure strategies and when we want to talk about mixed strategies, we shall say so explicitly.

Apart from the above definitions of a strategy, we shall also be interested in partial strategies where a strategy of any player i is a partial function from the set of histories to the set of actions A_i of player i. We adopt the convention that if a partial strategy s_i of a player i is not defined at a particular history ρ then she may play any action there. Thus a play ρ in the arena $\mathcal{A}$ is said to conform to strategy s_i if for all $k \geq 0$ and all length k and $k+1$ prefixes ρ_k and ρ_{k+1} respectively of ρ, $last(\rho_k) \xrightarrow{\mathbf{a}} last(\rho_{k+1})$ and s_i is defined on ρ_k implies $\mathbf{a}(i) = s_i(\rho_k)$.

For every player $i \in N$ we let Σ_i denote the set of all strategies of i. A strategy profile $(s_1, \ldots, s_n)$ is a subset of $\Sigma_1 \times \ldots \times \Sigma_n$.

Strategy tree

A strategy s_i of a player i in an arena $\mathcal{A}$ (turn-based or concurrent) can be viewed as a subtree of the extensive form arena $\mathcal{T}_\mathcal{A}$ corresponding to $\mathcal{A}$. We call this subtree the strategy tree of s_i and denote it as $\mathcal{T}_\mathcal{A}^{s_i}$. $\mathcal{T}_\mathcal{A}^{s_i}$ is obtained from $\mathcal{T}_\mathcal{A}$ by retaining only the plays that conform to s_i. Formally $\mathcal{T}_\mathcal{A}^{s_i} = (T^{s_i}, E^{s_i})$ such that

- $t_0 = (v_0, \epsilon) \in T^{s_i}$ is the root of $\mathcal{T}_\mathcal{A}^{s_i}$.
- For every vertex $t = (v, \mathbf{u})$ in T^{s_i}, let ρ be the path from t_0 to t and let $\pi(\rho)$ be the projection of this path to the first component. If $s_i(\pi(\rho)) = a$ then the children of t are all t' such that $t' = (v', \mathbf{ua})$ where $v \xrightarrow{\mathbf{a}} v'$ in $\mathcal{A}$ and $\mathbf{a}(i) = a$.
- Nothing else is in $\mathcal{T}_\mathcal{A}^{s_i}$.

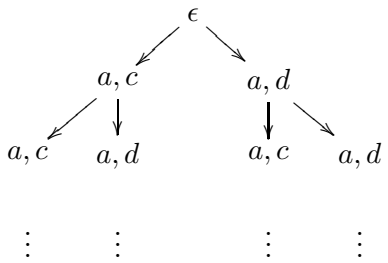

Figure 1.3: A strategy tree for player 1

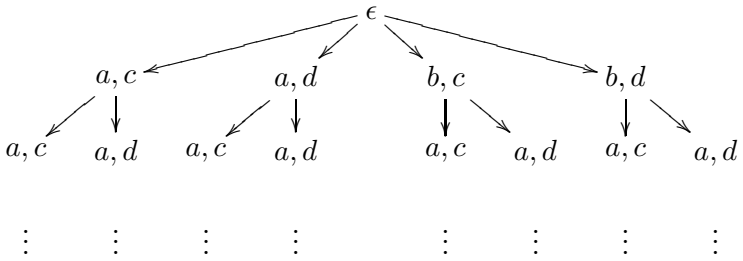

Figure 1.4: A partial strategy tree for player 1

Example 1.3 *In the extensive form game of Example 1.1, let s_1 be the strategy of player* 1 *which prescribes her to play the action a at every history. Then the strategy tree $\mathcal{T}_{\mathcal{A}}^{s_1}$ looks like Figure 1.3, which is a subtree of $\mathcal{T}_{\mathcal{A}}$.*

Example 1.4 *In the extensive form game of Example 1.1, let s_1' be the partial strategy for player* 1 *which is undefined at the empty history but prescribes her to play the action a for all successive histories. Then $\mathcal{T}_{\mathcal{A}}^{s_1'}$ looks as shown in Figure 1.4.*

Determinacy

Let $\mathcal{A} = (V, E)$ be a turn-based arena and let ϕ_i be the winning condition for player i. A strategy s_i for player i is called a winning strategy at a vertex $v \in V$ if for every play ρ that starts at v and conforms to s_i is winning for i, that is, we have $\rho \in \phi_i$. The biggest subset $W_i \subseteq V$ such that player i has a winning strategy at every vertex $v \in W_i$ is called the winning region for i. A two-player zero-sum turn-based game $(\mathcal{A}, \phi)$ (ϕ is the winning condition of player 0) is said to be (qualitatively) determined if for every vertex $v \in V$

it is the case that either player 0 or player 1 has a winning strategy at v. A classical problem in the theory of infinite two-player zero-sum games on graphical arenas is to determine given a game $(\mathcal{A}, \phi)$, if it is determined and if so to compute the winning regions and the winning strategies of both the players.

For concurrent games, such (pure-strategy) determinacy does not hold. For such games, it can be shown that randomised strategies are more powerful than pure strategies. Let $\mathcal{A} = (V, E)$ be a two-player zero-sum concurrent arena and let ϕ be the winning condition of player zero. Given a vertex $v \in V$, the maximum probability with which player 0 can ensure ϕ from v is called the value of the game at v for player 0 and is denoted $val_0(\phi)(v)$. Similarly the maximum probability with which player 1 can ensure $\bar{\phi} = P \setminus \phi$ from v is called the value of the game at v for player 1 and is denoted $val_1(\phi)(v)$. A concurrent two-player zero-sum game $(\mathcal{A}, \phi)$ is said to be (quantitatively) determined if for every vertex $v \in V$ it is the case that $val_0(\phi)(v) + val_1(\phi)(v) = 1$.

Equilibrium

In the non zero-sum setting, determinacy or winning regions cannot be defined. Various solution concepts have been explored in such a setting. Nash equilibrium is perhaps the most widely studied of these. Let $\mathcal{A} = (V, E)$ be an arena. Suppose the players have individual preferences over the various plays in this arena and suppose the preference relation of player i is denoted by $\sqsubseteq_i$. Let $\mathbf{s} = (s_1, s_2, \ldots, s_n)$ be a strategy profile of the players. We let $\mathbf{s}_{-i}$ be the profile $\mathbf{s}$ with the ith component removed and $(\mathbf{s}_{-i}, s)$ be the profile $\mathbf{s}$ except that the ith component is the strategy s. Given a strategy profile $\mathbf{s}$, we let $\rho(\mathbf{s})$ be the unique play in the arena conforming to $\mathbf{s}$.

A strategy s of player i is called a best response to a strategy profile $\mathbf{s}_{-i}$ of the other players if for every strategy s' of player i, $\rho(\mathbf{s}_{-i}, s') \sqsubseteq_i \rho(\mathbf{s}_{-i}, s)$. A strategy profile $\mathbf{s}$ is said to be a Nash equilibrium if for every i, s_i is a best response to s_{-i}.

Finite memory strategy

A strategy s of player i is said to be finite memory if it can be presented as a tuple (M, m^I, δ, g) where M is a finite set called the memory of the strategy, $m^I \in M$ is called the initial memory, $\delta : \mathbf{A} \times M \to M$ is called the memory update function, and $g : \mathbf{A} \times M \to A_i$ is called the output function such that if $\mathbf{a}_0\mathbf{a}_1 \ldots \mathbf{a}_k$ is a play and $m_0 m_1 \ldots m_{k+1}$ is a sequence determined by

$m_0 = m^I$ and $m_{j+1} = \delta(\mathbf{a}_j, m_j)$ then $s(\mathbf{a}_0\mathbf{a}_1 \dots \mathbf{a}_k) = g(\mathbf{a}_k, m_{k+1})$. The strategy s is said to be memoryless or positional if M is a singleton. Finite memory strategies can be modelled using finite state transducers.

1.2.8 Finite state transducer

A finite state transducer (FST) over input alphabet X and output alphabet Y is a tuple $\mathcal{Q} = (Q, I, \delta, f)$ where

- Q is the set of states,
- $I \subseteq Q$ is the set of initial states,
- $\delta : Q \times X \to 2^Q$ is the transition function and
- $f : Q \to Y$ is the output function.

A finite memory strategy, as defined above, can thus be naturally represented in terms of a finite state transducer. Let $s_i = (M, m^I, \delta, g)$ be a finite memory strategy for player i. s_i is represented by an FST $\mathcal{Q}_{s_i} = (Q, I, \delta, f)$ over input alphabet $\mathbf{A}$ and output alphabet A_i where the states Q of $\mathcal{Q}_{s_i}$ is equal to the memory M of the strategy s_i, the transition relation δ of $\mathcal{Q}_{s_i}$ is the same as the memory update function δ of s_i, the set of initial states I is equal to the singleton $\{m^I\}$ and the output function f is equal to g.

An FST $\mathcal{Q}_{s_i} = (Q, I, \delta, f)$ corresponding to a finite memory strategy s_i can be run on a subtree $\mathcal{T} = (T, E)$ of an extensive form game tree (unfolding). A run r of $\mathcal{Q}_{s_i}$ on $\mathcal{T}$ is a function $r : T \to Q$ that labels the vertices of $\mathcal{T}$ with states of $\mathcal{Q}_{s_i}$ and is defined inductively as:

- $r(t_0) \in I$.
- $r(t) = q$ and $t \xrightarrow{\mathbf{a}} t' \in \mathcal{T}$ implies $t' \in \delta(q, \mathbf{a})$.

A tree $\mathcal{T}$ is said to be accepted by $\mathcal{Q}_{s_i}$ if there exists a run r of $\mathcal{Q}_{s_i}$ on $\mathcal{T}$ such that for every $t \xrightarrow{\mathbf{a}} t'$, suppose $r(t) = q$, $t = (v, \mathbf{u})$ and suppose ρ is the path from t_0 to t. If $\pi(\rho)$ is the projection of this path to its first component, we have $\mathbf{a}(i) = s_i(\pi(\rho))$.

The language of $\mathcal{Q}_{s_i}$ is defined to be the set $\mathcal{L}(\mathcal{Q}_{s_i})$ of all the trees accepted by it. Note that the set $\mathcal{L}(\mathcal{Q}_{s_i})$ is the set of all strategy trees of the strategy s_i[2].

[2]This is a regular tree language in the parlance of automata theory

Product of transducers

Let $\mathcal{Q}_1 = (Q_1, I_1, \delta_1, f_1)$ and $\mathcal{Q}_2 = (Q_2, I_2, \delta_2, f_2)$ be two FSTs. The product of $\mathcal{Q}_1$ and $\mathcal{Q}_2$ is defined as $\mathcal{Q} = \mathcal{Q}_1 \times \mathcal{Q}_2 = (Q, I, \delta, f)$ where

- $Q = Q_1 \times Q_2$
- $I = I_1 \times I_2$
- $\delta = \{((q_1, q_2), \mathbf{a}, (q_1', q_2')) \mid (q_1, \mathbf{a}, q_1') \in \delta_1,\ (q_2, \mathbf{a}, q_2') \in \delta_2\}$
- $f : Q \to A_1$ or $f : Q \to A_2$ such that $f(q_1, q_2) = f_1(q_1)$ or $f(q_1, q_2) = f_2(q_2)$ depending on the operation at hand.

1.2.9 A modal logic

To reason about the properties of the various positions in an extensive form game, we develop a modal logic. The syntax of this logic is similar to temporal logic. However, its formulas are evaluated on nodes of extensive form game trees and hence they talk about a unique past and branching future. The syntax of this logic is formally given as:

$$\Phi ::= p \in \mathcal{P} \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \langle \mathbf{a} \rangle^- \varphi \mid \langle \mathbf{a} \rangle^+ \varphi \mid \diamondsuit \varphi$$

We have a couple of fragments of this logic. First, when we do not wish to talk about the future we have the logic Φ_- which is derived from Φ by dropping the modality $\langle \mathbf{a} \rangle^+ \varphi$. Similarly, when we do wish to talk about bounded future but not the past we have the logic Φ_+ which is derived from Φ by dropping $\langle \mathbf{a} \rangle^- \varphi$ and $\diamondsuit \varphi$.

We also use the standard abbreviations: $\boxminus \varphi \equiv \neg\diamondsuit\neg\varphi$, $\ominus \varphi \equiv \bigvee_{\mathbf{a} \in \mathbf{A}} \langle \mathbf{a} \rangle^- \varphi$ and $\bigcirc \varphi \equiv \bigvee_{\mathbf{a} \in \mathbf{A}} \langle \mathbf{a} \rangle^+ \varphi$.

A formula $\varphi \in \Phi$ is evaluated on the vertices of $\mathcal{T}_\mathcal{A}$. The truth of a formula φ at a vertex $t = (v, u) \in \mathcal{T}_\mathcal{A}$ is denoted by $t \models \varphi$ and is defined inductively as follows.

- $t \models p$ iff $p \in val(v)$.
- $t \models \neg\varphi$ iff $t \not\models \varphi$.
- $t \models \varphi_1 \vee \varphi_2$ iff $t \models \varphi_1$ or $t \models \varphi_2$.
- $t \models \langle \mathbf{a} \rangle^- \varphi$ iff there exists t' in $\mathcal{T}_\mathcal{A}$ such that $t' \xrightarrow{\mathbf{a}} t$ and $t' \models \varphi$.
- $t \models \langle \mathbf{a} \rangle^+ \varphi$ iff there exists t' in $\mathcal{T}_\mathcal{A}$ such that $t \xrightarrow{\mathbf{a}} t'$ and $t' \models \varphi$.
- $t \models \diamondsuit \varphi$ iff there exists an ancestor t' of t such that $t' \models \varphi$.

Fischer-Ladner closure, atom and atom graph

For a formula $\varphi \in \Phi$ we define its Fischer-Ladner closure $CL(\varphi)$ as follows. First we construct the subformula closure of φ, $CL'(\varphi)$ as:

- $\varphi \in CL'(\varphi)$.
- $\neg\varphi' \in CL'(\varphi)$ implies $\varphi' \in CL'(\varphi)$.
- $\varphi_1 \vee \varphi_2 \in CL'(\varphi)$ implies $\varphi_1 \in CL'(\varphi)$ and $\varphi_2 \in CL'(\varphi)$.
- $\langle \mathbf{a} \rangle^- \varphi' \in CL'(\varphi)$ implies $\varphi' \in CL'(\varphi)$.
- $\langle \mathbf{a} \rangle^+ \varphi' \in CL'(\varphi)$ implies $\varphi' \in CL'(\varphi)$.
- $\diamondsuit\varphi' \in CL'(\varphi)$ implies $\ominus\diamondsuit\varphi', \varphi' \in CL'(\varphi)$.

Finally let $CL(\varphi) = CL'(\varphi) \cup \{\neg\varphi' \mid \varphi' \in CL'(\varphi)\}$ where we identify $\neg\neg\varphi'$ with φ'.

A set $C \subseteq CL(\varphi)$ is called an atom (or a maximal propositionally consistent subset of $CL(\varphi)$) if:

- $\forall \varphi' \in CL(\varphi),\ \varphi' \in C$ iff $\neg\varphi' \notin C$.
- $\forall \varphi_1 \vee \varphi_2 \in CL(\varphi),\ \varphi_1 \vee \varphi_2 \in C$ iff ($\varphi_1 \in C$ or $\varphi_2 \in C$).
- $\forall \diamondsuit\varphi' \in CL(\varphi),\ \diamondsuit\varphi' \in C$ iff ($\varphi' \in C$ or $\ominus\diamondsuit\varphi' \in C$).

We denote by $AT(\varphi)$ the set of all atoms of φ. For any atom $C \in AT(\varphi)$ let $val(C) = C \cap \mathcal{P}$.

The atom graph $G_\varphi = (V_\varphi, E_\varphi)$ of a formula $\varphi \in \Phi$ is constructed as follows:

i. $V_\varphi = AT(\varphi)$.

ii. $E_\varphi \subseteq V_\varphi \times \mathbf{A} \times V_\varphi$ such that $(C, \mathbf{a}, C') \in E_\varphi$ if and only if for all $\langle \mathbf{a} \rangle^- \varphi' \in CL'(\varphi)$, $\langle \mathbf{a} \rangle^- \varphi' \in C' \Leftrightarrow \varphi' \in C$.

iii. $(C, \mathbf{a}, C') \in E_\varphi$ if and only if for all $\langle \mathbf{a} \rangle^+ \varphi' \in CL'(\varphi)$, $\langle \mathbf{a} \rangle^+ \varphi' \in C \Leftrightarrow \varphi' \in C'$.

The atom graph for a formula φ in the fragments Φ_- and Φ_+ can be suitably defined. A vertex C in the atom graph G_φ is (usually) designated an initial node only if it does not have any past requirement, that is, it does not have any formula of the form $\langle a \rangle^- \varphi'$ or $\langle \mathbf{a} \rangle^- \varphi'$. Hence note that for any $\diamondsuit\varphi' \in CL(\varphi)$, $\diamondsuit\varphi' \in C$ implies $\varphi' \in C$.

We call a subgraph $G'_\varphi = (V'_\varphi, E'_\varphi)$ of G_φ 'good' if for every $C \in V'_\varphi$, there exists $C' \in V'_\varphi$ such that C is reachable from C' and C' is initial.

Let $\mathcal{T}_{G'_\varphi}$ be the unfolding of a good subgraph G'_φ of the atom graph G_φ starting at an initial vertex C_0 of G'_φ. We have the following proposition

Proposition 1.5 *For any node $t = (C, u)$ of $\mathcal{T}_{G'_\varphi}$ and for every $\alpha \in CL(\varphi)$ we have $t \models \alpha$ iff $\alpha \in C$.*

Proof The proof is by induction on the structure of α.

- $\alpha = p \in \mathcal{P}$: Follows from definition since $t \models \alpha$ iff $p \in val(C)$ iff $p \in C$.
- $\alpha = \neg\alpha'$: $t \models \neg\alpha$ iff $t \not\models \alpha'$ iff $\alpha' \notin C$ iff $\neg\alpha' \in C$ [since C is an atom].
- $\alpha = \alpha_1 \vee \alpha_2$: $t \models \alpha$ iff $t \models \alpha_1$ or $t \models \alpha_2$ iff $\alpha_1 \in C$ or $\alpha_2 \in C$ iff $\alpha_1 \vee \alpha_2 \in C$ [since C is an atom].
- $\alpha = \langle \mathbf{a} \rangle^- \alpha'$: $t \models \langle \mathbf{a} \rangle^- \alpha'$ iff $t' \xrightarrow{\mathbf{a}} t \in \mathcal{T}_{G'_\varphi}$ and $t' \models \alpha'$ iff $\alpha' \in C'$ where $t' = (C', u')$ iff $\langle \mathbf{a} \rangle^- \alpha' \in C$ [by the construction of the atom graph].
- $\alpha = \langle \mathbf{a} \rangle^+ \alpha'$: Similar to the case for $\langle \mathbf{a} \rangle^- \alpha'$.
- $\alpha = \Diamond\alpha'$: $t \models \Diamond\alpha'$ implies there exists an ancestor t' of t such that $t' \models \alpha'$ Let $t' = (C', u')$ be the first such ancestor. We induct again on $|t| - |t'|$. $|t| - |t'| = 0$ implies $t \models \alpha''$ implies $\alpha' \in C$ by induction hypothesis which implies $\Diamond\alpha' \in C$ [since C is an atom]. $|t| - |t'| = k+1$ implies $t'' \models \Diamond\alpha'$ and $t \models \ominus(\Diamond\alpha')$ where $t'' = (C'', u'')$ is the parent of t. Then $\Diamond\alpha' \in C''$ [by the second induction hypothesis] which implies $\Diamond\alpha' \in C$ [by the construction of the atom graph].

  For the converse direction, suppose that $\Diamond\alpha' \in C$. Since G'_φ is a good subgraph, C is reachable from the initial vertex C_0. Hence, by the definition of the atom graph there exists a vertex C' on the path from C_0 to C such that $\alpha' \in C'$. Let C' be the last such vertex. We induct again on the distance between C' and C. Let d denote this distance. The base case is when $d = 0$. Then we have that $\alpha' \in C$ and hence $\Diamond\alpha' \in C$ [since C is an atom] which implies $t \models \Diamond\alpha'$ [by definition]. Let $d > 0$ and let $t'' = (C'', u'')$ be the parent of t. Since the distance between C' and C'' is $d - 1$ we can apply the second induction hypothesis to conclude that $t'' \models \Diamond\alpha'$. Hence $t \models \Diamond\alpha'$ [by definition].

□

1.2.10 Markov chain

A Markov chain is a tuple $\mathcal{M} = (V, E, \delta)$ where (V, E) is a directed graph and δ is a function $\delta : V \to \Delta(V)$ where $\Delta(V)$ is the set of probability distributions on V such that for $v \in V$, $\delta(v) = p_v$ if and only if $v' \notin vE$ implies $p_v(v') = 0$. Thus for every vertex $v \in V$, $\delta(v)$ gives a probability distribution p_v which gives the probabilities with which each outgoing edge of v is selected. A Markov chain is called finite if V is finite.

Given a Markov chain $\mathcal{M} = (V, E, \delta)$ and an initial vertex $v_0 \in V$ we can define a process on $\mathcal{M}$ as follows. Initially a token is placed at v_0. Whenever a token is in some vertex $v \in V$, a coin is tossed and the token is moved to a neighbour $v' \in vE$ with the probability $\delta(v)(v')$. If (V, E) has no dead-ends, then this process goes on forever.

Let $\mathcal{M} = (V, E, \delta)$ be a finite Markov chain and let the vertices V be ordered as $V = \{v_1, v_2, \ldots, v_{|V|}\}$. We can associate an $|V| \times |V|$ matrix M with $\mathcal{M}$ such that the ij th entry m_{ij} of M is the probability of going from the vertex v_i to v_j in the Markov chain $\mathcal{M}$, that is, $m_{ij} = \delta(i)(j)$. Such a matrix M is called the transition matrix of $\mathcal{M}$.

A vector $\mathbf{q} = (q_1, q_2, \ldots, q_{|V|})$ such that $\sum_{i=1}^{|V|} q_i = 1$ and $q_i \geq 0$ for all $i : \ 1 \leq i \leq |V|$ is said to a stationary distribution of a Markov chain $\mathcal{M}$ if it is a left eigenvector to the eigenvalue 1 of M, that is $qM = q$.

A Markov chain $\mathcal{M} = (V, E, \delta)$ is called irreducible if its graph is strongly connected. The period of a vertex v of a Markov chain is the gcd of all the cycles in (V, E) passing through v. A Markov chain is called aperiodic if it has period 1.

The following result is well known and can be found in any standard textbook on the topic.

Proposition 1.6 *We have*

i. Every finite Markov chain has a stationary distribution.

ii. Every irreducible and aperiodic Markov chain has a unique stationary distribution.

iii. Every irreducible and aperiodic Markov chain converges to its stationary distribution.

Thus a stationary distribution of an irreducible aperiodic Markov chain $\mathcal{M}$ represents what proportion of time the token will spend in every vertex of $\mathcal{M}$ in the above process.

1.3 Results in the literature

Classical strategic form games were introduced and studied by von Neumann and Morgenstern in their path-breaking work [vNM44], which really established the subject of game theory as a formal branch of mathematics. They proved the famous minmax theorem for two-player strategic form games and showed how to solve such games using linear programming. Nash, in his work [Nas50] proved that every strategic form multiplayer game has a Nash equilibrium in mixed strategies. His proof is of an existential nature and does not provide a procedure to compute the equilibrium. The computation of Nash equilibrium of strategic form games has attracted much attention and work of late. Notable among them are the works of Daskalakis, Papadimitriou and Goldberg [DGP06], Bernhard von Stengel et al [ARSvS10, vS10] etc.

Infinite games on graphs came to the foray mainly after a result of Büchi and Landweber [BL69] in the late sixties who showed that turn-based Muller games played on finite graphs are determined and the winning strategies of either player can be effectively synthesised in finite memory strategies. Martin [Mar75], in a deep result showed that every infinite turn-based game where the winning condition is Borel is determined. Since a Muller condition is also Borel, the determinacy of Muller games follows as a corollary of Martin's result. Gurevich and Harrington [GH82] cleverly used a data structure called the Latest Appearance Record (LAR) and re-proved the finite-memory determinacy of Muller games. Emerson and Jutla [EJ91] and Mostowski [Mos91] independently showed that turn-based parity games are determined in memoryless strategies. Zielonka [Zie98] used a tree representation of Muller objectives (now known as a Zielonka tree) and presented an elegant analysis of turn-based Muller and parity games. Using an insightful analysis of Zielonka's result, [DJW97] presented an optimal memory bound for winning strategies in turn-based Muller games. See [Tho90, Tho97] for beautiful surveys on the topic of infinite turn-based games with ω-regular winning conditions.

Mean-payoff games have also been widely studied in the literature. [EM79] showed that such games are memoryless determined. The value of a vertex $v \in V$, $val(v)$ of a mean-payoff game is the maximum limit-average reward that the 'max-player' can ensure when the play starts at v. A strategy s of the max-player at a vertex v is said to be optimal if it attains $val(v)$. Since such games are memoryless determined, every play finally settles down into a simple cycle and hence it is the cummulative payoffs of the simple cycles that determine the optimal strategies of the players. [ZP96] showed that the

the values of the vertices can be determined and optimal strategies can be effectively synthesised.

For the case of zero-sum multiplayer turn-based games, it has been shown in [CJM04] that every such game with a Borel winning condition has a Nash equilibrium. The existence of subgame perfect equilibrium for multiplayer turn-based games with zero-sum parity objectives was shown in [Umm06, GU08]. In [PS09], we show that in a multiplayer turn-based finite game where the players have preference over the various Muller sets of the vertices, a Nash equilibrium always exists and we give an effective procedure to compute such an equilibrium.

For the case of concurrent games, the literature is not so well developed. The existence of (quantitative) determinacy for concurrent games was shown again by Martin in [Mar98]. Concurrent games with reachability and parity objectives have been studied in [dAHK98, dAH00]. The values of concurrent games with parity objectives was characterised by quantitative μ-calculus formulae in [dAM01]. Chatterjee has extended a few of these results and proved many new ones. See, for instance, his thesis [Cha07] for a nice survey of the existing results on concurrent games and a collection of the results obtained by him.

Logical analysis of strategies in games has been extensively carried out and has a rich literature. Both modal and dynamic logic and temporal logic have been used to reason about strategies.

For the case of finite extensive form games, action indexed modal logics are well suited for logical analysis. Utilities can be coded up in terms of special propositions and the preference ordering is then induced by the implication available in the logic. Game trees themselves are taken as models of the logic. Adopting this approach, a characteristic formula for the backward induction procedure is exhibited in [Bon01]. Progressing further [HvdHMW03] show that the solution concept of subgame perfect equilibrium can be characterised in a modal logic framework. In [vB01, vB02] van Benthem argues that extensive form games can be thought of as process models along with special annotations identifying player nodes. A dynamic logic framework can then be used to describe complete strategies of players as well as reasoning about outcomes that can be ensured. Instead of coding objectives of players in terms of propositions, there have also been suggestions to incorporate elements of modal preference languages into the logic.

Various temporal logics have also been employed to reason about strategies in games. Notable among these is the work on alternating time temporal logic (ATL) [AHK02] which considers selective quantification over paths

that are possible outcomes of games in which players and an environment alternate moves. ATL reasons about structured games which are games on graphs where each node is associated with a single normal form game. Since the unfolding of the game structure encodes the past information, the logic itself can be extended with past modalities as well as knowledge modalities in order to reason about the history information and epistemic conditions used in strategising by players (see [vdHW02, JvdH04]). Extensions of ATL where strategies are allowed to be named and referred to in the formulas of the logic are proposed in [vdHJW05] and [WvdHW07]. ATL extended with the ability to specify actions of players in the formulas has been studied in [Ago06] and [Bor07].

Ramanujam and Simon [RS06, RS08b] give a logic to reason about structured strategies which uses the construct $\sigma \rightsquigarrow_i \beta$ which asserts that player i can play according to σ and ensure β. In [RS08a] they consider a dynamic logic in the lines of Parikh's game logic [Par85] where they reason about game-strategy pairs and their compositions rather than just composing games and analysing strategies separately. Typically, these logical studies involve decidability of satisfiability, complete axiomatisation and (relative) expressiveness of the logics. [Gho08] presents a complete axiomatisation of a logic describing both games and strategies in a dynamic logic framework where assertions are made about atomic strategies.

Part I

Introducing strategy switching in games

Chapter 2

Complexity of strategy switching

We start off by looking at turn-based finite extensive form games, the classical parity and Muller games but from a different point-of-view. We add the restriction that one of the players can apply strategies only from a given finite set S of strategies. In such a setting, many interesting questions naturally arise, viz., can an optimal strategy be obtained by 'composing' strategies from S? If so, what are the minimum number of such strategies required? What are the minimal number of strategy switches required in a winning play?

This chapter is intended as a warm up for things to follow. It shows that the questions surrounding strategy-switching are non-trivial and interesting. When the set of strategies a player can apply while playing is restricted, whether by bounding the memory she can use or the moves/strategies available to her etc., solving the game and computing optimal strategies can be more involved than when her strategy set is unrestricted. In some cases, an optimal strategy may not even exist and when it exists, it may be more complex.

2.1 Motivation

Consider the game of football. The scores of a match in the knock out stage may be tied in the 90 minutes or even after extra time and there may be a need for penalty shootout to decide the outcome. Now the manager would prefer players who are also good penalty-takers. But as it turns out, such players are a rare breed. A player might be exceptional on the

field, but might be a very poor penalty-taker. On the other hand, a player who's not quite skilled at regular play might be a skillful penalty-taker. Hence, the manager, towards the fag end of extra time, starts substituting his regular players with players who are good penalty-takers in anticipation of the penalty shootout. But this means the manager has to keep such players on the bench for the entire duration of the match and this involves a trade-off.

Thus the manager has to make a strategic decision before the match, when he announces his team. If he plays a good striker, rather than an expert penalty-taker, the team might be able to perform better in the regular 90 minutes itself. But there's also a chance of the match ending in a draw and going into the penalty-shootout stage. He will have to regret his decision then. The optimum strategy of the manager is to play players who are good at both regular match-play and penalty shootout. But he may not be able to employ this strategy because such players may just not be available! Hence, he has to play the optimum strategy by making substitutions towards the end of regular time as mentioned above. In other words, he has to compose two strategies from his set of available strategies: playing a regular player and playing a penalty-taker, to employ the optimum strategy. In this case the manager composes the strategies by switching from one to the other. However, one can imagine more innovative ways of composing strategies, like, deciding at the beginning of the game which strategy to employ and then playing accordingly, employing one strategy if certain condition holds of the game and if not, employing the other and so on. In this chapter we shall deal exclusively with switching strategies as a method of composing strategies. We shall study the other different and interesting ways of composing strategies in Chapter 4.

Examples such as the above are quite common in many games: a player has a limited set of strategies S at her disposal and composes strategies from S to play optimally or almost-optimally. A player would ideally like her set S to be small because strategies might be costly. But some strategies in S might be inevitable in the sense that she might not be able to play optimally without these strategies. In this chapter, we study these kind of questions: is it possible for a player to to play optimally by composing and switching between strategies from a fixed set S? If so, what is the minimum number of such strategies required? What is the minimal number of switches? In particular we are interested in the computational complexity of these questions when posed formally. We study the complexity of the above questions in the setting of two-player turn-based extensive-form, parity and Muller games and give various upper and lower bound results.

2.2 Extensive form games

We first study the complexity of the above problems for *two-player, turn-based, zero-sum, finite* extensive form games. As defined in section 1.2.2, an extensive form game $(\mathcal{T}, p)$ is a (finite) tree $\mathcal{T} = (T, E)$ where the set of vertices T is partitioned into $T = T_1 \cup T_2$. p is a function that assigns a tuple in $\{-1, 1\}^2$ to the leaf nodes, where 1 denotes a win and -1 denotes a loss for the corresponding player. At a leaf node $t \in T$ such that $p(t) = (x, y)$ we use the notations $p_1(t)$ and $p_2(t)$ to denote x and y respectively. Since we only consider zero-sum games, we have $p_1(t) + p_2(t) = 0$ for every leaf node $t \in T$.

As defined in Section 1.2.7, a positional strategy s_1 for player 1 in an initialised extensive form game $(\mathcal{T}, p, t_0)$ is a function $s_1 : T_1 \to A_1$ such that for every $t \in T_1$, $s_1(t) \in \Gamma_1(t)$ A strategy s_2 for player 2 is defined similarly.

Let Σ_1^{pos} and Σ_2^{pos} denote the set of positional strategies of player 1 and player 2 respectively. Given strategies $s_1 \in \Sigma_1^{pos}$ and $s_2 \in \Sigma_2^{pos}$ for players 1 and 2 and a vertex $t \in T$ we let $\rho^{s_1,s_2}(t)$ denote the unique play starting at t and conforming to s_1 and s_2. Let $p^{s_1,s_2}(t)$ denote the payoff at the leaf node of this play. As before, a strategy s_1 of player 1 is called winning if for every strategy s_2 of player 2, $p^{s_1,s_2}(t) = (1, -1)$. A winning strategy for player 2 is similarly defined.

Definition 2.1 *Given a subset $S \subseteq \Sigma_1^{pos}$ a strategy $s \in \Sigma_1^{pos}$ of player 1 is said to be built from S if for every node $t \in T$ $s(t) = a$ implies there exists $s' \in S$ such that $s'(t) = a$. If a strategy s is built from S then we can define a set of functions $\mathfrak{B}_s$ where for every $\beta \in \mathfrak{B}_s$, $\beta : T \to S$ such that $s(t) = \beta(t)(t)$ for every $t \in T$. Such a function $\beta \in \mathfrak{B}_s$ is called a certificate for s. The cardinality of the range of β is called the size of the certificate β.*

Thus, a certificate β for a strategy s gives a witness as to how the strategy s is composed using the strategies in the set S, i.e., given a node $t \in T$, which strategy of S is used.

Definition 2.2 *A play $\rho = t_0 \xrightarrow{a_1} t_1 \xrightarrow{a_2} \dots t_k$ is said to be built from S if for every $t_i \in T_1$, there exists $s \in S$ such that $s(t_i) = a_{i+1}$. If a play $\rho = t_0 \xrightarrow{a_1} t_1 \xrightarrow{a_2} \dots t_k$ is built from S then we can define a set of functions $\mathfrak{B}_\rho$ where for every $\beta \in \mathfrak{B}_\rho$, $\beta : [k] \to S$ such that for every $i \in [k]$, $s(t_i) = \beta(t_i)(t_i)$. Such a function $\beta \in \mathfrak{B}_\rho$ is called a certificate for ρ. The cardinality of the range of β is called the size of the certificate β.*

Definition 2.3 *Given a subset $S \subseteq \Sigma_1^{pos}$ and a play ρ of length k, if ρ is built from S and if β is a certificate, then a pair of indices $(i, j) : i < j \leq k$ is called a switch if $t_i, t_j \in T_1$, $t_l \notin T_1$ for any $j < l < k$ and $\beta(t_i) \neq \beta(t_j)$.*

Given a finite extensive form game $(\mathcal{T}, p)$, an initial node t_0 and a finite subset $S \subseteq s_1$ we wish to find out whether there exists a winning strategy s of player 1 such that s can be built from S. Also given a play ρ in $\mathcal{T}$ we wish to decide if ρ is built from S and if so, what is the minimum number of switches required. We study these questions below.

But first, we need to decide on a representation of these games and the size of such a representation.

Let the game tree $\mathcal{T} = (T, E)$ have n nodes and let $d = \max\{|A_1|, |A_2|\}$. Then every vertex $t \in T$ has at most d children. Also assume that the minimum number of children of any node is 2 (If a node t has a single child t', player 1 is forced to choose t'. Then we can simply verify if there exists a strategy $s \in S$ such that $s(t) = a$ where $t \xrightarrow{a} t'$). Hence, the depth of the tree is $\mathcal{O}(\log n)$. Also the number of leaves of the tree is $\mathcal{O}(\log n)$. Thus the specification of the payoff function requires size $\mathcal{O}(\log n)$. The number of actions available to a player at any vertex t is at most d and hence it takes size at most nd to represent the availability sets of the players. Since we are interested only in positional strategies, a strategy $s \in \Sigma_i$ of a player i is just a collection of n tuples of the form (t, a) such that $s_1(t) = a$. Thus a subset $S \subseteq \Sigma_i$ of size m requires size $\mathcal{O}(mn)$ to represent. Hence, the size of the representation is of the order of $n + 2nd + \log n + mn$. We denote this quantity by N.

Theorem 2.4 *Given a finite extensive form game $(\mathcal{T}, p)$, an initial node t_0 and a finite subset $S \subseteq \Sigma_1^{pos}$ the problem of determining whether there exists a winning strategy s of player 1 such that s can be built from S is ALOGSPACE-complete.*

Proof We first give an ALOGSPACE(N) algorithm for the above problem which we call COMPOSE-EXTENSIVE for convenience.

Algorithm 1 Deciding COMPOSE-EXTENSIVE

```
1: procedure DecideCompose(T, t_0, p, S = {s^1, ..., s^m})
2:     x ← BI(T, t_0, p)
3:     if x ≠ 1 then
4:         return "(T, p) is losing for player 1"
5:         exit
```

6: **else**
7: **repeat**
8: For every node $t \in T_1$ existentially guess a number $i : 1 \leq i \leq m$ and follow the edge $s^i(t)$
9: For every node $t \in T_2$ universally branch to every child of t
10: **until** a leaf node is reached
11: **return** YES only if for a leaf node t reached, $p_1(t) = 1$
12: **end if**
13: **end procedure**
14: **procedure** BI($\mathcal{T}, t_0, p$)
15: **repeat**
16: For every node $t \in T_1$ existentially guess an action $a \in \Gamma_1(t)$ and move to the child t' such that $t \xrightarrow{a} t'$
17: For every node $t \in T_2$ universally branch to every child of t
18: **until** a leaf node is reached
19: **if** for a leaf node t reached, $p_1(t) = 1$ **then**
20: **return** 1
21: **else**
22: **return** 0
23: **end if**
24: **end procedure**

The procedure DecideCompose has to remember $\log m$ bits for a guess. So DecideCompose takes $\log(m)$ space. Similarly the procedure BI has to remember $\log n$ bits for every existential guess. So the entire procedure takes $\log(mn)$ space. Thus COMPOSE-EXTENSIVE $\in$ ALOGSPACE(N). To see that the procedure DecideCompose is correct, first note that BI is the famous backward-induction procedure on the extensive form game tree $\mathcal{T}$. It checks whether player 1 has a winning strategy at all in the game $(\mathcal{T}, p)$. If player 1 has a winning strategy, then the remaining part of the procedure DecideCompose checks if such a strategy can be composed using strategies from S. To do so, it guesses a strategy s^i for every node of $\mathcal{T}$ (in step-8) and checks it against all possible strategies of player 2 (step 9).

Next, we show that COMPOSE-EXTENSIVE is ALOGSPACE hard. Let L be a language in logspace. Let $w \in \{0,1\}^*$ be a finite string. We show how to reduce w to an extensive form game $(\mathcal{T}, t_0, p)$ and a set S of strategies for player 1 such that $w \in L$ if and only if there exists a winning strategy for player 1 in S. Let M be an ALOGSPACE(N) machine that decides L. Our reduction simulates the run of M on w.

Step 1. Let (T, E) be the configuration graph of M on input w where T_1 is the set of existential configurations and T_2 is the set of universal configurations. Note that the number of such configurations is $\mathcal{O}(N)$.

Step 2. Let d be the maximum number of children of any player 1 node. Let $A_1 = A_2 = \{a_1, \ldots, a_d\}$. For every edge $(t, t') \in E$, such that t' is the ith neighbour of t, label (t, t') with a_i. Let $S = \{s^1, \ldots, s^d\}$. For every player 1 node t such that t is not a final configuration of M, define $s^j(t) = a_j$. If t has j neighbours such that $j < d$ then for every $i : j < i \leq d$, let $s^i(t) = a_d$. For every node t such that t is a final configuration of M, define t to be a leaf node.

Step 3. Label every leaf node t where t is an accepting configuration with the payoff (1,-1). Label all the other nodes with payoff (-1,1).

For the reduction to be complete, we need to show that the above steps can be carried out in LOGSPACE(N). For step 2, assuming that the vertices in T are ordered, the definition of the strategies $s^i \in S$ is implicit. To decide if it is a final configuration, we have to maintain one bit per node, which can be done in LOGSPACE(N). Finally, for step 3, it is again enough to maintain one bit x per node v where $x = 1$ iff t is an accepting configuration.

We now show that the above reduction is correct. That is, we have to show that player 1 has a winning strategy in $(\mathcal{T}, p)$ if and only if M has an accepting configuration on w. Suppose M has an accepting configuration. Then by our construction, this configuration is a leaf node of $\mathcal{T}$ which is labelled with $(1, -1)$. Now, since M is an alternating machine, for every existential configuration t, there exists a run from t which reaches an accepting configuration and for every universal configuration t, every run from t reaches an accepting configuration. Since, by construction we have assigned the existential configurations of M to player 1 and the universal ones to player 2, player 1 wins no matter which branch player 2 chooses, i.e., player 1 has a winning strategy. Moreover, since the set S has been constructed so as to cover all possible strategies of player 1, this winning strategy can be built from S.

Conversely, suppose player 1 has a winning strategy in $(\mathcal{T}, p)$ that can be built from S. By the same fact that we have assigned the existential configurations of M to player 1 and the universal ones to player 2, player 1 wins no matter which branch player 2 chooses. In other words, for every existential configuration t of M, there exists a run from t which reaches an accepting configuration and for every universal configuration t of M, every run from t reaches an accepting configuration. Hence M accepts w. □

Next analyse the running time complexity of minimising the number of strategies of S used by player 1 to play a winning strategy.

Theorem 2.5 *Given a finite extensive form game $(\mathcal{T}, p)$, an initial node t_0, a finite subset $S \subseteq \Sigma_1^{pos}$ and a natural number k in binary, the problem of determining whether there exists a winning strategy s of player 1 such that s has a certificate of size at most k is NP-complete.*

Proof We first give an NP algorithm for the above problem which we call #COMPOSE-EXTENSIVE for convenience.

Algorithm 2 Deciding #COMPOSE-EXTENSIVE

procedure DECIDE#COMPOSE($\mathcal{T}, t_0, p, S = \{s^1, \ldots, s^m\}, k$)
  $x \leftarrow \mathrm{BI}(\mathcal{T}, t_0, p)$
  if $x \neq 1$ **then**
    return "$(\mathcal{T}, p)$ is losing for player 1"
    exit
  else
    Guess a subset S' of S such that $|S'| \leq k$
    For every player 1 node t guess a strategy $s_t \in S'$ and mark t with s_t
    return VERIFY($\mathcal{T}, t_0, p, \{s_t\}_{t \in V}, x$)
  end if
end procedure
procedure BI($\mathcal{T}, t_0, p$)
  for $j \leftarrow d - 1$ to 0 **do** ▷ d is the depth of the tree $\mathcal{T}$
    for every node t of depth j **do**
      if $t \in T_1$ **then**
        if there exists a child t' of t such that $p_1(t') = 1$ **then**
          $p(t) \leftarrow p(t')$
        else
          $p(t) \leftarrow (-1, 1)$
        end if
      else
        if there exists a child t' of t such that $p_2(t') = 1$ **then**
          $p(t) \leftarrow p(t')$
        else
          $p(t) \leftarrow (1, -1)$
        end if
      end if

28: **end for**
29: **end for**
30: **return** $p_1(t_0)$
31: **end procedure**
32: **procedure** VERIFY($\mathcal{T}, t_0, p, \{s_t\}_{t\in T}, x$)
33: mark t_0
34: for every player 1 node t mark $s_t(t)$
35: for every player 2 node t mark t' for all the children t' of t
36: **if** for every marked leaf node t', $p_1(t') = 1$ **then**
37: **return** YES
38: **else**
39: **return** NO
40: **end if**
41: **end procedure**

Note that as the procedure VERIFY makes a single pass of the tree $\mathcal{T}$ in a top-down fashion, the procedure DECIDE#COMPOSE is non-deterministic polynomial time. To see the correctness of the above algorithm, first note that the procedure BI employs the classical backward induction algorithm to decide if player 1 has a winning strategy in the game $(\mathcal{T}, p)$. Step-7 of the algorithm then guesses a subset S' of the strategy set S and step-8 marks every node with one of the guessed strategies. The procedure VERIFY then verifies whether the marked strategies actually achieve a winning strategy. This is done by testing the strategy at each marked node (step-34) against all possible strategies of player 2 (step-35).

To show that #COMPOSE-EXTENSIVE is NP hard, we give a reduction from SET-COVER to #COMPOSE-EXTENSIVE. Let $(Z, \mathcal{Y} = \{Y_1, \ldots, Y_k\})$ be an instance of SET-COVER where $Z \subseteq \bigcup \mathcal{Y}$ is the target set and the elements of $\mathcal{Y}$ are the candidate sets. The intuition is that at every step, player 2 picks an element x from the target set Z and player 1 has to pick a candidate set. The best strategy for player 1 is to pick a candidate set Y_i such that $x \in Y_i$. If she does not do so, then the play moves to a vertex where player 1 loses immediately. Now, in the worst case, player 2 picks a different element of Z every time till all the elements of Z are exhausted. To cover these elements, player 1 can't but choose at least a set cover for Z. We formalise this intuition below. The arena $\mathcal{A} = (T, E)$ where

- $T = T_1 \cup T_2$ such that
  - $T_1 = \{(i, x) \mid 1 \leq i \leq |Z|,\ x \in Z\}$.

– $T_2 = \{t_0, t_L\} \cup \{(i, x, Y) \mid 1 \leq i \leq |Z|, x \in Z, Y \in \mathcal{Y}\}$.

- $$\begin{aligned} E = &\{(t_0, (1, x)) \mid (1, x) \in T_1\} \\ &\cup \{((i, x), (i+1, x, Y))\} \\ &\cup \{((i, x, Y), (i+1, y)) \mid x \in Y,\ i < |Z|\} \\ &\cup \{((i, x, Y), t_L) \mid x \notin Y\}. \end{aligned}$$

The payoff of every leaf node other than t_L is $(1, -1)$. The payoff of t_L is $(-1, 1)$. Thus t_L is losing for player 1 and every other leaf node is winning for her. The strategy set S is defined as $S = \{s^Y \mid Y \in \mathcal{Y}\}$ where $s^Y((i, x)) = (i + 1, x, Y)$ for all $(i, x) \in V_0$.

Let $(\mathcal{T}, p, t_0)$ be the extensive form game constructed above. From the argument preceding the construction, it is clear that $(Z, \mathcal{Y} = \{Y_1, \ldots, Y_k\})$ has a set cover of size k if and only if $(\mathcal{T}, p, t_0)$ has a winning strategy s for player 1 in S having a certificate of size at most k. □

Finally, given a play ρ in $\mathcal{T}$ we wish to minimise the number of switches between the strategies of S that are required to achieve ρ.

Theorem 2.6 *Given a finite extensive form game $(\mathcal{T}, p)$, a finite subset $S \subseteq \Sigma_1^{pos}$ and a play ρ in $\mathcal{T}$, we can decide whether ρ can be built from S and compute the minimum size of a certificate in PTIME.*

Proof For convenience, we call the above problem minSWITCH-EXTENSIVE. We give a greedy algorithm for it, prove its correctness and then show that it runs in time polynomial in the size of the input.

Algorithm 3 Deciding minSWITCH-EXTENSIVE

1: **procedure** DecideMinSwitch($\mathcal{T}$, $\rho = t_0 \xrightarrow{a_1} t_1 \xrightarrow{a_2} \ldots \xrightarrow{a_k} t_k$, $S = \{s^1, \ldots, s^m\}$, k)
2: **for** $i = 0$ to $k - 1$ **do**
3: Check if there exists $s \in S$ such that $s(t_i) = a_i$
4: If not, **return** "ρ cannot be built from S" and EXIT
5: **end for**
6: Let $\beta_\rho = \emptyset$
7: Start at t_0
8: **repeat**

9: For the next unmarked node t in ρ pick a strategy $s \in S$ that agrees with the maximum number of consecutive nodes along ρ starting at t. Suppose these nodes are $t_1, \ldots, t_k$ where $t_1 = t$. Mark $t_1, \ldots, t_k$ with s and set $\beta_\rho(t_i) = s$ for every $i : 1 \leq i \leq k$
10: **until** the leaf node of ρ is reached
11: **end procedure**

We claim that the certificate β_ρ generated by the above algorithm has the minimum number of switches. To see this suppose for a node t in ρ, the above greedy algorithm picks a strategy s whereas picking another strategy s' would have covered a greater number of consecutive nodes in ρ. This can happen only if the algorithm had already picked s for the parent t' (say) of t itself. In that case picking s' at t would involve a switch of $+1$. Now suppose s agrees with ρ till node u and s' agrees with ρ till node u' where u is an ancestor of u'. Then the greedy algorithm above can still pick s' for the child of u still increasing the number of switches by just 1. Continuing this way, we see that since the number of switches in the greedy algorithm is no more than that in the optimal, we conclude that the above algorithm returns a certificate β_ρ with the minimal number of switches.

To see that the algorithm runs in time polynomial in the size of the input, note that in the worst case, for every node t in ρ all the strategies in S agree with just one consecutive successor in ρ. Thus the algorithm has to examine m strategies at every node along ρ. Since ρ can be atmost $\mathcal{O}(\log n)$ long, the running time of the algorithm is $\mathcal{O}(mlogn)$. □

2.3 Parity games

We now move to the setting of infinite duration games on finite graphs where the players are restricted to play strategies only from a fixed finite set of strategies. We ask questions similar to the ones asked in the previous section for finite extensive form games: whether the player can play a winning strategy by composing and switching between strategies from S and if so, what is the minimum number of switches required etc.

Parity games enjoy positional determinacy (from every vertex $v \in V$, either player 1 or player 2 has a positional winning strategy) [Mos91, EJ91, Zie98]. Hence we work exclusively with positional strategies here. Let Σ_1^{pos} and Σ_2^{pos} be the set of positional strategies for player 1 and player 2 respectively. Given strategies $s_1 \in \Sigma_1^{pos}$ and $s_2 \in \Sigma_2^{pos}$ for players 1 and 2 and a vertex $v \in V$ we let $\rho^{s_1,s_2}(v)$ denote the unique play starting at v and con-

forming to s_1 and s_2. Since the strategies of both the players are positional, every play in a parity game ends up in a simple cycle.

Definition 2.7 *A simple cycle $v_1, \ldots, v_k$ where $v_1 = v_k$ is called good for player 1 and bad for player 2 if the maximum priority of all the vertices in the cycle is even, that is, $\max\{\chi(v_i) \mid 1 \leq i \leq k\}$ is even. Otherwise it is called good for player 2 and bad for player 1.*

Thus, a strategy s_1 of player 1 is winning if for every strategy s_2 of player 2 $\rho^{(s_1,s_2)}(v)$ ends up in a good cycle for player 1.

The notions of a strategy being built from a set $S \subseteq \Sigma_1^{pos}$, a certificate, a switch etc. are the same as those described in the previous section for extensive form games.

Note that for a graph of size n, the description of a parity game can be given in size $\mathcal{O}(n^2)$. Also a set $S \subseteq \Sigma_1^{pos}$ containing m strategies can be described in $\mathcal{O}(mn)$. We denote by N the size of the input which is the summation of the above quantities.

We now describe the problems we wish to study in the setting of parity games.

Theorem 2.8 *Given an arena $\mathcal{A}$, a set of priorities C, a priority function χ, an initial vertex v_0 and a subset $S \subseteq \Sigma_1^{pos}$ of positional strategies, the deciding if player 1 has a winning strategy s such that s can be built from S can done in time $\mathcal{O}(f(N))$ where $f(k)$ is the time taken to solve a parity game of size k.*

Proof Let us call the above problem COMPOSE-PARITY for convenience. We construct a new parity game $(\mathcal{A}', C, \chi)$ from $(\mathcal{A}, C, \chi)$ as follows: $\mathcal{A}' = (V', E')$ where $V' = V$ and $E' \subseteq E$ is constructed as follows. For vertices $v, v' \in V$ $(v, v') \in E'$ iff $(v, v') \in E$ and there exists a strategy $s \in S$ such that $s(v) = v'$. The function χ remains the same. $(\mathcal{A}', C, \chi)$ now is a parity game whose size in the worst case is the same as the original game $(\mathcal{A}, C, \chi)$. It is also clear that if player 1 has a winning strategy s in S from a vertex $v \in V$ in the game $(\mathcal{A}, C, \chi)$ then she has a winning strategy from $v \in V'$ in the game $(\mathcal{A}', C, \chi)$. □

Note that the converse need not hold. That is because suppose s is a winning strategy for player 1 in $(\mathcal{A}, C, \chi)$ and suppose $s(v) = v'$ for some $v \in V$. There may not exist $s' \in S$ such that $s'(v) = v'$.

Remark We can also give an easy reduction from the problem of deciding the winner in a parity game, PARITY to COMPOSE-PARITY as follows.

Let d be the maximum out-degree of the game graph. Define the set of strategies $S = \{s^1, \ldots, s^d\}$ as $s^i(v) =$ the ith neighbour of v. Then, by solving COMPOSE-PARITY, we can determine if a vertex v_0 is winning for player 1 and the certificate gives the winning strategy. Thus the two problems COMPOSE-PARITY and PARITY are equivalent in the sense that an algorithm to decide one also yields an algorithm to decide the other and vice-versa.

Theorem 2.9 *Let $\mathcal{A}$ be an arena, C be a set of priorities, χ be a priority function, v_0 be an initial vertex and $S \subseteq \Sigma_1^{pos}$ be a subset of positional strategies such that there exists a winning strategy s for player 1 which can be built from S. Then the problem of deciding if there exists a winning strategy s for player 1 that has a certificate of size at most k, k is in binary, is NP-Complete.*

Proof We call the above problem #COMPOSE-PARITY. We first give an NP algorithm for #COMPOSE-PARITY.

Algorithm 4 Deciding #COMPOSE-PARITY

1: **procedure** DECIDECOMPOSE($\mathcal{A}, C, \chi, v_0, S = \{s^1, \ldots, s^m\}, k$)
2: Guess a subset S' of S such that $|S'| = k$
3: For every player 1 node v, guess a strategy $s_v \in S'$
4: For every player 1 node v, remove every edge (v, v') such that $s_v(v) \neq v'$
5: **return** YES if and only if in the resulting graph every reachable cycle is good for player 1
6: **end procedure**

To show that #COMPOSE-PARITY is NP hard, we give a reduction from SET-COVER to #COMPOSE-PARITY. Let $(Z, \mathcal{Y} = \{Y_1, \ldots, Y_k\})$ be an instance of SET-COVER where $Z \subseteq \bigcup \mathcal{Y}$ is the target set and the elements of $\mathcal{Y}$ are the candidate sets. Just as in the case of extensive form games, the intuition is that at every step, player 2 picks an element x from the target set Z. The best strategy for player 1 is to pick a candidate set Y_i such that $x \in Y_i$. If she doesn't do so, then the play moves to a vertex with a large odd priority. Now, in the worst case, player 2 picks a different element of Z every time till a cycle is completed. To cover these elements, player 1 can't but choose at least a set cover for Z. We formalise this intuition below.

Let $\mathcal{A} = (V, E)$ where

- $V = V_1 \cup V_2$ where
  - $V_1 = \{(i, x) \mid 1 \leq i \leq |Z|,\ x \in Z\}$
  - $V_2 = \{v_0, v_\infty\} \cup \{(i, x, Y) \mid 1 \leq i \leq |Z|, x \in Z, Y \in \mathcal{Y}\}$.
- $$\begin{aligned} E = &\{(v_0, (1, x)) \mid (1, x) \in V_1\} \\ &\cup \{((i, x), (i + 1, x, Y))\} \\ &\cup \{((i, x, Y), (i + 1, y)) \mid x \in Y,\ i < |Z|\} \\ &\cup \{((i, x, Y), v_\infty) \mid x \notin Y\} \\ &\cup \{(v_\infty, v_0)\} \cup \{((|Z|, x, Y), v_0)\}. \end{aligned}$$

Let $C = \{c, c + 1\}$ such that c is even. The priority function is defined as $\chi(v) = c$ for all $v \in V \setminus \{v_\infty\}$ and $\chi(v_\infty) = c + 1$. The strategy set S is defined as $S = \{s^Y \mid Y \in \mathcal{Y}\}$ where $s^Y((i, x)) = (i + 1, x, Y)$ for all $(i, x) \in V_1$.

From the argument preceding the construction, it is clear that player 1 has a winning strategy with certificate of size k in the game $(\mathcal{A}, C, \chi)$ constructed above iff $(Z, \mathcal{Y} = \{Y_1, \ldots, Y_k\})$ has a set cover of size at most k. □

Remark Deciding whether PARITY is in PTIME is a long-standing open question. However, there are known subexponential time algorithms for PARITY [JPZ06, Sch08]. Thus assuming the hardest problems in NP do not have subexponential algorithms, #COMPOSE-PARITY is strictly harder than PARITY.

Theorem 2.10 *Given an arena $\mathcal{A}$, a set of priorities C, a priority function χ, an initial vertex v_0, a play ρ in $\mathcal{A}$ and a subset $S \subseteq \Sigma_1^{pos}$, we can decide whether ρ can be built from S and compute the minimum size of a certificate in PTIME.*

Proof Given a play ρ that eventually ends in a cycle, it is clear that a greedy algorithm similar to Algorithm 3 of the previous section decides the above problem in PTIME. □

2.4 Muller games

Finally, in this section, we turn to Muller games. It is known that Muller games are determined such that from every vertex, either player has a winning strategy using finite memory [BL69]. We thus deal with finite memory strategies in this section. Like in the previous sections, we restrict these games to the setting where player 1 is allowed to play strategies from a fixed finite set S. We ask similar questions: whether the player can play a winning strategy by composing and switching between strategies from S and if so what is the minimum number of switches required etc.

Let $\mathcal{A} = (V, E)$ be an arena equipped with the Muller winning condition where the Muller sets are given by $\mathcal{F}$. Let Σ_1^{fin} and Σ_2^{fin} denote the sets of finite memory strategies for player 1 and 2 respectively and let Σ_1^{pos} and Σ_2^{pos} be the sets of (all) strategies for player 1 and 2 respectively.

Definition 2.11 *A strategy s_1 of player 1 is called winning if for every strategy s_2 of player 2, $\inf(\rho^{(s_1,s_2)}(v)) \in \mathcal{F}$.*

Definition 2.12 *Given a subset $S \subseteq \Sigma_1^{fin}$ a strategy $s \in \Sigma_1^{pos}$ is said to be built from S if for every play ρ, $s(\rho) = a$ implies there exists $s' \in S$ such that $s'(\rho) = a$. If a strategy s is built from S then we can define a set of functions $\mathfrak{B}_s$ where for every $\beta \in \mathfrak{B}_s$, $\beta : P^{fin} \to S$ such that $s(\rho) = \beta(\rho)(\rho)$ for every $\rho \in P^{fin}$, where P^{fin} is the set of finite plays in $\mathcal{A}$. Such a function $\beta \in \mathfrak{B}_s$ is called a certificate for s. The cardinality of the range of β is called the size of the certificate β.*

Definition 2.13 *A play ρ is said to be built from S if for every prefix ρ_i of ρ, there exists $s \in S$ such that $s(\rho_i) = \rho(i+1)$. If a play ρ is built from S then we can define a set of functions $\mathfrak{B}_\rho$ where for every $\beta \in \mathfrak{B}_\rho$, $\beta : pref(\rho) \to S$ such that $s(\rho') = \beta(\rho')(\rho')$ for every $\rho' \in pref(\rho)$, where $pref(\rho)$ is the set of prefixes of ρ. Such a function $\beta \in \mathfrak{B}_\rho$ is called a certificate for ρ. The cardinality of the range of β is called the size of the certificate β.*

Definition 2.14 *Given a subset $S \subseteq \Sigma_1^{fin}$ a strategy $s \in \Sigma_1^{pos}$ is said to be built from S using finite memory if s has a certificate β_s given by a tuple $(M^\beta, \delta^\beta, g^\beta, m_I^\beta)$ where M^β is the memory of the certificate β^s, $\delta^\beta : M^\beta \times V \to M^\beta$ is the memory update, $g^\beta : M^\beta \times V \to S$ is the strategy update and $m_I^\beta \in M^\beta$ is the initial memory such that if $\rho = v_0 \dots v_k \in V^* V_0$ is a play and $m_0^\beta \dots m_{k+1}^\beta$ is a sequence determined by $m_0^\beta = m_I^\beta$ and $m_{i+i}^\beta = \delta^\beta(m_i^\beta, v_i)$ then $\beta_s(\rho) = g^\beta(m_{k+1}^\beta, v_k)$.*

We are now ready to answer the questions about Muller games that we posed in the beginning of this section. However, we first need to say how these games are presented and fix the size of the input. The size of the arena $\mathcal{A}$ is just $\mathcal{O}(n^2)$ where n is the number of vertices in $\mathcal{A}$. We do not take a stand on how the Muller sets in $\mathcal{F}$ are specified. They may be specified explicitly, in which case there are at most 2^n such sets or they may be specified implicitly, using a finite set of colours or even the Zielonka tree. The running time of solving a Muller game depends critically on the representation of these sets. See [Zie98, DH05, Hor08] for more on this. We assume that the strategies in the finite set S are specified as finite state transducers. Hence the representation of a strategy s with a memory of m states can be given size $\mathcal{O}(m^2)$. We denote the total size of the input by N.

Theorem 2.15 *Given an arena $\mathcal{A} = (V, E)$, Muller sets $\mathcal{F}$, an initial vertex $v_0 \in V$ and a finite subset $S \subseteq \Sigma_1^{fin}$ of finite memory strategies for player 1, it is decidable in time $\mathcal{O}(f(n \cdot m^{|S|}))$ if there exists a winning strategy s for player 1 such that s can be built from S, where $f(k)$ is the time required to solve a Muller game of size k and m is the maximum memory of any strategy in S. Moreover if there is a certificate for the winning strategy then there exists a certificate which is finite memory in S.*

Proof Let $S = \{s^1, \ldots, s^k\}$ where $s^i = (M^i, \delta^i, g^i, m_I^i)$. Define the graph $\mathcal{A} \times S$ as $\mathcal{A} \times S = (V', E', v_0')$ where:

- $V' = V \times M^1 \times \ldots \times M^k$.
- $E' \subseteq V' \times 2^{[k]} \times V'$ such that $(v, m_1, \ldots, m_k) \xrightarrow{X} (v', m_1', \ldots, m_k')$ if and only if $\delta^i(v, m_i) = m_i'$ for all $i : 1 \leq i \leq k$ and $g^\ell(v, m_l) = a$ such that $v \xrightarrow{a} v'$ for all $\ell \in X$.
- $v_0' = (v_0, m_I^1, \ldots, m_I^k)$.

Let $\mathcal{F}' \subseteq V'$ be defined as: for all $F \in \mathcal{F}$, $F' = \{(v, m_1, \ldots, m_k) \in V' \mid v \in F\} \in \mathcal{F}'$. Also let $V' = V_1' \cup V_2'$ such that $(v, m_1, \ldots, m_k) \in V_1'$ iff $v \in V_1$.

Then $(\mathcal{A} \times S, \mathcal{F}', v_0')$ is a Muller game, the size of which is $\mathcal{O}(n \cdot m^{|S|})$. Solve it in time $\mathcal{O}(f(n \cdot m^{|S|})$. We claim that player 1 has a winning strategy in $(\mathcal{A}, \mathcal{F}, v_0)$ built from S if and only if she has a winning strategy in $(\mathcal{A} \times S, \mathcal{F}', v_0')$.

Let s' be a winning strategy of player 1 in the game $(\mathcal{A} \times S, \mathcal{F}', v_0')$. Then for any history ρ in $\mathcal{A} \times S$ if $s'(\rho) = a$ then by construction, it must be the

case that there exists a strategy $s_i \in S$ such that $s_i(\rho') = a$ where ρ' is the history ρ projected to the first component. Player 1 can thus play according to such a strategy s_i for every history and win in $(\mathcal{A}, \mathcal{F}, v_0)$.

Conversely, suppose player 1 has a winning strategy s in $(\mathcal{A}, \mathcal{F}, v_0)$ such that s is built from S. Then for every history ρ in $\mathcal{A}$ there exists a strategy $s_i \in S$ such that $s_i(\rho) = s(\rho) = a$ (say). Let ρ' be the history in $\mathcal{A} \times S$ such that ρ' projected to the first component is ρ. Let $last(\rho') = (v, m_1, \ldots, m_k)$. By construction, we have that $(v, m_1, \ldots, m_k) \xrightarrow{X} (v', m_1', \ldots, m_k')$ such that $i \in X$ and $v \xrightarrow{a} v'$. Hence, player 1 can play a at history ρ' in $(\mathcal{A} \times S)$. Thus by playing as described, player 1 can win in the game $(\mathcal{A} \times S, \mathcal{F}', v_0')$.

Let $s_1' = (M, \delta, g, m_I)$ be a winning strategy for player 1 in the game $(\mathcal{A} \times S, \mathcal{F}', v_0')$. Note that the action update g can be viewed as a function $g : M \times V' \to 2^{[k]}$ We show how to obtain a finite memory certificate β for a winning strategy in s_1 in the game $(\mathcal{A}, \mathcal{F}, v_0)$. We define β to be the tuple $(M^\beta, \delta^\beta, g^\beta, m_I^\beta)$ where:

- $M^\beta = M^1 \times \ldots M^k \times M$.
- $\delta^\beta : M^\beta \times V \to M^\beta$ such that

  $$\delta^\beta(\langle m_1, \ldots, m_k, m\rangle, v) = \langle \delta^1(m_1, v), \ldots, \delta^k(m_k, v), \delta(m, \langle v, m_1, \ldots, m_k\rangle)\rangle$$

- $g^\beta : M^\beta \times V \to S$ where $g^\beta(\langle m_1, \ldots, m_k, m\rangle, v) \in g(m, \langle v, m_1, \ldots, m_k\rangle)$.
- $m_I^\beta = \langle m_I^1, \ldots, m_I^k, m_I\rangle$.

□

Note that the function g^β above outputs a strategy in S for every vertex-memory pair. The certificate β is nothing but a finite state automaton which takes as input a sequence of vertices and outputs one of many strategies at every memory state. We would like to minimise the number of total strategies of S used for a winning certificate. In other words, we would like to solve the following problem:

- #COMPOSE-MULLER $= \{(\bar{\beta}, k) \mid$ there exists a certificate that uses at most k strategies $\}$

where $\bar{\beta}$ is a certificate that gives all possible strategies of S applicable at every memory state. Formally, $\bar{\beta} = (M_{\bar{\beta}}, \delta_{\bar{\beta}}, g_{\bar{\beta}}, m_{\bar{\beta}}^I)$ where $g_{\bar{\beta}} : M_{\bar{\beta}} \times V \to 2^{[k]}$.

Theorem 2.16 *#COMPOSE-MULLER is NP-complete.*

Proof That #COMPOSE-MULLER is in NP is clear: Just guess a subset $S' \subseteq S$ of k strategies and verify if $g_{\bar{\beta}}$ prescribes at least one strategy in S' as output for every memory state in $M_{\bar{\beta}}$.

To show that #COMPOSE-MULLER is NP hard, we give a simple reduction again from SET-COVER. Let $(Z, \mathcal{Y} = \{Y_1, \ldots, Y_k\})$ be an instance of SET-COVER where $Z \subseteq \bigcup \mathcal{Y}$ is the target set and the elements of $\mathcal{Y}$ are the candidate sets. Let $M_{\bar{\beta}} = Z$ and $g_{\bar{\beta}}(x, v) = \{Y \in \mathcal{Y} |\ x \in Y\}$ for all v. That is, the memory is the set of elements of the target set Z and with each such memory x we associate all those candidate sets in $\mathcal{Y}$ that contain x. It is now clear that $(Z, \mathcal{Y} = \{Y_1, \ldots, Y_k\})$ has a set cover of size at most k if and only if there exists a certificate that uses k strategies. □

Chapter 3

When switching strategy comes with a cost

Imagine a firm which brings out its products in packs of 1 kilogram. But after a while, the firm observes the sales statistics of the various markets and realises that the buyers prefer smaller packs, say packs of 500 grams. Hence the firm decides to release packs of 500 grams as well. But now it has to invest some money in changing its infrastructure: it has to produce wrappers and boxes of 500 grams which involve a certain cost (manufacturing, printing etc.), it has to change its packaging and advertising policies and so on. But after a few days, if the firm observes a dip in its market shares and once again decides to revert to releasing packs of 1 kilogram, the above investment would be rendered fruitless and would result in a loss.

There are many such examples in day-to-day life where switching strategies involve a cost. In this chapter we look at games where a player has to incur a cost if she changes her strategy from one round to the next. Our model is that of infinite repeated strategic form games with discounted payoffs. In such a model, a player who switches her strategy from one round to the next does so only if she knows that her increase in payoff would compensate for the cost involved in the switch. Hence, the player has to make a decision about whether to switch. Moreover, her own switching might trigger other players to switch their strategies as well and this might result in a complicated and interesting recursive switching phenomenon. The questions to ask in this setting are: When should a player switch her strategy? What are the payoffs that can be supported, i.e., what are the feasible payoffs? How are equilibria affected by the switching cost? In this chapter we address these questions.

In particular, we first show that some of the classical equilibrium tuples need not be equilibria when a cost is involved for every strategy switch between successive rounds. We then show that by increasing the cost of switching strategies, players can be made to stick to their strategies for longer and longer time. We show that in general, there might exist equilibrium payoff tuples in the strategic form game that are not achievable in the repeated game with costs but one can get ϵ close to such equilibria given that the players negotiate prior to playing the game. Finally, we prove a folk theorem for these games.

3.1 The model

3.1.1 Repeated strategic form game

Let $\mathcal{G}$ be an n-player strategic form game, as defined in Section 1.2.1. That is, $\mathcal{G}$ is an n-dimensional payoff matrix. A repeated game is one where the game $\mathcal{G}$ is played repeatedly, in discrete time-units (rounds), possibly for an unbounded duration. There should be a way to accumulate the payoffs received by the players in the various rounds. Several such conventions have been studied in the literature. One such commonly accepted method is called the method of discounted payoff. In such a method, if the action tuples played by the players in rounds $1, 2, \ldots$ are $\mathbf{a}_1, \mathbf{a}_2, \ldots$, then the cumulative payoff to player i is given by

$$p_i = (1-\delta)[p_i(\mathbf{a}_1) + \delta p_i(\mathbf{a}_2) + \delta^2 p_i(\mathbf{a}_3) + \ldots]$$

where $\delta : 0 \leq \delta \leq 1$ is called the discounting factor. It represents the patience of the players. The closer δ is to 1, the more patient the players are. Intuitively, players value their payoffs in the current round more than what they will get in future rounds. Hence the payoffs in the future rounds are discounted.

Such repeated strategic form games with discounting (both finitely repeated and infinitely repeated) have been quite well-studied. There are many results about such games, which are usually called folk-theorems. A representative of such a theorem is one that states that any payoff in the feasible region (a region that guarantees some minimum payoff to all the players) can be supported in such games provided that the players are patient enough.

3.1.2 Repeated game with switching-cost

In our model of repeated strategic form games, every player incurs a cost c in every round $t+1$ if she changes her action (strategy) from what she played in round t. Thus if there are two players and the action tuples played in rounds t and $t+1$ are (a_t, b_t) and (a_{t+1}, b_{t+1}) respectively such that $a_t \neq a_{t+1}$ then her payoff in round $t+1$ is $p_1(a_{t+1}, b_{t+1}) - c$ discounted appropriately, where $p_1(a_{t+1}, b_{t+1})$ is her payoff corresponding to the action tuple (a_{t+1}, b_{t+1}).

3.1.3 Related work

Lippman and Wang study both finitely repeated [LW97] and infinitely repeated [LW09] games with switching-costs. In the infinitely repeated setting, suppose the sequence of action profiles chosen by the players is $\mathbf{a}_0, \mathbf{a}_1, \ldots$ and suppose the actions are changed only at intervals of length Δ. The payoff for player i they consider is given as

$$\sum_{t=0}^{\infty} \int_{t\Delta}^{(t+1)\Delta} e^{-rs} u_i(\mathbf{a}_t) ds - \sum_{t=0}^{\infty} e^{-rt\Delta} \epsilon I_i(\mathbf{a}_{t-1}, \mathbf{a}_t)$$

where $I_i(\mathbf{a}, \mathbf{a}') = 0$ if $\mathbf{a}(i) = \mathbf{a}'(i)$ and 1 otherwise and r is the continuous time discount rate. Carrying out the integration, putting $r = 1$ and setting $\delta = e^{-\Delta}$, we get

$$(1-\delta) \sum_{t=0}^{\infty} \delta^t u_i(\mathbf{a}_t) - \sum_{t=0}^{\infty} \delta^t \epsilon I_i(\mathbf{a}_{t-1}, \mathbf{a}_t).$$

In the above setting they first show that the set of equilibrium payoffs is exactly the usual folk theorem set if the switching cost is small relative to a round's worth of payoff but differs from the usual set if the cost is large relative to one round of payoff. Secondly, when one considers a sequence such that $\epsilon/(1-\delta)$ goes to infinity, one gets a limiting set of payoffs which differs from the folk theorem set in two ways. First, the payoff a player can guarantee herself is smaller with switching costs. Intuitively, if a player needs to randomise, the expected costs of switching actions makes this too costly. Second, the notion of feasibility changes as well in the limit as $\epsilon \to 0$. For example in the coordination game

3,3	0,0
0,0	1,1

the usual folk theorem set is all payoff vectors (u_1, u_2) where $u_1 = u_2$ and $.75 \leq u_i \leq 3$. But if $(\epsilon, \delta) \to (0, 1)$ with $\epsilon/(1-\delta) \to \infty$ along the sequence, the set of equilibrium payoffs converges to the set of all (u_1, u_2) such that $(0, 0) \leq (u_1, u_2) \leq (3, 3)$.

Our setting is one of discrete time: the players play the strategic form game once in every time unit and they receive payoffs according to the actions they play. The results we prove are similar to [LW09] but we work in a much simpler setting and hence the analyses are also simpler.

Switching costs in games have also been studied by Chakrabarti [Cha90] where he analyses infinitely repeated games with a more general 'inertia cost' and also by Dutta [Dut95] who studies switching costs in stochastic games. Their results and also some of the results of [LW09] are similar to the ones we prove, in that, they show that both individual rationality and feasibility must be redefined to take into account the switching costs.

3.2 Example and observations

To gain intuition into how the equilibria of infinitely repeated strategic form games might change when costs are added to strategy-switches of the players, we consider an example. Our payoff matrix for Prisoners' Dilemma is

	C	D
C	2,2	0,3
D	3,0	1,1

We shall use this matrix in the examples throughout this chapter.

We first show that when switching strategies involves cost, some of the classical equilibrium tuples in repeated Prisoners' Dilemma (RPD) may cease to be equilibria.

GRIM is a popular strategy in RPD. A player playing GRIM starts by co-operating and co-operates as long as her opponent co-operates. If her opponent defects first in round t, then she defects from round $t+1$ onwards for the rest of the game, irrespective of the action of her opponent. It is a relentless punishing strategy.

Consider now the strategy tuple (GRIM, GRIM). It is well known that (GRIM,GRIM) is an equilibrium tuple of the RPD. We show that it may not be an equilibrium tuple in our model of RPD where switching strategies between rounds involve a cost. To see this, suppose player 2 defects in round $t+1$. Then the sequence of strategy tuples is

$$\underbrace{(C,C),(C,C),\ldots,(C,C)}_{t},(C,D),(D,D),(D,D),\ldots$$

The sequence of utilities to player 1 is

$$\underbrace{2,2,\ldots,2}_{t},0,(1-c),1,1,\ldots$$

She receives a utility of $(1-c)$ in round $t+2$ where c is the cost incurred by switching from C to D. Her net discounted utility for a discount factor δ is

$$(1-\delta)[2+2\delta+2\delta^2+\ldots+2\delta^{t-1}+0\cdot\delta^t+(1-c)\delta^{t+1}+1\delta^{t+2}+1\delta^{t+3}+\ldots]$$

$$=2-2\delta^t+(1-c)\delta^{t+1}-(1-c)\delta^{t+2}+\delta^{t+2}$$

For this to be unprofitable for player 1 than the case where she keeps co-operating we must have

$$2-2\delta^t+(1-c)\delta^{t+1}+(1-c)\delta^{t+2}+\delta^{t+2}<(1-\delta)[2(1-\delta^t)/(1-\delta)+0]$$

which happens when

$$c>1+\delta^{t+2}/(1-\delta)\delta^{t+1}$$

Thus for a cost greater than $1+\delta^{t+2}/(1-\delta)\delta^{t+1}$ (GRIM,GRIM) no longer remains an equilibrium strategy tuple. Intuitively the cost involved in switching from co-operation to defection is too high to be compensated later by the gain in utility.

Using a similar analysis, we can show that for the strategy (TFT, TFT), where TFT stands for tit-for-tat [1], there exists a $c>0$ such that when the cost of switching strategies for the players is greater than or equal to c, it does not remain an equilibrium tuple any more.

We now show that as c increases, the players tend to stick to their current strategy for longer and longer time. To see this suppose a player initially starts by playing an action a which fetches her a payoff of 2. But after t rounds something happens (maybe the other players change their strategies) and her payoff drops to 1. She can switch to another action a' and restore

[1] It is the strategy where in a round, a player always plays the strategy played by her opponent in the previous round.

her payoff of 2. But that would involve a cost of c. Suppose she does so after k more rounds. Then her sequence of utilities is

$$\underbrace{2, 2, \ldots, 2}_{t}, \underbrace{1, 1, \ldots, 1}_{k}, (2-c), 2, 2, \ldots$$

Her net discounted utility for a discount factor of δ is

$$(1-\delta)[2(1-\delta^t)/(1-\delta) + \delta^t(1-\delta^k)/(1-\delta) + \delta^{t+k}(2-c) + 2\delta^{t+k+1}/(1-\delta)]$$

$$= [2(1-\delta^t) + \delta^t(1-\delta^k) + 2\delta^{t+k+1}] - c\delta^{t+k}(1-\delta)$$

Thus as c increases, k should also increase to keep the cost factor $c\delta^{t+k}(1-\delta)$ low. This means that the player tends to stick more and more to her current strategy as the cost of switching increases.

3.3 Cost of mixing

One common and quite well justified interpretation of mixed strategies is to view them as limiting behaviour in infinitely repeated games. In other words, if a player has pure strategy set Σ and she mixes with probabilities $\mathbf{x}$, where $\mathbf{x}(a) \geq 0,\ a \in \Sigma$ and $\sum_{a\in\Sigma} \mathbf{x}(a) = 1$, then

$$\lim_{t\to\infty} \frac{\text{number of times } a \text{ is played}}{t} = \mathbf{x}(a)$$

where t is the total number of rounds.

With this interpretation, in our formulation where switching strategies involves a cost, a mixed strategy also should involve a certain cost. This cost is the expected cost of all possible switches in the strategy.

To formalise this notion we need to fix our notations first. Suppose there are n players. Each player i has a pure strategy set Σ_i. We denote by $\mathbf{X}_i$ the set of mixed strategies of player i.

Given a tuple $\bar{\mathbf{x}} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ of mixed strategies the expected payoff of player i in any round $t+1$ is derived as follows. Since player i is playing a mixed strategy $\mathbf{x}_i$ the probability that she plays a particular strategy $a \in \Sigma_i$ in round t is $\mathbf{x}_i(a)$. The probability that player i plays a in round $t+1$ as well is again $\mathbf{x}_i(a)$ and this event is independent of her playing a in round t. Thus the probability that she plays a strategy different from a in round $t+1$ is $1-\mathbf{x}_i(a)$. Thus the probability that player i switches from a to some

other strategy a' in round $t+1$ is $\mathbf{x}_i(a)[1-\mathbf{x}_i(a)]$. Summing over all $a \in \Sigma_i$ we have the probability that she switches strategies in round $t+1$ as

$$\sum_{a\in\Sigma_i} \mathbf{x}_i(a)[1-\mathbf{x}_i(a)] = \sum_{a\in\Sigma_i} \mathbf{x}_i(a) - \sum_{a\in\Sigma_i} \mathbf{x}_i(a)^2 = 1 - \sum_{a\in\Sigma_i} \mathbf{x}_i(a)^2$$

Thus her expected cost in each round is

$$c\Big[1 - \sum_{a\in\Sigma_i} \mathbf{x}_i(a)^2\Big]$$

Hence the expected utility of player i in every round $t > 1$ is given by

$$u_i(\bar{\mathbf{x}}) = \sum_{a_1\in\Sigma_1} \dots \sum_{a_n\in\Sigma_n} p_i(a_1,\dots,a_n) \prod_{j=1}^{n} \mathbf{x}_j(a_j) - c\Big[1 - \sum_{a\in\Sigma_i} \mathbf{x}_i(a)^2\Big] \quad (3.1)$$

Note that for any pure strategy, i.e., when $\mathbf{x}_i(a) = 1$ and $\mathbf{x}_i(a') = 0$ for all $a' \neq a$, the cost factor $c[1 - \sum_{a\in\Sigma_i} \mathbf{x}_i(a)^2]$ vanishes. Also note that the cost is maximum when $\mathbf{x}_i(a) = 1/|\Sigma_i|$ for all $a \in \Sigma_i$.

For mixed strategies incurring such costs, equilibrium tuples in a strategic form game may no longer remain so if the game is repeated infinitely. Consider, for instance, the game of matching pennies

	H	T
H	1,-1	-1,1
T	-1,1	1,-1

Both players mixing with probabilities $(0.5, 0.5)$ is a mixed strategy equilibrium for this game and the expected payoff is 0. But what if the players play this strategy repeatedly? For a cost c, the expected utility of each player is

$$0 - c[1 - \{(0.5)^2 + (0.5)^2\}] = 0 - c[1 - 0.5] = -0.5c$$

Thus her net discounted utility for a discount factor of δ is

$$(1-\delta)[0 + \delta(-0.5c) + \delta^2(-0.5c) + \dots]$$

$$= -0.5c\delta$$

For this to be unprofitable for a player than her worst pure strategy we must have

$$-0.5c\delta < -1$$

which implies

$$c > 2/\delta$$

Thus for any value of the cost c which is greater than $2/\delta$, mixing with probability $(0.5, 0.5)$ is no longer an equilibrium in the repeated game.

But can an equilibrium tuple $\mathbf{u}$ of the strategic form game be attained in the repeated game at all? We answer this and other questions in the next section.

3.4 Main results

Proposition 3.1 *A strategic form game $\mathcal{G}$ repeated infinitely where switching strategies is associated with a cost and where the payoffs are given by Equation 3.1 may not have an equilibrium.*

Proof Consider the game of matching pennies

	H	T
H	1,-1	-1,1
T	1,-1	11,1

Suppose player 1 mixes with probabilities x and $1-x$ while player 2 mixes with probabilities y and $1-y$. The utility of player 1 is given by Equation 3.1 as

$$xy - x(1-y) + y(1-x) + (1-x)(1-y) - c[1 - \{x^2 + (1-x)^2]$$

which we denote by X and is equal to

$$X = 4xy - 2x - 2y + 1 - 2cx + 2cx^2$$

Similarly the utility of player 2, denoted by Y is given as

$$Y = -4xy + 2x + 2y - 1 - 2cy + 2cy^2$$

These surfaces are shown in Figure 3.1 as functions of x and y for $c = 10$.

Differentiating X w.r.t x for fixed y we have

$$\frac{\partial X}{\partial x} = 4y - 2 - 2c + 4cx$$

Differentiating again we have

$$\frac{\partial^2 X}{\partial x^2} = 4c > 0$$

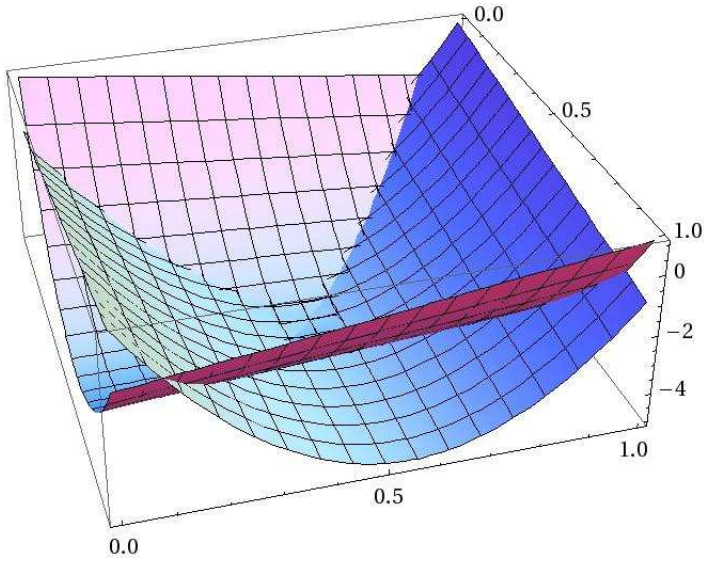

Figure 3.1: The payoff curves of players 1 and 2 for $c = 10$

And similarly for Y when differentiated w.r.t y keeping x fixed. These surfaces are concave and the maxima occur at the boundaries. Thus for any interior point $(x, y),\ 0 < x < 1,\ 0 < y < 1$, either of the players always has an incentive to deviate towards one of the boundaries.

But what about the boundary points? Are they in equilibrium? For $x = 0, 1 < y < 1$ or $x = 1, 0 < y < 1$, player 2 always has an incentive to deviate because of the same concavity of the surface Y. Similarly for $y = 0, 0 < x < 1$ or $y = 1, 0 < x < 1$. player 1 has an incentive to deviate. Thus the only points left to consider are the corner points $(0, 0), (0, 1), (1, 0), (1, 1)$. We look at what happens at $(0, 0)$. The other points are symmetrical.

At $x = 0, y = 0,\ X = 1, Y = -1$. But then player 2 has an incentive to deviate to $y = 1$ because at $y = 1, Y = 1$.

Actually, when we consider only the corner points, the game is just matching pennies without any costs. And we already know that this game does not have a pure strategy equilibrium.

This proves the proposition. □

3.4.1 Pre-play agreement

We now show that if the players are allowed to negotiate before the game starts, and if everyone sticks to her promise, then any ϵ-Nash equilibrium tuple of the strategic form game can be attained in the repeated game for $\epsilon > 0$. The idea is to delay the first switch for as long as it takes to bring down the cumulative discounted cost below ϵ. Pre-play agreement is required so that the other players do not punish her for doing so.

Theorem 3.2 *For any $\epsilon > 0$ an ϵ-Nash equilibrium of a strategic form game $\mathcal{G}$ can be achieved when the game $\mathcal{G}$ is repeated infinitely often and where the players incur a cost of $c > 0$ for every strategy switch provided pre-play agreements are allowed and the players stick to such agreements.*

Proof Let $\mathbf{p} = (p_1, \ldots, p_n)$ be an equilibrium payoff tuple of $\mathcal{G}$ and let $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ be the mixed strategy tuple that achieves $\mathbf{p}$.

For every i, for the cumulative cost of switching to be less than ϵ, the first switch by player i should occur at least after k_i rounds where k_i is such that

$$c\delta^{k_i} < \epsilon$$

and $\delta :\ 0 < \delta < 1$ is the discount factor. This implies

$$k_i > \frac{\log \epsilon - \log c}{\log \delta}$$

The values of k_i for every player i are the quantities that every player must agree upon during the pre-play negotiation.

Thus if each player i played strategy $a \in \Sigma_i$ in round 0, she should play a for at least k_i consecutive rounds and then randomise according to a public randomisation device ω such that $\bar{\mathbf{x}}(\omega) = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$. Since $(\lim_{t\to\infty} k_i/t) \to 0$ for every i, the stated strategy profile achieves the required ϵ-Nash equilibrium. □

3.4.2 A folk theorem

Let $\mathcal{G}$ be a strategic form game with n players where each player i has a strategy set Σ_i.

Definition 3.3 *We define the following quantities:*

(i) Let V be the convex linear combination of the pure strategy payoff vectors. That is, $V =$ convex hull $\{p(\mathbf{a}) \mid \mathbf{a} \in \prod_{i=1}^{n} \Sigma_i\}$.

(ii) Player i's reservation payoff $\underline{p}_i = \min_{\mathbf{a}_{-i} \in \Sigma_{-i}} \{\max_{a_i \in \Sigma_i} p_i(a_i, \mathbf{a}_{-i})\}$. *Let* $\mathbf{m}_i \in \Sigma_{-i}$ *be the profile of the other players that realises* $\underline{p}_i$.

(iii) Let U *be the set of feasible strictly individually rational payoffs defined as* $U = \{\mathbf{v} \in V \mid \forall i,\ \mathbf{v}(i) > \underline{p}_i\}$.

(iv) Given a cost $c \geq 0$ *define the set* C *as* $C = \{\mathbf{v} \mid \mathbf{v}(i) = u_i(\mathbf{x}),\ \mathbf{x} \in \mathbf{X}_1 \times \ldots \times \mathbf{X}_n\}$ *where* $u_i(\mathbf{x})$ *is given by Equation 3.1.*

We shall show that if the players are patient enough, any payoff vector in the set $C \cap U$ can be supported when the game $\mathcal{G}$ is repeated infinitely.

But first we see how the sets C, U and V defined above look like. Figures 3.2 and 3.3 below show these sets for the RPD for different values of the cost c. Note that C is not a convex set and hence $C \cap U$ is not convex either. Also $C \cap U \subseteq C \cap V$ is never empty as it always contains the payoff vectors corresponding to the pure strategy tuples.

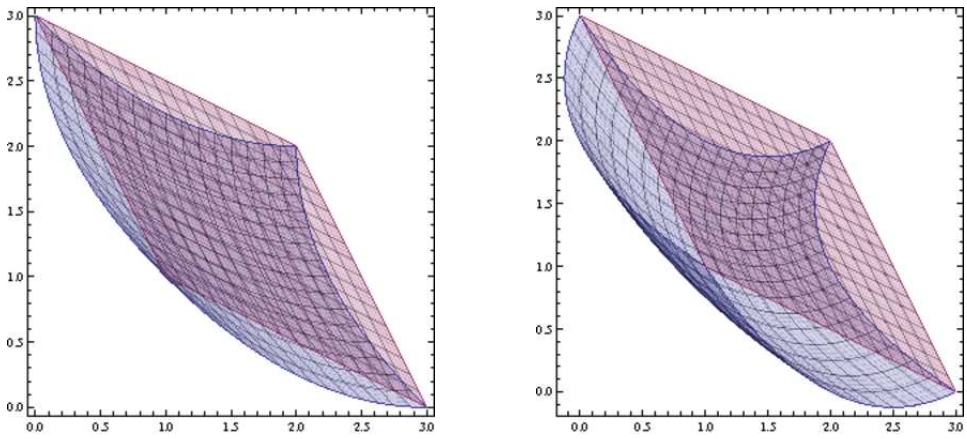

Figure 3.2: The sets V, U and C for $c = 0.5$ and 1 respectively from left to right

Theorem 3.4 (folk theorem) *For every* $\mathbf{v} \in C \cap U$, *there exists a* $\underline{\delta} < 1$ *such that for all* $\delta \in (\underline{\delta}, 1)$ *there is a Nash equilibrium of the repeated game* $\mathcal{G}$ *with switching costs having payoffs* $\mathbf{v}$.

The proof constructs strategies that are relentless. A player who deviates from her prescribed strategy is minimaxed forever. We use the following notation in the proof. A randomisation device (a coin, a die etc.) is called public when its outcome is visible to all the players in the game. When the

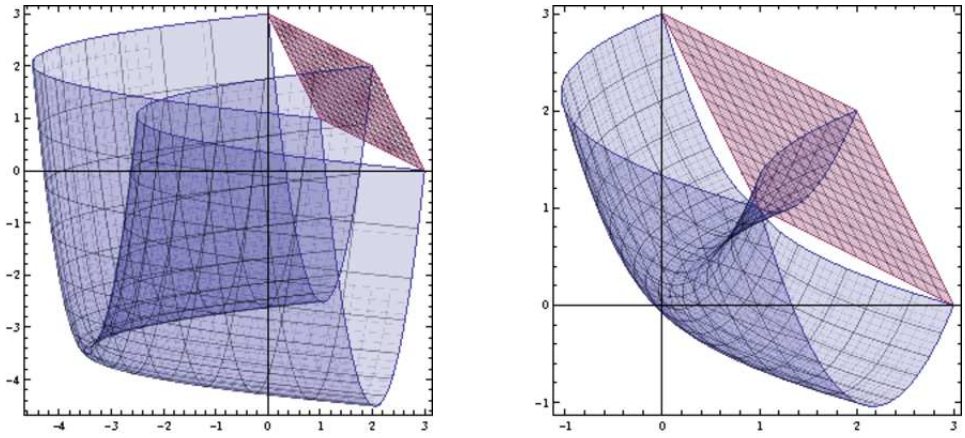

Figure 3.3: The sets V, U and C for $c = 3$ and 10 respectively from left to right

outcome ω of such a device is used to play a mixed strategy, we denote the mixed strategy by $\mathbf{x}^\omega$.

Proof Let $u(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \mathbf{v}$. Such an $\bar{\mathbf{x}} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ exists by the definition of $C \cap U$.

Consider the following strategy for each player i: Play $\mathbf{x}_i$ in round 1 and continue to play $\mathbf{x}_i$ as long as (i) the realised strategy profile in the previous round was $\bar{\mathbf{x}}$ or (ii) the realised strategy profile in the previous round differed from $\bar{\mathbf{x}}$ in two or more components. If in some previous round, player i was the only player not to follow $\bar{\mathbf{x}}$, then each player j plays $\mathbf{m}_i(j)$ for the rest of the game.

Suppose in round t a public randomisation device outputs ω but player i deviates from $\mathbf{x}_i^\omega$. Then she obtains at most

$$(1-\delta^t)\mathbf{v}(i) + \delta^t(1-\delta)\max_{\mathbf{a}} p_i(\mathbf{a}) + \delta^{t+1}\underline{p}_i$$

For this to be unprofitable, δ should be so high that this payoff is strictly smaller than the payoff from sticking to the strategy prescribed by ω. That is,

$$(1-\delta^t)\mathbf{v}(i)+\delta^t(1-\delta)\max_{\mathbf{a}} p_i(\mathbf{a})+\delta^{t+1}\underline{p}_i < (1-\delta^t)\mathbf{v}(i)+\delta^t(1-\delta)u_i(\mathbf{a}^\omega)+\delta^{t+1}\mathbf{v}(i)$$

Now since ω might prescribe the worst possible strategy to player i, we have:

$$\delta^t(1-\delta)\max_{\mathbf{a}} p_i(\mathbf{a}) + \delta^{t+1}\underline{p}_i < \delta^t(1-\delta)\min_{\mathbf{a}} p_i(\mathbf{a}) + \delta^{t+1}\mathbf{v}(i)$$

which implies

$$(1-\delta)\max_{\mathbf{a}} p_i(\mathbf{a}) + \delta^t \underline{p}_i < (1-\delta)\min_{\mathbf{a}} p_i(\mathbf{a}) + \delta^t \mathbf{v}(i)$$

For each player i define the critical level $\underline{\delta}_i$ to be the solution of the equation

$$(1-\underline{\delta}_i)\max_{\mathbf{a}} p_i(\mathbf{a}) + \underline{\delta}_i^t \underline{p}_i = (1-\underline{\delta}_i)\min_{\mathbf{a}} p_i(\mathbf{a}) + \underline{\delta}_i^t \mathbf{v}(i)$$

Since $\underline{p}_i \leq \mathbf{v}(i)$ and $\max_{\mathbf{a}} p_i(\mathbf{a}) \geq \min_{\mathbf{a}} p_i(\mathbf{a})$, the solution to this equation always exists with $0 < \underline{\delta}_i < 1$. Take $\underline{\delta} = \max_i \underline{\delta}_i$. □

We thus see that with the modified notion of individually rational payoffs, taking into account the cost of strategy switching, any tuple of such payoffs is achievable provided the players are patient enough.

Chapter 4

Specifying strategy switches

In Chapter 2 we studied games where the players are restricted to use strategies from a fixed finite set S. A question one may ask at this point is how this set S is presented. Although S is finite, it may be a humongous object. We need a systematic way to present the set S. Secondly, so far we have only looked at composing strategies by switching between them. We mentioned in the introduction of Chapter 2 that there may be other ways of composing strategies viz., chosing between two strategies at the beginning of the game and playing accordingly, observing the outcome of the game so far and then employing a certain strategy and so on. We need a way to formally describe such composition of strategies. Logic comes to our rescue at this point. Using logic, we can specify the set S using formulae from the logical syntax and also compose strategies in a logical and intuitive manner.

In this chapter we study strategies that are specified logically. We introduce a logic for a process-like notion of strategy in which switching strategies by players and the rationale for such switching may be specified and structurally composed. However, unlike Chapter 2, we do not deal with complexity theoretic issues here. Rather, we study a more fundamental question of game theory - the stability question: given a game arena and strategy specifications, whether a particular objective is eventually attained if the players play according to these specifications and if so then whether players settle down to strategies which do not involve any further switching.

We further introduce the notion of probabilistic switching and a similar syntax for specifying such switching. We then show how to analyse eventual behaviour and outcome in such a model. This chapter is loosely based on the paper [PRS09b].

4.1 Overview

In an ideal world, once a game is completely specified, along with the players' preferences, we know what we can (or cannot) predict about rational play, and hence actual plays and strategies followed by players are not especially interesting in themselves. The situation is entirely different when players' abilities to strategise or to consider possible futures is limited. In this case players form partial plans, make observations as play progresses and extend or revise their plans dynamically. The process is epistemic, as each player is aware that other players are also dynamically updating or refining their strategies, and such strategising is mutually dependent [vB07]. In such situations, strategies are structured, much like processes and are (de)composed similarly. Moreover, all strategies are not equal in some sense: partial plans that make sense in the early stages of play may be ruled out by rational play later on.

Consider the game of cricket[1]. A bowler, starting on his run-up, considers: Should I bowl on the batsman's off-side or leg-side? Should I bowl a short-pitch ball? Should I bowl a slower one? Since he mis-hit the last bouncer I bowled to him, should I bowl one again? The batsman, on his part, considers as he takes his stance: If he bowls on my legs, should I pelt him for a boundary and reveal my strength off that flank? Or should I play it safe and settle for a single? I have already hit two boundaries in this over; if I hit him for too many runs, will he be taken off the attack?

In an ideal world, both bowler and batsman would have perfect information not only about each other's prowess but also about the nature of the pitch, and would play optimal mixed strategies, since they could go through all the reasoning above before a single ball is ever bowled. We could compute equilibria and predict rational cricket play.

Not only is the actual game far from ideal, it is also more interesting. If we are interested in predicting, in addition to outcomes, also how the play is likely to progress (at some partial play), we need to correspondingly look not just at which strategies are available to players, but also how they select a strategy from among many. Such considerations naturally lead to partial strategies, and the notion of switching between (partial) strategies.

In such a view, a player enters the game arena with information on the game structure and on other players' skills, as well as an initial set of possible strategies to employ. As the play progresses, she makes observations and

[1]Wikipedia-level understanding of cricket http://en.wikipedia.org/wiki/Cricket is enough to understand the points being made here, though some knowledge of cricket would surely help.

accordingly revises strategies, switches from one to another, perhaps even devises new strategies that she hadn't considered before. The dynamics of such interaction eventually leads to some strategies being eliminated, and some becoming stable.

Such considerations can be entirely eliminated by taking into account all possible futures while strategising. However, such omniscient strategising may be impossible, even in principle, for finitary agents (who have access only to finite resources). Dynamical system models of social interaction and negotiations have for long considered such switching behaviour ([SP00], [Hor05]). We would like to study such switching behaviour from a logical and computational perspective.

We ask the stability question of such a model: "Does the play finally settle down to some subset of the entire arena?", "Can a player ensure certain objectives using a strategy which does not necessitate switching between several strategies?" Such questions are especially relevant in the context of bargaining and negotiations, as evidenced in many political contexts.

4.2 Preliminaries

In Section 1.2 we introduced the notions of game arenas, unfoldings, partial and total strategies and strategy-trees. The models we work with in this chapter are concurrent-move game arenas. However, the entire analysis goes through for turn-based arenas as well. First we need to define and develop a few more notions in addition to those defined in Section 1.2.

4.2.1 Strategy for a specified history

Let $\mathcal{A} = (V, E)$ be an arena. For a vertex $v_0 \in V$ and a play ρ in the arena starting at v_0, let $H[\rho] \subseteq H$ be the set of plays with prefix ρ. That is a play $\rho' \in H[\rho]$ if and only if ρ is a prefix of ρ'. A partial strategy with history ρ, denoted $s[\rho]$ is a partial function:

$$s[\rho] : H[\rho] \rightharpoonup A_i.$$

such that for every play ρ' with prefix ρ, $s[\rho](\rho') \in \Gamma_i(last(\rho'))$. The strategy tree $\mathcal{T}_{\mathcal{A}}^{s[\rho]}$ of $s[\rho]$ has root $t_0 = (last(\rho), \mathbf{u}(\rho))$ and is defined inductively similar to the description given in Section 1.2.7.

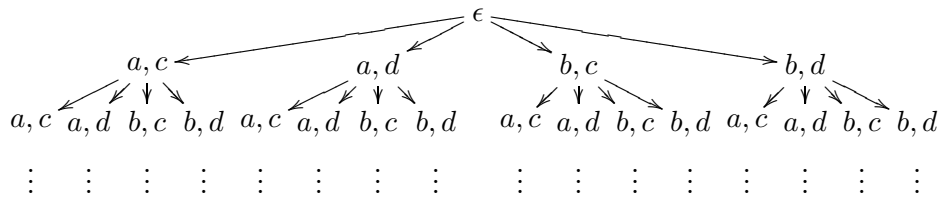

Figure 4.1: Extensive form game tree

4.2.2 Composition of strategies

Let $\mathcal{A} = (V, E)$ be an arena, $v_0 \in V$ be an initial vertex and let s_1 and s_2 be two strategies of player i. Suppose player i starts playing with strategy s_1 and after k rounds ($k \geq 0$), she decides to use the strategy s_2 for the rest of the game. The resulting prescription, which we denote by ${s_1}^k s_2$, is also a strategy. It may be viewed as a composition of the strategies s_1 and s_2.

Definition 4.1 *The strategy tree $\mathcal{T}_{\mathcal{A}}^{{s_1}^k s_2}$ is derived from $\mathcal{T}_{\mathcal{A}}^{s_1} = (T_1, E_1)$ and $\mathcal{T}_{\mathcal{A}}^{s_2}$ as follows. First prune $\mathcal{T}_{\mathcal{A}}^{s_1}$ at depth k and call the resulting (finite) tree $\mathcal{T}_{\mathcal{A},k}^{s_1}$. Then to every leaf vertex $t = (v, \mathbf{u})$ of $\mathcal{T}_{\mathcal{A},k}^{s_1}$, append the strategy tree $\mathcal{T}_{\mathcal{A}}^{s_2[\rho(v_0,\mathbf{u})]}$ of $s_2[\rho(v_0, \mathbf{u})]$.*

Example 4.2 *Recall the extensive form game of Example 1.1 (shown again in Figure 4.1). Let s_a be the strategy of player 1 which prescribes her to play a for every move and let s_b be the one which prescribes her to play b for every move. Then ${s_a}^2 s_b$ is the strategy which prescribes her to play a for the first two moves and then to switch to playing b's. The strategy trees are shown in Figure 4.2.*

This operation can be lifted to sets of strategies in the usual way. Let Σ_1 and Σ_2 be sets of strategies. Then $\Sigma_1{}^k\Sigma_2 = \{{s_1}^k s_2 \mid s_1 \in \Sigma_1 \text{ and } s_2 \in \Sigma_2\}$.

4.2.3 Back and forth between partial and total strategies

Let $\mathcal{A}$ be an arena. A partial strategy may be viewed as a set of (total) strategies. Given the strategy tree $\mathcal{T}_{\mathcal{A}}^{s}$ for a partial strategy s of player i we obtain a set of trees $Tot_i(\mathcal{T}_{\mathcal{A}}^{s})$ of total strategies of player i as follows. $\mathcal{T} = (T, E) \in Tot_i(\mathcal{T}_{\mathcal{A}}^{s})$ if and only if the root of $\mathcal{T}$ is the same as that of $\mathcal{T}_{\mathcal{A}}^{s}$ and

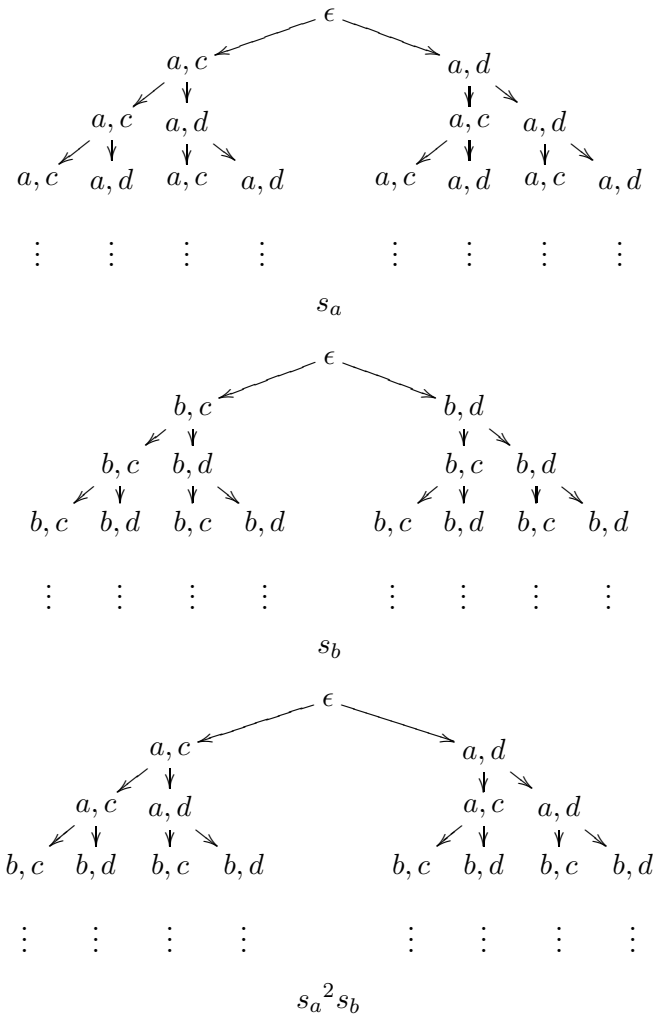

Figure 4.2: Composition of s_a and s_b to get ${s_a}^2 s_b$

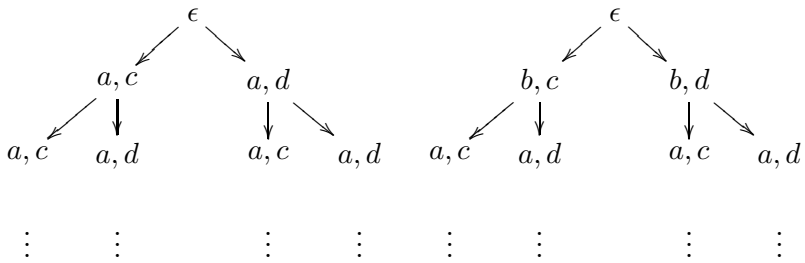

Figure 4.3: Partial strategy to total strategies

- If $(v, \mathbf{u}) \in \mathcal{T}$ then $(v', \mathbf{ua}) \in \mathcal{T}$ is a child of $(v, \mathbf{u})$ if and only if $(v', \mathbf{ua}) \in \mathcal{T}^s_{\mathcal{A}}$ and for all children $(v_1, \mathbf{ua}_1)$, $(v_2.\mathbf{ua}_2)$ of $(v, \mathbf{u})$ we have $\mathbf{a}_1(i) = \mathbf{a}_2(i)$.

For any history ρ, the set $Tot_i(\mathcal{T}^{s[\rho]}_{\mathcal{A}})$ of (total) strategy trees for the partial strategy $s[\rho]$ of player i may be defined similarly.

Example 4.3 *The partial strategy s'_1 for player 1 of Example 1.4, which is undefined at the empty history but prescribes her to play the action a for all successive histories corresponds to the two total strategies as shown in Figure 4.3.*

It is also convenient to define a reverse map $\mathcal{TP}_i$ for every player i.

Definition 4.4 *$\mathcal{TP}_i$ for every player $i \in N$ are defined as:*

- *Given a set $\mathfrak{T}$ of (total) strategy trees of player i for history ρ, $\mathcal{TP}_i(\mathfrak{T})$ is the partial strategy tree (T, E) with root $(last(\rho), \mathbf{u}(\rho))$ such that*
  - *$t \in T$ if and only if $t \in \mathcal{T}$ for some $\mathcal{T} \in \mathfrak{T}$ and*
  - *t' is a child of t in (T, E) if and only if t' is a child of t in some tree $\mathcal{T} \in \mathfrak{T}$.*

Example 4.5 *If we apply $\mathcal{TP}_i$ to the two total strategies of Example 4.3, we get back the partial strategy of Example 1.4 (shown again in Figure 4.4).*

Remark The notion of composition of strategies is also present in the theory of game semantics of programs, a work pioneered by Abramsky and Jagadeesan and independently by Hyland and Ong [AJ94, HO94]. There a

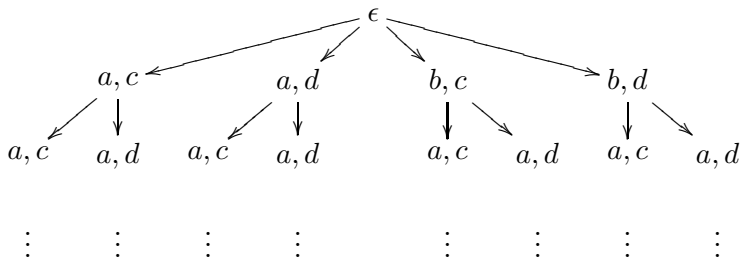

Figure 4.4: Partial strategy of Example 1.4

program is viewed as a game between a process and a context. Composition of programs thus naturally leads to the composition of strategies of the process and the context in these programs. See, for eg., Russ Harmer's Ph.D thesis for a beautiful survey of the results on game semantics.

4.3 Strategy specifications

We present a syntax to specify partial strategies and their composition in a structural manner. A player i 'strategises', that is, composes several strategies to build new strategies, based on her observation of the outcomes of the game so far.

4.3.1 Syntax

By an atomic strategy of player i we mean a strategy which dictates her to play the same action at all positions. We denote atomic strategies by $s_a,\ a \in A_i$.

The strategy set Σ_i of player i is obtained by combining her moves using a number of operators. Σ_i is defined as

$$\Sigma_i ::= \phi \mid s_a,\ a \in A_i \mid s_1 \cup s_2 \mid s_1{}^\frown s_2 \mid (s_1 + s_2) \mid \varphi?s$$

where $\varphi \in \Phi$ [Section 1.2.9].

We now give the intuitive meanings followed by a formal semantics of the strategy building operators. The intuitive meaning of the operators are given as:

- ϕ is the 'null' strategy which is undefined everywhere. In other words, ϕ is the strategy in which player i plays any move at any position.

- $s_a,\ a \in A_i$ is the atomic strategy where player i plays the action a at each move.

- $s_1 \cup s_2$ means that the player plays according to the strategy s_1 or the strategy s_2.

- $s_1{}^\frown s_2$ means that the player plays according to the strategy s_1 and then after some history, switches to playing according to s_2. The position at which she makes the switch is not fixed in advance.

- $(s_1 + s_2)$ says that at every point, the player can choose to follow either s_1 or s_2.

- $\varphi?s$ says at every history, the player tests if the property φ holds of that history. If it does then she plays according to s.

Remark Note that in the syntax of Φ, we do not have the corresponding indefinite future time operator $\Diamond$. This is because, as mentioned earlier, according to our view, players in the real world have bounded memory and hence are unable to strategise about the entire future. But, players do strategise based on their expectations about some bounded amount of future, and hence the operator $\langle \mathbf{a} \rangle^+ \varphi$.

Examples of strategy specifications

In the cricket example, let the bowler's set of atomic strategies be given as $\Sigma_{bowler} = \{\sigma_{short}, \sigma_{good}, \sigma_{outside-off}, \sigma_{legs}\}, \sigma_{bouncer}$ which corresponds to bowling a short-pitch, good length, off-side, leg-side and a bouncer ball respectively.

Let $p_{(short,sixer)}$ be the observable which says that the outcome of a short ball is a sixer. Then the following specification says that the bowler keeps bowling short balls or bouncers till he is hit for a sixer.

- $\neg \Diamond p_{(short,sixer)}?(\sigma_{short} \cup \sigma_{bouncer})$

The specification $\sigma_{short}{}^\frown \sigma_{legs}{}^\frown \sigma_{good}$ for the bowler says that he starts by bowling short-pitch balls and after some point he switches to bowling at the batsman's legs and again switches to bowling good-length balls.

4.3.2 Semantics

We now give the formal semantics of the strategy specifications. Let $\mathcal{A}$ be an arena, v_0 be an initial node and $\mathcal{T}_\mathcal{A} = (T, E)$ be the unfolding of $\mathcal{A}$. Let $\mathfrak{T}_i$ denote the set of total strategy trees of player i. The semantics of a strategy specification $s \in \Sigma_i$ is a function $[\![\cdot]\!]_\mathcal{A} : \Sigma_i \times T \to 2^{\mathfrak{T}_i}$. That is, each specification at a vertex $t = (v, \mathbf{u})$ of the tree $\mathcal{T}_\mathcal{A}$ is associated with a set of total strategy trees after history $\rho(v_0, \mathbf{u})$.

For any $t = (v, \mathbf{u}) \in T$, $[\![\cdot]\!]_\mathcal{A}$ is defined inductively as follows:

- $[\![\phi, t]\!]_\mathcal{A} = Tot_i(\mathcal{T}_\mathcal{A}^{\rho(v_0,\mathbf{u})})$ where $\mathcal{T}_\mathcal{A}^{\rho(v_0,\mathbf{u})}$ is the play $\rho(v_0, \mathbf{u})$ followed by the subtree of $\mathcal{T}_\mathcal{A}$ starting at $\rho(v_0, \mathbf{u})$.

- $[\![s_a, t]\!]_\mathcal{A} = Tot_i(\mathcal{T}_\mathcal{A}^{s_a[\rho(v_0,\mathbf{u})]})$ where $s_a[\rho(v_0, \mathbf{u})](\rho') = a$ for all plays ρ' in $\mathcal{A}$ such that $\rho(v_0, \mathbf{u})$ is a prefix of ρ' and $a \in \Gamma_i(last(\rho))$. $s_a[\rho(v_0, \mathbf{u})](\rho')$ is undefined otherwise.

- $[\![s_1 \cup s_2, t]\!]_\mathcal{A} = [\![s_1, t]\!]_\mathcal{A} \cup [\![s_2, t]\!]_\mathcal{A}$.

- Let $|\mathbf{u}| = l$. $[\![s_1{}^\frown s_2, t]\!]_\mathcal{A} = \bigcup_{k \geq l}([\![s_1, t]\!]_\mathcal{A}{}^k [\![s_2, t]\!]_\mathcal{A})$.

- $[\![(s_1 + s_2), t]\!]_\mathcal{A} = \bigcup_{k_1,k_2,\ldots}(([\![s_1, t]\!]_\mathcal{A}{}^{k_1} [\![s_2, t]\!]_\mathcal{A})^{k_2} [\![s_1, t]\!]_\mathcal{A}) \cdots$ where $k < k_1 < k_2 \ldots$.

- $[\![\psi?s, t]\!]_\mathcal{A}$ is obtained from $[\![s, t]\!]_\mathcal{A}$ and $\mathcal{T}_\mathcal{A}$ as follows. Let $\mathcal{TP}_i([\![s, t]\!]_\mathcal{A}) = \mathcal{T}_\mathcal{A}^{s[\rho(v_0,\mathbf{u})]}$ be the partial strategy tree of $s[\rho(v_0, \mathbf{u})]$. Then $[\![\psi?s, t]\!]_\mathcal{A}$ is a set of trees such that for every tree $\mathcal{T} \in [\![\psi?s, t]\!]_\mathcal{A}$ and for every node $t' = (v, \mathbf{u}) \in \mathcal{T}$, if $t' \models \psi$ then the children of t' are those that correspond to the strategy s. The consistency condition is always maintained, in that, the moves played on all the branches should be the same. Formally, $\mathcal{T} \in [\![\psi?s, t]\!]_\mathcal{A}$ if and only if:

  - $t \in \mathcal{T}$ is the root.
  - If $t' = (v', \mathbf{u}') \in \mathcal{T}$ and $t' \models \psi$ then $t'' = (v'', \mathbf{u}'\mathbf{a}) \in \mathcal{T}$ if and only if $t'' \in \mathcal{T}_\mathcal{A}^{s[\rho(v_0,\mathbf{u})]}$ and for all $t_1 = (v_1, \mathbf{u}'\mathbf{a}_1)$ and $t_2 = (v_2, \mathbf{u}'\mathbf{a}_2)$ in $\mathcal{T}$, $\mathbf{a}_1(i) = \mathbf{a}_2(i)$. If $t' = (v', \mathbf{u}') \in \mathcal{T}$ and $t' \not\models \psi$ then $t'' = (v'', \mathbf{u}'\mathbf{a}) \in \mathcal{T}$ if and only if $t'' \in \mathcal{T}_\mathcal{A}$ and for all $t_1 = (v_1, \mathbf{u}'\mathbf{a}_1)$ and $t_2 = (v_2, \mathbf{u}'\mathbf{a}_2)$ in $\mathcal{T}$, $\mathbf{a}_1(i) = \mathbf{a}_2(i)$.

4.4 Transducer lemma

In this section we state and prove a lemma which ties strategy specifications to finite state transducers. Thus given strategy specifications of the players, we can represent them using finite state transducers. These transducers can then be used to answer questions regarding the eventual behaviour of the players and the stability of the game.

Lemma 4.6 *Given an initialised arena $(\mathcal{A}, v_0)$, a player i and a strategy specification $s \in \Sigma_i$, we can construct a finite state transducer $\mathcal{Q}_s$ such that for all $\mathcal{T} \in \tau^i_{\mathcal{A}}$ we have $\mathcal{T} \in [\![s, t_0]\!]_{\mathcal{A}}$ iff $\mathcal{T} \in \mathcal{L}(\mathcal{Q}_s)$, where $t_0 = (v_0, \epsilon)$.*

Proof
Step 1: Given a strategy specification $s \in \Sigma_i$ of player i, we inductively construct a transducer $\mathcal{Q}_s$, with input alphabet $\mathbf{A}$ and output alphabet A_i.

As the transducers we construct run on the arena $\mathcal{A} = (V, E)$ and the set of vertices V come with a valuation on the observables $\mathcal{P}$, we require that the states of our transducers also have valuations on these observables. We inductively build this valuation into the states of the transducers along with their constructions. Thus a transducer for us is now a tuple $\mathcal{Q} = (Q, I, \delta, f, val)$ where Q, I, δ and f are as before and $val : Q \to 2^{\mathcal{P}}$ is a valuation function on the states. We now proceed with the inductive construction.

- $s \equiv \phi$: $\mathcal{Q}_s = (Q, I, \delta, f, val)$ where
  - $Q = A_i \times 2^{\mathcal{P}}$
  - $I = Q$
  - $\delta = Q \times \prod_{i \in N} A_i \times Q$
  - $f(a, X) = a$
  - $val(a, X) = X$.
- $s \equiv s_a,\ a \in A_i$: $\mathcal{Q}_s = (Q, I, \delta, f, val)$ where
  - $Q = 2^{\mathcal{P}}$
  - $I = Q$
  - $\delta = \{(q, \mathbf{a}, q') \mid q, q' \in Q, \mathbf{a} \in \mathbf{A}\}$
  - $f(q) = a$ for all $q \in Q$ and
  - $val(q) = q$.

- $s \equiv s_1 \cup s_2$: The transducer $\mathcal{Q}_s$ for s is a union of the transducers $\mathcal{Q}_{s_1}$ and $\mathcal{Q}_{s_2}$. Let $\mathcal{Q}_{s_1} = (Q_1, I_1, \delta_1, f_1, val_1)$ and $\mathcal{Q}_{s_2} = (Q_2, I_2, \delta_2, f_2, val_2)$. Then $\mathcal{Q}_s = (Q, I, \delta, f, val)$ where
  - $Q = Q_1 \cup Q_2$
  - $I = I_1 \cup I_2$
  - $\delta = \delta_1 \cup \delta_2$
  - $f(q) = \begin{cases} f_1(q_1) & \text{if } q \in Q_1 \\ f_2(q_2) & \text{if } q \in Q_2 \end{cases}$
  - $val(q) = \begin{cases} val_1(q_1) & \text{if } q \in Q_1 \\ val_2(q_2) & \text{if } q \in Q_2. \end{cases}$
- $s \equiv s_1 ^\frown s_2$: The transducer $\mathcal{Q}_s$ for s is a product of the transducers $\mathcal{Q}_{s_1}$ and $\mathcal{Q}_{s_2}$. $\mathcal{Q}_s$ initially starts by outputting the output of $\mathcal{Q}_{s_1}$ and then non-deterministically switches to that of $\mathcal{Q}_{s_2}$. Let $\mathcal{Q}_{s_1} = (Q_1, I_1, \delta_1, f_1, val_1)$ and $\mathcal{Q}_{s_2} = (Q_2, I_2, \delta_2, f_2, val_2)$. Then $\mathcal{Q}_s = (Q, I, \delta, f, val)$ where
  - $Q \subseteq Q_1 \times Q_2 \times \{1, 2\}$ such that $\{(q_1, q_2)\} \times \{1, 2\} \in Q$ if and only if $val_1(q_1) = val_2(q_2)$
  - $I = (I_1 \times I_2 \times \{1\}) \cap Q$
  - $\begin{aligned} \delta = \; & \{((q_1, q_2, 1), \mathbf{a}, (q_1', q_2', 1)) \mid (q_1, \mathbf{a}, q_1') \in \delta_1, (q_2, \mathbf{a}, q_2') \in \delta_2\} \\ & \cup \{((q_1, q_2, 1), \mathbf{a}, (q_1', q_2', 2)) \mid (q_1, \mathbf{a}, q_1') \in \delta_1, (q_2, \mathbf{a}, q_2') \in \delta_2\} \\ & \cup \{((q_1, q_2, 2), \mathbf{a}, (q_1', q_2', 2)) \mid (q_1, \mathbf{a}, q_1') \in \delta_1, (q_2, \mathbf{a}, q_2') \in \delta_2\} \end{aligned}$
  - $f(q_1, q_2, 1) = f_1(q_1), \;\; f(q_1, q_2, 2) = f_2(q_2)$
  - $val(q_1, q_2, 1) = val(q_1, q_2, 2) = val_1(q_1) = val_2(q_2)$.
- $s \equiv (s_1 + s_2)$: The transducer $\mathcal{Q}_s$ for s is again a product of the transducers $\mathcal{Q}_{s_1}$ and $\mathcal{Q}_{s_2}$. $\mathcal{Q}_s$ at every step non-deterministically chooses to output the output of either $\mathcal{Q}_{s_1}$ or $\mathcal{Q}_{s_2}$. Formally, let $\mathcal{Q}_{s_1} = (Q_1, I_1, \delta_1, f_1, val_1)$ and $\mathcal{Q}_{s_2} = (Q_2, I_2, \delta_2, f_2, val_2)$. Then $\mathcal{Q}_s = (Q, I, \delta, f, val)$ where
  - $Q \subseteq Q_1 \times Q_2 \times \{1, 2\}$ such that $\{(q_1, q_2)\} \times \{1, 2\} \in Q$ if and only if $val_1(q_1) = val_2(q_2)$
  - $I = (I_1 \times I_2 \times \{1, 2\}) \cap Q$
  - $\begin{aligned} \delta = \; & \{((q_1, q_2, 1), \mathbf{a}, (q_1', q_2', 1)) \mid (q_1, \mathbf{a}, q_1') \in \delta_1, (q_2, \mathbf{a}, q_2') \in \delta_2\} \\ & \cup \{((q_1, q_2, 1), \mathbf{a}, (q_1', q_2', 2)) \mid (q_1, \mathbf{a}, q_1') \in \delta_1, (q_2, \mathbf{a}, q_2') \in \delta_2\} \\ & \cup \{((q_1, q_2, 2), \mathbf{a}, (q_1', q_2', 2)) \mid (q_1, \mathbf{a}, q_1') \in \delta_1, (q_2, \mathbf{a}, q_2') \in \delta_2\} \\ & \cup \{((q_1, q_2, 2), \mathbf{a}, (q_1', q_2', 1)) \mid (q_1, \mathbf{a}, q_1') \in \delta_1, (q_2, \mathbf{a}, q_2') \in \delta_2\} \end{aligned}$

- $f(q_1, q_2, 1) = f_1(q_1),\ f(q_1, q_2, 2) = f_2(q_2)$
- $val(q_1, q_2, 1) = val(q_1, q_2, 2) = val_1(q_1) = val_2(q_2)$.

- $s \equiv \varphi?s'$: Let $\mathcal{Q}_{s'} = (Q', I', \delta', f', val')$, $G_\varphi = (V_\varphi, E_\varphi)$ be the atom graph of φ. Then $\mathcal{Q}_s = (Q, I, \delta, f, val)$ is constructed from $\mathcal{Q}_{s'}$ and G_φ as follows
  - $Q \subseteq Q' \times AT(\varphi) \times A_i$ such that $(q, C, a) \in Q$ iff $val'(q) = C \cap \mathcal{P}$
  - $I \subseteq (I' \times AT(\varphi) \times A_i) \cap Q$ such that $(q, C, a) \in I$ iff C is an initial node of the atom graph G_φ
  - $\delta = \{((q, C, a), \mathbf{a}, (q', C', a')) \mid (q, \mathbf{a}, q') \in \delta',\ C \xrightarrow{\mathbf{a}} C'\}$
  - $f(q, C, a) = \begin{cases} f'(q) & \text{if } \varphi \in C \\ a & \text{otherwise} \end{cases}$

Step 2: We need to show the following:

- For all $\mathcal{T} \in \mathfrak{T}_i,\ t = (v, \mathbf{u}) \in \mathcal{T}, \mathcal{T} \in [\![s, t]\!]_\mathcal{A}$ iff $\mathcal{T} \in \mathcal{L}(\mathcal{Q}^t_s)$ where $\mathcal{Q}^t_s$ is the same as $\mathcal{Q}_s$ but with start states being the set of all states reachable on $\mathbf{u}$ in $\mathcal{Q}_s$.

We show this by induction on the structure of s:

- $s \equiv \phi$: By definition, $\mathcal{T} \in [\![\phi, t]\!]_\mathcal{A}$ if and only if $\mathcal{T}$ is a total strategy tree derived from $\mathcal{T}_\mathcal{A}^{\rho(v_0, \mathbf{u})}$) where $\mathcal{T}_\mathcal{A}^{\rho(v_0, \mathbf{u})}$ is the play $\rho(v_0, \mathbf{u})$ followed by the subtree of $\mathcal{T}_\mathcal{A}$ starting at $\rho(v_0, \mathbf{u})$. Now by construction of $\mathcal{Q}_s = (Q, I, \delta, f, val)$, there is a state in Q corresponding to every action $a \in A_i$ and the function δ is the total function $Q \times \prod_{i \in N} \times Q$. Hence we have $\mathcal{T} \in [\![\phi, t]\!]_\mathcal{A}$ iff $\mathcal{T} \in \mathcal{L}(\mathcal{Q}^t_s)$.

- $s \equiv s_a$: By definition, $\mathcal{T} \in [\![s, t]\!]_\mathcal{A}$ if and only if for all $t = (v, \mathbf{ua}) \in \mathcal{T}$, $\mathbf{a}(i) = a$. Now by construction of $\mathcal{Q}_s = (Q, I, \delta, f, val)$, $f(q) = a$ for every $q \in Q$ and hence we have $\mathcal{T} \in [\![s, t]\!]_\mathcal{A}$ iff $\mathcal{T} \in \mathcal{L}(\mathcal{Q}^t_s)$.

- $s \equiv s_1 \cup s_2$: Note that $\mathcal{L}(\mathcal{Q}^t_{s_1}) \cup \mathcal{L}(\mathcal{Q}^t_{s_2}) = \mathcal{L}(\mathcal{Q}^t_{s_1 \cup s_2})$ by the construction of $\mathcal{Q}_{s_1 \cup s_2}$. Thus, $\mathcal{T} \in [\![s_1 \cup s_2, t]\!]_\mathcal{A}$ iff $\mathcal{T} \in [\![s_1, t]\!]_\mathcal{A} \cup [\![s_2, t]\!]_\mathcal{A}$ iff $\mathcal{T} \in [\![s_1, t]\!]_\mathcal{A}$ or $\mathcal{T} \in [\![s_2, t]\!]_\mathcal{A}$ iff $\mathcal{T} \in \mathcal{L}(\mathcal{Q}^t_{s_1})$ or $\mathcal{T} \in \mathcal{L}(\mathcal{Q}^t_{s_2})$ iff $\mathcal{T} \in \mathcal{L}(\mathcal{Q}^t_{s_1}) \cup \mathcal{L}(\mathcal{Q}^t_{s_2})$ iff $\mathcal{T} \in \mathcal{L}(\mathcal{Q}^t_{s_1 \cup s_2})$.

- $s \equiv s_1 \frown s_2$: $\mathcal{T} \in [\![s_1 \frown s_2, t]\!]_\mathcal{A}$ implies, by the definition of $[\![s_1 \frown s_2, t]\!]_\mathcal{A}$, that there exists $\mathcal{T}'$ where $\mathcal{T}' \in [\![s_1, t]\!]_\mathcal{A}$ and a $k \geq |t|$ such that $\mathcal{T}$ pruned at depth k is equal to $\mathcal{T}'$ pruned at depth k and there exists

$\mathcal{T}''$ where $\mathcal{T}'' \in [\![s_2, t]\!]_{\mathcal{A}}$ such that $\mathcal{T}''$ from depth $k+1$ is the same as $\mathcal{T}$ from depth $k+1$. By the induction hypothesis, $\mathcal{T}' \in \mathcal{L}(\mathcal{Q}^t_{s_1})$ and $\mathcal{T}'' \in \mathcal{L}(\mathcal{Q}^t_{s_2})$. Thus the run of $\mathcal{Q}_{s_1{}^\frown s_2}$ where the transducer makes a switch from mirroring the output of $\mathcal{Q}_{s_1}$ to that of $\mathcal{Q}_{s_2}$ after reading the kth input, is the accepting run on $\mathcal{T}$.

Conversely, suppose $\mathcal{T} \in \mathcal{L}(\mathcal{Q}^t_{s_1{}^\frown s_2})$. Let r be the accepting run of $\mathcal{Q}_{s_1{}^\frown s_2}$ on $\mathcal{T}$. Then, there exists $k \geq |t|$ such that there exists $\mathcal{T}'$ where $\mathcal{T}' \in \mathcal{L}(\mathcal{Q}^t_{s_1})$ and $\mathcal{T}''$ where $\mathcal{T}'' \in \mathcal{L}(\mathcal{Q}^t_{s_2})$ such that for all t', $|t| \leq |t'| \leq k, t' \in \mathcal{T}'$ if and only if $t' \in \mathcal{T}$ and for all $t'', |t''| > k, t'' \in \mathcal{T}''$ if and only if $t'' \in \mathcal{T}$. By the induction hypothesis, $\mathcal{T}' \in [\![s_1, t]\!]_{\mathcal{A}}$ and $\mathcal{T}'' \in [\![s_2, t''']\!]_{\mathcal{A}}$ for all $t''' \in \mathcal{T}'$. Hence $\mathcal{T} \in [\![s_1{}^\frown s_2, t]\!]_{\mathcal{A}}$.

- $s \equiv (s_1 + s_2)$: By definition,

$$\bigcup_{k_1,k_2,\ldots} ((([\![s_1,t]\!]_{\mathcal{A}}{}^{k_1}[\![s_2,t]\!]_{\mathcal{A}})^{k_2}[\![s_1,t]\!]_{\mathcal{A}})\cdots \text{ where } k < k_1 < k_2 \ldots$$
$$= \bigcup_{i\geq 0} \bigcup_{k_1,k_2,\ldots,k_i} (\ldots(([\![s_1,t]\!]_{\mathcal{A}}{}^{k_1}[\![s_2,t]\!]_{\mathcal{A}})^{k_2}[\![s_1,t]\!]_{\mathcal{A}})\ldots^{k_i}[\![s_1,t]\!]_{\mathcal{A}})$$
$$\text{where } k < k_1 < k_2 \ldots < k_i.$$

Now, similar to the case for $s_1{}^\frown s_2$, we can show that for every $i \geq 0$, there exist instances $k_1 < k_2 \ldots < k_i$ at which the transducer $\mathcal{Q}^t_{(s_1+s_2)}$ alternates between the outputs of $\mathcal{Q}^t_{s_1}$ and $\mathcal{Q}^t_{s_2}$. Now since $\mathcal{L}(\mathcal{Q}^t_{(s_1+s_2)}) = \bigcup_{i\geq 0}\bigcup_{k_1,k_2,\ldots,k_i} \mathcal{L}(\mathcal{Q}^t_{s_1{}^{k_1}s_2{}^{k_2}\ldots{}^{k_i}s_x})$, $x \in \{1,2\}$, it follows from the above equation that $\mathcal{T} \in [\![(s_1+s_2), t]\!]_{\mathcal{A}}$ iff $\mathcal{T} \in \mathcal{L}(\mathcal{Q}^t_{(s_1+s_2)})$.

- $s \equiv \varphi?s'$: Let $\mathcal{Q}^t_s = (Q, I, \delta, f, val)$ and $\mathcal{Q}^t_{s'} = (Q', I', \delta', f', val')$ be the transducers for s and s' respectively. Let $\mathcal{T}$ and $\mathcal{T}'$ be such that $\mathcal{T} \in [\![s, t]\!]_{\mathcal{A}}$ and $\mathcal{T}' \in [\![s', t]\!]_{\mathcal{A}}$ and for all $t' \in \mathcal{T}_{\mathcal{A}}$, if φ holds at t' then $t' \in \mathcal{T}$ if and only if $t' \in \mathcal{T}'$. We have to show that $\mathcal{Q}^t_s$ has an accepting run on $\mathcal{T}$. By the induction hypothesis, $\mathcal{Q}_{s'}$ has an accepting run r' on $\mathcal{T}'$. We construct the run r of $\mathcal{Q}^t_s$ on $\mathcal{T}$ as follows. $r(t_0) = q_0 \in I$. Let $t', t'' \in \mathcal{T}$ such that $|t'| \geq |t|$, t'' is a child of t and $r'(t') = q', r'(t'') = q''$. If $t' \models \varphi$, then by the construction of $\mathcal{Q}^t_s$, we have $r(t'') = r'(t'') = q''$. If φ doesn't hold at t', put $r(t'') = r'(t'') = q''$. This is possible since r' is a valid run on $\mathcal{T}'$ and hence $q' \to q''$. It is clear that the above run accepts $\mathcal{T}$.

Conversely, suppose $\mathcal{T} \in \mathcal{L}(\mathcal{Q}^t_s)$ and let r be an accepting run of $\mathcal{Q}^t_s$ on $\mathcal{T}$. We need to show that there exists $\mathcal{T}' \in \mathcal{L}(\mathcal{Q}^t_{s'})$ such that for all $t' \in \mathcal{T}_{\mathcal{A}}$, $t' \in \mathcal{T}$ if and only if $t' \in \mathcal{T}'$. We make the following claim.

Claim 4.7 *For all $t \in \mathcal{T}_\mathcal{A}$ and for all $\varphi' \in CL(\varphi)$, if $r(t) = (q, C, a)$ then $\varphi' \in C$ iff $t \models \varphi'$.*

Assuming claim, by the construction of the transducer $\mathcal{Q}_s^t$, we have that if φ holds at t'' then $f(r(t'')) = f'(r'(t''))$. Therefore there must exist $\mathcal{T}' \in [\![s', t]\!]_\mathcal{A}$ such that for all $t' \in \mathcal{T}_\mathcal{A}$, $t' \in \mathcal{T}$ if and only if $t' \in \mathcal{T}'$. If φ does not hold at t'' then $f(r(t'')) = a''$ where $r(t'') = (q'', C'', a'')$. Since the state (q'', C'', a'') is a nondeterministic choice of the transducer $\mathcal{Q}_s$, this holds for every $a'' \in A_i$. Hence $\mathcal{T} \in [\![s, t]\!]_\mathcal{A}$ by the definition of $[\![s, t]\!]_\mathcal{A}$.

It now remains to prove claim 4.7 which we do below.

Proof of Claim 4.7 The proof proceeds by induction on the structure of φ'.

- $\varphi' = p \in \mathcal{P}$: Follows from definition since in the construction we ensured that $(q, C, a) \in Q$ iff $val(q) = C \cap \mathcal{P}$.
- $\varphi' = \neg\varphi''$: $t \models \neg\varphi'$ iff $t \not\models \varphi''$ iff $\varphi'' \notin C$ iff $\neg\varphi'' \in C$ [since C is an atom].
- $\varphi' = \varphi_1 \vee \varphi_2$: $t \models \varphi'$ iff $t \models \varphi_1$ or $t \models \varphi_2$ iff $\varphi_1 \in C$ or $\varphi_2 \in C$ iff $\varphi_1 \vee \varphi_2 \in C$ [since C is an atom].
- $\varphi' = \langle \mathbf{a} \rangle^- \varphi''$: $t \models \langle \mathbf{a} \rangle^- \varphi''$ iff $t' \xrightarrow{\mathbf{a}} t \in \mathcal{T}_\mathcal{A}$ and $t' \models \varphi''$ iff $\varphi'' \in r(t') = (q', C', a')$ iff $\langle \mathbf{a} \rangle^- \varphi'' \in r(t)$ since C is an atom and $(q', C', a') \xrightarrow{\mathbf{a}} (q, C, a)$ iff $q' \xrightarrow{\mathbf{a}} q$ and $C' \xrightarrow{\mathbf{a}} C$ by construction.
- $\varphi' = \langle \mathbf{a} \rangle^+ \varphi''$: Similar to the case for $\langle \mathbf{a} \rangle^- \varphi''$.
- $\varphi' = \diamondminus\varphi''$: $t \models \diamondminus\varphi''$ iff there exists an ancestor t' of t such that $t' \models \varphi''$ Let t' be the first such ancestor. We induct again on $|t| - |t'|$. $|t| - |t'| = 0$ iff $t \models \varphi''$ iff $\varphi'' \in r(t)$ by induction hypothesis where iff $\diamondminus\varphi'' \in C$ [since C is an atom]. $|t| - |t'| > 0$ iff $t'' \models \diamondminus\varphi''$ and $t \models \ominus(\diamondminus\varphi'')$ where t'' is the parent of t iff $\diamondminus\varphi'' \in C''$ where $r(t'') = (q'', C'', a'')$ iff $\diamondminus\varphi'' \in C$ [since C is an atom].

  For the converse direction, suppose that $\diamondminus\varphi'' \in C$. Since, by the construction of the atom graph $G_{\varphi''}$, the initial nodes do not have any formula of the form $\ominus\alpha$, there must exist a vertex C' on the path from an initial vertex C_0 to C in $G_{\varphi''}$ such that $\varphi'' \in C'$. Hence there must exist a node t' in $\mathcal{T}_\mathcal{A}$ such that $r(t') = (q', C', a')$. Let t' be the last such node. We induct again

on the distance between t' and t. Let d denote this distance. The base case is when $d = 0$. Then we have that $\varphi'' \in C$ and hence $\diamondminus\varphi'' \in C$ [since C is an atom] which implies $t \models \diamondminus\varphi''$ [by definition]. Let $d > 0$ and let t'' be the parent of t and let $r(t'') = (q'', C'', a'')$. Since the distance between t' and t'' is $d-1$ we can apply the second induction hypothesis to conclude that $t'' \models \diamondminus\varphi''$. Hence $t \models \diamondminus\varphi''$ [by definition].

□

This finishes the proof of the claim and along with it, the proof of the lemma.

□

We have the following immediate corollary to the above lemma.

Corollary 4.8 *The strategies of player i that can be specified using the syntax of Σ_i are finite memory.*

Proof Every such strategy s can be represented by a finite state transducer $\mathcal{Q}_s$ as described in Lemma 4.6. □

Remark Note that Proposition 1.5, which we use in the proof of Lemma 4.6, is a truth lemma and we can go ahead and prove a completeness theorem for our logic. However we refrain from doing so as we feel the logic is not interesting enough for such an exercise. A more interesting logic is one where we can make statements of the form "strategy s ensures outcome α". Such a logic has been studied in [RS06] where they show a finite model property and prove a completeness theorem. What is more interesting for our logic are questions concerning the eventual stability of games which we address below.

4.5 Stability

In this section we answer questions about the dynamics and the eventual behaviour of the model of concurrent games with strategy specifications that we have introduced and described in the previous sections.

4.5.1 Substrategy

Given a strategy specification $s \in \Sigma_i$ of player i, the set of substrategies of s, $sub(s)$ is defined to be the least set satisfying:

- $s \in sub(s)$.
- $s_1 \cup s_2 \in sub(s)$ implies $s_1, s_2 \in sub(s)$.
- $s_1 ^\frown s_2 \in sub(s)$ implies $s_1, s_2 \in sub(s)$.
- $(s_1 + s_2) \in sub(s)$ implies $s_1, s_2 \in sub(s)$.
- $\varphi ? s' \in sub(s)$ implies $s' \in sub(s)$.

We call a strategy specification s 'switch-free' if it does not have any of the $\frown$ or the $+$ constructs. Let $sf(sub(s))$ be the set of switch-free substrategies of $sub(s)$. Note that $sf(sub(s))$ is a finite set for a given s.

4.5.2 Eventual behaviour

We would like to analyse what are the properties that eventually hold in the game when the players play according to their strategy specifications. For that we define the notion of 'stability' of a property (formula). Let α be a formula from the syntax Φ_+ (Section 1.2.9). We say α is stable in a subarena $\mathcal{Z} = (V_\mathcal{Z}, E_\mathcal{Z})$ of $\mathcal{A}$ if $t \models \varphi$ for all nodes $t \in \mathcal{T}_\mathcal{Z}$ for unfoldings $\mathcal{T}_\mathcal{Z}$ starting at every node $z \in V_\mathcal{Z}$. One way of checking the stability of a formula α in $\mathcal{Z}$ is by the following procedure.

Procedure 1

- We construct the subformula closure $CL'(\varphi)$ of φ.
- We define the modal-depth $md(\varphi')$ of a subformula $\varphi' \in CL'(\varphi)$ inductively as follows:
  - $md(p) = 0$.
  - $md(\neg\varphi') = md(\varphi')$.
  - $md(\varphi_1 \vee \varphi_2) = \max\{md(\varphi_1), md(\varphi_2)\}$.
  - $md(\langle\mathbf{a}\rangle^{-}\varphi') = md(\varphi') + 1$.
  - $md(\langle\mathbf{a}\rangle^{+}\varphi') = md(\varphi') + 1$.
- Let $\mathbf{m} = md(\varphi)$. For $i = 0$ to m, for every $\varphi' \in CL'(\varphi)$ such that $md(\varphi') = i$, we label a node $v \in V$ with φ' iff $v \models \varphi'$. This can be done by checking v, all nodes in $vE_{\mathcal{Z}}$ and all nodes v' such that $v \in v'E_{\mathcal{Z}}$.
- Finally we check if v is labelled with φ.

Thus if every node $v \in V_{\mathcal{Z}}$ is labelled with φ then we conclude that φ is stable in $\mathcal{Z}$.

Now given a game arena $\mathcal{A}$ and strategy specifications of the players, we may ask whether a certain objective $\alpha \in \Phi_+$ is attained if the players play according to these specifications. When the objective α is attained it is also meaningful to ask whether a particular player plays according to some substrategy that does not involve any switching.

Let $(\mathcal{A}, v_0)$ be the initialised arena where $\mathcal{A} = (V, E)$ and $s \in \Sigma_i$. Let $\mathcal{Q}_s = (Q, I, \delta, f, val)$ be the transducer for s. We construct a subarena of $\mathcal{A}$ which corresponds to the moves dictated by the strategy s. For that purpose, we define the restriction of $\mathcal{A}$ with respect to $\mathcal{Q}_s$, denoted by $\mathcal{A} \restriction \mathcal{Q}_s$. Note that by definition, if the move a is not available to player i at any point, then the strategy s is undefined. Formally, $\mathcal{A} \restriction \mathcal{Q}_s = (\mathcal{A}', v_0')$ where $\mathcal{A}' = (V', E')$ such that

- $V' \subseteq V \times Q$ such that $(v, q) \in V'$ iff $val(v) = val(q)$.
- $(v_1, q_1) \xrightarrow{\mathbf{a}} (v_2, q_2)$ iff
  - $v_1 \xrightarrow{\mathbf{a}} v_2$, $q_1 \xrightarrow{\mathbf{a}} q_2$, $f(q_1) = \mathbf{a}(i)$ and $\mathbf{a}(i) \in \Gamma_i(v_1)$ or

- $v_1 \xrightarrow{\mathbf{a}} v_2$, $q_1 \xrightarrow{\mathbf{a}} q_2$ and $\mathbf{a}(i) \notin \Gamma_i(v_1)$.

- $v'_0 = (\{v_0\} \times I) \cap V'$.

Note that due to the Transducer Lemma, for any $v' = (v, q) \in V'$, $w'E'$ are the moves dictated by the strategy tuple $(s_1, s_2, \ldots, s_n)$ of the players at the node v'.

Theorem 4.9 *Given an initialised arena $(\mathcal{A}, v_0)$ with a valuation val of the observables, an objective $\alpha \in \Phi_+$ and strategy specifications $s_1, \ldots, s_n$ for players 1 to n, the following questions are decidable:*

1. *If every player plays according to her specification, is the objective α eventually stable?*

2. *If the objective α is eventually stable, does the strategy of player i play according to a substrategy that does not involve switching?*

Proof

1. We construct the graph $(\mathcal{A}', v'_0) = (\cdots((\mathcal{A} \restriction \mathcal{Q}_{s_1}) \restriction \mathcal{Q}_{s_2} \cdots) \restriction \mathcal{Q}_{s_n})$. For every maximal connected component $\mathcal{Z}$ of $\mathcal{A}'$ reachable from every initial state in v'_0, we check if α is stable in $\mathcal{Z}$ using Procedure 1. If so, we output YES.

2. To solve this question, we need to keep track of the switches ($\frown$'s) in strategies that the player i already makes till the play reaches a maximal connected component $\mathcal{Z}$. That is because, if player i has $s \frown s'$ among the substrategies of her strategy specification s_i, and she has already made a switch from s to s' before the play reaches $\mathcal{Z}$, she cannot later use s. We also have to keep track of the current state of each of the transducers till that point.

   To achieve this we do the following. We associate an index id_i with every substrategy of the specification s_i of player i, $id_i : sub(s_i) \to |sub(s_i)|$. id_i can be lifted to any subset of S of $sub(s_i)$ as $id_i(S) = \{id_i(s) \mid s \in S\}$.

   To keep track of the relevant substrategies of s_i at every history t, we need to equip each transducer $\mathcal{Q}_s$, used to construct $\mathcal{Q}_{s_i}$ with an output function O_i which gives the indices of the substrategies whose transducers are being simulated by $\mathcal{Q}_{s_i}$ at t and also their current states.

Let $\mathcal{T}_{\mathcal{A}} = (T, E)$. Let $\mathcal{Q}(sub(s_i)) = \{\mathcal{Q}_s \mid s \in sub(s_i)\}$ be the transducers corresponding to all the substrategies of s_i that are used to construct $\mathcal{Q}_{s_i}$. Let $\widetilde{Q}$ be the union of the set of states of the transducers in $\mathcal{Q}(sub(s_i))$. Formally

$$O_i : T \times \mathcal{Q}(sub(s_i)) \to 2^{id(sub(s_i)) \times 2^{\widetilde{Q}}}$$

We want O_i to satisfy:

(∗) $\forall t, \forall \mathcal{Q}_s \in \mathcal{Q}(sub(s_i)),\ O_i(t, \mathcal{Q}_s) = X$ implies X is the set of the indices of the substrategies whose transducers are being simulated by $\mathcal{Q}_{s_i}$ at t and their current states. Furthermore, $(m, Q) \in O_i(t, \mathcal{Q}_s)$ implies Q is a subset of the set of states of $\mathcal{Q}_s$.

Assume for the time being that we have such a function O_i. Let $id(O_i(t, \mathcal{Q}_s))$ be the set $O_i(t, \mathcal{Q}_s)$ projected to the first component of each tuple, i.e., it is the set of all the indices present in this set.

Step 1: Check if the objective α is eventually attained (item 1). If not output NO and exit.

Step 2: For every maximal connected component $\mathcal{Z}$ and for every initial node in v_0' and every minimal path ρ that reaches $\mathcal{Z}$ let $t = (last(\rho), \mathbf{u}(\rho))$. Repeat:

Loop 1 Let $sf_t(sub(s_i)) = \{s \in sf(sub(s_i)) \mid id(s) \in id(O_i(t, \mathcal{Q}_{s_i}))\}$. $sf_t(sub(s_i))$ are the switch-free substrategies of s_i whose indices are present in $O_i(t, \mathcal{Q}_{s_i})$. Let $Q_1^t, \ldots, Q_n^t$ be the states of $\mathcal{Q}_{s_1}, \ldots, \mathcal{Q}_{s_n}$ resp. at t. For each $s \in sf_t(sub(s_i))$ repeat:

Loop 2 Let (m, Q) be the element corresponding to s in $O_i(t, \mathcal{Q}_{s_i})$. Construct $\mathcal{Q}_{s_i}$ but with start states being Q and call it $\mathcal{Q}'_{s_i}$. For all $j \neq i$ let the start states of $\mathcal{Q}_{s_j}$ be Q_j^t resp. Denote these transducers by $\mathcal{Q}'_{s_j}$. Let

$$(\mathcal{A}'', v_0'') = ((\cdots((\cdots(((\mathcal{A}, last(\rho)) \restriction \mathcal{Q}'_{s_1}) \restriction \mathcal{Q}'_{s_2}) \cdots) \restriction \mathcal{Q}'_s) \cdots) \restriction \mathcal{Q}'_{s_n})$$

Let $\mathcal{Z}'$ be the maximal connected component of $\mathcal{A}''$ to which v_0' belongs. Check if α is stable in $\mathcal{Z}'$ using Procedure 1. If not, return NO and exit.

Return YES.

It is now left to define O_i. We do it inductively as follows:

- $O_i(t, \mathcal{Q}_{s_a}) = (id(s_a)$, current state set of $\mathcal{Q}_{s_a})$.
- $$O_i(t, \mathcal{Q}_{s_1 \cup s_2}) = \{(id(s_1 \cup s_2),\ \text{current state set of } \mathcal{Q}_{s_1 \cup s_2})\} \cup \begin{cases} O_i(t, \mathcal{Q}_{s_1}) & \text{if } \mathcal{Q}_{s_1 \cup s_2} \text{ is simulating } \mathcal{Q}_{s_1} \text{at } t \\ O_i(t, \mathcal{Q}_{s_2}) & \text{if } \mathcal{Q}_{s_1 \cup s_2} \text{ is simulating } \mathcal{Q}_{s_2} \text{at } t \end{cases}$$
- $$O_i(t, \mathcal{Q}_{s_1 \cap s_2}) = \{(id(s_1 \cap s_2),\ \text{current state set of } \mathcal{Q}_{s_1 \cap s_2})\} \cup O_i(t, \mathcal{Q}_{s_1}) \cup O_i(t, \mathcal{Q}_{s_2})$$
- $$O_i(t, \mathcal{Q}_{s_1 ^\frown s_2}) = \{(id(s_1 ^\frown s_2),\ \text{current state set of } \mathcal{Q}_{s_1 ^\frown s_2})\} \cup \begin{cases} O_i(t, \mathcal{Q}_{s_1}) & \text{if } \mathcal{Q}_{s_1 ^\frown s_2} \text{ is simulating } \mathcal{Q}_{s_1} \text{at } t \\ O_i(t, \mathcal{Q}_{s_2}) & \text{if } \mathcal{Q}_{s_1 ^\frown s_2} \text{ is simulating } \mathcal{Q}_{s_2} \text{at } t \end{cases}$$
- $$O_i(t, \mathcal{Q}_{s_1 + s_2}) = \{(id(s_1 + s_2),\ \text{current state set of } \mathcal{Q}_{s_1 + s_2})\} \cup O_i(t, \mathcal{Q}_{s_1}) \cup O_i(t, \mathcal{Q}_{s_2})$$
- $O_i(t, \mathcal{Q}_{\varphi ? s'}) = \{(id(\varphi ? s'),\ \text{current state of } \mathcal{Q}_{\varphi ? s'})\} \cup O_i(\mathcal{Q}_{s'})$

Now an easy induction on the substrategies of s_i shows that O_i defined above meets requirement $(*)$.

□

4.5.3 Complexity

We now analyse the running time of the procedure described in Theorem 4.9. For a game arena $\mathcal{A}$ whose size is m and a strategy specification s of length $|s|$, let $\mathcal{Q}_s$ denote the transducer for s. It can be verified from the construction of $\mathcal{Q}_s$ that its size is $\mathcal{O}(2^{|s|})$. The size of the restricted graph $|\mathcal{A} \restriction \mathcal{Q}_s|$ then is $\mathcal{O}(m \cdot 2^{|s|})$. Thus the size of $\mathcal{A}'$, $|\mathcal{A}'|$ is $\mathcal{O}(m \cdot 2^{np})$ where p is the maximum length of a specification formula s_1 or s_2 or ... or s_n. Now since checking for the maximal connected component of a graph can be done is time polynomial in the size of the graph, step 1 takes time $\mathcal{O}(m \cdot 2^{np})$. There can be at most m^m minimal paths from v_0' to $\mathcal{Z}$ and hence loop 1 repeats at most m^m times. There are $\mathcal{O}(|s|)$ substrategies in $sf_t(sub(s_i))$ and so loop 2 repeats $\mathcal{O}(|s|) = \mathcal{O}(p)$ times. Each iteration of loop 2 takes $\mathcal{O}(q \cdot m \cdot 2^{np})$ time, where $q = |\alpha|$. Hence the complexity of the entire procedure is $\mathcal{O}(m \cdot 2^{np}) + \mathcal{O}(m^m \cdot pq \cdot 2^{np})$ which is $\mathcal{O}(m^m \cdot pq \cdot 2^{np})$.

4.6 Probabilistic switching

Consider the game of cricket again. A bowler who has been pelted for a six every time he has bowled short to the batsman may change his strategy and bowl good length deliveries instead. But will he change his strategy entirely? Will he not retain a small probability of still bowling a short-pitched delivery just to create an element of surprise for the batsman and to keep him guessing? Will the batsman who has charged down the track to every over-pitched delivery still not retain a small probability of staying put and coaxing the bowler to bowl a short delivery? Such strategising is common in every game. In real-life games it is seen that players do not combine or switch strategies deterministically but in a randomised fashion. The probabilities may be determined by the player's previous experience, the conditions at hand and various other things. In this section we study strategy specifications for players where the operators introduced for combining strategies are probabilistic. We then show that the eventual outcome of such games can be predicted.

4.6.1 Syntax and semantics

The set of strategies Σ_i of player i is built using the following syntax

$$\Sigma_i ::= a \in A_i \mid s_1 {}^\frown{}^p s_2 \mid s_1 +^{p,q} s_2 \mid \varphi ?^p s_1; s_2$$

for all $0 \leq p, q \leq 1$.

We now give the semantics of the above operations:

- $s \equiv s_1 {}^\frown{}^p s_2$: The player starts off by playing s_1 and in every round switches to s_2 with probability p independently of prior switches. The process is then a Bernoulli's trial and we have the following facts:

  1. Probability that the player switches in $\leq n$ rounds is $1-(1-p)^n$.
  2. Expected number of rounds it takes for the player to switch is $1/p$.
  3. Probability that player plays s_1 in n rounds is $(1-p)^n$

  and so on.

- $s \equiv s_1 +^{p,q} s_2$: In every round $t+1$ if the player played s_1 in round t then she switches to s_2 with probability p and if she played s_2 in round t then she switches to s_1 with probability q. Let $X_i,\ i \geq 1$ be

the random variable that takes values s_1 or s_2. It denotes the strategy played in round i. We then have:

$$prob(X_1 = s_1) = 1,\ prob(X_1 = s_2) = 0.$$

$$\begin{aligned}
prob(X_2 = s_1) &= prob(X_2 = s_1 \mid X_1 = s_1) + prob(X_2 = s_1 \mid X_1 = s_2) \\
prob(X_2 = s_1) &= prob(X_2 = s_1 \cap X_1 = s_1) + prob(X_2 = s_1 \cap X_1 = s_2) \\
&= prob(s_1 s_1) + prob(s_1 s_2) \text{(Simplifying notation)} \\
&= (1-p) \cdot 1 + 0 \\
&= (1-p)
\end{aligned}$$

Similarly, $prob(X_2 = s_2) = p$.

$$\begin{aligned}
prob(X_3 = s_1) &= prob(s_1 s_1 s_1) + prob(s_1 s_1 s_2) + prob(s_1 s_2 s_2) + prob(s_1 s_2 s_1) \\
&= prob(X_3 = s_1 \mid (X_2 = s_1 \cap X_1 = s_1) \text{ or } (X_2 = s_1 \cap X_1 = s_2)) \\
&+ prob(X_3 = s_1 \mid (X_2 = s_2 \cap X_1 = s_2) \text{ or } (X_2 = s_2 \cap X_1 = s_1)) \\
&= prob(X_3 = s_1 \mid X_2 = s_1) + prob(X_3 = s_1 \mid X_2 = s_2) \\
&= (1-p) \cdot prob(X_2 = s_1) + q \cdot prob(X_2 = s_2) \\
&= (1-p)^2 + pq.
\end{aligned}$$

Similarly, $prob(X_3 = s_2) = 2p - pq - p^2 = 1 - prob(X_3 = s_1)$.

Thus generalising, we have the following recursive equations:

$$\begin{aligned}
prob(X_n = s_1) &= prob(X_n = s_1 \mid X_{n-1} = s_1) prob(X_{n-1} = s_1) \\
&+ prob(X_n = s_1 \mid X_{n-1} = s_2) prob(X_{n-1} = s_2) \\
&= prob(X_n = s_1 \ \cap \ X_{n-1} = s_1) \\
&+ prob(X_n = s_1 \ \cap \ X_{n-1} = s_2) \\
&= prob(X_n = s_1) \cdot prob(X_{n-1} = s_1) \\
&+ prob(X_n = s_1) \cdot prob(X_{n-1} = s_2) \\
&= (1-p) \cdot prob(X_{n-1} = s_1) + q \cdot prob(X_{n-1} = s_2).
\end{aligned} \tag{4.1}$$

Similarly,

$$prob(X_n = s_2) = (1-q) \cdot prob(X_{n-1} = s_2) + p \cdot prob(X_{n-1} = s_1). \tag{4.2}$$

As a sanity check

$$prob(X_n = s_1) + prob(X_n = s_2) = [1 - q + q]prob(X_{n-1} = s_2) + [1 - p + p]prob(X_{n-1} = s_1) = 1.$$

- $s \equiv \varphi ?^p s_1; s_2$: At every position the player checks if φ holds of that history. If it does, then the player plays s_1 with probability p and s_2 with probability $1 - p$. s_2 is the default strategy. Thus if φ does not hold then the player plays s_2.

4.6.2 Probabilistic transducer

A probabilistic transducer over input alphabet X and output alphabet Y is a tuple $\mathcal{Q} = (Q, I, \delta, f)$ where Q is a finite set of states $I \subseteq Q$ is the set of initial states, $\delta : Q \times X \to \Delta(Q)$ where $\Delta(Q)$ is the set of probability distributions over the set Q and $f : Q \to Y$ is the output function. Note that, we use the same notation $\mathcal{Q}$ to denote both probabilistic and non-probabilistic finite state transducers. As we shall exclusively deal with just one type of transducers during the course of a subtopic, this will not create a confusion.

4.6.3 Transducer construction

Given a strategy specification s for player i, we wish to construct a probabilistic finite state transducer $\mathcal{Q}_s$ oven input alphabet $\mathbf{a}$ and output alphabet A_i such that for every history ρ, and for every $a \in A_i$, $prob(s(\rho) = a) = p$ if and only if the probability that $\mathcal{Q}_s$ outputs a on history ρ is p. We carry out the construction inductively. As our transducers run on the arena $\mathcal{A} = (V, E)$ where the vertices come with valuations for the observables in $\mathcal{P}$, as in the case of non-probabilistic transducers, we need to equip our probabilistic transducers with a valuation function which gives the valuation of the observables in every state of the transducer. Hence our probabilistic transducer is a tuple $\mathcal{Q} = (Q, I, \delta, f, val)$ where Q, I, δ, f are as above and $val : Q \to 2^{\mathcal{P}}$ is the valuation function. We again build this function inductively.

- For the atomic strategy $s \equiv s_a, a \in A_i$, $\mathcal{Q}_s = (Q, I, \delta, f, val)$ where
  - $Q = 2^{\mathcal{P}}$
  - $I = Q$

- $\delta(q, \mathbf{a})(q') = 1/|Q|$ for every $q, q' \in Q,\ \mathbf{a} \in \mathbf{A}$
- $f(q) = a$ for every $q \in Q$
- $val(q) = q$

Let transducers $\mathcal{Q}_{s_1}$ and $\mathcal{Q}_{s_2}$ for strategies s_1 and s_2 be given where $\mathcal{Q}_{s_1} = (Q_1, I_1, \delta_1, f_1, val_1)$ and $\mathcal{Q}_{s_2} = (Q_2, I_2, \delta_2, f_2)$. Then

- $\mathcal{Q}_{s_1 \frown^p s_2} = (Q, I, \delta, f, val)$ where
  - $Q \subseteq Q_1 \times Q_2 \times \{1, 2\}$ such that $(q_1, q_2) \times \{1, 2\} \in Q$ if and only if $val_1(q_1) = val_2(q_2)$
  - $I = (I_1 \times I_2 \times \{1, 2\}) \cap Q$.

  Let $X \subseteq (Q_1 \times Q_2)$ be the event $X = \{(q_1, q_2) \mid val_1(q_1) = val_2(q_2)\}$. Let $(q_1, q_2) \in (Q_1 \times Q_2)$, $\mathbf{a} \in \mathbf{A}$. Then

$$prob(X \mid (q_1, q_2), \mathbf{a}) = \sum_{(q_1', q_2') \in X} \delta_1(q_1, \mathbf{a})(q_1') \cdot \delta_2(q_2, \mathbf{a})(q_2').$$

  - $\delta(\langle q_1, q_2, 1\rangle, \mathbf{a})(q_1', q_2', 2) = p\ \dfrac{\delta_1(q_1, \mathbf{a})(q_1') \cdot \delta_2(q_2, \mathbf{a})(q_2')}{prob(X \mid (q_1, q_2), \mathbf{a})}.$

    $\delta(\langle q_1, q_2, 1\rangle, \mathbf{a})(q_1', q_2', 1) = (1 - p)\dfrac{\delta_1(q_1, \mathbf{a})(q_1') \cdot \delta_2(q_2, \mathbf{a})(q_2')}{prob(X \mid (q_1, q_2), \mathbf{a})}.$

    $\delta(\langle q_1, q_2, 2\rangle, \mathbf{a})(q_1', q_2', 1) = 0.$

    $\delta(\langle q_1, q_2, 2\rangle, \mathbf{a})(q_1', q_2', 2) = \dfrac{\delta_1(q_1, \mathbf{a})(q_1') \cdot \delta_2(q_2, \mathbf{a})(q_2')}{prob(X \mid (q_1, q_2), \mathbf{a})}.$
  - $f(q_1, q_2, i) = f_i(q_i)$.
  - $val(q_1, q_2, i) = val_1(q_1) = val_2(q_2)$.

We first check that the transducer $\mathcal{Q}_{s_1 \frown^p s_2}$ is well-defined. For every state of the form $(q_1, q_2, 1) \in Q$, and $\mathbf{a} \in \mathbf{A}$, we have

$$\begin{aligned}
&\sum_{(q_1', q_2', i) \in Q} \delta(\langle q_1, q_2, 1\rangle \mathbf{a})(q_1', q_2', i) \\
&= \sum_{(q_1', q_2', 1) \in Q} \delta(\langle q_1, q_2, 1\rangle \mathbf{a})(q_1', q_2', 1) + \sum_{(q_1', q_2', 2) \in Q} \delta(\langle q_1, q_2, 1\rangle \mathbf{a})(q_1', q_2', 2) \\
&= \frac{1}{prob(X \mid (q_1, q_2), \mathbf{a})} \Bigg[\sum_{(q_1', q_2', 1) \in Q} (1 - p) \cdot \delta_1(q_1, \mathbf{a})(q_1') \cdot \delta_2(q_2, \mathbf{a})(q_2')
\end{aligned}$$

$$
\begin{aligned}
&+ \sum_{(q_1',q_2',2)\in Q} p \cdot \delta_1(q_1,\mathbf{a})(q_1') \cdot \delta_2(q_2,\mathbf{a})(q_2')] \\
&= \frac{1}{prob(X \mid (q_1,q_2),\mathbf{a})}\Bigg[\sum_{\substack{q_1'\in Q_1, q_2'\in Q_2 \\ val_1(q_1)=val_2(q_2)}} (1-p) \cdot \delta_1(q_1,\mathbf{a})(q_1') \cdot \delta_2(q_2,\mathbf{a})(q_2') \\
&+ \sum_{\substack{q_1'\in Q_1, q_2'\in Q_2 \\ val_1(q_1)=val_2(q_2)}} p \cdot \delta_1(q_1,\mathbf{a})(q_1') \cdot \delta_2(q_2,\mathbf{a})(q_2')] \\
&= \frac{1}{prob(X \mid (q_1,q_2),\mathbf{a})}\Bigg[\sum_{(q_1',q_2')\in X} (1-p) \cdot \delta_1(q_1,\mathbf{a})(q_1') \cdot \delta_2(q_2,\mathbf{a})(q_2') \\
&+ \sum_{(q_1',q_2')\in X} p \cdot \delta_1(q_1,\mathbf{a})(q_1') \cdot \delta_2(q_2,\mathbf{a})(q_2')] \\
&= \frac{1}{prob(X \mid (q_1,q_2),\mathbf{a})}[(1-p) \cdot prob(X \mid (q_1,q_2),\mathbf{a}) + p \cdot prob(X \mid (q_1,q_2),\mathbf{a})] \\
&= 1
\end{aligned}
$$

And similarly for states of the form $(q_1, q_2, 2)$.

We now prove the correctness of the above construction. We need to prove the following claim:

Claim 4.10 *Starting from an initial state $\langle q_1, q_2, 1\rangle$ and given $\mathbf{u} = \mathbf{a}_1\mathbf{a}_2 \ldots \mathbf{a}_n$, the probability that a state of the form $\langle q_1', q_2', 2\rangle$ is reached in n steps is $(1-p)^{n-1}p$ and the probability that a state of the form $\langle q_1', q_2', 1\rangle$ is reached in n steps is $(1-p)^n$.*

Proof We prove this by induction on n. For $n = 1$, the probability that a state of the form $\langle q_1', q_2', 2\rangle$ is reached is

$$
\begin{aligned}
&p \sum_{(q_1',q_2',2)\in Q} \delta(\langle q_1,q_2,1\rangle,\mathbf{a}_1)(q_1',q_2',2) \\
&= \frac{p}{prob(X \mid (q_1,q_2),\mathbf{a}_1)} \sum_{\substack{q_1'\in Q_1, q_2'\in Q_2 \\ val_1(q_1)=val_2(q_2)}} \delta_1(q_1,\mathbf{a}_1)(q_1') \cdot \delta_2(q_2,\mathbf{a}_1)(q_2') \\
&= \frac{p}{prob(X \mid (q_1,q_2),\mathbf{a}_1)} \sum_{(q_1',q_2')\in X} \delta_1(q_1,\mathbf{a}_1)(q_1') \cdot \delta_2(q_2,\mathbf{a}_1)(q_2') \\
&= \frac{p}{prob(X \mid (q_1,q_2),\mathbf{a}_1)}\big[prob(X \mid (q_1,q_2),\mathbf{a}_1)\big] \\
&= p = (1-p)^{1-1}p
\end{aligned}
$$

Similarly, the probability that a state of the form $\langle q_1', q_2', 1\rangle$ is reached is $(1-p) = (1-p)^1$.

Assume now that the property holds for $n = k$ steps. In step $k+1$, the probability of reaching a state of the form $\langle q_1'', q_2'', 2\rangle$ from a state of the form $\langle q_1', q_2', 1\rangle$ is p (by construction and by the argument for the base case). Hence the probability of reaching a state of the form $\langle q_1'', q_2'', 2\rangle$ from any state of the form $\langle q_1, q_2, 1\rangle$ in $k+1$ steps
$= p\cdot$[the probability of reaching a state of the form $\langle q_1', q_2', 1\rangle$ in k steps]
$= p \cdot (1-p)^k$.

By a similar argument, the probability of reaching a state of the form $\langle q_1'', q_2'', 1\rangle$ in $k+1$ steps is $(1-p)^{k+1}$. □

- $\mathcal{Q}_{s_1 +^{p,q} s_2} = (Q, I, \delta, f, val)$ where
  - $Q \subseteq Q_1 \times Q_2 \times \{1,2\}$ such that $(q_1, q_2) \times \{1,2\} \in Q$ if and only if $val_1(q_1) = val_2(q_2)$
  - $I = (I_1 \times I_2 \times \{1,2\}) \cap Q$.

  Once again, let $X \subseteq (Q_1 \times Q_2)$ be the event $X = \{(q_1, q_2) \mid val_1(q_1) = val_2(q_2)\}$. Let $(q_1, q_2) \in (Q_1 \times Q_2)$, $\mathbf{a} \in \mathbf{A}$. Then

$$prob(X \mid (q_1, q_2), \mathbf{a}) = \sum_{(q_1', q_2') \in X} \delta_1(q_1, \mathbf{a})(q_1') \cdot \delta_2(q_2, \mathbf{a})(q_2').$$

  - $\delta(\langle q_1, q_2, 1\rangle, \mathbf{a})(q_1', q_2', 2) = p\ \dfrac{\delta_1(q_1, \mathbf{a})(q_1') \cdot \delta_2(q_2, \mathbf{a})(q_2')}{prob(X \mid (q_1, q_2), \mathbf{a})}$.
    $\delta(\langle q_1, q_2, 1\rangle, \mathbf{a})(q_1', q_2', 1) = (1-p)\dfrac{\delta_1(q_1, \mathbf{a})(q_1') \cdot \delta_2(q_2, \mathbf{a})(q_2')}{prob(X \mid (q_1, q_2), \mathbf{a})}$.
    $\delta(\langle q_1, q_2, 2\rangle, \mathbf{a})(q_1', q_2', 1) = q\ \dfrac{\delta_1(q_1, \mathbf{a})(q_1') \cdot \delta_2(q_2, \mathbf{a})(q_2')}{prob(X \mid (q_1, q_2), \mathbf{a})}$.
    $\delta(\langle q_1, q_2, 2\rangle, \mathbf{a})(q_1', q_2', 2) = (1-q)\dfrac{\delta_1(q_1, \mathbf{a})(q_1') \cdot \delta_2(q_2, \mathbf{a})(q_2')}{prob(X \mid (q_1, q_2), \mathbf{a})}$.
  - $f(q_1, q_2, i) = f_i(q_i)$.
  - $val(q_1, q_2, i) = val_1(q_1) = val_2(q_2)$.

We now prove the correctness of the above construction. Let Y_n be the event of being in a state of the form $(q_1, q_2, 1)$ after n rounds and let $\overline{Y_n}$ be the event of being in a state of the form $(q_1, q_2, 2)$ after n rounds. Let Z_n be the state of $\mathcal{Q}_{s_1 +^{p,q} s_2}$ after n rounds. We need to show that our construction

satisfies the equations 4.1 and 4.2. That is, we need to to prove the following claim:

Claim 4.11

$$prob(Y_n) = (1-p)prob(Y_{n-1}) + q \cdot prob(\overline{Y_{n-1}}) \text{ and}$$
$$prob(\overline{Y_n}) = (1-q)prob(\overline{Y_{n-1}}) + p \cdot prob(Y_{n-1})$$

Proof We show this by induction on n. By construction we have $prob(Y_0) = 1$ and $prob(\overline{Y_0}) = 0$. For $n = 1$, let $\mathbf{a} \in \mathbf{A}$ be given. By construction

$$\begin{aligned}
prob(Y_1) &= \sum_{(q_1,q_2,i),(q_1',q_2',1)\in Q} prob(Z_0 = (q_1,q_2,i)) \cdot \delta(\langle q_1,q_2,i\rangle, \mathbf{a})(q_1',q_2',1) \\
&= \sum_{(q_1,q_2,1),(q_1',q_2',1)\in Q} prob(Z_0 = (q_1,q_2,1)) \cdot \delta(\langle q_1,q_2,1\rangle, \mathbf{a})(q_1',q_2',1) \\
&+ \sum_{(q_1,q_2,2),(q_1',q_2',1)\in Q} prob(Z_0 = (q_1,q_2,2)) \cdot \delta(\langle q_1,q_2,2\rangle, \mathbf{a})(q_1',q_2',1) \\
&= \sum_{(q_1,q_2,1)\in Q} prob(Z_0 = (q_1,q_2,1)) \sum_{(q_1',q_2',1)\in Q} \delta(\langle q_1,q_2,1\rangle, \mathbf{a})(q_1',q_2',1) \\
&\text{(the second term is 0)} \\
&= \sum_{(q_1,q_2,1)\in Q} \frac{prob(Z_0 = (q_1,q_2,1))(1-p)}{prob(X \mid (q_1,q_2),\mathbf{a})} \sum_{(q_1',q_2',1)\in Q} \delta_1(q_1,\mathbf{a})(q_1') \cdot \delta_2(q_2,\mathbf{a})(q_2') \\
&= \sum_{(q_1,q_2,1)\in Q} \frac{prob(Z_0 = (q_1,q_2,1))(1-p)}{prob(X \mid (q_1,q_2),\mathbf{a})} \sum_{\substack{q_1'\in Q_1, q_2'\in Q_2 \\ val_1(q_1)=val_2(q_2)}} \delta_1(q_1,\mathbf{a})(q_1') \cdot \delta_2(q_2,\mathbf{a})(q_2') \\
&= \sum_{(q_1,q_2,1)\in Q} \frac{prob(Z_0 = (q_1,q_2,1))(1-p)}{prob(X \mid (q_1,q_2),\mathbf{a})} prob(X \mid (q_1,q_2),\mathbf{a}) \\
&= (1-p)prob(Y_0).
\end{aligned}$$

And similarly for $prob(\overline{Y_1})$. Assume now the hypothesis holds for $n = k$. For $n = k+1$ let $\mathbf{u} = \mathbf{a}_1\mathbf{a}_2 \ldots \mathbf{a}_{k+1}$ be given and let $\mathbf{a}_{k+1} = \mathbf{a}$. Then

$$\begin{aligned}
prob(Y_{k+1}) &= \sum_{(q_1,q_2,1),(q_1',q_2',1)\in Q} prob(Z_k = (q_1,q_2,1)) \cdot \delta(\langle q_1,q_2,1\rangle, \mathbf{a})(q_1',q_2',1) \\
&+ \sum_{(q_1,q_2,2),(q_1',q_2',1)\in Q} prob(Z_k = (q_1,q_2,2)) \cdot \delta(\langle q_1,q_2,2\rangle, \mathbf{a})(q_1',q_2',1) \\
&= \sum_{(q_1,q_2,1)\in Q} \sum_{(q_1',q_2',1)\in Q} prob(Z_k = (q_1,q_2,1)) \cdot \delta(\langle q_1,q_2,1\rangle, \mathbf{a})(q_1',q_2',1)
\end{aligned}$$

$$+ \sum_{(q_1,q_2,2)\in Q} \sum_{(q_1',q_2',1)\in Q} prob(Z_k = (q_1,q_2,2)) \cdot \delta(\langle q_1,q_2,2\rangle, \mathbf{a})(q_1',q_2',1)$$

$$= \sum_{(q_1,q_2,1)\in Q} \frac{prob(Z_k = (q_1,q_2,1))(1-p)}{prob(X \mid (q_1,q_2),\mathbf{a})} \sum_{(q_1',q_2',1)\in Q} \delta_1(q_1,\mathbf{a})(q_1') \cdot \delta_2(q_2,\mathbf{a})(q_2')$$

$$+ \sum_{(q_1,q_2,2)\in Q} \frac{prob(Z_k = (q_1,q_2,2)) \cdot q}{prob(X \mid (q_1,q_2),\mathbf{a})} \sum_{(q_1',q_2',1)\in Q} \delta_1(q_1,\mathbf{a})(q_1') \cdot \delta_2(q_2,\mathbf{a})(q_2')$$

$$= \sum_{(q_1,q_2,1)\in Q} \frac{prob(Z_k = (q_1,q_2,1)) \cdot (1-p)}{prob(X \mid (q_1,q_2),\mathbf{a})} \sum_{\substack{q_1'\in Q_1, q_2'\in Q_2 \\ val_1(q_1)=val_2(q_2)}} \delta_1(q_1,\mathbf{a})(q_1') \cdot \delta_2(q_2,\mathbf{a})(q_2')$$

$$+ \sum_{(q_1,q_2,2)\in Q} \frac{prob(Z_k = (q_1,q_2,2)) \cdot q}{prob(X \mid (q_1,q_2),\mathbf{a})} \sum_{\substack{q_1'\in Q_1, q_2'\in Q_2 \\ val_1(q_1)=val_2(q_2)}} \delta_1(q_1,\mathbf{a})(q_1') \cdot \delta_2(q_2,\mathbf{a})(q_2')$$

$$= (1-p) \sum_{(q_1,q_2,1)\in Q} prob(Z_k = (q_1,q_2,1)) \frac{prob(X \mid (q_1,q_2),\mathbf{a})}{prob(X \mid (q_1,q_2),\mathbf{a})}$$

$$+q \sum_{(q_1,q_2,2)\in Q} prob(Z_k = (q_1,q_2,2)) \frac{prob(X \mid (q_1,q_2),\mathbf{a})}{prob(X \mid (q_1,q_2),\mathbf{a})}$$

$$= (1-p) prob(Y_k) + q \cdot prob(\overline{Y_k})$$

And similarly for $prob(\overline{Y_{k+1}})$. □

- $\mathcal{Q}_{\varphi ?^p s_1;s_2}$: Let $G_\varphi = (V_\varphi, E_\varphi)$ be the atom graph of φ and let $I_\varphi \subseteq V_\varphi$ be the initial states of G_φ. Then $\mathcal{Q}_{\varphi ?^p s_1;s_2} = (Q, I, \delta, f, val)$ where
  - $Q \subseteq Q_1 \times Q_2 \times V_\varphi \times \{1,2\}$ such that $(q_1,q_2,C,i) \in Q$ iff $val_1(q_1) = val_2(q_2) = C \cap \mathcal{P}$.
  - $I = (I_1 \times I_2 \times I_\varphi \times \{2\}) \cap Q$.
  - If it is not the case that $C \xrightarrow{\mathbf{a}}_\varphi C'$ then $\delta(\langle q_1,q_2,C,i\rangle,\mathbf{a})(q_1',q_2',C',j) = 0$ for all $(q_1,q_2,C,i),(q_1',q_2',C',j) \in Q$.

    Otherwise let $X \subseteq (Q_1 \times Q_2 \times V_\varphi)$ be the event $X = \{(q_1,q_2,C) \mid val_1(q_1) = val_2(q_2) = C \cap \mathcal{P}\}$. Let $(q_1,q_2,C) \in (Q_1 \times Q_2 \times V_\varphi)$, $\mathbf{a} \in \mathbf{A}$. Then

    $$prob(X \mid (q_1,q_2,C),\mathbf{a}) = \sum_{(q_1',q_2',C')\in X} \delta_1(q_1,\mathbf{a})(q_1') \cdot \delta_2(q_2,\mathbf{a})(q_2') \cdot [C \xrightarrow{\mathbf{a}}_\varphi C']$$

    where $[C \xrightarrow{\mathbf{a}}_\varphi C']$ is an indicator random variable which is equal to 1 when $C \xrightarrow{\mathbf{a}}_\varphi C'$ and 0 otherwise. Now let $C \xrightarrow{\mathbf{a}}_\varphi C'$. Then

$$\delta(\langle q_1, q_2, C, 1\rangle, \mathbf{a})(q_1', q_2', C', 1) = p \cdot \frac{\delta_1(q_1, \mathbf{a})(q_1') \cdot \delta_2(q_2, \mathbf{a})(q_2')}{prob(X \mid (q_1, q_2, C), \mathbf{a})}, \varphi \in C$$
$$= 0 \text{ otherwise.}$$

$$\delta(\langle q_1, q_2, C, 1\rangle, \mathbf{a})(q_1', q_2', C', 2) = (1-p)\frac{\delta_1(q_1, \mathbf{a})(q_1') \cdot \delta_2(q_2, \mathbf{a})(q_2')}{prob(X \mid (q_1, q_2, C), \mathbf{a})}, \varphi \in C$$
$$= 1 \text{ otherwise.}$$

$$\delta(\langle q_1, q_2, C, 2\rangle, \mathbf{a})(q_1', q_2', C', 1) = p \cdot \frac{\delta_1(q_1, \mathbf{a})(q_1') \cdot \delta_2(q_2, \mathbf{a})(q_2')}{prob(X \mid (q_1, q_2, C), \mathbf{a})}, \varphi \in C$$
$$= 0 \text{ otherwise.}$$

$$\delta(\langle q_1, q_2, C, 2\rangle, \mathbf{a})(q_1', q_2', C', 2) = (1-p) \cdot \frac{\delta_1(q_1, \mathbf{a})(q_1') \cdot \delta_2(q_2, \mathbf{a})(q_2')}{prob(X \mid (q_1, q_2, C), \mathbf{a})}, \varphi \in C$$
$$= 1 \text{ otherwise.}$$

- $f(q_1, q_2, C, i) = f_i(q_i)$.
- $val(q_1, q_2, C, i) = val_1(q_1) = val_2(q_2) = C \cap \mathcal{P}$.

The well-definedness of the construction is again similar to the previous constructions. A play ρ in the arena $\mathcal{A}$ corresponds to a unique vertex $C_\rho \in V_\varphi$ in the atom graph G_φ. We know from Proposition 1.5 that

$$(last(\rho), \mathbf{u}(\rho)) \models \varphi \text{ if and only if } \varphi \in C_\rho$$

From this, the correctness of the construction of $\mathcal{Q}_{\varphi ?^p s_1; s_2}$ is immediate.

4.6.4 Eventual dynamics

Given an initialised arena $(\mathcal{A}, v_0)$ where $\mathcal{A} = (V, E)$ and the strategy transducer $\mathcal{Q}_s = (Q, I, \delta, f, val)$ constructed as above for a probabilistic strategy specification $s \in \Sigma_i$ of player i, we define the restriction of $\mathcal{A}$ with respect to $\mathcal{Q}_s$ as $\mathcal{A} \restriction \mathcal{Q}_s = (\langle V', E'\rangle, v_0')$ where

- $V' \subseteq V \times Q$ such that $(v, q) \in V'$ iff $val(v) = val(q)$.

- $v'_0 = (\{v_0\} \times I) \cap V'$.

let $X \subseteq (V \times Q)$ be the event $X = \{(v,q) \mid val(v) = val(q)\}$. Let $(v,q) \in (V \times Q)$, $\mathbf{a} \in \mathbf{A}$. Then

$$prob(X \mid (v,q), \mathbf{a}) = \sum_{(v',q') \in X, \mathbf{a}(i) = f(q)} \delta(q, \mathbf{a})(q') \cdot [v \xrightarrow{\mathbf{a}} v']$$

where $[v \xrightarrow{\mathbf{a}} v']$ is an indicator random variable which is equal to 1 when $v \xrightarrow{\mathbf{a}} v'$ and 0 otherwise.

-

$$\begin{aligned} prob(\langle v,q\rangle, \mathbf{a}, \langle v',q'\rangle) &= 0 \text{ if it is not the case that } v \xrightarrow{\mathbf{a}} v' \\ &= 0 \text{ if } \mathbf{a}(i) \neq f(q) \\ &= \delta(q,\mathbf{a})(q')/prob(X \mid (v,q), \mathbf{a}) \text{ otherwise.} \end{aligned}$$

Thus given an initialised arena $(\mathcal{A}, v_0)$, and strategy specifications $s_1, s_2, \ldots, s_n$ for players 1 to n, we can define $(\mathcal{A}^*, v_0^*) = (\ldots((\mathcal{A} \restriction \mathcal{Q}_{s_1}) \restriction \mathcal{Q}_{s_2}) \ldots \restriction \mathcal{Q}_{s_n})$. Then by construction $\mathcal{A}^*$ is a finite markov chain and we can do all the usual analyses for markov chains on $\mathcal{A}^*$. In particular, as we know that every finite markov chain has a stationary distribution, Proposition 1.6, this distribution gives the eventual dynamics of the game.

Although computing the stationary distribution of a general Markov chain is a hard problem, there are several algorithms known for approximating the stationary distribution. Perhaps the most famous of these algorithms is the Markov Chain Monte Carlo (MCMC) algorithm [JS96].

Remark Like in the non-probabilistic case, we could have added the following rule $s_1 \cup^p s_2$, $0 \leq p \leq 1$ to Σ_i for combining strategies s_1 and s_2. $s_1 \cup^p s_2$ would have the semantics that player i initially chooses strategy s_1 with probability p and s_2 with probability $1-p$ and plays according to the chosen strategy. While constructing the transducer $\mathcal{Q}_{s_1 \cup^p s_2}$ for $s_1 \cup^p s_2$ from the transducers $\mathcal{Q}_{s_1}$ and $\mathcal{Q}_{s_2}$ for strategies s_1 and s_2 respectively, would have to add ϵ-transitions from the initial states to the initial states of $\mathcal{Q}_{s_1}$ and $\mathcal{Q}_{s_2}$ with probabilities p and $1-p$ respectively. But now, $s_1 \cup^p s_2$ in turn, may also be combined with other strategies using the given operators. One way to deal with the ϵ-transition in $\mathcal{Q}_{s_1 \cup^p s_2}$ is to add a dummy initial state to every constructed strategy-transducer with ϵ transitions to the actual initial states and to lift the probabilities associated with the ϵ-transitions accordingly. Care has to be taken even while taking the restriction of a

strategy $s \in \Sigma_i$ with the arena $\mathcal{A}$ because the ϵ-transitions have to be taken in account now.

Thus to integrate the $\cup^p$ operator into the syntax, a lot of extra book-keeping needs to be done not only for the strategy-transducer of $s_1 \cup^p s_2$ but also for all the other transducer constructions. We avoid doing so in the interest of readability with the assurance that the underlying concepts are the same.

Part II

Structure of strategy switching: rationale, consequences and visibility

Chapter 5

Dynamic restriction of choice

In Chapter 3 we studied games where switching strategies involve a cost. However, in general, playing a strategy or even maintaining a strategy itself might involve a cost. If not many players play a strategy, it may not be worthwhile to maintain the strategy. For eg., the government provides public transport to its citizens. But if not many people utilise the facility, the government runs the risk of incurring a loss. It may then be forced to increase the fares or maybe even withdraw the facility altogether. Thus, in many social situations, it is seen that the game arena does not remain static but keep changing dynamically. Some actions, positions etc. that were available to players initially may cease to be available as and when the game progresses. Players are then forced to change their strategies and this in-turn disrupts the balance of the available strategies. Hence this change is intrinsic and is brought about by the players playing the game.

In this chapter we look at such games that change intrinsically based on the actions / strategies played by the players. There is an implicit player - the society, who maintains the available actions of the players and incurs certain costs in making them available. If and when it feels that an action a is being played by a small number of players and/or it becomes too expensive for it to maintain the action a, it removes a from the set of available actions. This results in a change in the game and the players have to strategise afresh taking this change into account. We specify the restrictions of the society as well as the strategies of the players using a logical syntax similar to that of Chapter 4. We are interested in finding out what properties of the game and which of the players' actions remain stable eventually.

We then study the converse question: which actions of the players should the society restrict and how should it restrict them so that the social cost

is minimised in the eventuality? We address this question both in the case when the players are maximisers and when they play according to strategy specifications. [1]

5.1 Motivation

Outfitting oneself in India before the 1980s was an elaborate affair. One had to buy cloth from a store, decide on the design and style of the garment, and get it stitched by a tailor. Along the way, many personal preferences and current fashion trends would play a role. But gradually, as ready-made garments came into the market, it became clear that they were a cheaper and quicker option, though this severely limited one's say in the finer design details. As more people bought readmits, they became cheaper still, and with fewer customers, tailors had to charge more to sustain themselves. Today, there are very few practising tailors, and getting one's clothes custom made is a luxury. Whether this choice will even be available a few generations from now is unclear.

Economists are, of course, well aware of such phenomena. The availability of individual choices is, in general, determined by choices by the society as a whole, and in turn, social choices are influenced by patterns of individual choices. In this process, the set of choices may expand or contract over time.

However, there is a political or philosophical value attached to availability of individual choices. A strategy s_a may be justified by the presence of another option s_b but if eventually s_b is forced out, the rationale for s_a may disappear, though s_a is the only one present. In a world where all possible eventual consequences can be computed, the cost of such disappearance of choices can also be taken into account, but (as we see in the case of environment conservation) realisation typically comes post-facto.

To see this, consider a toll booth on a busy road which is manually operated. A vehicle driving through has to stop, tender cash and only then is allowed to proceed. Hence it is suggested that toll collection be RFID based. A vehicle equipped with RFID can speed through an automatic lane, and the requisite amount is debited from the bank account of the owner of the vehicle. While this is welcome, protesters point to loss of privacy, since the movements of the car owner can then be tracked. RFID is defended on the grounds that anyone worried about privacy can always use the lane with the manual booth. Thus speed and privacy are traded off against each

[1] This chapter is loosely based on the paper [PRS09a].

other and the RFID system is introduced. Gradually, as more people use the fast lanes, only one lane is operated manually, and there comes a point when the manual booth is removed on the grounds that it is too expensive to maintain. Interestingly, there is almost no public debate when this is done.

What happened to the trade-off between speed and privacy? It can be argued that a strategy valued by so few is socially a luxury to maintain, but the point remains that the rationale and terms of debate have changed entirely. While the question of whether this is acceptable or not is interesting for political philosophers, we suggest that there is at least a clear case for models that compute such eventual consequences of social decisions.

The general situation is as follows. At every stage, an individual has certain choices to make. But making a choice also comes with a cost which is associated with that choice and which the individual has to incur in making the choice. On the other hand, society also incurs a certain cost in making these choices available to individuals. This cost is a function of the choices being provided as well as the profile of choices made by individuals.

From time to time, based on the history of choice profiles and predictions of the future, society revises the choices it provides to individuals as well as the cost individuals have to incur to make these choices. This in turn has an effect on individuals' strategies, who switch between available choices. The dynamics of this back and forth process can be quite interesting and complicated.

Indeed, the decision on whether a facility should be provided as a part of social infrastructure (as opposed to being individually maintained, based on affordability) is based on such computations. Society may well decide that it is in its interest to ensure that everyone gets access to a facility, however uniform, rather than having a range of choices available but only to a subset of people. (As an example, consider Singapore offering free island-wide WiFi connectivity for four years.)

In game theoretic models of such social phenomena, social rules are considered as game forms, and individual behaviour is regulated using payoffs. Rule changes are considered to be exogenous, and correspond to change of payoff matrices. In evolutionary game theory, rules are considered as game equilibria: individuals following rules are players, and the desired properties of rules are given by equilibrium strategies, thus describing enforced rules. However what we discuss here is endogenous dynamics of these rules that takes into account the fact that individual behaviour and rules operate mutually and concurrently. In this sense, individual rationality and social rationality are mutually dependent, and what we seek to study are

the patterns of reasoning that inform such dependence.

We thus consider game forms which change dynamically, but according to pre-specified rules stated in a formal logic. If players had unlimited computational power, they could strategise about all possible game changes as well, but we consider players with bounded computational ability, who formulate initial strategic plans and revise them during course of play, based on observation (as described in Chapter 4). Such switching is again described logically. This, in turn, determines applicability of game changing rules, and so on. We can then ask, in this model, which action choices are eventually stable (in the sense that no further game changes will eliminate them), and under what conditions. We may also ask if a player eventually gets removed by the dynamics of the game, if eventually a particular action tuple becomes the only choice available for ever or if the cost stabilises to some specific amount. We show that these questions are algorithmically solvable.

We then look at the quantitative aspect of the choice-restriction phenomenon. We consider anonymous games where the cost incurred by the society in a particular round is given by a function of the action distribution of the players. The cumulative cost is the limit-average (mean-payoff) of these costs. We then ask whether it is possible for the society to synthesise rules for removal of actions of the players so that the eventual social cost is less than a certain threshold. We show that such synthesis is possible and that the rules require only finite memory.

5.2 Dynamic game restriction

The crucial elements for defining game restrictions are: *when* a restriction is to be carried out in the course of play, and *what* the effects of a restriction are. We choose a very simple answer to the latter, namely to eliminate a subset of choices at selected game positions, that is, to restrict the set of actions available to a player. The former is treated logically, to be defined in the next section, by tests for logical conditions.

Formally the restriction is triggered by a rule of the form $r = pre \supset \mathcal{A}'$ where *pre* is a precondition which is interpreted on partial plays and $\mathcal{A}'$ is a restriction of the arena. For an arena $\mathcal{A}$ and a partial (finite) play $\rho \in \mathcal{A}$, we say that the rule $r = pre \supset \mathcal{A}'$ is enabled at $(\mathcal{A}, \rho)$ if the following conditions hold.

- The partial play ρ conforms to the precondition *pre*.
- The arena $\mathcal{A}' = (V', E')$ is a sub-arena of $\mathcal{A}$.

- $last(\rho) \in V'$.

When the rule $r = pre \supset \mathcal{A}'$ is applied to a partial play ρ, the game proceeds to the new arena $\mathcal{A}'$ starting at the node $last(\rho)$.

5.2.1 Induced game tree

The restriction rules are specified along with the initial game arena. Let $R = \{r_1, \ldots, r_m\}$ be a finite set of restriction rules. For an arena $\mathcal{A}$, let $SA(\mathcal{A})$ denote the set of all subarenas of $\mathcal{A}$. Given an initialised arena $(\mathcal{A}, v_0)$ and a finite set of rules R, the extensive form game tree is the (infinite) tree $\mathcal{T}_{\mathcal{A}}(R) = (T, E)$ where $T \subseteq (V \times \mathbf{A}^* \times SA(\mathcal{A}))$ and is defined inductively as follows:

- $t_0 = (v_0, \epsilon, \mathcal{A})$ is the root.

- At any node $t = (v, \mathbf{u}, \mathcal{A}')$ of the tree, check if for a rule $(r_j = pre_j \supset \mathcal{A}_j) \in R$, it is the case that $t \models r_j$. If more than one rule is enabled at t then choose any one of them, say $pre_j \supset \mathcal{A}_j$. Let $\mathcal{A}_j = (V_j, E_j)$.

  - If $pre_j \supset \mathcal{A}_j$, then the subtree starting from t is the unfolding of the arena $\mathcal{A}_j$ from the vertex v. Note that $v \in V_j$ since we have ensured that $v = last(\rho(v_0, \mathbf{u})) \in V_j$.
  - If there is no such rule then the children of t are the same as those of t in the unfolding of $\mathcal{A}$ and the edge labels are also the same.

5.2.2 Strategising by players

In the presence of dynamic game restriction operations, the players can keep track of the restriction rules which are triggered by observing the history of play and adapt their strategies based on this information. A strategy specification for a player i would therefore be of the form $pre \supset a$ where, as earlier, pre is a precondition which is interpreted on partial plays and $a \in A_i$. The specification asserts that if a play ρ conforms to the precondition pre, then the action a is taken by the player.

Note that a strategy specification of this form is partial, since it does not constrain game positions at which the precondition does not hold; the player is free to choose any enabled action.

5.3 Logical specifications

In this section we show how game arena restrictions can be specified in a succinct manner in terms of homomorphisms and that restriction preconditions can be represented in a simple tense logic formalism.

5.3.1 Homomorphisms

Let $A = \bigcup_{i \in N} A_i$ be the set of actions of all the players. A homomorphism is a function $h : 2^A \to 2^A$ such that $h(X) \subsetneq X$. Given an arena $\mathcal{A}$ and a homomorphism h, the restriction of $\mathcal{A}$ with respect to h, $\mathcal{A}|_h$ is defined as follows. Every edge-label $\mathbf{a} = (a_1, \ldots, a_n)$ where there exists $j \in N$ such that $a_j \notin h(X)$ is removed from the arena. If the process leaves an edge without any label, the edge itself is removed.

A homomorphism on an arena is thus nothing but the removal of one (or more) action(s) from the set of available actions of the players. Thus, in order to describe a homomorphism, it is enough to specify the action(s) to be removed. However, given an action we may not wish to remove the action from an individual's choice at all possible points but only at selective ones. This can be achieved by associating the restriction with certain observables of the players.

5.3.2 Strategies and restrictions

We now formally describe how we can specify these restrictions that the society imposes on the actions of the players. Let $bool(\mathcal{P})$ be the set of boolean formulas over $\mathcal{P}$ (i.e. built using the syntax $p \in \mathcal{P} \mid \neg\beta \mid \beta_1 \vee \beta_2$). We also use the following abbreviations: $\top \equiv p \vee \neg p$ and $\bot \equiv p \wedge \neg p$. The truth of a formula $\beta \in bool(\mathcal{P})$ at a game position v, denoted $v \models \beta$ is defined as follows:

- $v \models p \in \mathcal{P}$ iff $p \in val(v)$.
- $v \models \neg\beta$ iff $v \not\models \beta$.
- $v \models \beta_1 \vee \beta_2$ iff $v \models \beta_1$ or $v \models \beta_2$.

Given an arena $\mathcal{A}$ the restriction rules imposed by the society consists of a collection of specification of the form $\psi \supset h$, where ψ is a precondition specification and h is a specification of the homomorphism. The formal syntax and semantics is presented below

Syntax of homomorphism specifications

Homomorphisms are specified using the following syntax:

$$\mathfrak{h} ::= h_{\beta:a} \mid h_1 \wedge h_2, \text{ where } a \in A \text{ and } \beta \in \mathit{bool}(\mathcal{P}).$$

Semantics

For an arena $\mathcal{A}$ and a homomorphism specification h, we define the restriction of $\mathcal{A}$ with respect to h (denoted $\mathcal{A}|_h$) inductively as follows:

- $h \equiv h_{\beta:a}$: $\mathcal{A}|_{h_{\beta:a}}$ is derived from $\mathcal{A} = (V, E)$ as follows. For every vertex $v \in V$ such that $v \models \beta$, the action a is removed from the set of available actions of all the players. That is, the new set of available actions of every player i becomes $\Gamma_i(v) \setminus \{a\}$.

- $h \equiv h_1 \wedge h_2$: $\mathcal{A}|_{h_1 \wedge h_2} = (\mathcal{A}|_{h_1})|_{h_2}$

Note that using the above notation the removal of all 'a' actions in the arena can be specified by $h_{\top:a}$ and the removal of a player i from the arena by $\bigwedge_{a \in A_i} h_{\top:a}$.

Syntax and semantics of restriction precondition

A restriction precondition ψ comes from the syntax Φ (see Section 1.2.9) using which we can specify properties of the indefinite past and bounded future. ψ is evaluated on $\mathcal{T}_{\mathcal{A}}(R)$ as usual.

The modality $\boxminus\psi'$ makes assertion about the unbounded past, it specifies the transitive closure of the one step past operator. We can define the corresponding construct for future, $\Box\psi'$ which makes assertions about unbounded future. The technical results go through even with the addition of this construct. However, for the applications we have in mind, this construct is not required.

Strategy Specifications

The strategy of players depend on properties of the history of the play. These can therefore be specified as a collection of formulae of the form $\varphi \supset a$ where $\varphi \in \Phi_-$ (Section 1.2.9).

5.3.3 Capturing costs in the logical formalism

Following a strategy induces a certain cost for the player. The distribution of strategies chosen by players carry a social cost. We first take an abstract view of costs associated with individual players and social costs associated with providing facilities. The social cost typically depends on the history of the choices made by players in the past. When the social cost crosses some pre-defined threshold, it might be socially optimal to make certain facilities part of the common infrastructure which reduces the individual costs.

When the costs arise from a fixed finite set, they can be coded up using propositions in the logical framework on the lines of [Bon01]. The cost c (say) can be represented using the proposition p_c and orderings are inherited from the implication available in the logic. Furthermore, costs can be dependent on the actions enabled at a game position. This can also be easily represented in the logical formalism by making use of the one step future modality.

5.3.4 Examples

Revisiting the tailor example, suppose there are two players 1 and 2. Each of them have two choices initially: t for going to a tailor and r for opting for ready-made. Suppose initially the social cost is 5 units. Suppose the cost functions are as follows: the cost of going to a tailor is 2/5 times the social cost and the cost of going for a ready-made is 3/5 times the social cost. Also suppose that initially both players play t. Player 1 has the condition that if at any point, the cost of t becomes 2.5 or more then she switches to r and player 2 has the condition that if at any point, the cost of t becomes 3 or more then she switches to r. Suppose the propositions for the social costs are $\{p_4, p_5, p_7, p_8\}$ where p_4 is supposed to mean that the social cost is 4 units and so on. Then the strategy of player 1 is:

$$\Sigma_1 = \{\ominus(p_4 \vee p_5) \supset t, \ominus(p_7 \vee p_8) \supset r\}$$

and that of player 2 is:

$$\Sigma_2 = \{\ominus(p_4 \vee p_5 \vee p_7) \supset t, \ominus p_8 \supset r\}.$$

Now suppose after 2 moves, the social cost rises to 7. This is modelled in the arena by having paths $v_0 \xrightarrow{(\cdot,\cdot)} v_1 \xrightarrow{(\cdot,\cdot)} v_2$ where $p_7 \in val(v_2)$. Then player 1 switches to play r. Also suppose the social cost increases to 8 when 1 of the players play r. This is modelled in the arena by having paths $v \xrightarrow{(r,\cdot)/(\cdot,r)} v'$

where $p_8 \in val(v')$. Then player 2 also switches to r. Further suppose if the social cost increases to 8 then the society decides to do away with all the tailors. This is given by the restriction specification $\ominus p_8 \mapsto h_{\top:t}$.

5.3.5 Stability

Let $(\mathcal{A}, v_0)$ be an initialised arena, R be a finite set of game restriction rules, $\{\Sigma_i\}_{i\in N}$ be a finite set of strategy specifications or each player $i \in N$. Let α be a formula from the syntax:

$$\alpha ::= \alpha \in bool(\mathcal{P}) \mid \langle \mathbf{a} \rangle^+ \alpha$$

We say α is stable in $(\mathcal{A}, R, \{\Sigma_i\}_{i\in N})$ if there exists a sub-arena $\mathcal{A}'$ such that for all game positions $t \in \mathcal{T}_{\mathcal{A}'}$, we have: $t \models \alpha$. Thus stability with respect to an observable property captures the existence of a subarena to which the game stabilises under the dynamics specified by R and $\{\Sigma_i\}_{i\in N}$. For the applications we consider, we do not require the full power of temporal logic for α.

5.4 Decidability

In this section we present the main result of this chapter. The question addressed here is representative of the kind of questions one can ask and prove of the model.

Theorem 5.1 *Given an initialised arena $(\mathcal{A}, v_0)$, a finite set of restriction rules R, a finite set of strategy specifications $\{\Sigma_i\}_{i\in N}$ and a formula α, the following question is decidable:*

- *Is α stable in $(\mathcal{A}, R, \{\Sigma_i\}_{i\in N})$?*

Proof Outline The proof proceeds in three steps. In Step 1, we construct a finite state automaton for each of the restriction rules. This automaton runs on the arena and keeps track of the restriction preconditions that are satisfied. It maintains a set X which is the set actions that are not available to the players anymore. Whenever a precondition for some restriction rule holds, it updates the set X by adding the action that is dictated by that rule.

In Step 2, we construct finite state transducers for each of the players. These again run on the arena and output actions according to the strategy specifications of the players. However, if an action is not available, that

is, if it is present in the current set of restricted actions, X, they non-deterministically output any available action.

In Step 3, we construct a master transducer that simulates all of these transducers in parallel. Finally in Step 4, we take the restriction of the arena with respect to the master transducer and infer about the stability of α. $\square$

Proof Let $R = \{(\psi_1 \supset h_1), \ldots, (\psi_m \supset h_m)\}$ be the set of restriction rules and $\Sigma_i = \{(\varphi^i_1 \supset a^i_1), \ldots, (\varphi^i_{k_i} \supset a^i_{k_i})\}$ be the strategy specification for each player i. Let $CL(\alpha)$ denote the sub-formula closure of a temporal formula α. For a homomorphism specification h, let $\ell(h)$ denote the set of all atomic homomorphism specifications in h. Let $H = \{h_1, \ldots, h_m\}$ and let $\ell(H) = \ell(h_1) \cup \ldots \cup \ell(h_m)$. The proof is carried out in the following steps.

Step 1. For each of the restriction rules $\psi_j \supset h_j$, we construct a finite state automaton $\mathcal{Q}_j = (Q_j, \delta_j, I_j, F_j)$ over the alphabet $2^{\mathcal{P}}$. $\mathcal{Q}_j$ is just the atom graph of ψ_j (Section 1.2.9). The set of final states $F_j = \{q \in Q_j \mid \psi_j \in q\}$. We then construct the restriction automaton $\mathcal{Q}$ which runs $\mathcal{Q}_1, \ldots, \mathcal{Q}_m$ in parallel. In addition, it also keeps track of the set $X \subseteq 2^{\ell(H)}$ of atomic homomorphisms which are enabled. The set X is updated by the behaviour of the individual automata $\mathcal{Q}_j$. At any point when the automaton $\mathcal{Q}_j$ indicates that ψ_j holds, the rule is triggered and $\ell(h_j)$ is added to X. A formal definition of the automaton $\mathcal{Q}$ is given below.

The restriction automaton $\mathcal{Q}$ is a tuple $\mathcal{Q} = (Q, \delta, I)$ over the alphabet $\prod_{i \in N} A_i$ where

- $Q \subseteq \prod_{j=1}^{m} Q_j \times 2^{\ell(H)}$ such that $(q_1, \ldots, q_m, X) \in S$ iff $q_1 \cap \mathcal{P} = \ldots = q_m \cap \mathcal{P}$.
- $I = (I_1 \times \ldots \times I_m \times \{\emptyset\}) \cap Q$. That is, at the initial state, all the actions of every player is available.
- $(\langle q_1, \ldots, q_m, X\rangle, \mathbf{a}, \langle q'_1, \ldots, q'_m, X'\rangle) \in \delta$ iff $(q_j, \mathbf{a}, q'_j) \in \delta_j \ \forall j : 1 \leq j \leq m$ and
  - $X' = X \cup \ell(h_k)$ if $q_k \in F_k$. That is, if the kth restriction has been enabled then the automaton keeps track of it by adding it to the set X.
  - $X' = X$ otherwise.

For a state $q = (q_1, \ldots, q_m, X) \in Q$, we let $val(q) = q_1 \cap \mathcal{P} = \ldots = q_m \cap \mathcal{P}$.
Step 2. For each of the strategy specifications $\varphi^i_j \supset a^i_j$, we first construct a finite state automaton $\mathcal{S}_{\varphi^i_j} = (S_{\varphi^i_j}, \delta_{\varphi^i_j}, I_{\varphi^i_j}, F_{\varphi^i_j})$ which keeps track

of whether φ^i_j holds at a game position. As earlier, the automaton $\mathcal{S}_{\varphi^i_j}$ is just the atom graph of φ^i_j.

For every player i, we construct a finite state transducer $\mathcal{S}_i$ which generates the strategy of i in conformance with the specifications Σ_i. It simulates the automata $\Sigma_{\varphi^i_j}$ for all j. It also simulates the restriction automaton $\mathcal{Q}$ in parallel. At every position suppose $\varphi^i_{j_1}, \ldots, \varphi^i_{j_l}$ holds at that position. $\mathcal{S}_i$ chooses one of $\varphi^i_{j_1}, \ldots, \varphi^i_{j_l}$ non-deterministically, say $\varphi^i_{j_*}$. $\mathcal{S}_i$ then outputs action $a^i_{j_*}$ iff $a^i_{j_*}$ has not already been restricted by the restriction automaton $\mathcal{Q}$. Otherwise it non-deterministically outputs any available action. However, if $l = 0$, that is, none of the φ^i_j's hold, then again it non-deterministically outputs any available action.

The formal automaton construction of $\mathcal{S}_i$ is given below. $\mathcal{S}_i$ is a tuple $\mathcal{S}_i = (S_i, \delta_i, I_i, f_i)$ over input alphabet $\prod_{i \in N} A_i$ and output alphabet A where S_i is the set of states, δ_i is the transition relation, I_i is the initial state and f_i is the output function. The output function f_i generates the strategy of player i.

- $S_i \subseteq \prod_{j=1}^{k_i} S_{\varphi^i_j} \times Q \times A_i$ where Q is the state space of $\mathcal{Q}$ such that $(s_1, \ldots, s_{k_i}, q, a) \in S_i$ where $q = (q_1, \ldots, q_m, X)$ iff
  - $s_1 \cap \mathcal{P} = \ldots = s_{k_i} \cap \mathcal{P} = val(q)$.
  - $a \in A_i \setminus X$ such that $a = a^i_j$ iff s_j is a final state of $\mathcal{S}_{\varphi^i_j}$.
- $I_i = (\prod_{j=1}^{k_i} I_{\varphi^i_j} \times I \times A_i) \cap S_i$
- $f_i(s_1, \ldots, s_{k_i}, q, a) = a$
- $(\langle s_1, \ldots, s_{k_i}, q, a\rangle, \mathbf{a}, \langle s'_1, \ldots, s'_{k_i}, q', a'\rangle) \in \delta_i$ iff $(s_i, \mathbf{a}, s'_i) \in \delta_{\varphi^i_j}$, $\forall j$: $1 \leq j \leq k_i$ and $(r, \mathbf{a}, r') \in \delta$.

Step 3. A transducer $\mathcal{S}$ simulates all the $\mathcal{S}_j$s, $1 \leq j \leq n$, in parallel. That is, $\mathcal{S}$ is a product of all the $\mathcal{S}_j$s. Its output are action tuples which are the actions output by the individual transducers, $\mathcal{S}_j$'s. The restriction automaton $\mathcal{Q}$ operates on the output of $\mathcal{S}$. Finally a master transducer $\mathcal{M}$ simulates $\mathcal{Q}$ and $\mathcal{S}$ in parallel. $\mathcal{M}$ is a product of $\mathcal{Q}$ and $\mathcal{S}$ and its output is the same as that of $\mathcal{S}$.

Figure 5.1 shows the interdependence between the various automata.

Step 4. Let $\mathcal{M} = (M, \delta, I, f)$ be the master transducer constructed as above. For $q \in M$, we say that $val(q) = P$ iff for each component q_i of q,

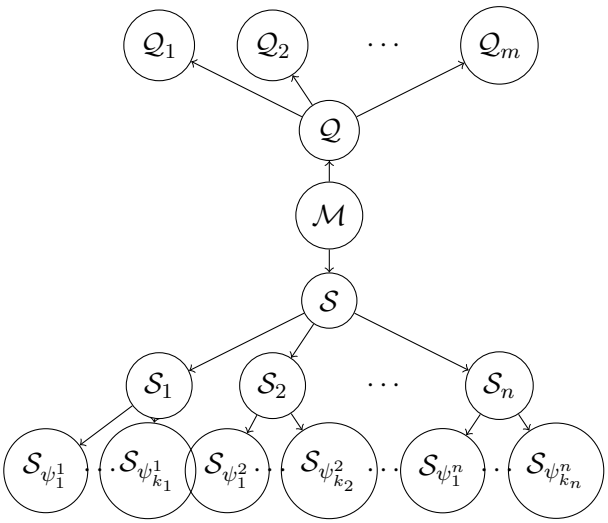

Figure 5.1: Interdependencies among the various automata

which are states of the restriction automaton $\mathcal{Q}$ and the strategy transducer $\mathcal{S}$, $q_i \cap \mathcal{P} = P$.

We define the restriction of the arena with respect to $\mathcal{M}$, $\mathcal{A} \restriction \mathcal{M}$ as follows. $\mathcal{A} \restriction \mathcal{M} = (V', E')$ with initial vertex set v_0' where

- $V' \subseteq V \times M$ such that $(v, q) \in V'$ iff $val(v) = val(q)$.
- $E' \subset V' \times V'$ such that $(v, q) \xrightarrow{\mathbf{a}} (v', q')$ iff $v \xrightarrow{\mathbf{a}} v'$ and $(q, \mathbf{a}, q') \in \delta$ and $f(q)(i) = \mathbf{a}(i)$ for all $1 \leq i \leq n$ and $val(v') = val(q')$.
- $v_0' = (\{v_0\} \times I) \cap V'$.

To answer the stability question, we construct the restricted graph $\mathcal{A} \restriction \mathcal{M}$ as described above. We then

- Check if there is a maximal connected component F in $\mathcal{A} \restriction \mathcal{M}$ and whether all paths starting from all initial vertices reach F. If not, then we output 'NO' and quit.
- Check if α holds at all the game positions in F. If so output 'YES', else output 'NO'.

Correctness: Let r be a run of any of the automata/transducers constructed above, corresponding to a precondition φ, on the tree unfolding $\mathcal{T}_{\mathcal{A}}$

of the arena (Section 1.2.8). Then we can show by an argument, similar to the proof of Claim 4.7, that

Claim 5.2 *For all $t \in \mathcal{T}_{\mathcal{A}}$ and for all $\varphi' \in CL(\varphi)$, $\varphi' \in val(r(t))$ iff $t \models \varphi'$.*

Hence the restriction automaton $\mathcal{Q}$ correctly keeps track of all the restriction-precondition. Thus each of the strategy-automata $\mathcal{S}_i$ outputs an action a corresponding to a strategy specification such that a has not been restricted and is available to the player i. □

Corollary 5.3 *Given an arena $\mathcal{A}$ and specifications R and $\{\Sigma_i\}_{i\in N}$, the following questions are decidable:*

1. *Does player i eventually get removed by the dynamics of the game?*
2. *Does a particular action tuple $\mathbf{a}$ become the only choice available for ever?*

Proof In each case, we come up with a formula α using the coding mentioned in Section 5.3.3 such that answering the question amounts to checking the stability of α in $(\mathcal{A}, R, \{\Sigma_i\}_{i\in N})$, which is decidable by Theorem 5.1.

For (1), we can code the positions of player i using a proposition $turn_i$ and check if $\alpha = \neg turn_i$ is stable in $(\mathcal{A}, R, \{\Sigma_i\}_{i\in N})$. This asks whether it is the case that the rules of the society and the behaviour of other players drive a particular player out of the game. The negation of this question can also be answered: Does player i survive till the end of the game?

For (2), we check if $\alpha = \langle \mathbf{a} \rangle^+ \top \wedge \bigwedge_{\mathbf{a}_* \neq \mathbf{a}} \langle \mathbf{a}_* \rangle^+ \bot$ is stable in $(\mathcal{A}, R, \{\Sigma_i\}_{i\in N})$. This corresponds to deciding whether the action tuple $\mathbf{a}$ eventually becomes part of the social infrastructure. The choices available to players disappear in such a scenario. □

Consequences of Theorem 5.1

Theorem 5.1 implies that comparison between game restriction rules in terms of their imposed social cost is possible. Suppose the 'type' of players is known in terms of the strategy specification employed (note that we do not insist on knowing the exact strategy) and we have two sets of game restriction rules R_1 and R_2. It is possible to compute the social cost with respect to R_1 and R_2 and deduce which is better suited. From the players' perspective, if the game restriction rules are known and the type of other players are known, then they can compare between their strategy specifications. For instance,

in the tailor example, this process might help a tailor to adapt better to the competition from ready-made manufacturers. He might be able to change his service into something of a hybrid form where the basic stitching itself is mechanised with respect to a fixed range of sizes. However, certain specific personalisation can be done by employing fewer number of workers, thereby being cost efficient.

5.5 Quantitative objectives

In this section we change our model to one where the costs (both social and individual) are given by certain functions instead of being coded up as propositions as before. We then ask whether it is possible for the society to restrict the actions of the players in such a way that the social cost stays within a certain threshold. We first develop some preliminaries.

5.5.1 Preliminaries

As before we let $N = \{1, 2, \ldots, n\}$ be the set of players. However we assume that the players have a common action set A, that is, $A_1 = \ldots = A_n = A$.

We study anonymous games [Blo99, Blo00, DP07, BFH09] because as argued in Chapter 1, in large games, the payoffs are usually dependent on the 'distribution' of the actions played by the players rather than the action profiles themselves. Moreover, in such games the payoffs are independent of the identities of the players. An action distribution is a tuple $\mathbf{y} = (y_1, y_2, \ldots, y_{|A|})$ such that $y_i \geq 0,\ \forall i$ and $\sum_{i=1}^{|A|} y_i \leq n$. Let $\mathbf{Y}$ be the set of all action distributions. Given an action profile $\mathbf{a}$, we let $\mathbf{y}(\mathbf{a})$ be its corresponding action distribution, that is, $\mathbf{y}(\mathbf{a})(k)$ gives the number of players playing the kth action in A.

Now as the payoffs are dependent on the action-distribution of the players, we convert the arena $\mathcal{A}$ to a new arena $\mathcal{A}[\mathbf{Y}]$ so that the payoffs can be assigned to the vertices of the arena. $\mathcal{A}[\mathbf{Y}]$ is defined as $\mathcal{A}[\mathbf{Y}] = (V[\mathbf{Y}], E[\mathbf{Y}])$ as follows:

- $V[\mathbf{Y}] = V \times \mathbf{Y}$
- $E[\mathbf{Y}] \subseteq V[\mathbf{Y}] \times \mathbf{A} \times V[\mathbf{Y}]$ such that $(v_1, \mathbf{y}_1) \xrightarrow{\mathbf{a}} (v_2, \mathbf{y}_2)$ iff $v_1 \xrightarrow{\mathbf{a}} v_2$ and $\mathbf{y}(\mathbf{a}) = \mathbf{y}_2$.
- Delete all vertices in $V[\mathbf{Y}]$ that do not have any incoming or outgoing edges.

As we shall exclusively deal with the expanded arena $\mathcal{A}[\mathbf{Y}]$ in the entire development, we denote $\mathcal{A}[\mathbf{Y}] = (V[\mathbf{Y}], E[\mathbf{Y}])$ by just $\mathcal{A} = (V, E)$ and assure that it will not result in any confusion. A tuple $(\mathcal{A}, v_0)$ where $\mathcal{A}$ is an arena and v_0 is a distinguished vertex is called an initialised arena.

Every player i has a function $f_i : \mathbf{Y} \to \mathbb{Q}$ which can be seen as the payoff of i for a particular distribution. There is also a function $f : \mathbf{Y} \to \mathbb{Q}$ which can be viewed as the cost incurred by the society for maintaining the actions. These functions can be lifted to the vertices of $V[\mathbf{Y}]$ as $f(v, \mathbf{y}) = f(\mathbf{y})$.

5.5.2 Rule synthesis

In this section we investigate if it is possible for the society to impose restrictions in such a way that the social cost stays within a certain threshold. We look at two variations of our model:

a. At the beginning of each round the society chooses an order for the n players and makes it known to them. The players then choose their actions according to this order.

b. The players play according to strategy specifications (as in the previous sections). The society, at any point, can restrict some action $a \in A$ of the players, in that, it can make the action a unavailable.

In (a), we wish to investigate if it is possible for the society to pick the orders of actions of the players in such a way that the eventual social cost is within a certain threshold. In (b), we want to find out if the society can restrict the action of the players based on certain rules so that the same effect is obtained.

Restriction of order

The game proceeds in rounds. At the beginning of every round, the society chooses an order for the players to play and makes it known to them. The players choose actions in that particular order. These actions define a tuple $\mathbf{a} \in \mathbf{A}$ and the play moves along the edge labelled $\mathbf{a}$ to the next vertex. This process goes on forever. Given an initial vertex v_0 this defines an infinite play $\rho = v_0 \xrightarrow{\mathbf{a}_1} v_1 \xrightarrow{\mathbf{a}_2} \dots$ in the arena. We study the discounted-payoff Player i gets:

$$p_i(\rho) = \lim_{n\to\infty} \inf \frac{1}{n} \sum_{j=1}^{n} f_i(v_j).$$

Similarly the society incurs a cost of:

$$c(\rho) = \lim_{n\to\infty} \inf \frac{1}{n} \sum_{j=1}^{n} f(v_j).$$

There is a threshold cost θ. The aim of each player i is to play in such a way that the quantity p_i is maximised and the aim of the society is to choose the orders in such a way that the quantity $c(\rho)$ is always less than θ for every play ρ no matter what actions the players play according to the order it selects. We are interested in the following question:

Question *What orders can the society choose so that the social cost c always remains less than the threshold θ?*

We first define a normalised version of the game where we subtract θ from the cost associated with every vertex of the arena $\mathcal{A}$. In other words, we define a new function $\tilde{f}$ from f such that $\tilde{f}(v) = f(v) - \theta$ for every $v \in V$. For a play ρ in $\mathcal{A}$ we let

$$\tilde{c}(\rho) = \lim_{n\to\infty} \inf \frac{1}{n} \sum_{j=1}^{n} \tilde{f}(\rho(j)).$$

Note that for any play ρ in $\mathcal{A}$ $c(\rho) < \theta$ iff $\tilde{c}(\rho) < 0$.

Now, to answer the above question, we first define a tree unfolding $\mathcal{T}_{\mathcal{A}}$ of the initialised arena $(\mathcal{A}, v_0)$. The unfolding is slightly different from the one described in Section 1.2.5. It also takes into account the order in which the players choose their actions. $\mathcal{T}_{\mathcal{A}}$ is constructed inductively, the set of nodes being:

$$T \subseteq (V \times \{soc\}) \cup (V \times N \times \pi(N))$$

where $\pi(N)$ is the set of permutations of 2^N (the subsets of N) such that $(v, j, w) \in \mathcal{T}_{\mathcal{A}}$ only if $j = w(1)$. The root node is (v_0, soc). Suppose $\mathcal{T}_{\mathcal{A}}$ has been constructed till level i. Consider an unprocessed node at level i.

- If this node is of the form (v, soc) and if (v, soc) has an identical ancestor already present in $\mathcal{T}_{\mathcal{A}}$ constructed so far, then we call (v, soc) a leaf node and do not process is any further. Otherwise the set of children of (v, soc) correspond to all the permutations (orders) for the following round. In other words, the set of children of (v, soc) are of the form $(v, j, w) \in T$ where $w \in \pi(N)$.

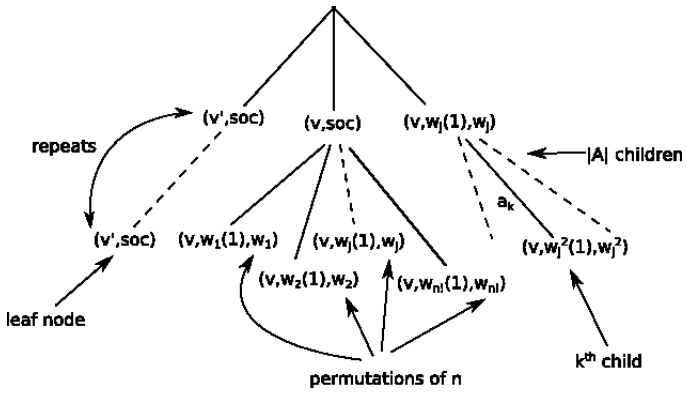

Figure 5.2: The unfolding

- If the node is of the form (v, j, w) and $|w| > 1$ then its children correspond to all the possible actions that j can choose. That is, it has $|A|$ many children of the form $(v, k, w^2) \in T$. The edge from (v, j, w) to the ℓth child is labelled with $a_\ell \in A$. If $|w| = 1$ then again (v, j, w) has $|A|$ children, the ℓth edge being labelled with $a_\ell \in A$ such that the following holds. (v', soc) is a child of (v, j, w) if and only if the actions corresponding to the path from (v, soc) to (v', soc) in the tree give the action tuple $\mathbf{a}$ and v' is the neighbour of v along the edge labelled $\mathbf{a}$.

See Figure 5.2 for an illustration. The above procedure is repeated until all the branches have seen a leaf node and there are no more nodes to process. Note that as the set T is finite, the procedure does terminate. We then define a backward induction procedure on $\mathcal{T}_\mathcal{A}$ as follows. In the process, we construct another tree $\mathcal{T}^*_\mathcal{A}$ which is a subtree of $\mathcal{T}_\mathcal{A}$ and which gives the orders that the society can choose so that the social cost always remains less than the threshold.

Procedure 2

- Label the leaf nodes with tuples from $\mathbb{Q}^{n+1}$ as follows. For every leaf node (v, soc) there exists an identical ancestor. This means that on this branch, the game has settled down to a simple cycle involving the vertex v. Let $C = v_0 \xrightarrow{\mathbf{a}_1} v_1 \xrightarrow{\mathbf{a}_2} \ldots \xrightarrow{\mathbf{a}_k} v_k$ where $v_0 = v_k = v$ be this cycle. Label (v, soc) with $(p_0(C), p_1(C), \ldots, p_{n+1}(C))$ where:

$$p_0(C) = \sum_{j=1}^{k} \tilde{f}(v_j)$$

and

$$p_i(C) = \sum_{j=1}^{k} f_i(v_j),\ i \in N.$$

- For a non-leaf node, suppose all its children have been labelled.
  - If the non-leaf node is of the form (v, j, w) let L be the set of labels of its children. Let $L_j = \{p_j \mid (p_0, \ldots, p_n) \in L\}$. Let $m_j = \max L_j$ and let $(p_0, \ldots, p_n) \in L$ be such that $p_j = m_j$. Label (v, soc) with $(p_0, \ldots, p_n)$.
  - If the non-leaf node is of the form (v, soc) let L be the set of labels of its children. Let $L_{<0} = \{p_0 \mid (p_0, \ldots, p_n) \in L \text{ and } p_0 < 0\}$ and let $L_{\geq 0} = L \setminus L_{<0}$. If $L_{<0} \neq \emptyset$ then label (v, soc) with $(p_0, \ldots, p_n) \in L_{<0}$. Otherwise label (v, soc) with $(p_0, \ldots, p_n) \in L_{\geq 0}$. Delete the subtrees rooted at every child of (v, soc) whose label is in the set $L_{\geq 0}$.

The above backward induction procedure generates a subtree $\mathcal{T}^*_{\mathcal{A}}$ which is the strategy of the society, in that, $\mathcal{T}^*_{\mathcal{A}}$ gives all the orders that the society can choose so that the social cost always remains less than the threshold. When a play reaches a vertex $v \in V$, the society chooses a permutation w of the players which dictated by $\mathcal{T}^*_{\mathcal{A}}$, in that, there is a node $(v, soc) \in \mathcal{T}^*_{\mathcal{A}}$ such that it has a child of the form (v, j, w).

Proposition 5.4 *$\mathcal{T}^*_{\mathcal{A}}$ gives all the orders that the society can choose so that the social cost c always remains less than the threshold θ.*

Proof The fact that it is enough to unfold the arena till a cycle is completed and that it is enough for the players to play memoryless strategies follows from [EM79]. Then it is clear that if the society enforces the orders as given by $\mathcal{T}^*_{\mathcal{A}}$ and the players play to maximise their own respective payoffs, then the social cost remains less than θ. Conversely, suppose the society enforces an order that is not given by $\mathcal{T}^*_{\mathcal{A}}$. Then again the correctness of the backward induction procedure and the fact that the players play memoryless strategies implies that since the players play to maximise their own respective payoffs, the game settles down to a cycle such that the social cost is greater than θ. □

The following corollary is immediate from the above proposition.

Corollary 5.5 *When the society can specify the order in which the players choose their actions, it has a finite memory strategy to ensure that the social cost remains within a given threshold.*

Restriction of action

In this section we study if and how the society can restrict the actions of the players so as to keep the social cost within a certain threshold. The players play according to strategy specifications. Let A be the common set of actions for the players. The specifications of player i is a set Σ_i whose elements are of the form $\varphi_i \supset a$ where $a \in A$ and φ_i is a temporal formula from a syntax similar to Φ_- except that the atomic formulas also have propositions of the form $p_i(x) < d,\ d \in \mathbb{Q}$.

A formula φ is evaluated on finite paths (plays/histories) which are nodes of the unfolding of the arena $\mathcal{T}_{\mathcal{A}}$ as described in Section 1.2.9. The truth of the atomic proposition $p_i < d$ at a node t, $t \models p_i$ is defined as:

- $t \models p_i < d$ iff
  - there exists a prefix t' of t such that $\mathit{last}(\rho(t')) = \mathit{last}(\rho(t)) = v$ and
  - let t'' be the longest prefix of t such that $\mathit{last}(\rho(t'')) = v$ and let C be the cycle from t'' to t. Then we have $\sum_{j=1}^{|C|} f_i(C(j)) < d$.

That is, $p_i < d$ holds as a node t if the play has settled down to a cycle C on that branch and the cumulative reward to player i from that cycle is less than d.

Among other possible propositions, the set $\mathcal{P}$ has propositions of the form $p_{a\times}$ which if true at a vertex v says that the action a is not available at v. Using these propositions, players can form observables like

$$\diamondsuit(p_i(x) \geq d \wedge p_{a\times}) \supset a'$$

which says that if in the past the play settled down to a loop with utility greater than d and the society restricted the action a then play a'.

Each player i has a set Σ_i of strategy specifications of the form $\varphi \supset a$. The players play according to their strategy specifications. For a history ρ and for a player i, if there exists a specification $\varphi \supset a \in \Sigma_i$ such that $\rho \models \varphi$ then she plays action a. Otherwise she plays any action from her set of actions A non-deterministically.

The society can remove certain actions from the available set of actions of the players at a particular vertex v. The removal of an action a results in the following. For every edge in the arena, if the edge has a label $\mathbf{a}$ such that $\mathbf{a}(i) = a$ for some $i \in N$ then the label $\mathbf{a}$ is removed. If this results in an edge without a label, then the edge itself is removed. Like in the previous case, the aim of the society is to restrict the actions of the players in such a way as to make the game eventually settle down so that the social cost is less than a certain threshold.

We thus wish to generate a set of rules (a finite state automaton) $\mathcal{M}$ for the society, such that if it restricts actions of the players as prescribed by $\mathcal{M}$ then the game always settles down so that the social cost is less than a threshold even though the players play according to their specifications.

Given a formula $\varphi \in \Phi_-$, let $AT(\varphi)$ denote the set of atoms of φ. We first construct a set $AT_{up}(\varphi)$ from $AT(\varphi)$ as follows. For every $D \in AT(\varphi)$ we do the following:

- Let $D = D_1 \sqcup D_2$ where D_1 consists only of formulae of the form $\langle \mathbf{a} \rangle^- \varphi'$. If $D_1 = \emptyset$ then we let $D \in AT_{up}(\varphi)$. Otherwise
- For every $\langle \mathbf{a} \rangle^- \varphi' \in D_1$ we define the following sets:
  - $D_{\langle \mathbf{a} \rangle^- \varphi'} = \{\langle \mathbf{a} \rangle^- \varphi'\}$.
  - $D'_{\langle \mathbf{a} \rangle^- \varphi'} = \{\neg \langle \mathbf{b} \rangle^- \varphi' \mid \langle \mathbf{b} \rangle^- \varphi' \in D_1,\ \mathbf{b} \neq \mathbf{a}\}$.
  - $D'' \subseteq D_2$ such that D'' is a maximal subset of D_2 that is consistent with $D_{\langle \mathbf{a} \rangle^- \varphi'} \cup D'_{\langle \mathbf{a} \rangle^- \varphi'}$.

  We let $D_{\langle \mathbf{a} \rangle^- \varphi'} \cup D'_{\langle \mathbf{a} \rangle^- \varphi'} \cup D'' \in AT_{up}(\varphi)$.

Every set $D \in AT_{up}(\varphi)$ contains at most one formula of the form $\langle \mathbf{a} \rangle^{-} \varphi'$. We call such a set a unique-past atom.

We construct another set $AT_{up-c}(\varphi)$ which are the set of unique past atoms without the (cost) formulae of the form $p_i < d$. Formally, for every $D \in AT_{up-c}(\varphi)$, let $D' = D \setminus (\{p_i < d \mid i \in N,\ p_i \in D\} \cup \{\neg(p_i < d) \mid i \in N,\ p_i \in D\})$. For all maximal subsets D'' of D' such that D'' does not have any inconsistency, let $D'' \in AT_{up-c}(\varphi)$.

Let the modal depth of a formula φ, $md(\varphi)$ be defined as in Section 4.5.2 inductively as:

- $md(p) = 0$.
- $md(\neg\varphi') = md(\varphi')$.
- $md(\varphi_1 \vee \varphi_2) = \max\{md(\varphi_1), md(\varphi_2)\}$.
- $md(\langle \mathbf{a} \rangle^{-} \varphi') = md(\varphi') + 1$.
- $md(\Diamond\varphi') = md(\varphi') + 1$.

Let $M_i = \max_{\varphi \supset a \in \Sigma_i}\{md(\varphi)\}$ and $M = \max_i M_i$.

We now describe an unfolding of the initialised arena $(\mathcal{A}, v_0)$ combined with the unique-past atoms of the strategy specifications. The unfolding is a tree $\mathcal{T}_{\mathcal{A}}$ which is built in stages and is described as below.

Intuitively, the purpose of Stage 1 is to unfold the arena according to the specifications of the players so as to reach a cycle on every branch. Once a cycle C is reached on a branch, the formulas of the form $p_i < d$ will start to hold true or false. We then move to Stage 2 to unfold the game again till the modalities involving the formulas of the form $p_i < d$ come into effect. Stage 1 and 2 are repeated alternatively till cycles are reached on every branch. Note that it is not enough to stop at the cycles of Stage 1 since the modalities involving the formulas $p_i < d$ have not yet come into effect. Also note that, it is enough to stop when a cycle is reached involving nodes of both Stage 1 and Stage 2 since we have considered all the possible ways that any modal formula can or cannot be satisfied. This is because, the number of possible nodes of the tree is finite and also each formula is of bounded modal-depth.

Stage 1: The vertices of $\mathcal{T}_{\mathcal{A}}$ in stage 1 is the set

$$T_1 = (V \times \prod_{i \in N} \bigcup_{\varphi \supset a \in \Sigma_i} 2^{AT_{up-c}(\varphi)} \times \{plr, soc\} \times (A \cup \epsilon)) \cup \epsilon.$$

1. The root node is ϵ (level 0).

2. The nodes at level 1 are of the form $(v_0, \mathscr{C}_1, \ldots, \mathscr{C}_n, soc, \epsilon)$ such that for every $i \in N$, for every restriction $\varphi \supset a \in \Sigma_i$ and for every atom $D \in AT_{up-c}(\varphi)$, D is a component of $\mathscr{C}_i$ iff D doesn't have any formula of the form $\ominus\varphi'$ and $D \cap \mathcal{P} = val(v_0)$.

3. Every node $(v_0, \mathscr{C}_1, \ldots, \mathscr{C}_n, soc, \epsilon)$ of level 1 has $k+1$ children where the ith child, $i \in [k]$, is $(v_0, \mathscr{C}_1, \ldots, \mathscr{C}_n, plr, a_i)$ and the $k+1$th child is $(v_0, \mathscr{C}_1, \ldots, \mathscr{C}_n, plr, \epsilon)$. Intuitively, the ith child represents the situation where the society has banned the action a_i from the set of available actions of the players. The $k+1$th child represents the situation where the society hasn't applied any restriction.

4. Let $(v_0, \mathscr{C}_1, \ldots, \mathscr{C}_n, plr, a')$ be a node at level 1. It's children are determined as follows. For every player i construct a set $A_i \subseteq A$ as: if there exists $\varphi \supset a$ such that $\varphi \in D \in \mathscr{C}_i$ then let A'_i be the set of all such a's. If $a' \in A'_i$ then let $A_i = (A'_i \setminus \{a'\}) \cup \{\epsilon\}$. Otherwise let $A_i = A'_i$. Finally, if there does not exist $\varphi \supset a$ such that $\varphi \in D \in \mathscr{C}_i$ then let $A_i = A$. Now for every $\mathbf{a} \in \prod_{i \in N} A_i$, $(v_0, \mathscr{C}_1, \ldots, \mathscr{C}_n, plr, a')$ has a child $(v_1, \mathscr{C}'_1, \ldots, \mathscr{C}'_n, soc, \epsilon)$ such that

   - $v_0 \xrightarrow{\mathbf{a}} v_1$,
   - For every i for every $D \in \mathscr{C}_i$ and $D' \in \mathscr{C}'_i$ and for every $\varphi \supset a \in \Sigma_j$, $\langle \mathbf{a} \rangle^- \varphi' \in D'$ iff $\varphi' \in D$,
   - For every $D' \in \mathscr{C}'_i$, $D' \cap \mathcal{P} = val(v_1)$.

Note that as the set T_1 is finite, a node has to repeat along every branch. Steps 3 and 4 are repeated till on every branch a node repeats. This indicates that the game has settled down to a cycle on every branch. This completes the description of Stage 1. See Figure 5.3 for an illustration.

Stage 2: We now unfold the game starting from every leaf node of Stage 0 enough number of times so that any modality involving formulas of the form $p_i < d$ can take into effect. For that purpose, we mimic the cycle already reached on every branch in Stage1. As and when some new modal formula is satisfied, the subtree changes according to the actions specified by the modal formula.

The vertices of $\mathcal{T}_\mathcal{A}$ in this stage is the set

$$T_2 = (V \times \prod_{1 \in N} \bigcup_{\varphi \supset a \in \Sigma_i} 2^{AT_{up}(\varphi)} \times \{plr\} \times (A \cup \epsilon)) \cup \epsilon.$$

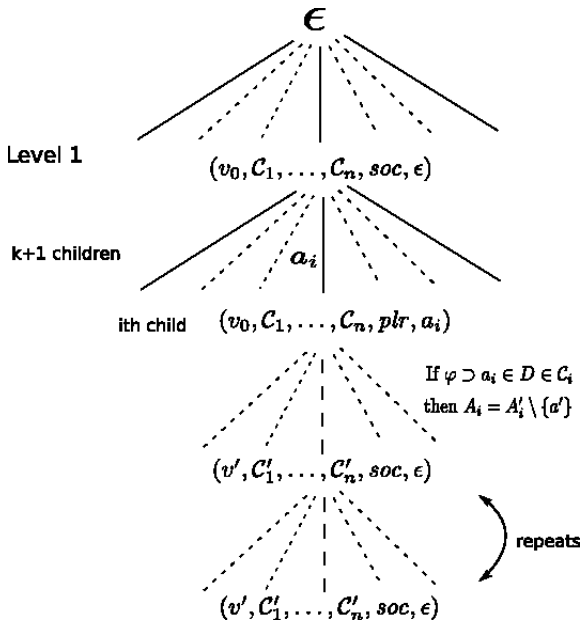

Figure 5.3: Stage 1

The leaf nodes of Stage 1 are modified to constitute the level 0 nodes of Stage 2 (because now we have to keep track of the formulas of the form $p_i < d$). Every leaf node $(v, \mathscr{C}_1, \ldots, \mathscr{C}_n, soc, \epsilon)$ represents a cycle that the game has settled down to. Suppose the cycle for the leaf $(v, \mathscr{C}_1, \ldots, \mathscr{C}_n, soc, \epsilon)$ is C. We denote by $C(v, \mathscr{C}_1, \ldots, \mathscr{C}_n, soc, \epsilon)$ the identical ancestor of $(v, \mathscr{C}_1, \ldots, \mathscr{C}_n, soc, \epsilon)$ with the greatest depth.

Let $(u, \mathscr{C}_1', \ldots, \mathscr{C}_n', plr, a)$ be the parent of $(v, \mathscr{C}_1, \ldots, \mathscr{C}_n, soc, \epsilon)$ and suppose the edge from $(u, \mathscr{C}_1', \ldots, \mathscr{C}_n', plr, a)$ to $(v, \mathscr{C}_1, \ldots, \mathscr{C}_n, soc, \epsilon)$ was labelled $\mathbf{a}$. Replace $(v, \mathscr{C}_1, \ldots, \mathscr{C}_n, soc, \epsilon)$ with another vertex $(v, \mathscr{C}_1'', \ldots, \mathscr{C}_n'', soc, \epsilon)$ such that for every $i \in N$ and for every $\varphi \supset a \in \Sigma_i$, for every $D \in AT_{up}(\varphi)$, $D \in \mathscr{C}_i''$ iff

- $D \cap \mathcal{P} = val(v)$,
- For every $p_i < d$ in the subformula closure of φ, $p_i < d \in D$ iff $p_i(C) < d$,
- For every $D' \in \mathscr{C}_i$, $\langle \mathbf{a} \rangle^- \varphi' \in D$ iff $\varphi' \in D'$.

These constitute the Level 0 nodes of stage 2. See Figure 5.4 for an illustration.

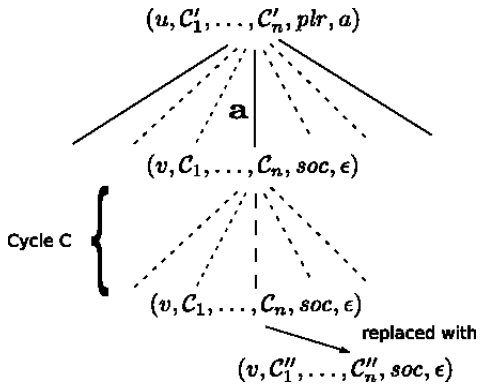

Figure 5.4: Stage 2, Level 0

Level 1: Every node $(v, \mathscr{C}''_1, \ldots, \mathscr{C}''_n, soc, \epsilon)$ of level 0 has a single child determined as follows. Let $(v, \mathscr{C}'_1, \ldots, \mathscr{C}'_n, plr, a)$ be the unique ancestor of $(v, \mathscr{C}''_1, \ldots, \mathscr{C}''_n, plr, a)$ which is a child of $C(v, \mathscr{C}''_1, \ldots, \mathscr{C}''_n, soc, \epsilon)$. Modify every set $\mathscr{C}'_i$ to $\mathscr{C}''_i$ as described above. The child of $(v, \mathscr{C}''_1, \ldots, \mathscr{C}''_n, soc, \epsilon)$ is the node $(v, \mathscr{C}''_1, \ldots, \mathscr{C}''_n, plr, a)$. This constitutes level 1.

Level 2 and above: For the nodes of Level 2 and above, we first check if any new modal formula is satisfied. If not, then we let the players play the same action as was played by them on the cycle involving that branch and wait.

The children of every node $(v, \mathscr{C}_1, \ldots, \mathscr{C}_n, plr, a')$ at level 1 (the Level 2 nodes) are determined as follows. Let $I \subseteq N$ be the set of indices such that for every player $i \in I$, there exists $\varphi \supset a$ such that $\varphi \in D \in \mathscr{C}_i$. Let A_i be the set of all such a's. For $i \in N \setminus I$, let $A_i = A$.

If $I \neq \emptyset$ then for every $\mathbf{a} \in \prod_{i \in N} A_i$, $(v, \mathscr{C}_1, \ldots, \mathscr{C}_n, plr, a')$ has a child $(v', \mathscr{C}'_1, \ldots, \mathscr{C}'_n, soc, \epsilon) \in T_1$ such that

- $v \xrightarrow{\mathbf{a}} v'$,
- For every i for every $D \in \mathscr{C}_i$ and $D' \in \mathscr{C}'_i$ and for every $\varphi \supset a \in \Sigma_i$, $\langle \mathbf{a} \rangle^- \varphi' \in D'$ iff $\varphi' \in D$,
- For every $D' \in \mathscr{C}'_j$, $D' \cap \mathcal{P} = val(v')$.

Call each such child a leaf node of stage 2.

If no new restriction is applicable, then we wait. In other words, on every such branch, we copy the actions played previously by the players in the corresponding cycle. Formally, if $I = \emptyset$, let $\mathbf{a}$ be the label of the outgoing edge from the node $C(v, \mathscr{C}_1, \ldots, \mathscr{C}_n, plr, a')$ to the ancestor of

$(v, \mathscr{C}_1, \ldots, \mathscr{C}_n, plr, a')$. Then $(v', \mathscr{C}'_1, \ldots, \mathscr{C}'_n, soc, \epsilon)$ is a child of $(v, \mathscr{C}_1, \ldots, \mathscr{C}_n, plr, a')$ iff

- $v \xrightarrow{\mathbf{a}} v'$,
- For every i for every $D \in \mathscr{C}_i$ and $D' \in \mathscr{C}'_i$ $\langle \mathbf{a} \rangle^- \varphi' \in D'$ iff $\varphi' \in D$,
- For every $D' \in \mathscr{C}'_i$, $D' \cap \mathcal{P} = val(v')$.

For every non-leaf node, the above process is repeated till $2M$ steps (Since M is the maximum modal depth of any formula in the strategy specifications. The factor of 2 is due to the alternation between a society node and a player node in the unfolding). This constitutes stage 2.

Stages 1 and 2 are repeated in alternation till along each branch, a node repeats. This completes our description of the tree $\mathcal{T}_{\mathcal{A}}$.

Let θ be the threshold value such that the society wishes that the game eventually settles down so that the social cost is less than θ. We now construct an automaton $\mathcal{M}$ from $\mathcal{T}_{\mathcal{A}}$ which is the strategy automaton for the society as follows. First, we construct a subtree $\mathcal{T}^*_{\mathcal{A}}$ of $\mathcal{T}_{\mathcal{A}}$ using the following backward induction procedure.

Procedure 3

- We label the leaf nodes with either 0 or 1 as follows. Let t be a leaf node. Let $C(t)$ be its identical ancestor and let C be the cycle from $C(t)$ to t. Label t with 1 only if $\tilde{c}(C) < 0$. Otherwise label it with 0.
- Suppose all the children of a node t has been labelled. Delete all the children with label 0. If there is no remaining child of t then label it with 0. Otherwise label it with 1.

The above procedure thus generates a subtree $\mathcal{T}^*_{\mathcal{A}}$ of $\mathcal{T}_{\mathcal{A}}$. We then construct the strategy automaton $\mathcal{M}$ from $\mathcal{T}^*_{\mathcal{A}}$. Note that every leaf node of $\mathcal{T}^*_{\mathcal{A}}$ is part of a cycle. Thus to construct $\mathcal{M}$, for every leaf node t of $\mathcal{T}^*_{\mathcal{A}}$ such that t' is the parent of t, and $t' \xrightarrow{\mathbf{a}} t$ we make a loop $t' \xrightarrow{\mathbf{a}} C(t)$ in $\mathcal{M}$. The initial state of $\mathcal{M}$ is the root of $\mathcal{T}^*_{\mathcal{A}}$ and its transition relation is given by the parent-child structure of $\mathcal{T}^*_{\mathcal{A}}$.

Note that the subtree $\mathcal{T}^*_{\mathcal{A}}$ obtained from Procedure 3 maybe empty in which case the society cannot force the play to settle down to a social cost less than θ by removing one action at a time from the vertices. But if $\mathcal{T}^*_{\mathcal{A}}$ is nonempty, then we have:

Proposition 5.6 *By playing the bounded memory strategy $\mathcal{M}$, the society can make the game eventually settle down to a social cost less than the threshold θ given that the players play according to their strategy specifications.*

Proof For every play ρ in the arena that is consistent with the strategy specifications we can associate a node $t(\rho)$ in $\mathcal{T}_{\mathcal{A}}$. Note that then it is sufficient to prove:

Claim 5.7 *For all i, for all $\varphi \supset a \in \Sigma_i$ and for all φ' in the subformula closure of φ, $\rho \models \varphi'$ iff $\varphi' \in t(\rho)$.*

That is because all the actions that a player i can play by testing the truth of the specification pre-conditions φ in Σ_i is present in $\mathcal{T}_{\mathcal{A}}$ and all such actions that lead to unfavourable cycles are removed in the construction of $\mathcal{M}$ from $\mathcal{T}_{\mathcal{A}}$.

To prove Claim 5.7 we proceed by induction on the structure of a subformula φ'. The base cases $\varphi' \equiv p \in \mathcal{P}$ and $\varphi' \equiv p_i(x) < d$ are immediate from the construction of $\mathcal{T}_{\mathcal{A}}$. $\rho \models \varphi_1 \vee \varphi_2$ iff $\rho \models \varphi_1$ or $\rho \models \varphi_2$ iff $\varphi_1 \in t(\rho)$ or $\varphi_2 \in t(\rho)$ iff $\varphi_1 \vee \varphi_2$ in $t(\rho)$, the corresponding component being an atom. $\rho \models \neg\varphi'$ iff $\rho \not\models \varphi'$ iff $\varphi' \notin t(\rho)$ iff $\neg\varphi' \in t(\rho)$, since the corresponding component is an atom.

Suppose there exists a play ρ in the arena such that $\rho \models \langle \mathbf{a} \rangle^{-} \varphi'$. We know that the modal depth of $\langle \mathbf{a} \rangle^{-} \varphi'$ is at most M and hence the modal depth of φ' is at most $M - 1$. Since we have unfolded the cycles in the even stages $2M$, the even stages are effectively of length M (since each vertex along a branch is repeated twice, one for the society and the other for the players). Now, $\rho \models \langle \mathbf{a} \rangle^{-} \varphi'$ iff $\rho = \rho' \xrightarrow{\mathbf{a}} v$ and $\rho' \models \varphi'$ [by induction hypothesis which can be applied because of the above observation] iff $\varphi' \in t(\rho')$ iff $t(\rho') \xrightarrow{\mathbf{a}} t(\rho)$ in $\mathcal{T}_{\mathcal{A}}$ (by construction) iff $t(\rho) \models \langle \mathbf{a} \rangle^{-} \varphi'$ (by construction).

Finally, suppose $\rho \models \diamondminus \varphi'$ then there exists a prefix ρ' of ρ such that $\rho \models \varphi'$. We do a second induction on $\ell = |\rho| - |\rho'|$. If $\ell = 0$ then $\rho \models \varphi'$ iff $\varphi' \in t(\rho)$ iff $\diamondminus \varphi' \in t(\rho)$. Now suppose $\ell > 0$ in that case $|\rho'| < |\rho|$. Then $\rho \models \diamondminus \varphi'$ iff $\rho'' \models \diamondminus \varphi'$ where ρ'' is a prefix of ρ and $|\rho''| = |\rho| - 1$. Then by the second induction hypothesis, $\diamondminus \varphi' \in t(\rho'')$ iff $\diamondminus \varphi' \in t(\rho)$ since $t(\rho)$ is a child of $t(\rho'')$ by construction of $\mathcal{T}_{\mathcal{A}}$. □

As an immediate corollary to the above proposition we have:

Corollary 5.8 *Let $(\mathcal{A}, v_0)$ be an initialised arena where $\mathcal{A} = (V, E)$ and A is the common set of actions of the players. Given strategy specifications*

$\{\Sigma_i\}_{i\in N}$ for the players and given a function $f : V \to \mathbb{Q}$ for the social cost, if the society can force the game to eventually settle down so that the social cost is less than θ then it can do so using a finite memory strategy, the memory being

$$\Big(V \times \prod_{1\in N} \bigcup_{\varphi \supset a \in \Sigma_i} 2^{AT_{up}(\varphi)} \times \{plr, soc\} \times (A \cup \epsilon)\Big) \cup \epsilon$$

We thus see that if the society imposes restrictions, on the order of play or the availability of actions, based on the finite state automata derived in the above proofs, then it can ensure that the social cost always remains within a certain threshold. These automata are finite memory strategies which can be seen as rules for the society for applying the restrictions. Note that although we carried out our analysis for limit-average payoffs, a similar analysis also goes through in the setting of discounted payoffs.

Chapter 6

Imitation as a strategy

In the setting of large games, where players have limited resources and computational power, they strategise dynamically as the game progresses, based on their observations of the outcomes. In such games players often play based on certain heuristics. Also, players belong to a fixed number of 'types' where the players of a certain type employ a certain kind of strategy/heuristic. In this chapter, we explore the possibility of imitation as a viable strategy. In our setup, there are two types of players: optimisers and imitators. The optimising players play finite memory strategies and the imitators play according to specifications given by automata. We present algorithmic results on the eventual survival of types. [1]

6.1 Overview

Imitation is an important heuristic studied by game theorists in the analysis of large games, in both extensive form games with considerable structure, and repeated normal form games with a large number of players. One reason for this is that notions of rationality underlying solution concepts are justified by players' assumptions about how other players play, iteratively. In such situations, players' knowledge of the types of other players alters game dynamics. Skilled players can then be imitated by less skilled ones, and the former can then strategise about how the latter might play. In games with a large number of players, both strategies and outcomes are studied using distributions of player types.

The dynamics of imitation, and strategising of optimizers in the presence of imitators can give rise to interesting consequences. For instance, in the

[1] This chapter is based on the results from [PR10].

game of chess, if the player playing white somehow knows that her opponent will copy her move for move then the following simple sequence of moves allows her to checkmate her opponent [2]:

```
1.e3 e6 2.Qf3 Qf6 3.Qg3 Qg6 4.Nf3 Nf6 5.Kd1 Kd8 6.Be2 Be7
7.Re1 Re8 8.Nc3 Nc6 9.Nb5 Nb4 10.Qxc7#
```

On the other hand, we can have the scenario where every player is imitating someone or the other and the equilibrium attained may be highly inefficient. This is usually referred to as 'herd behaviour' and has been studied for instance in [Ban92].

In an ideal world, where players have unbounded resources and computational ability, each of them can compute their optimal strategies and play accordingly and thus we can predict optimal play. But in reality, this is seldom the case. As we already argued in Chapter 1, in real life, players are limited in their resources, in computational ability and their knowledge of the game. Hence, in large games it is not possible for such players to compute their optimal strategies beforehand by considering all possible scenarios that may arise during play. Rather, they observe the outcome of the game and then strategise dynamically. They apply various heuristics which they know, maybe from their experience or by observing other players, to have performed well. In such a setting again, imitation types make sense.

A resource bounded player may attach some cost to strategy selection. For such a player, imitating another player who has been doing extensive research and computation may well be worthwhile, even if her own outcomes are less than optimal. What is lost in sub-optimal outcomes may be gained in avoiding expensive strategisation.

Thus, in a large population of players, where resources and computational abilities are asymmetrically distributed, it is natural to consider a population where the players are predominantly of two kinds: optimisers and imitators.[3] Asymmetry in resources and abilities can then lead to different types of imitation and thus ensure that we do not end up with 'herd behaviour' of the kind referred to above. Mutual reasoning and strategising process between optimizers and imitators leads to interesting questions for game dynamics in these contexts.

[2]This is called 'monkey-chess' in chess parlance.

[3]There would also be a third kind of players, randomisers, who play any random strategy, but we do not consider such players in this exposition.

6.1.1 Related work

Imitation is typically modelled in the dynamical systems framework in game theory. Schlag ([Sch98]) studies a model of repeated games where a player in every round samples one other player according to some sampling procedure and then either imitates this player or sticks to her own move. He shows that the strategy where a player imitates the sampled player with a probability that is proportional to the difference in their payoffs, is the one that attains the maximum average payoff in the model. He also gives a simple counterexample to show that the naïve strategy of 'imitate if better' may not always be improving. Banerjee ([Ban92]) studies a sequential decision model where each decision maker may look at the decisions made by the previous decision makers and imitate them. He shows that the decision rules that are chosen by optimising individuals are characterised by herd behaviour, i.e., people do what others are doing rather than using their own information. He also shows that such an equilibrium is inefficient. Levine and Pesendorfer ([LP07]) study a model where existing strategies are more likely to be imitated than new strategies are to be introduced.

The common framework in all of the above studies is repeated non-zero-sum normal form games where the questions asked of the model are somewhat different from standard ones on equilibria. Since all players are not optimizers, we do not speak of equilibrium profiles as such but optimal strategies for optimizers and possibly suboptimal outcomes for imitators. In the case of imitators, since they keep switching (imitate i for 2 moves, j for 3 moves, then again i for 1 move, etc.) studies consider stability of imitation patterns, what types of imitation survive eventually, since these would in turn determine play by optimizers and thus stable subgames, thus determining stable outcomes. Note that, as in the example of chess above, imitation and hence the study of system dynamics of this kind makes equal sense in large *turn-based* extensive form games among resource bounded players as well.

For finitely presented infinite games the stability questions above can be easily posed and answered in automata theoretic ways, since typically finite memory strategies suffice for optimal solutions, and stable imitation patterns can be analysed algorithmically. Indeed, this also provides a natural model for resource bounded players as finite state automata.

6.1.2 What we study

In this chapter, we consider games of unbounded duration on finite graphs among players with overlapping objectives where the population is divided into players who optimise and others who imitate. Unbounded play is natural in the study of imitation as a heuristic, since 'losses' incurred per move may be amortised away and need not affect eventual outcomes very much. Imitator types specify how and who to imitate and are given using finite state transducers. Since plays eventually settle down to connected components, players' preferences are given using orderings on Muller sets [PS09]. We study turn-based games so as to use the set of techniques already available for the analysis of such games.

In this setting we address the following questions:

- If the optimisers and the imitators play according to certain specifications, is a global outcome eventually attained?
- What sort of imitative behaviour (subtypes) eventually survive in the game?
- How worse-off are the imitators from an equilibrium outcome?

6.2 Preliminaries

In this chapter we study games on finite turn-based arenas, as defined in Chapter 1. $N = \{1, 2, \ldots, n\}$ is the set of players and we assume that all the players have a common set of actions A, that is, $A_1 = A_2 = \ldots = A_n = A$.

As usual, for a vertex $v \in V$, let vE denote its set of neighbours: $vE = \{v' | (v, a, v') \in E \text{ for some } a \in A\}$. For $v \in V_i$, $1 \leq i \leq n$ and $a \in A$, we let $v[a] = \{v' | (v, a, v') \in E\}$. $v[a]$ is either empty when a is not available to the player i at v, i.e., $a \notin \Gamma_i(v)$. Otherwise $v[a]$ is the singleton $\{v'\}$. In the latter case, we often say a is enabled at v and write $v[a] = v'$. For $u \in A^*$, we can similarly speak of u being enabled at v and define $v[u]$ so that when $v[u] = \{v'\}$, there is a path in the graph from v to v' such that u is the sequence of move labels of edges along that path. Given $v \in V$ and $u \in A^*$, if any u-labelled path exists in the graph, it is unique. On the other hand, given any sequence of vertices that correspond to a path in the graph, there may be more than one sequence of moves that label that path.

The notion of a play in the the initialised arena $(\mathcal{A}, v_0)$ is defined as in Chapter 1. With each player i, we associate a total pre-order $\sqsubseteq_i \subseteq (2^V \times 2^V)$. This induces a total preorder on plays as follows: $u \sqsubseteq_i u'$ iff $\inf(u) \sqsubseteq_i \inf(u')$.

Thus an n-player game is given by a tuple $(\mathcal{A}, v_0, \sqsubseteq_1, \ldots, \sqsubseteq_n)$, consisting of an n-player initialised game arena and the players' preferences.

The notion of a strategy for a player is defined as in Chapter 1. Formally, a strategy s_i for player i is a partial function

$$s_i : VA^* \rightharpoonup A$$

where $s_i(vu)$ is defined if $v[u]$ is defined and $v[u] \in V_i$, and if $(v[u])[s_i(vu)]$ is defined.

The notions of a finite memory strategy, a memoryless strategy, best response and Nash equilibrium are as given in Chapter 1.

6.3 Specification of Strategies

We now describe how the strategies of the imitator and optimiser types are specified.

6.3.1 Imitator Types

An imitator type is specified by a finite state transducer which advises the imitator whom to imitate when using memory states for switching between imitating one player or another. When deciding not to imitate any other player, we assume that the type advises what to play using a memoryless strategy.

An imitator type τ_j for player j is a tuple $(M, \pi, \mu, \delta, m_0)$ where M is the finite set denoting the memory of the strategy, $m_0 \in M$ is the initial memory, $\delta : A \times M \to M$ is the memory update function, $\pi : V \to A$ is a positional strategy such that for any $v \in V$, $\pi(v)$ is enabled at v, and $\mu : M \to [n]$ is the imitation map.

Given τ_j as above, define a strategy s_j for player j as follows. Let $v \in V$ and $u = a_1 \ldots a_k \in A^*$ is a partial play from v such that $v[u]$ is defined and $v[u] \in V_j$. Let $m_{i+1} = \delta(a_{i+1}, m_i)$ for $0 \leq i < k$. Then $s_j(vu) = a_\ell$, if a_ℓ is the last $\mu(m_k)$ move in the given play and a_ℓ is enabled at vu, and $s_j(vu) = \pi(v[u])$, otherwise.

Note that the type specification only specifies whom to imitate, and how it decides whom to imitate but is silent on the rationale for imitating a player or switching from imitating x to imitating y. In general an imitator would have a set of observables, and based on observations of game states made during course of play, would decide on whom to imitate when. Thus imitator specifications could be given by past-time formulas in a simple propositional

modal logic. With any such formula we can associate an imitation type transducer as defined above. We have already discussed this approach in Chapters 4 and 5.

The following are some examples of imitating strategies that can be expressed using such automata:

1. Imitate player 1 for 3 moves and then keep imitating player 4 forever.
2. Imitate player 2 till she receives the highest payoff. Otherwise switch to imitating player 3.
3. Non-deterministically imitate player 4 or 5 forever.

For convenience of the subsequent technical analysis, we assume that an imitator type $\tau = (M, \pi, \mu, \delta, m_0)$ in game arena $\mathcal{A} = (V, E)$ and an action set A is presented as a finite state transducer $\mathcal{R}_\tau = (M', \delta', g', m_I)$ where

- $M' = V \times M \times A^{[n]}$.
- $\delta' : A \times M' \to M'$ such that $\delta'(a, \langle v, m, (a_1, \ldots, a_n)\rangle) = \langle v', m', (a_1, \ldots, a_{i-1}, a, a_{i+1}, \ldots, a_n)\rangle$ such that $v \xrightarrow{a} v'$, $\delta(a, m) = m'$ and $v \in V_i$.
- $g' : V \times M' \to A$ such that $g'(v, \langle v, m, (a_1, \ldots, a_n)\rangle) = a_i$ iff $\mu(m) = i$ and a_i is enabled at v. Otherwise $g'(v, \langle v, m, (a_1, \ldots, a_n)\rangle) = \pi(v)$.
- $m_I = \langle v_0, m_0, (a_1, \ldots, a_n)\rangle$ for some $(a_1, \ldots, a_n) \in A^{|n|}$.

Figure 1 below depicts an imitator strategy where a player imitates player 1 for two moves and then player 2 for one move and then again player 1 for two moves and so on. She just plays the last move of the player she is currently imitating. Suppose there are a total of p actions, that is, $|A| = p$. She remembers the last move of the player she is imitating in the states m_1 to m_p, and when it is her turn to move, plays the corresponding action.

Given an FST $\mathcal{R}_\tau$ for an imitator type τ, we call a strongly connected component of $\mathcal{R}_\tau$ a subtype of $\mathcal{R}_\tau$. We will often refer to the strategy s_j induced by the imitator type $\mathcal{R}_\tau$ for player j as $\mathcal{R}_\tau$, when the context is clear.

We define the notion of an imitation equilibrium which is a tuple of strategies for the optimisers such that none of the optimisers can do better by unilaterally deviating from it given that the imitators stick to their specifications.

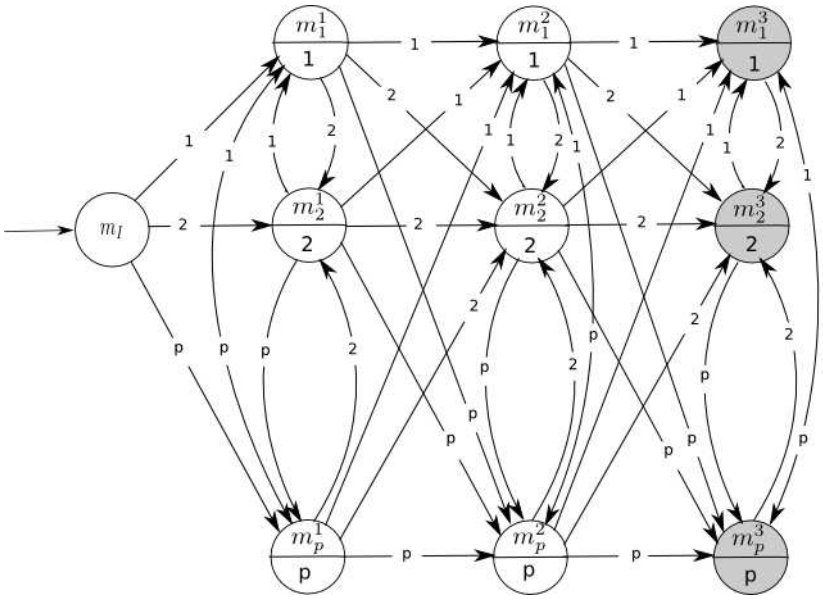

Figure 6.1: An imitator strategy

Definition 6.1 *In the game* $(\mathcal{A}, v_0, \sqsubset_1, \ldots, \sqsubset_n)$*, given that the imitators* $r+1, \ldots, n$ *play strategies* $\tau_{r+1}, \ldots, \tau_n$*, a profile of strategies* $\mathbf{s} = (s_1, \ldots, s_r)$ *of the optimisers is called an imitation equilibrium if for every optimiser* i *and for every other strategy* s_i' *of* i*,* $\inf(\rho_{(\mathbf{s}_{-i}, s_i')}) \sqsubseteq_i \inf(\rho_{\mathbf{s}})$.

Remark Note that an imitation equilibrium $\mathbf{s}$ may be quite different from a Nash equilibrium $\mathbf{s}'$ of the game $(\mathcal{A}, v_0, \sqsubset_1, \ldots, \sqsubset_n)$ when restricted to the first r components. In a Nash equilibrium the imitators are not restricted to play according to the given specifications unlike in an imitation equilibrium. In the latter case, the optimisers, in certain situations, may be able to exploit these restrictions imposed on the imitators (as in the example of 'monkey-chess' discussed in Section 6.1).

6.3.2 Optimiser specifications

One of the motivations for an imitator to imitate an optimiser is the fact that an optimiser plays to get best results. To an imitator, an optimiser appears to have the necessary resources to compute and play the best strategy and hence by imitating such a player she cannot be much worse off. But what kind of strategies do the optimisers play on their part?

In the next section, we show that if the optimisers know the types (the FSTs) of each of the imitators, then it suffices for them to play finite memory strategies. Of course, this depends on the solution concept: Nash equilibrium is defined for strategy profiles, we need to particularise them for applying only to optimizers.

Thus in the treatment below, we consider only finite memory strategies for the optimisers.

6.4 Main results

In this section, we first show that it suffices to consider finite memory strategies for the optimisers. Then we go on to address the questions raised towards the end of Section 6.1.

First we define a product operation between an arena and a finite memory strategy.

6.4.1 Product operation

Let $(\mathcal{A}, v_0)$ be an initialised arena and s be a finite memory strategy given by the FST $\mathcal{Q}_s = (M, \delta, g, m_I)$. We define $\mathcal{A} \times \mathcal{Q}_s$ to be the graph $(\mathcal{A}', v_0')$ where $\mathcal{A}' = (V', E')$ such that

- $V' = V \times M$,
- $v_0' = (v_0, m_I)$, and
- – If $g(v, m)$ is defined then $(v, m) \xrightarrow{a} (v', m')$ iff $\delta(a, m) = m'$, $v \xrightarrow{a} v'$ and $g(v, m) = a$.
  – If $g(v, m)$ is not defined then $(v, m) \xrightarrow{a} (v', m')$ iff $\delta(a, m) = m'$ and $v \xrightarrow{a} v'$.

Proposition 6.2 *Let $(\mathcal{A}, v_0)$ be an arena and s be a finite memory strategy. Then $\mathcal{A} \times \mathcal{Q}_s$ is an arena, that is, there are no dead ends.*

Proof Let $(\mathcal{A}', v_0') = \mathcal{A} \times \mathcal{Q}_s$. $\delta : A \times M \to M$ being a function, $\delta(a, m)$ is defined for every $a \in A$ and $m \in M$. Also by the definition of $\mathcal{A}$, for every vertex $v \in V$ there exists an action $a \in A$ enabled at v and a vertex $v' \in V$ such that $v \xrightarrow{a} v'$. Thus for every vertex $(v, m) \in V'$,

- if $g(v, m)$ is not defined then corresponding to every enabled action $a \in A$ there exists $(v', m') \in V'$ such that $(v, m) \xrightarrow{a} (v', m')$,

- if $g(v, m)$ is defined then by definition the unique action $a = g(v, m)$ is enabled at v. Hence, there exists $(v', m') \in V'$ such that $(v, m) \xrightarrow{a} (v', m')$.

□

Thus taking the product of the arena with a finite memory strategy s_i of player i does the following. For a vertex $v \in V_i$, it retains only the outgoing edge that is labelled with the action specified by the corresponding memory state of s_i. For all other vertices $v \notin V_i$, it retains all the outgoing edges.

Proposition 6.3 *Let $(\mathcal{A}, v_0)$ be an arena and $s_1, \ldots, s_n$ be finite memory strategies. Then $\mathcal{A} \times \mathcal{Q}_{s_1} \times \ldots \times \mathcal{Q}_{s_n}$ is an arena, that is, there are no dead ends.*

Proof Follows from Proposition 6.2 by induction on n. □

6.4.2 Equilibrium

Of the n players let the first r be optimisers and the rest $n-r$ be imitators. Let $\tau_{r+1}, \ldots, \tau_n$ be the types of the imitators $r+1, \ldots, n$. We transform the game $(\mathcal{A}, v_0, \sqsubset_1, \ldots, \sqsubset_n)$ with n players to a game $(\mathcal{A}', v_0', \sqsubset_1', \ldots, \sqsubset_{r+1}')$ with $r+1$ players in the following steps:

1. Construct the graph $(\mathcal{A}', v_0') = ((V', E'), v_0')$ as $\mathcal{A}' = \mathcal{A} \times \mathcal{R}_{\tau_{r+1}} \times \cdots \times \mathcal{R}_{\tau_n}$.

2. Let $V' = V_1' \cup \ldots \cup V_r' \cup V_{r+1}'$ such that for $i : 1 \leq i \leq r$, $(v, m_{r+1}, \ldots, m_n) \in V_i'$ iff $v \in V_i$. And $(v, m_{r+1}, \ldots, m_n) \in V_{r+1}'$ iff $v \in V_{r+1} \cup \ldots \cup V_n$. There are $r+1$ players such that the vertex set V_i' belongs to player i. Thus we introduce a dummy player, the $r+1$th player, who owns all the vertices $(v, m_{r+1}, \ldots, m_n) \in V'$ such that v was originally an imitator vertex in V. By construction, we know that every vertex $(v, m_{r+1}, \ldots, m_n) \in V_{r+1}'$ has an unique outgoing edge $(v, m_{r+1}, \ldots, m_n) \xrightarrow{a} (v', m_{r+1}', \ldots, m_n')$. Thus the dummy player $r+1$ has no choice but to play this edge always. He has a unique strategy in the arena $\mathcal{A}'$: at every vertex of V_{r+1}', play the unique outgoing edge.

3. Lift the preference orders of the players 1 to r to subsets of V' as follows. A subset W of V' corresponds to the Muller set $F(W) = \{v \mid (v, m_{r+1}, \ldots, m_n) \in W\}$ of $\mathcal{A}$. For every player $i : 1 \leq i \leq r$, for $W, W' \subseteq V'$, $W \sqsubseteq_i' W'$ if and only if $F(W) \sqsubseteq_i F(W')$.

Since the player $r+1$ has a unique strategy and plays it always, his preference ordering doesn't matter in the game. However, for consistency, we assign the preference of an arbitrary imitator (say imitator n) in the game $(\mathcal{A}, v_0, \sqsubset_1, \ldots, \sqsubset_n)$ to the $r+1$th player in the game $(\mathcal{A}', v_0', \sqsubset_1', \ldots, \sqsubset_{r+1}')$. That is, for $W, W' \subseteq V'$, $W \sqsubseteq_{r+1}' W'$ if and only if $F(W) \sqsubseteq_n F(W')$.

The game $(\mathcal{A}', v_0', \sqsubset_1', \ldots, \sqsubset_{r+1}')$ is a turn based game with $r+1$ players (the optimisers and the dummy) such that each player i has a preference ordering $\sqsubseteq_i'$ over the Muller sets of V'. Such a game was called a generalised Muller game in [PS09].

Let L be the set

$$L = \{l \in (V' \cup \{\sharp\})^{|V'|+1} \mid |l|_\sharp = 1 \wedge \forall v \in V' \, (|l|_v = 1)\}$$

where $|l|_v$ denotes the number of occurrences of v in l. We have

Theorem 6.4 ([PS09]) *The game $(\mathcal{A}', v_0', \sqsubset_1', \ldots, \sqsubset_{r+1}')$ has a Nash equilibrium in finite memory strategies, the memory being L.*

Proof Outline The proof proceeds by unfolding the arena $\mathcal{A}$ keeping track of the Latest Appearance Record (LAR) of the visited vertices at each node of the unfolding. The play settles down to a cycle on each branch of the unfolding and the LAR gives the Muller set that the cycle corresponds to. The unfolding can be stopped at that point. A backward induction procedure on the thus generated finite unfolding according to the preferences of the players gives the equilibrium strategy tuple. □

Now let $\mathbf{s}' = (s_1', \ldots, s_r', s_{r+1}')$ be a Nash equilibrium tuple for $r+1$ players in the game $(\mathcal{A}', v_0', \sqsubset_1', \ldots, \sqsubset_{r+1}')$. We now construct a finite memory imitation equilibrium tuple $\mathbf{s} = (s_1, \ldots, s_r)$ for the r optimisers in the game $(\mathcal{A}, v_0, \sqsubset_1, \ldots, \sqsubset_n)$.

For the optimiser $i : 1 \leq i \leq r$, let $s_i' = (L, \delta', g', l_I')$. Define $s_i = (M, \delta, g, l_I)$ to be a finite memory strategy in the game $(\mathcal{A}, v_0, \sqsubset_1, \ldots, \sqsubset_n)$ as

- $M = M_{r+1} \times \ldots \times M_n \times L$ where M_i, $r+1 \leq i \leq n$ is the memory of strategy τ_i of imitator i.
- $\delta : A \times M \to M$ such that $\delta(a, \langle m_{r+1}, \ldots, m_n, l\rangle) = \langle m_{r+1}', \ldots m_n', \delta'(a, l)\rangle$ where $m_i' = \delta_i(a, m_i)$, $r+1 \leq i \leq n$ such that δ_i is the memory update of strategy τ_i.

- $g : V \times M \to A$ such that $g(v, \langle m_{r+1}, \ldots, m_n, l\rangle) = g'(\langle v, m_{r+1}, \ldots, m_n\rangle, l)$.
- $l_I = \langle m_I^{r+1}, \ldots m_I^n, l_I'\rangle$ where $m_I^i,\ r+1 \leq i \leq n$ is the initial memory of strategy τ_i.

We then have:

Theorem 6.5 $\mathbf{s} = (s_1, \ldots, s_r)$ *is an imitation equilibrium in* $(\mathcal{A}, v_0, \sqsubset_1, \ldots, \sqsubset_n)$.

Proof Suppose not and suppose player i has an incentive to deviate to a strategy s' in $(\mathcal{A}, v_0, \sqsubset_1, \ldots, \sqsubset_n)$. Let $u \in A^\omega$ be the unique play consistent with the tuple $\mathbf{s}$ where the imitators stick to their strategy tuple $(\tau_{r+1}, \ldots, \tau_n)$. Let $u' \in A^\omega$ be the unique play consistent with the tuple $(\mathbf{s}_{-i}, s')$ (that is when player i has deviated to the strategy s') where again the imitators stick to their strategy tuple $(\tau_{r+1}, \ldots, \tau_n)$. Let l be the first index such that $u(l) \neq u'(l)$. Then, $v_0[u_{l-1}] \in V_i$, (where u_{l-1} is the length $l-1$ prefix of u). That is, the vertex $v_0[u_{l-1}]$ belongs to optimiser i since everyone else sticks to her strategy.

Now consider what happens in the game $(\mathcal{A}', v_0', \sqsubset_1', \ldots, \sqsubset_{r+1}')$ when all the optimisers except i play the strategies $s_1', \ldots, s_{i-1}', \ldots, s_{i+1}', \ldots, s_r'$. If the optimiser i mimics strategy s' for $l-1$ moves in the game then the play is exactly u_{l-1} and reaches a vertex $(v, m_{r+1}, \ldots, m_n) \in V_i'$ where $v = v_0[u_{l-1}]$. By construction of the product, all the actions enabled at v in the arena $\mathcal{A}$ are also enabled in the arena $\mathcal{A}'$. Hence the optimiser i can play $u(l)$. By similar arguments, optimiser i can mimic the strategy s' in the arena $\mathcal{A}'$ forever.

Thus by mimicking s' in the game $(\mathcal{A}', v_0', \sqsubset_{r+1}', \ldots, \sqsubset_n')$, the optimiser i can force a more preferred Muller set. But this contradicts the fact that $\mathbf{s}'$ is an equilibrium tuple in the game $(\mathcal{A}', v_0', \sqsubset_{r+1}', \ldots, \sqsubset_n')$. □

6.4.3 Stability

Finally, we address the questions asked in Section 6.1. Given a game $(\mathcal{A}, v_0, \sqsubset_1, \ldots, \sqsubset_n)$ with optimisers and imitators where the optimisers play finite memory strategies and the imitators play imitative strategies specified by k finite state transducers we wish to find out:

- If a certain strongly connected component W of $\mathcal{A}$ is where the play eventually settles down to.
- What subtypes eventually survive.

- How worse-off is imitator i from an equilibrium outcome.

We have the following theorem:

Theorem 6.6 *Let $(\mathcal{A}, v_0, \sqsubset_1, \ldots, \sqsubset_n)$ be a game with n players where the first r are optimisers playing finite memory strategies $s_1, \ldots, s_r$ and the rest $n-r$ are imitators playing imitative strategies $\tau_{r+1}, \ldots, \tau_n$ where every such strategy is among k different types. Let W be a strongly connected component of $\mathcal{A}$. The following questions are decidable:*

(i) Does the game eventually settle down to W?

(ii) What subtypes of the k types eventually survive?

(iii) How worse-off is imitator i from an equilibrium outcome?

Proof Construct the arena $(\mathcal{A}', v_0') = \mathcal{A} \times \mathcal{Q}_{s_1} \times \ldots \times \mathcal{Q}_{s_r} \times \mathcal{R}_{\tau_{r+1}} \times \ldots \times \mathcal{R}_{\tau_n}$.

(i) For the strongly connected component S in $(\mathcal{A}', v_0')$ that is reachable from v_0', let S be subgraph induced by the set $\{v \mid (v, m_1, \ldots, m_n) \in S'\}$. Collapse the vertices of S that have the same name and call the resulting graph S''. Check if S'' is the same as W and output YES if so.

(ii) For the strongly connected component S in $(\mathcal{A}', v_0')$ that is reachable from v_0' do the following:

- For $i: \ r+1 \leq i \leq n$ take the restriction of S to the ith component for every $(v, m_1, \ldots, m_n) \in S$. Let S_i denote this restriction.
- Collapse vertices with the same name in S_i. Let S_i' be this new graph.
- Check if S_i' is a subtype of s_i. If so output S_i'.

(iii) Compute a Nash equilibrium $\mathbf{s}$ of the game $(\mathcal{A}, v_0, \sqsubset_1, \ldots, \sqsubset_n)$ using the procedure described in [PS09]. Let S' be the reachable strongly connected component of the arena $(\mathcal{A}', v_0')$. Restrict S' to the first component and call it S. Let $F = occ(S)$. Compare F with $\inf(\rho_{\mathbf{s}})$ according to the preference ordering $\sqsubset_i$ of imitator i.

□

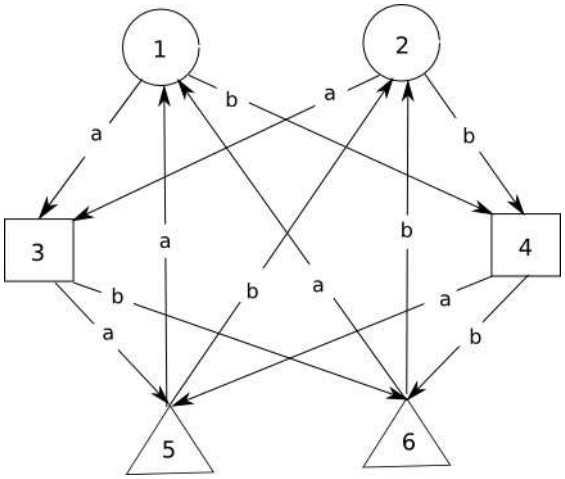

Figure 6.2: The arena $\mathcal{A}$

6.4.4 An Example

Let us look at an example illustrating the concepts of the previous section. Consider 3 firms A, B and C. Each firm has a choice of producing 2 products, product a or product b repeatedly, i.e., potentially infinitely often. In every batch each of them can decide to produce either of the products.

Now firm A is a large firm with all the technical knowhow and infrastructure and it can change between its choice of production in consecutive batches without much increase in cost. On the other hand, the firms B and C are small. For either of them, if in any successive batch it decides to change from producing a to b or vice-versa, there is a high cost incurred in setting up the necessary infrastructure. Whereas, if it sticks to the product of the previous batch, the infrastructure cost is negligible. Thus in the case where it switches between products in consecutive batches, it is forced to set the price of its product high. This actually favours firm A as it can always set its product at a reasonable price since it is indifferent between producing either of the two products in any batch.

The demand in the market for a and b keeps changing. Firm A being the bigger firm has the resources and knowhow to analyse the market and anticipate the current demand and then produce a or b accordingly. Also assume that firm A is the first to put its product out in the market. Thus it is tempting for firms B and C to imitate A. But in doing so they run the risk of setting the prices of their products too high and incurring a loss.

We model this situation in the form of the arena $\mathcal{A}$ shown in Figure 2

where the nodes of firm A, B and C are denoted as $\bigcirc$, $\square$ and $\triangle$ respectively. The preferences of each of the firms for the relevant connected components when the market demand is low are given as:

$$\{1,2,3,4,5,6\} \sqsupset_A X, \text{ for } X \subsetneq \{1,2,3,4,5,6\}$$

$$\{1,3,5\} \sqsupset_B \{1,4,5\} \sqsupset_B \{1,3,5,4\} \sqsupset_B \{2,3,6,4\} \sqsupset_B Y, \text{ for any other } Y \subsetneq \{1,2,3,4,5,6\}$$

$$\{1,3,5\} \sqsupset_C \{1,4,5\} \sqsupset_C \{2,3,6,4\} \sqsupset_C \{1,3,5,4\} \sqsupset_C Z, \text{ for any other } Z \subsetneq \{1,2,3,4,5,6\}$$

Thus firm A prefers the larger set $\{1,2,3,4,5\}$ to the smaller ones while B and C prefer the smaller sets. But when the market demand is high their preferences are given as:

$$\{1,2,3,4,5,6\} \sqsupset_i X, \text{ for } X \subsetneq \{1,2,3,4,5,6\} \text{ and } i \in \{A,B,C\}$$

That is, all of them prefer the larger set.

Now if A produces a and b in alternate batches and B and C imitate A, then we end up in the component $\{1,2,3,4,5,6\}$ which is profitable for A but less so for B and C when the market demand is not so high. But when the demand is high, the component $\{1,2,3,4,5,6\}$ is quite profitable even for B and C and thus in this case, imitation is a viable strategy for them.

Chapter 7

Neighbourhood structure in games

In this chapter, we study repeated normal form games where the players are arranged in a neighbourhood structure. The structure is given by a graph G whose nodes are players and edges denote visibility. The neighbourhoods are maximal cliques in G. The game proceeds in rounds where in each round the players of every clique X of G play a strategic form game among each other. A player at a node v strategies based on what she can observe, i.e., the strategies and the outcomes in the previous round of the players at vertices adjacent to v. Based on this, the player may switch strategies in the same neighbourhood, or migrate to a different neighbourhood. We introduce a simple modal logic, similar to the one in Chapter 4 to specify the player types.

We show that given the initial neighbourhood graph and the types of the players in the logic, we can effectively decide if the game eventually stabilises. We prove a characterisation result for these games for arbitrary types using potentials. We then offer some applications to the special case of weighted co-ordination games where we can compute bounds on how long it takes to stabilise. [1]

7.1 Overview

In Indian towns, it is still possible to see vegetable sellers who carry vegetables in baskets or pushcarts and set up shop in some neighbourhood. The

[1] The results in this chapter appear in the paper [PR11].

location of their 'shop' changes dynamically, based on the seller's perception of demand for vegetables in different neighbourhoods in the town, but also on who else is setting up shop near her, and on her perception of how well these (or other) sellers are doing. Indeed, when she buys a lot of vegetables in the wholesale market, the choice of her 'product mix' as well as her choice of location are determined by a complex rationale. While the prices she quotes do vary depending on general market situation, the neighbourhoods where she sells also influence the prices significantly: she knows that in the poorer neighbourhoods, her buyers cannot afford to pay much. She can be thought of as a small player in a large game, one who is affected to some extent by play in the entire game, but whose strategising is local where such locality is itself dynamic.

In the same town, there are other, relatively better off vegetable sellers who have fixed shops. Their prices and product range are determined largely by wholesale market situation, and relatively unaffected by the presence of the itinerant vegetable sellers. If at all, they see themselves in competition only against other fixed-shop sellers. They can be seen as big players in a large game.

What is interesting in this scenario is the movement of a large number of itinerant vegetable sellers across the town, and the resultant increase and decrease in availability of specific vegetables as well as their prices. We can see the vegetable market as composed of dynamic neighbourhoods that expand and contract, and the dynamics of such a structure dictates, and is in turn dictated by the strategies of itinerant players. [2]

Such division into neighbourhoods need not be spatial or physical, but only logical. Consider, for instance, online stores such as Amazon, eBay, Yahoo Shopping, Rediff Shopping etc. Sellers put their items up for sale on one or more of these stores based on the demand of these items and the outcomes so far. A seller who puts her item up on eBay today may very well switch to Amazon tomorrow if the demand there is higher. The buyers, on their part, would generally want the best price on offer. Hence a buyer who bought an item from Amazon today might buy another of the same kind tomorrow from eBay.

In large games, a flat structure of all players as "equals" hides important detail: neither does a player consider the detailed play of every other, nor does a player consider all other players to be of one 'average' type. We suggest that it is useful to group players into logical neighbourhoods in such

[2] In fact, the movement of these sellers may further depend on the cost of transport between these places because of small profit margins.

games: within a neighbourhood players strategise interpersonally; across neighbourhoods their visibility, and hence strategising, is limited. In the latter situation, heuristic play becomes significant. Moreover, game dynamics alters neighbourhood structure, and conversely.

Though we speak of dynamic neighbourhood structure, we note that static neighbourhood structure makes sense as well. For instance, consider the game of chess. A player can be a grandmaster, a national master, a professional or an amateur. It is generally the case that the grandmasters play among themselves, the national masters play each other and so on. Moreover the lesser non-professional players are also constrained by time, location, resources etc. Thus, for instance, a medium rated player in New Delhi would usually take part in tournaments in and around New Delhi. But how do these players strategise? The same medium rated player may not be able to take part in a tournament in Moscow (say), but that doesn't prevent her from following what is going on in that particular tournament. If a player in the tournament in Moscow is faring well by playing the Hungarian defence, our player in New Delhi may well employ the same strategy in her tournament in the hope of doing better.

It can be meaningfully argued that the games in New Delhi, Dortmund and Wijk an Zee are all subgames in one large game, in the sense that strategising and play in one is influenced by play in the other and become part of communal memory. Once again, a neighbourhood substructure abstracts such influence in the large game.

Similar structuring is seen in many other games. In football, for instance, every team all over the world participates only in three or four different leagues each year: the English Premier League, La Liga, Serie A, Bundesliga etc. But every team closely follows the unfolding of play in the other leagues and strategises based not only on the outcomes of its own league but also on those of the others. Here again, the neighbourhoods may change dynamically as the game progresses. These changes are brought about by teams/players switching allegiances. A player playing in league 1 today may think that his strategy and style of play is more suited for a different league and that he can do much better there. Hence he might join the latter league tomorrow.

In this chapter, we study large games in which players are arranged in certain neighbourhoods. The neighbourhood structure is given by a finite undirected graph where the vertices of the graph represent players and edges represent their visibility. The cliques in the graph represent the different neighbourhoods of players. We prove a characterization theorem on such games. Then we study weighted co-ordination anonymous games. We consider both the variations: neighbourhood structures that are static as

well as the dynamic ones (where the neighbourhood structure changes after every round).

Our model is that of an infinite repeated game. In every round every player plays a strategic form game with the players of her clique. The players are among a fixed set of types, which determine their strategies. In every strategic form game in a neighbourhood, the payoffs of the players are determined by the action profile of the players of that neighbourhood in that particular round.

We are interested in the dynamics of such games and their eventual stability. What action profiles, strategies, configurations etc. eventually arise? We call a game eventually stable if eventually a set of configurations repeat cyclically forever (e.g. a set of localities in the town for the vegetable seller in an Indian town). This set might be a singleton in which case the actions of the players don't change anymore; the configuration is static. We are also interested in how long it takes for such a game to eventually stabilise. We show the following:

- We define a simple modal logic, like the logic of Chapter 4, in which a player's rationale for type switching can be specified. When the types of the players are specified in this logic, we show that it can be effectively decided whether the game eventually stabilises.
- When the types of the players are unknown, we show that one can associate a potential with every configuration such that the potential becomes constant if and only if the game eventually stabilises.
- We study an application to weighted co-ordination games and explore the consequences when the players play simple imitative strategies. We show that in such cases the game always stabilises and one can compute an upper bound on the number of rounds needed to attain stability.

A valid objection at this point, at least in the case of static neighbourhoods, is the following. If the players of every neighbourhood play normal form games in every round among themselves, why is it not the case that the expected outcome is the Nash equilibrium of every normal form game in every neighbourhood? There are two explanations for this. First, since the model is that of repeated normal form, there might be action tuples other than the Nash equilibrium tuples that are in equilibrium (as for example in tit-for-tat in repeated prisoners' dilemma). But the more potent argument, the one already discussed extensively in the introductory chapter, is

that when the game is large, players hardly have the expertise, knowhow or even resources to compute and play the Nash equilibrium tuple. They play based on heuristics and employ simple strategies such as imitation, tit-for-tat, follow-the-leader etc. Hence the outcome may be much more varied than the Nash equilibrium tuples. Thus, we feel that a more natural question to ask in the setting of large games is on the dynamics of the game given the types of the players and their eventual stability and also on what configurations eventually arise. We have discussed this at length in the introductory chapter.

Related work

We study games where the players are represented by the vertices of a graph and the edges of the graph give the other players they can interact with. Such games have been studied, for instance, in [KLS01b, KLS01a, EGG06, EGG07]. They analyse games where the payoff of players depend only on her own action and the actions of her adjacent players as given by the graph structure. They study the existence and computation of Nash equilibria in such games. Young ([PY93, PY00]) studies how innovations spread through society by observation and interaction. He too models the interaction structure of the players by a finite undirected graph. Imitation dynamics in congestion games have been studied, for instance, in [ARV06, AFBH08], where they study asymptotic time complexities of the convergence or non-convergence to Nash equilibrium in congestion games when the players play imitative strategies.

In the weighted co-ordination games we study here, in every round and in every neighbourhood, the payoffs of the strategic form games do not depend on the actual action profile of the players in the neighbourhood but only on the distribution of the actions. As the actions come from a common set, such a distribution, in every round, is well-defined and non-trivial. Games where the payoffs depend on the action profile of the players are called anonymous games and have been extensively studied in the literature. See, for instance, [Blo99, Blo00, DP07, BFH09] and the references therein.

7.2 The model

As usual, $N = \{1, 2, \ldots, n\}$ be the set of players. The players are arranged in a neighbourhood structure given by a simple undirected graph $G = (V[G], E[G])$ without self loops called the neighbourhood graph. Every vertex of G stands for a player and we use the letters i, j, k etc. to denote

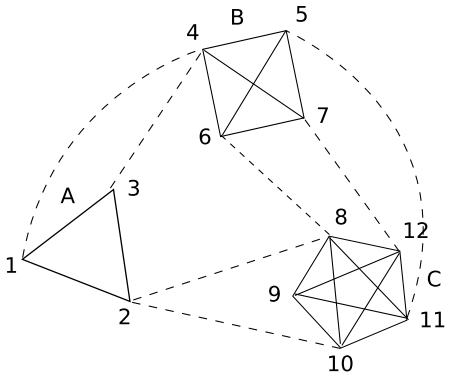

Figure 7.1: A neighbourhood graph. A, B, C are the neighbourhoods (maximal cliques in the graph) and $1, 2, \ldots, 12$ are the players (vertices)

both the vertices of the graph and players from N. The neighbourhood graph G is topologically described as follows.

Let $clq[G]$ be the set of maximal cliques of G. For simplicity, we assume that the maximal cliques are non-intersecting. The entire analysis goes through even if we drop this assumption. These cliques are the neighbourhoods of the players. Moreover, a vertex i in any clique X may have edges to vertices in some other clique X'. For a player, i these edges give the visibility structure of i. Thus the player i can view the moves and outcomes of all the players that are in her clique and also that of some players from other neighbourhoods.

Remark Note that the neighbourhood graph G is different from the game arenas dealt with in the previous chapters. G simply gives the way in which the players are spatially or logically arranged. As described below, the players play simple strategic form games and hence the arena is just the strategic form payoff matrix.

We assume that the players have a common set of actions, that is, $A_1 = \ldots = A_n$ We denote this set by A and let $A = \{a_1, a_2, \ldots, a_{|A|}\}$. Given a neighbourhood graph G, we denote by $X[G](i)$ the maximal clique (neighbourhood) that player i belongs to. As usual, we let $iE[G] = \{j \mid (i,j) \in E[G]\}$ be the set of vertices adjacent to i, that is $iE[G]$ is the set of players visible to player i. Note that $iE[G] \cap X[G](i) = X[G](i) \setminus \{i\}$. Let $nbd[G](i)$ be the set of neighbourhoods visible to player i. These are the neighbourhoods, at least one player of which i can view. Thus $nbd[G](i) = \{X[G](i) \mid j \in iE[G]\}$. See Figure 7.1 for an example.

The type of a player, like in Chapter 6, specifies how she strategises.

These are functions that will be defined below, but we assume a set Γ of player types, and a type-map $typ : N \to \Gamma$. As a rule, $|\Gamma| << |N|$, reflecting the intuition that in a large game, although the number of players may be large, there are only a few player types. We use γ, γ' etc. to range over Γ, and specify typ by an n-tuple $\langle \gamma_1, \ldots, \gamma_n \rangle$.

To talk about the outcomes of the game, we use a propositional language as before. Fix $\mathcal{P}$, a countable set of atomic propositions. $\mathcal{P}$ consists of propositions which stand for statements of the form:

- action a is played,
- payoff is greater than a threshold c,
- payoff is greater than all neighbours,

and so on. Every game involves only a finite set $P \subseteq \mathcal{P}$ of these propositions. The game proceeds in rounds. In every round k, the players of every neighbourhood play a normal form game among themselves. The outcome of the entire game in that round k, is thus the outcomes of these normal form games.

Since the games are large, it is natural that the outcome in any round does not depend on the identity of the players or the profile of actions played by them. Rather in any round k, given the neighbourhood graph G_k for that round, the payoffs of the players depend only on the distribution of the actions in the various neighbourhoods given by G_k. Hence, we consider anonymous games.

As defined in Chapter 5, an action distribution for a neighbourhood X of size k is an $|A|$ tuple of integers $\mathbf{y} = (y_1, \ldots, y_{|A|})$ such that $y_j \geq 0$ and $\Sigma_{j=1}^{k} y_j = k, \quad 1 \leq j \leq |A|$. That is, the jth component of $\mathbf{y}$ gives the number of players in the neighbourhood X who play action a_j. Let $\mathbf{Y}[k]$ denote the set of all action distributions of a neighbourhood of size k and let $\mathbf{Y} = \bigcup_{k=1}^{n} \mathbf{Y}[k]$.

We have an outcome function $out : \mathbf{Y} \to 2^P$ which gives the truth of the outcome propositions P at any neighbourhood X of size k according to the action distribution of the players of that neighbourhood.

Now given a neighbourhood graph G, we can lift out to a valuation function at the vertices of G: $val_{out}[G] : N \to 2^P$ $val_{out}[G](i)$ gives the truth of the propositions which talk about the outcomes of $\{i\} \cup nbd[G](i)$.

Thus formally, a game $\mathcal{G}$ is a tuple $\mathcal{G} = (typ, P, out)$, where typ is a type map, P a subset of $\mathcal{P}$ and out an outcome function. A configuration of the game is a pair $c = (G, \mathbf{a})$ where G is a neighbourhood graph and $\mathbf{a} \in A^n$ is

an action profile. Let C be the set of all configurations. Note that the size of C, that is, the total number of configurations, is $\binom{n}{2} \times |A|^n$.

When an initial configuration c_0 is specified, we call the pair $(\mathcal{G}, c_0)$ an initialised game. A play (history) in an initialised game $(\mathcal{G}, c_0)$, where $c_0 = (G_0, \mathbf{a}_0)$, is a sequence $\rho = (G_0, \mathbf{a}_0), \ldots, (G_k, \mathbf{a}_k),\ k > 0$, of configurations, where for all $i \geq 0$, $V[G_i] = V[G_0]$. Let H denote the set of all histories. We call a game **static neighbourhood** if, in every history in H, for all $i \geq 0$, $G_i = G_0$; otherwise it is a **dynamic neighbourhood** game.

Given a neighbourhood graph G, and a player $i \in N$, a choice for player i is a pair (X, a) where $X \in nbd[G](i)$ and $a \in A$. Let $\chi[G](i)$ denote the set of choices of i in G. A type γ is then a map $\gamma : H \to 2^{(2^N \times A)}$ such that for all $\rho \in H$, $\gamma(\rho) \subseteq \chi[G](i)$ where $G = G_{|\rho|}$.

In a static neighbourhood game, players cannot switch neighbourhoods between rounds (but can switch strategies). Thus in a static neighbourhood game, given a neighbourhood graph G, if $(X, a) \in \chi[G](i)$, then $X = X[G](i)$. However, in a dynamic neighbourhood game, a player i can decide to move to a different neighbourhood from round k to round $k+1$ provided the new neighbourhood is in $nbd[G](i)$. Thus in the dynamic neighbourhood game, the underlying neighbourhood graph keeps changing.

We say that a history $\rho = (G_0, \mathbf{a}_0), \ldots, (G_m, \mathbf{a}_m)$ is coherent with respect to the player types $\langle \gamma_1, \ldots, \gamma_n \rangle$ if the following conditions hold. Let ρ_k be the length k prefix of ρ. Then for every $k :\ 0 \leq k < m$:

- For every $i \in N$, if $\mathbf{a}_{k+1}(i) = a$ then $(X, a) \in \gamma_i(\rho_k)$.
- Given that $G_k = (V, E_1)$ and $G_{k+1} = (V, E_2)$ there exists a choice tuple $\langle (X_1, a_1), \ldots, (X_n, a_n) \rangle$ such that for all i, $(X_i, a_i) \in \gamma_i(\rho_k)$, and for all $l, m \in N$:
  - $(l, m) \in E_2 \setminus E_1$ implies $m \in X_i$, and
  - $(l, m) \in E_1 \setminus E_2$ implies $m \notin X_i$.

In other words, every player i that joins a neighbourhood X in round $k+1$, has an edge in the neighbourhood graph G_{k+1} to all the other players who also decide to join (or stay put) in the same neighbourhood X in round $k+1$. In addition, her visibility structure changes, in that, she may be able to view the outcomes and actions of new players in some neighbourhood other than X after joining X whereas, some of the players that she could view in round k, may not be visible to her anymore. Note that the process is non-deterministic. See Figure 7.2 for an example.

Some typical types of players are

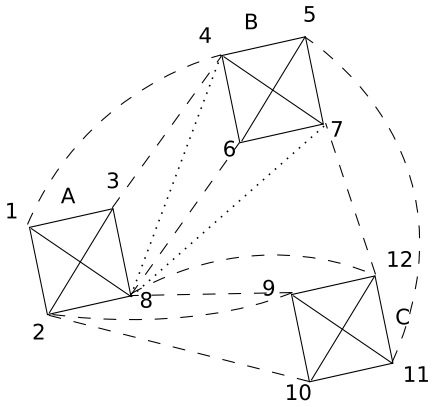

Figure 7.2: The neighbourhood graph of figure 7.1 after player 8 has joined the neighbourhood of players 1,2 and 3. The dashed edges are the visibility of player 8 retained from her old neighbourhood and the dotted ones are the players newly visible to her.

- Play the action played by the maximum number of visible players in the previous round.
- Play the action played by the player who, among the visible ones, received the maximum payoff in the last round.
- Play the action played by the player who, among the visible ones in the last round, received the maximum average payoff in the previous k rounds.
- Switch from the current neighbourhood to a neighbourhood where a player of the same type received a higher payoff in the previous round.

and so on.

Let $c, c' \in C$ be configurations. c' is said to follow c, denoted $c \to c'$ if c' is derived from c as above (that is, when all players play according to their types specified by the game). Let $\to^*$ be the transitive closure of the follows relation. The graph $\mathcal{C} = (C, \to)$ will be referred to as the configuration graph of the game. Let $c \in C$; we then speak of $\mathcal{T}_{\mathcal{C}}(c)$, the tree unfolding of the configuration graph from c. $\mathcal{T}_{\mathcal{C}}(c) = (T, E)$ is an infinite tree where the nodes are labelled with configurations from C. For a node $t \in T$, we let $c(t)$ denote this configuration.

7.3 Types

We have spoken of players switching strategies or migrating to other neighbourhoods. In general, this is to improve payoffs over the course of play. We introduce a logical language to talk about the types of the players. The syntax should be able to specify the properties of the games that the players observe and the actions they play based on these observations.

Let X be a countable set of variables. Let the terms of the logic be defined as:

$$\tau ::= i \mid x,\ i \in N,\ x \in X$$

That is, a term is either a player (vertex) or a variable (which takes players as its values). Then the types of the players are built using the following syntax:

$$\begin{aligned}\Phi ::= &\tau_1 = \tau_2 \mid \tau_1 \leftrightarrow \tau_2 \mid [\tau_1, \tau_2] \mid p@\tau, p \in \mathcal{P} \mid \neg\varphi \mid \\ &\varphi_1 \vee \varphi_2 \mid \ominus\varphi \mid \bigcirc\varphi \mid \Diamond\!\!\!\!-\varphi \mid \Diamond\varphi \mid \exists x \cdot \varphi(x)\end{aligned}$$

where τ_1 and τ_2 are terms.

Intuitively, $\tau_1 \leftrightarrow \tau_2$ is intended to mean that players $\pi(\tau_1)$ and $\pi(\tau_2)$ are visible to each other in the neighbourhood structure. Or in other words, there is an edge between $\pi(\tau_1)$ and $\pi(\tau_2)$ in the neighbourhood graph G. $[\tau_1, \tau_2]$ holds when the players $\pi(\tau_1)$ and $\pi(\tau_2)$ are in the same neighbourhood.

Formally, let $(\mathcal{G}, c_0)$ be an initialised game. The formulas in Φ are evaluated at the nodes of $\mathcal{T}_{\mathcal{C}}(c_0)$. The truth of a formula $\varphi \in \Phi$ at a node $t \in \mathcal{T}_{\mathcal{C}}(c_0)$ is denoted by $t \models \varphi$ and is defined inductively as follows. Let $c(t) = (G, \mathbf{a})$ be the configuration associated with t. The truth of the atomic formulas $\tau_1 = \tau_2, \tau_1 \leftrightarrow \tau_2$ and $[\tau_1, \tau_2]$ are derived from $c(t)$:

- $t \models \tau_1 = \tau_2$ iff $\pi(\tau_1) = \pi(\tau_2)$.
- $t \models \tau_1 \leftrightarrow \tau_2$ iff $(\pi(\tau_1), \pi(\tau_2)) \in E[G]$.
- $t \models [\tau_1, \tau_2]$ iff $\exists X \in clq[G]$ such that $\pi(\tau_1) \in X$ and $\pi(\tau_2) \in X$.

For the rest of the formulas, we define truth by:

- $t \models p@\tau$ iff $p \in val_{out}[G](\pi(\tau))$.
- $t \models \neg\varphi$ iff $t \not\models \varphi$.
- $t \models \varphi_1 \vee \varphi_2$ iff $t \models \varphi_1$ or $t \models \varphi_2$.

- $t \models \ominus\varphi$ iff t is not the root of $\mathcal{T}_{\mathcal{C}}(c_0)$ and $t' \models \varphi$ where t' is the parent of t in $\mathcal{T}_{\mathcal{C}}(c_0)$.
- $t \models \bigcirc\varphi$ iff there exists a child t' of t in $\mathcal{T}_{\mathcal{C}}(c_0)$ such that $t' \models \varphi$.
- $t \models \diamondsuit\varphi$ iff there exists an ancestor t' of t in $\mathcal{T}_{\mathcal{C}}(c_0)$ such that $t' \models \varphi$.
- $t \models \Diamond\varphi$ iff there exists a successor t' of t in $\mathcal{T}_{\mathcal{C}}(c_0)$ such that $t' \models \varphi$.
- $t \models \exists x \cdot \varphi(x)$ iff there exists $j \in N$ such that $t \models \varphi[j/x]$.

Above, $\varphi[j/x]$ denotes the result of replacing every free occurrence of x by j. The notions of satisfiability, validity etc are standard. Note that the following formula is valid:

$$\exists x \bigcirc \varphi(x) \equiv \bigcirc \exists \varphi(x)$$

The following are examples of some typical types that can be specified in the logic:

- Play action a and b alternatively:

  $$(\ominus p_a @ i \supset p_b @ i) \wedge (\ominus p_b @ i \supset p_a @ i)$$

  where p_a and p_b stand for "play action a" and "play action b" respectively.

- Play the action played by the visible player who received the maximum payoff in the previous round:

  $$\exists x (i \leftrightarrow x \wedge r @ x \wedge p_a @ x) \supset \bigcirc (p_a @ i)$$

  where r and p_a stand for "payoff is greater than that of all neighbours of i" and "plays action a (for some $a \in A$)" respectively.

- If there exists a player j within the visibility of i who is in a different neighbourhood X' but plays the same action and gets a better payoff in round k, player i joins the neighbourhood X' of such a player with the maximum such payoff.

  $$\forall x ((i \leftrightarrow x \wedge q @ x \wedge r @ y \wedge \neg [i, x]) \supset \bigcirc [i, x])$$

  where q and r are propositions which say, "payoff is greater than that of i" and "payoff is greater than that of all neighbours of i".

and so on.

Call two formulas $\varphi_1, \varphi_2 \in \Phi$ equivalent, denoted $\varphi_1 \equiv \varphi_2$ if $t \models \varphi_1$ if and only if $t \models \varphi_2$ for all $t \in \mathcal{T}_{\mathcal{C}}(c_0)$. Since π is a fixed map and every neighbourhood graph is finite, we can show the following:

Proposition 7.1 *Every formula $\varphi \in \Phi$ is equivalent to a quantifier free formula $\varphi' \in \Phi$.*

Proof Since

$$\exists x \varphi(x) \equiv \bigvee_{i \in V} \varphi[i/x]$$

the proposition follows by an easy induction on the structure of φ. □

Remark Note that the logic is a standard modal logic on trees, extended to speak of players and neighbourhoods. We do not initate a logical study of neighbourhood switching, but use standard logical machinery to specify a wide variety of rules that constitute the rationale of players for switching strategies or neighbourhoods. The expressiveness of the logic has a critical bearing on game dynamics and hence needs a more careful study. Note that the modalities are branching (as they are interpreted on tree nodes); path connectives like *until* would be meaningful but require a different technical development.

7.4 Stationariness

In this section we study the dynamics of the games with neighbourhood structures. We are interested in finding out what kind of neighbourhood structures eventually arise and whether the players settle down to playing in such a way that the neighbourhood structure and the actions do not change any further. We look at games where the types of the players are given as formulas. We show that in this case, it is decidable whether the game becomes eventually stationary for the notion of stationariness that we shall define presently.

Definition 7.2 *A configuration c is said to be stationary if $c' = c$ for all $c \to^* c'$.*

Definition 7.3 *A game is said to be eventually stationary if it always reaches a stable configuration.*

7.4.1 Types specified as formulas

Let Γ_Φ be a subset of types where every type $\gamma \in \Gamma_\Phi$ is specified as a formula in Φ. Let $\langle \gamma_1, \ldots, \gamma_n \rangle$ be the types of the players where $\gamma_i \in \Gamma_\Phi$ for all $i \in N$. Given neighbourhood graph G and such a type specification, what does it mean for players to play according to their types? Note that the configuration transition relation, $c \to c'$, is derived from player types. Hence, we say that the tree unfolding $\mathcal{T}_\mathcal{C}(c_0)$, where c_0 is the initial configuration, conforms to the specification $\langle \gamma_1, \ldots, \gamma_n \rangle$ when $t_0 \models \gamma_1 \wedge \ldots \wedge \gamma_n$ where t_0 is the root of $\mathcal{T}_\mathcal{C}(c_0)$.

The implication of such a definition of conformance is as follows: suppose that the types of two or more players are inconsistent. For example, it can be that the types γ_i and γ_j of players i and j are $\bigcirc[i,j]$ and $\neg \bigcirc [i,j]$ respectively. But in that case the formula $\gamma_1 \wedge \ldots \wedge \gamma_n$ is unsatisfiable and hence there is no successor configuration at the node where this formula must hold. This is equivalent to the convention that the game terminates immediately in such situations.

We first study the stationariness of games when the types are specified as formulas from the syntax Φ. That is, for every $i \in N$, $\gamma_i \in \Gamma_\Phi$. In this case we have:

Theorem 7.4 *Let $(\mathcal{G}, c_0)$ be an initialised game where $c_0 = (G_0, \mathbf{a}_0)$. Let $\gamma_1, \ldots, \gamma_n$ be the types of the players specified as formulas in Φ. Then it can be effectively decided whether the game becomes eventually stationary.*

Proof We assume, using Proposition 7.1 that for all $i \in N$, γ_i is quantifier free. Let for $i \in N$, $CL(\gamma_i)$ be the subformula closure of γ_i and $AT(\gamma_i)$ be the set of atoms (Section 1.2.9) of the type γ_i. We construct a graph $\mathcal{A} = (V(\mathcal{A}), E(\mathcal{A}))$ (similar to an atom graph) as follows:

- $V(\mathcal{A}) \subseteq (C \times \prod_{i \in N} AT(\gamma_i))$ such that

  $$(c, \langle D_1, \ldots, D_n \rangle) \in V(\mathcal{A})$$

  iff for all $i \in N$, $D_i \cap \mathcal{P} = val_{out}[G](i)$ where $c = (G, \mathbf{a})$.

- A node $w = (c, \langle D_1, \ldots, D_n \rangle) \in V[\mathcal{A}]$ is called *initial* if $c = c_0$ and for all $i \in N$, D_i does not have any formula of the form $\ominus\alpha$. Let $init(\mathcal{A})$ be the set of initial nodes.

- A node $w = (c, \langle D_1, \ldots, D_n \rangle) \in V[\mathcal{A}]$ is called *final* if c is stationary and for all $\Diamond\beta \in D_i, \beta \in D_i$, and for all $\bigcirc\beta \in D_i,\ \beta \in D_i$.

- For $w, w' \in V(\mathcal{A})$ such that

$$w = (c, \langle D_1, \ldots, D_n \rangle)$$
$$w' = (c', \langle D'_1, \ldots, D'_n \rangle)$$

  $(w, w') \in E(\mathcal{A})$ iff

  - For all $i \in N$,
    - for all $\ominus\alpha \in CL(\gamma_i)$, if $\alpha \in D_i$ then $\ominus\alpha \in D'_i$,
    - for all $\bigcirc\alpha \in CL(\gamma_i)$, if $\bigcirc\alpha \in D_i$ then $\alpha \in D'_i$,
    - for all $\Diamond\alpha \in CL(\gamma_i)$, if $\Diamond\alpha \in D_i$ then $\alpha \in D'_i$ or $\Diamond\alpha \in D'_i$ and
    - for all $\diamondminus\alpha \in CL(\gamma_i)$, if $\diamondminus\alpha \in D'_i$ then $\alpha \in D_i$ or $\Diamond\alpha \in D_i$.

We call a subgraph $\mathcal{A}'$ of $\mathcal{A}$ *good* if for every node $w = (c, \langle D_1, \ldots, D_n \rangle) \in \mathcal{A}'$:

1. there exists $w' = (c', \langle D'_1, \ldots, D'_n \rangle) \in \mathcal{A}'$ reachable in $\mathcal{A}'$ from w such that w' is final, and
2. there exists $w' = (c', \langle D'_1, \ldots, D'_n \rangle) \in \mathcal{A}'$ such that w is reachable in $\mathcal{A}'$ from w' and w' is initial.

Let $reach(\mathcal{A})$ be the subgraph of $\mathcal{A}$ generated by all the configurations reachable from $init(\mathcal{A})$ in $\mathcal{A}$.

Let $\mathcal{A}'$ be a good subgraph of $\mathcal{A}$ and w_0 be an initial node. Let $\mathcal{T}_{\mathcal{A}}(w_0)$ be the tree unfolding of $\mathcal{A}$ from w_0. $\mathcal{T}_{\mathcal{A}}(w_0)$ is an infinite tree with nodes labelled with elements from $V[\mathcal{A}]$. Let t be a node of $\mathcal{T}_{\mathcal{A}}(w_0)$ such that t is labelled with $(c, \langle D_1, \ldots, D_n \rangle)$. We can show that:

Claim 7.5 *For every $i \in N$, for every $\alpha \in CL(\gamma_i)$, $t \models \alpha$ iff $\alpha \in D_i$.*

Proof The proof proceeds by induction on the structure of α:

- $\alpha \equiv p@\tau$: Follows immediately since we have ensured in the construction of $\mathcal{T}_{\mathcal{A}}(w_0)$ that $(c, \langle D_1, \ldots, D_n \rangle) \in V(\mathcal{A})$ iff for all $i \in N$, $D_i \cap \mathcal{P} = val_{out}[G](i)$ where $c = (G, \mathbf{a})$.
- $\alpha \equiv \neg\varphi$: $t \models \neg\varphi$ iff $t \not\models \varphi$ iff $\varphi \notin D_i$ iff $\neg\varphi \in D_i$ (since D_i is an atom).
- $\alpha \equiv \varphi_1 \vee \varphi_2$: $t \models \varphi_1 \vee \varphi_2$ iff $t \models \varphi_1$ or $t \models \varphi_2$ iff $\varphi_1 \in D_i$ or $\varphi_2 \in D_i$ iff $\varphi_1 \vee \varphi_2 \in D_i$ (since D_i is an atom).
- $\alpha \equiv \ominus\varphi$: $t \models \ominus\varphi$ iff $t' \models \varphi$ where $t' = (c', \langle D'_1, \ldots, D'_n \rangle)$ is the parent of t iff $\varphi \in D'_i$ iff $\ominus\varphi \in D_i$ (by the construction of $\mathcal{A}$).

- $\alpha \equiv \bigcirc\varphi$: similar to above.
- $\alpha \equiv \diamondsuit\varphi$: $t \models \diamondsuit\varphi$ iff there exists an ancestor $t' = (c', \langle D'_1, \ldots, D'_n\rangle)$ of t such that $t' \models \varphi$ iff $\varphi \in D'_i$. We do a second induction on the distance between t' and t. The base case is when the distance is 0 and then $\varphi \in D_i$ and hence $\diamondsuit\varphi \in D_i$ (since D_i is an atom). If the distance is $k+1$ then $\diamondsuit\varphi \in D''_i$ such that $t'' = (c'', \langle D''_1, \ldots, D''_n\rangle)$ is the parent of t and hence $\diamondsuit\varphi \in D_i$
- $\alpha \equiv \Diamond\varphi$: similar to above.

□

Thus by the above claim, for a formula $\alpha \in CL(t_i)$ for some $i \in N$, to check if $t \models \alpha$ it is enough to check if $\alpha \in D_i$.

Claim 7.6 *The game is eventually stationary if and only if reach*$(\mathcal{A})$ *has a good subgraph.*

Let us assume the claim. We see that the construction of the configuration graph $\mathcal{A}$, the reachable subgraph *reach*$(\mathcal{A})$ and the checking of whether *reach*$(\mathcal{A})$ has a good subgraph can all be effectively done. Hence the theorem follows.

Proof of Claim 7.6 Suppose the game eventually stabilises. Then by definition, for all $w \in$ *init*$(\mathcal{A})$, there exists a node $w' = (c', \langle D'_1, \ldots, D'_n\rangle)$ reachable from w such that c' does not change from then on when the players play according to their types. This is ensured by the goodness conditions 1 and 2.

Conversely, suppose *reach*$(\mathcal{A})$ has a good subgraph. From the construction of the configuration graph $\mathcal{A}$, we know that if the players play according to their types, from every initial node in *init*$(\mathcal{A})$, a configuration w is reached such that the goodness conditions 1 and 2 are satisfied. These conditions imply that the game eventually stabilises. □

This ends the proof of the theorem. □

From the above proof, we also have the following:

Corollary 7.7 *The satisfiability problem for the logic* Φ *is decidable.*

Proof Given an initialised game $(\mathcal{G}, c_0)$ and a formula $\varphi \in \Phi$, we construct a graph $\mathcal{A}'$, similar to $\mathcal{A}$ in the proof above. $\mathcal{A}'$ is a product of the configurations of $(\mathcal{G}, c_0)$ and the atoms of φ. A node (c, D) in $\mathcal{A}'$ is called initial

if $c = c_0$ and D does not have any formula of the form $\ominus\beta$. A node (c, D) in $\mathcal{A}'$ is called final if for all $\Diamond\beta \in D, \beta \in D$ and for all $\bigcirc\beta \in D, \beta \in D$. It is then clear that checking whether φ is satisfiable amounts to checking whether there exists a final node reachable from an initial node in $\mathcal{A}'$. □

7.5 Unknown types

We have seen above that the restricted expressiveness of types specified by the logic gives us an algorithm for checking stationariness. Can we say anything about the stationariness of games where the types of the players are not known? In general, types may depend on history and hence require unbounded memory. However, we can characterise these games in terms of potentials á lá Monderer and Shapley [MS96]. We show that such games eventually stabilise if and only if they are "well-behaved" in terms of the potentials of configurations.

Since the types of the players are arbitrary, we do not require the logical language to specify them. Hence the utility of the normal form games is given as a function

$$p_a : \mathbf{Y} \to \mathbb{Q}$$

for every $a \in A$. For any neighbourhood X of size k and given a distribution $\mathbf{y}$ of actions of the players in X, $p_a(\mathbf{y})$ gives the payoff to all the players in X who play action a.

A game now is a tuple $\mathcal{G} = (typ, \{p_a\}_{a \in A})$ and an initialised game is a pair $(\mathcal{G}, c_0)$ where c_0 is a configuration from the set of configurations C where a configuration as before, is a neighbourhood graph labelled with the actions of the players. The configuration graph $\mathcal{C}$ and the tree unfolding of $\mathcal{C}$ from a configuration $c \in C$ is denoted as $\mathcal{T}_{\mathcal{C}}(c)$ and is defined as before.

7.5.1 Types of types

Just like finite memory strategies (Section 1.2.7), a type γ of a player is said to be finite memory if there exists a finite set M, the memory of the type, $m^I \in M$, the initial memory and functions $\delta : C \times M \to M$, the memory update function and $g : C \times M \to 2^{(2^N \times A)}$, the choice function where for every history $\rho = c_0 \dots c_k \in H$ if $m_0 \dots m_{k+1}$ is a sequence determined by $m_0 = m^I$ and $m_{i+1} = \delta(c_i, m_i)$ then $\gamma(\rho) = g(c_k, m_{k+1})$.

A type γ is memoryless if M is a singleton. A memoryless type only depends on the current configuration. That is, if for $\rho, \rho' \in H$ if $last(\rho) = last(\rho')$ then $\gamma(\rho) = \gamma(\rho')$.

Memoryless types

Theorem 7.8 *Let $(\mathcal{G}, c_0)$ be an initialised game such that the type of every player is memoryless. $(\mathcal{G}, c_0)$ is eventually stationary if and only if we can associate a potential ϕ_k with every round k such that if the game moves to a different configuration from round k to round $k+1$ then $\phi_{k+1} > \phi_k$ and the maximum possible potential of the game is bounded.*

Proof One direction is trivial: if such a potential exists, then the tree of possible configurations is finite, stabilising at leaf nodes.

For the other direction, assume that the game eventually stabilises. Then $\mathcal{T}_{\mathcal{C}}(c_0)$ has the following properties:

1. From the definition of stationariness (Definition 7.3) every branch of $\mathcal{T}_{\mathcal{C}}(c_0)$ eventually ends in a path such that the associated configuration does not change. That is for every branch b of $\mathcal{T}_{\mathcal{C}}(c_0)$, there exists a node t at a finite depth such that every successor of t has a unique child and for every successor t' of t, $c(t) = c(t')$. Call t a leaf node and remove the subtree of $\mathcal{T}_{\mathcal{C}}(c_0)$ rooted at t. After removing such a subtree from every branch of $\mathcal{T}_{\mathcal{C}}(c_0)$ we get a finite tree $\mathcal{T}_{\mathcal{C}}^{fin}(c_0) = (T_{fin}, E_{fin})$.

2. Along every branch of $\mathcal{T}_{\mathcal{C}}^{fin}(c_0)$, for every node t on that branch, there does not exist an ancestor t' of t such that $c(t) = c(t')$. Otherwise, we would have a cycle on the configuration $c(t)$ and since the types are memoryless, this would contradict the assumed eventual stationariness of the game.

We now assign a potential ϕ to every node of $\mathcal{T}_{\mathcal{C}}^{fin}(c_0)$ such that when ϕ is lifted to the configurations C of the game, ϕ is unique for every configuration $c \in C$. The potential ϕ assigned inductively.

1. For the root node, t_0 say, let $\phi(t_0) = 1$ and let $C_0 = \{t_0\}$.

2a. Suppose C_k has been constructed where C_k is a prefix closed set of nodes of $\mathcal{T}_{\mathcal{C}}^{fin}(c_0)$. Let $\phi_{\max} = \max\{\phi(t) \mid t \in C_k\}$. That is, $\phi_{\max}$ is the maximum of all the potentials assigned to a node so far. To make the rest of the proof notationally convenient, we define a few subsets of T_{fin} below:

   - The *boundary* of C_k, $B(C_k)$ are the nodes in C_k which have a child outside C_k, i.e., $B(C_k) = \{t \in C_k \mid \exists t' \in T_{fin},\ t \to t',\ t' \notin C_k\}$.

- The *interface* of C_k is the set $I(C_k) = \{t \in T_{fin} \mid t' \to t,\ t' \in B(C_k)\}$. That is, the interface of C_k are the nodes that are just outside the boundary.
- Let the *clearance* of C_k be the set $clear(C_k) = T_{fin} \setminus \{C_k \cup I(C_k)\}$. Thus the clearance of C_k are all the nodes of T_k that are still to be assigned a potential and do not belong to the interface.

We shall assign the next higher potential to one of the nodes in the interface of C_k. For that, we claim that there exists a node t in the interface of C_k, $t \in I(C_k)$, such that there does not exist any node in the clearance with the same configuration. That is, there does not exist $t' \in clear(C_k)$ such that $c(t) = c(t')$. We set $\phi(t) = \phi_{\max} + 1$.

2b. For all $t' \in I(C_k)$ such that $c(t) = c(t')$, we let $\phi(t') = \phi(t)$. That is, for every other node t' in the interface of C_k with the same configuration as the node t just processed, we assign the same potential to t' as that of t. This is required as the potential of every configuration should be unique. Finally, we go to the next stage by updating C_k to C_{k+1} as $C_{k+1} = C_k \cup \{t\} \cup \{t' \in I(C_k) \mid c(t) = c(t')\}$. In other words, we add to C_k all the nodes that have been newly assigned a potential and obtain C_{k+1}. Note that C_{k+1} remains prefix closed in the process, since we are only adding nodes that are in the interface of C_k.

After a potential has been assigned to all the nodes in $\mathcal{T}_{\mathcal{C}}^{fin}(c_0)$, we lift it back to the configurations of C as: for every $c \in C$, $\phi(c) = \phi(t)$, $t \in T_{fin}$ such that $c(t) = c$. Note that the potentials have been so assigned that they satisfy the condition $c(t) \to c(t')$ implies $\phi(t) < \phi(t)$. Also note that step 2b ensures that every configuration c receives a unique potential. To complete the proof we have to show that step 2a can always be performed.

Suppose, for contradiction, that step 2a cannot be performed for a prefix closed set C_k during the induction. That is, suppose for all $t \in I(C_k)$ there exists $t' \in clear(C_k)$ such that $c(t) = c(t')$.

Now let $t_1 \in I(C_k)$. By our assumption, there exists $t' \in clear(C_k)$ such that $c(t') = c(t_1)$. Let t_1' be such a node. Let t_2 be the ancestor of t_1' such that $t_2 \in I(C_k)$. Again by assumption there exists $t'' \in clear(C_k)$ such that $c(t'') = c(t_2)$. Let t_2' be such a node and let t_3 be the ancestor of t_2' such that $t_3 \in I(C_k)$. Continuing this way we have a sequence

$$t_1, t_1', t_2, t_2', t_3, t_3', \ldots$$

Now as the tree $\mathcal{T}_{\mathcal{C}}^{fin}(c_0)$ is finite, it is finitely branching. Hence the above process cannot go on forever and a configuration has to repeat. Suppose t_r

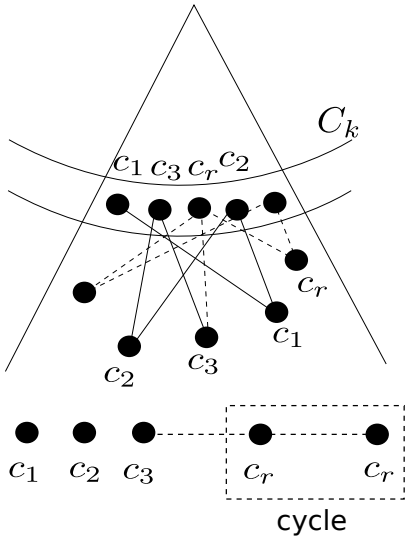

Figure 7.3: Step 2a of the proof of Theorem 7.8

be such that the ancestor of t_r in $I(C_k)$ is t_m for some $m < r$. But now since the types of the players are memoryless, this means that $t_m, t_r, t_{r-1}, \ldots, t_m$ forms a cycle of configurations when the players play according to their types. This violates property 2 of the tree unfolding $\mathcal{T}_{\mathcal{C}}(c_0)$ of the game as mentioned above (see figure 7.3). □

Stability

The notion of stationariness introduced in the previous section is a bit too rigid. As in the example of the vegetable seller in section 7.1, a periodic visit to markets A, B and C in that order is an instance of stable behaviour for us. We wish to capture such a behaviour in our notion of stability.

Here we introduce another notion of stable behaviour which we call eventual stability.

Definition 7.9 *A set C of configurations is called stable if C is either a simple cycle with respect to the follows relation $\rightarrow$ or a singleton. We say that a game eventually stabilises if it always ends in a stable set of configurations.*

Note that we do not allow complex cycles in a stable set of configurations because then it would make even non-deterministic plays stable. If

a complex cycle C consists of two simple cycles C_1 and C_2 then the players can eventually settle down to C even by playing C_1 and C_2 without any particular order. But for a simple cycle C, though the players have a non-deterministic choice of whether to remain in C or to exit C, note that once they exit C they cannot come back to it again. Thus, they are eventually either in a (simple) cyclic play or the configuration of the game doesn't change anymore, both of which are accepted notions of stability for us. But, of course, it is just a matter of choice. One may have other, perfectly justifiable notions of stability.

General types

In this subsection, we prove a theorem similar to Theorem 7.8 for the case when the types of the players are unknown but arbitrary (not just memoryless).

Theorem 7.10 *Let $(\mathcal{G}, c_0)$ be an initialised game. $(\mathcal{G}, c_0)$ eventually stabilises if and only if we can associate a potential ϕ_k with every round k such that the following holds:*

1. *If the game has not yet stabilised in round k then there exists a round $k' > k$ such that $\phi_{k'} > \phi_k$.*
2. *There exists $k_0 \geq 0$ such that for all $k, k' > k_0$, $\phi_k = \phi_{k'}$. That is, the potential of the game becomes constant eventually.*
3. *The maximum potential of the game is bounded.*

Proof For the non-trivial direction, assume that the game eventually stabilises. Then from the definition of eventual stability (Definition 7.9 the tree $\mathcal{T}_{\mathcal{C}}(c_0)$ has the following property. Along every branch b of $\mathcal{T}_{\mathcal{C}}(c_0)$ there exists a node t at a finite depth such that b is just a path from t onwards and either of the following holds:

- b ends in a self-loop: that is, for every successor t' of t, $c(t') = c(t)$. Call t a leaf node and remove the subtree rooted at t.
- b ends in a simple cycle of configurations: that is the following holds. t has a successor t' along b such that $c(t) = c(t')$. Let $t_{\min}$ be the least such successor and let t'' be the parent of $t_{\min}$. Let ρ be the path from t to t'' in $\mathcal{T}_{\mathcal{C}}(c_0)$. It is the case that from t, the branch b is just a sequence of sets of nodes $\rho\rho_1\rho_2\ldots$ such that $|\rho| = |\rho_1| = |\rho_2| = \ldots$ and

for every $i \geq 1$ and every $j : 0 \leq j < |\rho|$, $c(\rho_i(j)) = c(\rho(j))$. In other words the configurations along ρ keep repeating in b forever from t in the same order.

Call t a leaf node and remove the subtree rooted at t.

After the above procedure we have a finite tree $\mathcal{T}_{\mathcal{C}}^{fin}(c_0) = (T_{fin}, E_{fin})$. We assign a potential ϕ to every node of $\mathcal{T}_{\mathcal{C}}^{fin}(c_0)$ such that when ϕ is lifted back to the configurations C of the game, the requirements of the theorem are satisfied. The potential ϕ is assigned inductively.

Initially let $C_0 = \emptyset$ and $\phi^0_{\max} = 0$. Suppose C_k has been constructed where C_k is a prefix closed set of nodes of $\mathcal{T}_{\mathcal{C}}^{fin}(c_0)$ and let $\phi^k_{\max}$ be the maximum potential of any node in C_k. Let $I(C_k) = \{t \in T_{fin} \mid t \notin C_k, t' \rightarrow t, t' \in C_k\}$ be the interface of C_k as in the proof of Theorem 7.8. We construct a set of *critical* nodes $crit(C_k) \subseteq T_{fin} \setminus C_k$ inductively as follows.

- Let $crit^0(C_k) = \{t\}$ where $t \in I(C_k)$ is an arbitrary node.

- Suppose $crit^i(C_k)$, $i \geq 0$ has been constructed. If there exists $t \in crit^i(C_k)$ be such that there exists $t' \in T_{fin} \setminus (C_k \cup crit^i(C_k))$ with $c(t') = c(t)$ then we construct $crit^{i+1}(C_k)$ as follows. We let $T_t \subseteq T_{fin} \setminus C_k$ be $T_t = \{t' \in T_{fin} \setminus (C_k \cup crit^i(C_k)) \mid c(t') = c(t)\}$. Let $closure(T_t)$ be the upward closure of the nodes in T_t till the interface of C_k. That is $closure(T_t) = \{t' \in T_{fin} \setminus C_k \mid \exists t'' \in T_t, t' \text{ is an ancestor of } t''\}$. We let $crit^{i+1}(C_k) = crit^i(C_k) \cup closure(T_t)$.

Since $\mathcal{T}_{\mathcal{C}}^{fin}(c_0)$ is finite, there exists a $j \geq 0$ such that $crit^{j+1}(C_k) = crit^j(C_k)$. We set $crit(C_k) = crit^j(C_k)$. Put $\phi(t) = \phi^k_{\max} + 1$ for every $t \in crit(C_k)$ and set $C_{k+1} = C_k \cup crit(C_k)$. Note that C_{k+1} is a prefix-closed set and for every node $t \in C_{k+1}$, there does not exist $t' \in T_{fin} \setminus C_{k+1}$, such that $c(t') = c(t)$ (otherwise t' would have been added to $crit(C_k)$ while processing t).

After ϕ has been assigned to all the nodes in $\mathcal{T}_{\mathcal{C}}^{fin}(c_0)$, we let for every $c \in C$, $\phi(c) = \phi(t)$, $t \in T_{fin}$ such that $c(t) = c$. Note that the process of saturation in the construction of the critical sets ensures that if a node t is assigned a potential at some iteration then all nodes t' such that $c(t') = c(t)$ are assigned the same potential in the same iteration. This guarantees the uniqueness of the potential for every configuration. Also the assignment of the potentials in a top-down fashion on the unfolding $\mathcal{T}_{\mathcal{C}}^{fin}(c_0)$ of the configuration graph, makes sure that the other requirements of the theorem are satisfied. □

Finite memory types

From the proof of Theorem 7.10 we easily have the following theorem:

Theorem 7.11 *Let $(\mathcal{G}, c_0)$ be an initialised game. If $(\mathcal{G}, c_0)$ eventually stabilises then the types of all the players are finite memory.*

Proof Assume that $(\mathcal{G}, c_0)$ stabilises. Let $\mathcal{T}_{\mathcal{C}}^{fin}(c_0)$ be the finite tree as constructed in the proof of theorem 7.10. Then the required finite memory type of each player i is γ_i such that the memory of γ_i is the set of nodes T_{fin} of $\mathcal{T}_{\mathcal{C}}^{fin}(c_0)$. The initial memory is the root of $\mathcal{T}_{\mathcal{C}}^{fin}(c_0)$. The memory update function is given by the edge relation in $\mathcal{T}_{\mathcal{C}}^{fin}(c_0)$ and the choice function g_i at a memory node $t = (G, c)$ is given by $g_i(c, t) = \{(X, a) \mid (X, a)$ corresponds to the choice of i at a child t' of $t\}$. □

7.6 Application to weighted co-ordination games

To gain intuition into the dynamics of these games with neighbourhood structures and to see how the results above can be applied, in this section, we study a special case of such games which we call weighted co-ordination games under the assumption that all the players play simple imitative strategies.

Co-ordination games appear everywhere in game-theory in various disguises. It is the simultaneous and private selection of the moves/strategies by the players that makes these games interesting. Many of the common strategic form games are of this flavour. For example, in Prisoners' Dilemma, it is best for both the prisoners to co-ordinate and co-operate. In Bach and Stravinsky, the couple would rather co-ordinate and stay together than be selfish and be separated. In the weighted version of co-ordination games, there are $n \geq 2$ players who simultaneously and privately wish to co-ordinate. In our case, as we shall see presently, the size of a neighbourhood may affect the amount of co-ordination in that neighbourhood. So we normalise the payoffs with respect to the neighbourhood sizes and hence the term *weighted* co-ordination game.

First, we formally define these games. For simplicity we assume that the action set of every player is binary, that is $A = \{0, 1\}$. The payoff of the players are determined by the amount of co-ordination in the neighbourhood they belong to. More precisely, let X be a neighbourhood and let $X_0[G_k]$ be the set of players who play the action 0 in round k and $X_1[G_k]$ be the

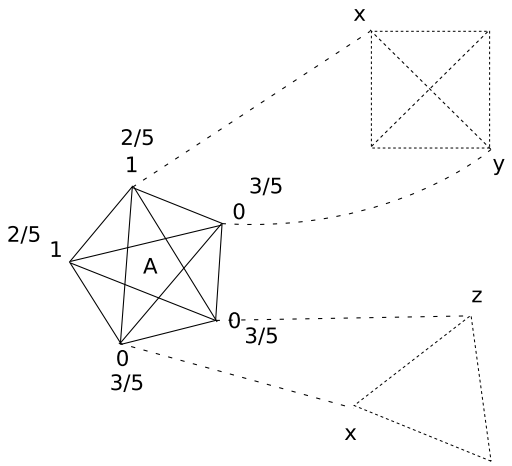

Figure 7.4: An example of a weighted co-ordination game. The players playing 1 in the neighbourhood A receive a payoff of 2/5 whereas those playing 0 receive 3/5.

set of players who play the action 1. Then the payoff to a player i in X who plays 0 is given as

$$p_0(|X_0[G_k]|, |X_1[G_k]|) = \frac{|X_0|}{|X|}$$

and the payoff of a player j in X who plays 1 is given as

$$p_1(|X_0[G_k]|, |X_1[G_k]|) = \frac{|X_1|}{|X|} = 1 - p_0(|X_0[G_k]|, |X_1[G_k]|)$$

See Figure 7.4 for an example.

We show that when all the players are of a simple imitative type (to be defined presently), we can associate a potential to every configuration c which is bounded and such that if c' follows c then its potential is strictly greater than c. Hence by Theorem 7.8, such games always stabilise. We can also give an upper bound on the number of rounds required to attain stability.

We first describe a simple imitative type t. We assume that every player is of type t. Suppose player i plays a in round k where $a \in \{0, 1\}$. If in round k player i receives a payoff less than 0.5 and there exists a player j visible to her who in round k received the maximum payoff among all the players visible to her, then in round $k+1$ i plays the action of j. This type t is given by the following formula:

$$\alpha_s \equiv \forall x.(p@i \wedge i \leftrightarrow x \wedge q@x \wedge r@x \wedge p_a@x) \supset \bigcirc(p_a@i)$$

where p, q, r and p_a are propositions which say:

- p: payoff is less than 0.5.
- q: payoff is greater than that of i.
- r: payoff is greater than that of all neighbours of i.
- p_a: plays action a (for some $a \in A$).

If it is a dynamic neighbourhood game, the type is the following: if there exists a player j within the visibility of i who is in a different neighbourhood X' but plays the same action and gets a better payoff in round k, player i joins the neighbourhood X' of such a player with the maximum such payoff. This is given by a very similar formula:

$$\alpha_d \equiv \forall x.(p@i \wedge (i \leftrightarrow x \wedge q@x \wedge r@x \wedge \neg[i,x]) \supset \bigcirc[i,x]$$

Theorem 7.12 *Let $(\mathcal{G}, c_0)$ be a game with initial neighbourhood graph G and a static neighbourhood structure and let all the players be of the same type t defined by α_s. Let m be the number of neighbourhoods (cliques) and $M = \max_{X \in clq(G)} |X|$. Then the game always stabilises and it does so in at most mM steps.*

Proof At any round k define the potential of a neighbourhood X to be

$$\phi_k(X) = \max\{|X_0|, |X_1|\}$$

Define the potential of the game in round k to be

$$\phi_k = \Sigma_X \phi_k(X)$$

Note that in any neighbourhood X, the players with the higher payoffs never change their actions. Hence any neighbourhood X where in round k either $|X_0| > |X_1|$ or $|X_1| > |X_0|$, X is already stable or $\phi_{k+1}(X) > \phi_k(X)$ (since more and more players toggle to the action with the higher payoff).

Now consider a neighbourhood X where in round k, $|X_0| = |X_1|$. Then either in round $k+1$, $|X_0| > |X_1|$ or $|X_1| > |X_0|$ and we are in the previous case or exactly equal number of players of X switch from 0 to 1 and 1 to 0 in round $k+1$. We show that such a thing happens only for boundedly many steps.

The worst case arises when there are two players $i, j \in X$ such that in round k the action played by player i, $a(i) = 0$ and that played by player

j, $a(j) = 1$ and both of them toggle their action in every subsequent round. But this means that in round k, i is adjacent to a neighbourhood $X[i]$ such that $p_1^k(X[i]) > p_0^k(X)$ and j is adjacent to a neighbourhood $X[j]$ such that $p_0^k(X[j]) > p_1^k(X)$. But since i and j again toggle their actions in round $k+2$, it means that in round $k+1$, i is adjacent to a neighbourhood $X'[i]$ such $p_1^{k+1}(X'[i]) > p_0^{k+1}(X)$ and j is adjacent to a neighbourhood $X'[j]$ such that $p_0^{k+1}(X'[j]) > p_1^{k+1}(X)$ and so on. Thus these neighbourhoods $X[i], X'[i], X[j], X'[j], \ldots$ are not stable and hence by the previous argument, $\phi_k(X[i]) > \phi_{k-1}(X[i]), \phi_k(X[j]) > \phi_{k-1}(X[i]), \phi_{k+1}(X'[i]) > \phi_k(X'[i]), \phi_{k+1}(X'[j]) > \phi_k(X'[i]), \ldots$. Hence the potential in round $k+1$ is strictly greater than that in round k.

Now, since $\max_k \phi_k = mM$ and they type α_s is finite memory, Theorem 7.10 implies that the game stabilises in at most mM steps with the following possible configurations:

- Some of the neighbourhoods are part of a maximal alternating chunk.
- For some neighbourhoods X, $p_0(X) = p_1(X)$.
- For some neighbourhoods X, $p_0(X) = 1$ or $p_1(X) = 1$.

□

Theorem 7.13 *Let $(\mathcal{G}.c_0)$ be an n-player game with initial neighbourhood graph G and a dynamic neighbourhood structure and let all the players be of the same type t defined by α_d. Then the game always stabilises in at most $n^{n(n+1)/2}$ steps.*

Proof The possible payoffs of a player in any round are

$$1/n, 2/n, \ldots, 1/(n-1), 2/(n-1), \ldots, 1$$

We arrange them in ascending order. We now define the potential of a payoff p in round k inductively as follows:

$$\phi_k(1/n) = 1,\ \phi_k(p') = \phi_k(p) + 1$$

where p is the immediate predecessor of p' in the ordering of the payoffs. These potentials can be lifted to the vertices (players) as $\phi_k(i) = \phi_k(p_{a(i)}(X(i)))$. We define the potential of a neighbourhood X in round k as $\phi_k(X) = \Sigma_{i \in X} \phi_k(i)$. Finally we define the potential of the game in round

k as $\phi_k = \Sigma_X \phi_k(X)$ where the sum is over the set of all neighbourhoods (cliques) in the neighbourhood graph G_k for round k.

Now let us look at what happens in each round k. Either for every (relevant) pair of neighbourhoods X and X', $p_0^k(X) = p_0^k(X')$ and $p_1^0(X) = p_1^0(X')$ and the configuration is already stable or otherwise there exists a neighbourhood X such that either $p_0^k(X)$ or $p_1^0(X)$ is the minimum of the payoffs of all the players. Without loss of generality suppose $p_0^k(X)$ is the minimum payoff. In that case $p_1^k(X) = 1 - p_0^k(X)$ is the maximum payoff. Moreover, we can also assume that X is such that there exists a neighbourhood X' adjacent to X such that $p_0^k(X') > p_0^k(X)$ [otherwise the configuration is already stable]. Now by the way the function $\phi_k(\cdot)$ has been defined, $\phi_k(X)$ will be the maximal of all the potentials of all the neighbourhoods reachable from X. That is for all reachable neighbourhoods X' where neither $p_0^k(X')$ nor $p_1^k(X')$ is the minimum, it is the case that $\phi_k(X') < \phi_k(X)$.

As a result in round $k+1$, at least 1 player from X_0 switches to an adjacent neighbourhood X'. Also since $p_1^k(X)$ was maximum, the players of X_1 do not switch neighbourhoods (but of course other players from different neighbourhoods who play 1 may join). Thus, $\phi_{k+1}(X) > \phi_k(X)$. Now it might be the case that $\phi_{k+1}(X') < \phi_k(X')$. But notice that the potential function has been so defined that since $\phi_k(X) > \phi_k(X')$, the increase in $\phi_k(X)$ to $\phi_{k+1}(X)$ is strictly greater than the decrease in $\phi_k(X')$ to $\phi_{k+1}(X')$. Hence the overall potential of the game increases from round k to $k+1$, $\phi_{k+1} > \phi_k$.

Thus for an unstable configuration the overall potential always increases in the next round.

Now, the total number of different payoffs is at most $n(n+1)/2$ and the maximum possible potential of a node is

$$\underbrace{1 + n + n(n+1) + n[n(n+1)+1] + \ldots}_{n(n+1)/2}$$

which is at most $n^{n(n+1)/2}$. Hence the maximum potential of the game is at most $n \times n^{n(n+1)/2}$.

Also as shown above, the if the configuration changes from round k to $k+1$ then the potential always strictly increases. Finally, since the type α_d is finite memory, Theorem 7.10 implies that the game always stabilises.

Moreover, as the potential increases by at least n in every round, the number of steps required for the game to stabilise is at most $n^{n(n+1)/2}$. □

Remark When there is an upper bound on the size of any clique (say M) in a dynamic neighbourhood structure, then the number of steps to stability can be bounded in terms of M. In the worst case, this makes no difference, but may be significant in some applications.

Chapter 8

Conclusions, extensions and future directions

We hope to have convinced the reader of the importance of looking at strategies as structured objects rather than viewing them as black-boxes. Moreover, when strategies are built up structurally, switching constitutes an integral part of such composition. We also hope to have made the point that when strategies are thus viewed, the equilibria and stability analyses of games, especially large games, can be less imposing and can provide various insights into the structure of the game dynamics.

In this concluding chapter, we look at possible extensions to the work presented in the previous chapters.

8.1 Reducing the number of players

As we pointed out in various places in the previous chapters, large games lack some of the desirable qualities of the traditional games: viz., perfect information, bounded number of players, unbounded computational resource etc. However, there is usually one feature of most of these games that may make them amenable to tractable analysis. As pointed out earlier, esp. in Chapters 6 and 7, in such games there are usually a finite number t of types such that $t << n$ where n is the number of players. Hence, it would be nice if one could carry out all the analysis using only the t types and then lift the results to the entire game.

In this section, we give some pointers to that direction. We study situations when such an analysis can be carried out when the types of the players are specified using finite state transducers.

Determinisation of finite state transducer

We wish to define the product of a strategy transducer $\mathcal{Q}$ with itself. And as we shall notice presently, we need the states of the product transducer to be of the form (q, q). One way to guarantee this is to ensure that the transducer $\mathcal{Q}$ itself is deterministic. We first see how to determinise such a transducer.

As usual let $N = \{1, 2, \ldots, n\}$ be the set of players and assume that the players have a common action set A. Let $\mathbf{A} = A^n$. Let $\mathcal{Q} = (Q, \delta, I, f)$ be an FST with input alphabet $\mathbf{A}$ and output alphabet A. We determinise $\mathcal{Q}$ to obtain another FST $\mathcal{Q}' = (Q', \delta', q_0', f')$ with input and output alphabets $\mathbf{A}$ and 2^A respectively as follows. (Q', δ', q_0') is the determinisation of the automaton (Q, δ, I) and $f' : Q' \to 2^A$ where

$$f'(X) = \{f(q) \mid q \in X\}.$$

Definition 8.1 *Let $\mathcal{Q}_1 = (Q_1, \delta_1, q_o^1, f_1)$ and $\mathcal{Q}_2 = (Q_2, \delta_2, q_0^2, f_2)$ be deterministic FSTs. Let $\mathcal{Q}_1 \times \mathcal{Q}_2 = (Q, \delta, q_0)$ be the product of the automata (Q_1, δ_1, q_0^1) and (Q_2, δ_2, q_0^2), that is:*

- $Q = Q_1 \times Q_2$,
- $q_0 = (q_0^1, q_0^2)$,
- $\langle (q_1, q_2), \mathbf{a}, (q_1', q_2') \rangle \in \delta$ *iff* $(q_1, \mathbf{a}, q_1') \in \delta_1$ *and* $(q_2, \mathbf{a}, q_2') \in \delta_2$.

Delete all the unreachable states of $\mathcal{Q}_1 \times \mathcal{Q}_2$ and call the resulting automaton $\mathcal{Q}'$. Then $\mathcal{Q}_1 \otimes \mathcal{Q}_2 = (\mathcal{Q}', f)$ where $f : Q \to 2^A \times 2^A$ such that $f(q_1, q_2) = (f(q_1), f(q_2))$.

Definition 8.2 *Let $\mathcal{Q} = (Q, \delta, q_0, f)$ be a deterministic FST. Let $\mathcal{Q} \times \mathcal{Q}$ be the product of (Q, δ, q_0) with itself. Delete all the unreachable states of $\mathcal{Q} \times \mathcal{Q}$ and call the resulting automaton $\mathcal{Q}' = (Q', \delta', q_0')$. Then $\mathcal{Q} \otimes \mathcal{Q} = (\mathcal{Q}', f')$ where $f' : Q' \to 2^A$ such that $f'(q, q) = f(q)$.*

This is well-defined because of the following claim.

Claim 8.3 *The only states in $\mathcal{Q} \otimes \mathcal{Q}$ are of the form (q, q).*

Proof Suppose not and suppose there exists a state in $\mathcal{Q} \otimes \mathcal{Q}$ of the form (q, q') such that $q \neq q'$. Since (q, q') is reachable from (q_0, q_0), there exists a path from (q_0, q_0) to (q, q') labelled by u say. Suppose (q, q') is the first such state along this path (otherwise repeat the argument for the

first such state) and suppose $|u| = k$. Then $(q_0, q_0) \stackrel{u}{\rightarrow} (q, q')$ and hence $(q_0, q_0) \stackrel{u(1)\ldots u(k-1)}{\rightarrow} (q_{k-1}, q_{k-1}) \stackrel{u(k)}{\rightarrow} (q, q')$. But this means $q_{k-1} \stackrel{u(k)}{\rightarrow} q$ and $q_{k-1} \stackrel{u(k)}{\rightarrow} q'$ contradicting the determinacy of $\mathcal{Q}$. □

We next develop notions of when an FST captures the behaviour of another and when two FSTs are equivalent.

Definition 8.4 *Let $\mathcal{Q}_1 = (Q_1, \delta_1, q_0^1, f_1)$ and $\mathcal{Q}_2 = (Q_2, \delta_2, q_0^2, f_2)$ be deterministic FST's. $\mathcal{Q}_1 \sqsubseteq \mathcal{Q}_2$ if there exists functions $h : Q_2 \rightarrow Q_1$ and $g : A_2 \rightarrow A_1$ where A_2 and A_1 are output alphabets of $\mathcal{Q}_2$ and $\mathcal{Q}_1$ respectively, such that*

- $h(q_0^2) = q_0^1$,
- $(q_2, \mathbf{a}, q_2') \in \delta_2$ *iff* $(h(q_2), \mathbf{a}, h(q_2')) \in \delta_1$,
- $f_2(q) = g(f_1(h(q)))$.

$\mathcal{Q}_1 \equiv \mathcal{Q}_2$ if $\mathcal{Q}_1 \sqsubseteq \mathcal{Q}_2$ and $\mathcal{Q}_2 \sqsubseteq \mathcal{Q}_1$.

Definition 8.5 *Let (G_1, v_0^1) and (G_2, v_0^2) be two initialised directed edge labelled graphs where $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. $G_1 \sqsubseteq G_2$ if there exists $h : V_2 \rightarrow V_1$ such that $h(v_0^2) = v_0^1$ and $v_2 \stackrel{a}{\rightarrow} v_2'$ iff $h(v_2) \stackrel{a}{\rightarrow} h(v_2')$. $G_1 \equiv G_2$ if $G_1 \sqsubseteq G_2$ and $G_2 \sqsubseteq G_1$.*

Claim 8.6 $\mathcal{Q}_1 \otimes \mathcal{Q}_2 \equiv \mathcal{Q}_2 \otimes \mathcal{Q}_1$

Proof Let $\mathcal{Q}_1 = (Q_1, \delta_1, q_0^1, f_1)$ and $\mathcal{Q}_2 = (Q_2, \delta_2, q_0^2, f_2)$ be deterministic FST's. Let $\mathcal{Q}_1 \otimes \mathcal{Q}_2 = (Q, \delta, q_0, f)$ and let $\mathcal{Q}_2 \otimes \mathcal{Q}_1 = (Q', \delta', q_0', f')$. We show $\mathcal{Q}_1 \otimes \mathcal{Q}_2 \sqsubseteq \mathcal{Q}_2 \otimes \mathcal{Q}_1$. Let $h(q_2, q_1) = (q_1, q_2)$ and $g(X_2, X_1) = (X_1, X_2)$. Now, $\langle (q_2, q_1), \mathbf{a}, (q_2', q_1') \rangle \in \delta$ iff $(q_2, \mathbf{a}, q_2') \in \delta_1$ and $(q_1, \mathbf{a}, q_1') \in \delta_2$ iff $\langle (q_1, q_2), \mathbf{a}, (q_1', q_2') \rangle \in \delta'$ i.e., $\langle h(q_2, q_1), \mathbf{a}, h(q_2', q_1') \rangle \in \delta'$. Let $f'(q_2, q_1) = (X_2, X_1)$, i.e., $f_2(q_2) = X_2$, $f_1(q_1) = X_1$. Thus, by definition, $f(q_1, q_2) = (X_1, X_2)$, that is, $f'(q_2, q_1) = g(f(h(q_2, q_1)))$.

The other direction is symmetrical where the function h' (say) is given as $h'(q_2, q_1) = (q_1, q_2)$. □

Note that in the above proof, since $h' = h^{-1}$ we have that $\mathcal{Q}_1 \otimes \mathcal{Q}_2$ is, in fact, isomorphic to $\mathcal{Q}_2 \otimes \mathcal{Q}_1$, the isomorphism being h.

Claim 8.7 $\mathcal{Q} \otimes \mathcal{Q} \equiv \mathcal{Q}$

Proof Let $\mathcal{Q} = (Q, \delta, q_0, f)$ and $\mathcal{Q} \otimes \mathcal{Q} = (Q', \delta', q_0', f')$. By Claim 8.3 the states of $\mathcal{Q} \otimes \mathcal{Q}$ are only of the form (q, q). To see that $\mathcal{Q} \otimes \mathcal{Q} \sqsubseteq \mathcal{Q}$ let $h(q) = (q, q)$ and $g(X) = X$. Then we have $(q, \mathbf{a}, q') \in \delta$ iff $\langle (q, q), \mathbf{a}, (q', q') \rangle \in \delta'$, i.e., $\langle h(q), \mathbf{a}, h(q') \rangle \in \delta'$. And $f(q) = X$ iff $f'(q, q) = X$, i.e., $f(q) = g(f'(h(q)))$. To see that $\mathcal{Q} \sqsubseteq \mathcal{Q} \otimes \mathcal{Q}$, let $h'(q, q) = q$ and $g'(X) = X$. The argument is then similar to above. □

Once more note that in the above proof, since $h' = h^{-1}$ we have that $\mathcal{Q} \otimes \mathcal{Q}$ is isomorphic to $\mathcal{Q}$, the isomorphism being h.

Let $(\mathcal{A}, v_0)$ be an initialised arena where $\mathcal{A} = (V, E)$. Suppose every player is one of t types where each type is specified as an FST, $\mathcal{Q}$ say. We define the restriction of $\mathcal{A}$ with respect to $\mathcal{Q}$, denoted by $\mathcal{A} \otimes \mathcal{Q}$, similar to the restriction operation $\mathcal{A} \restriction \mathcal{Q}$ defined in Chapters 4,5 and 6, as follows:

Definition 8.8 *Let $\mathcal{Q} = (Q, \delta, q_0, f)$ be an FST of type i. Let $\mathcal{A} \otimes \mathcal{Q} = (\mathcal{A}', v_0')$ such that*

- $V' = V \times Q$,
- $v_0' = (v_0, q_0)$,
- $(v, q) \xrightarrow{\mathbf{a}} (v', q')$ *iff* $v \xrightarrow{\mathbf{a}} v'$, $(q, \mathbf{a}, q') \in \delta$ *and* $\mathbf{a}(i) \in f(q)$.

Definition 8.9 *Let $\mathcal{Q}_1 \otimes \cdots \otimes \mathcal{Q}_n = (Q, \delta, q_0, f)$. $\mathcal{A} \otimes (\mathcal{Q}_1 \otimes \cdots \otimes \mathcal{Q}_n) = (\mathcal{A}' v_0')$ such that $\mathcal{A}' = (V', E')$*

- $V' = V \times Q$,
- $v_0' = (v_0, q_0)$,
- $(v, q) \xrightarrow{\mathbf{a}} (v', q')$, *where v is of type i, iff* $v \xrightarrow{\mathbf{a}} v'$, $(q, \mathbf{a}, q') \in \delta$ *and* $a(i) \in (f(q))(i)$, *for all* $i \in N$.

Claim 8.10 $\mathcal{A} \otimes (\mathcal{Q}_1 \otimes \cdots \otimes \mathcal{Q}_n) \equiv (\cdots((\mathcal{A} \otimes \mathcal{Q}_1) \otimes \mathcal{Q}_2) \cdots \otimes \mathcal{Q}_k)$

Proof Let $(\mathcal{Q}_1 \otimes \cdots \otimes \mathcal{Q}_n) = (Q, \delta, q_0, f)$ and let $\mathcal{A} \otimes (\mathcal{Q}_1 \otimes \cdots \otimes \mathcal{Q}_n) = (\mathcal{A}_1, v_0^1)$ and $(\cdots((\mathcal{A} \otimes \mathcal{Q}_1) \otimes \mathcal{Q}_2) \cdots \otimes \mathcal{Q}_k) = (\mathcal{A}_2, v_0^2)$. Now as per definition

- The set of states of both $\mathcal{A}_1$ and $\mathcal{A}_2$ are the same.
- If $(v, q) \in \mathcal{A}_1$, then $(v, q) \xrightarrow{\mathbf{a}} (v', q')$ iff $v \xrightarrow{\mathbf{a}} v'$, $(q, \mathbf{a}, q') \in \delta$ and $a(i) \in f(q)(i)$ for all $i \in N$. And if $(v, q) \in \mathcal{A}_2$ then while the constructing $(\cdots(\mathcal{A} \otimes \mathcal{Q}_1) \otimes \cdots \mathcal{Q}_k)$, at the ith step we would have $a(i) \in f(q)(i)$ for all $i \in N$ and at the end of the construction we would have $(v, q) \xrightarrow{\mathbf{a}} (v', q')$ iff $v \xrightarrow{\mathbf{a}} v'$ and $(q, \mathbf{a}, q') \in \delta$.

□

Let $\mathcal{A} = (V, E)$ be an arena and $\mathcal{P}$ be a set of atomic propositions and $val : V \to 2^{\mathcal{P}}$ be a valuation function. Let α be a formula from the syntax Φ_+ (section 1.2.9). As in Section 4.5.2, we say α is stable in a subarena $\mathcal{Z} = (V_{\mathcal{Z}}, E_{\mathcal{Z}})$ of $\mathcal{A}$ if $t \models \varphi$ for all nodes $t \in \mathcal{T}_{\mathcal{Z}}$ for unfoldings $\mathcal{T}_{\mathcal{Z}}$ starting at every node $z \in V_{\mathcal{Z}}$.

Proposition 8.11 *Let $\mathcal{Q}, \mathcal{Q}_1$ and $\mathcal{Q}_2$ be FSTs for types of players. We have*

(i) α is stable in $\mathcal{A} \otimes \mathcal{Q}$ iff α is stable in $\mathcal{A} \otimes (\mathcal{Q} \otimes \mathcal{Q})$.

(ii) α is stable in $\mathcal{A} \otimes \mathcal{Q}_1 \otimes \mathcal{Q}_2$ iff α is stable in $\mathcal{A} \otimes \mathcal{Q}_2 \otimes \mathcal{Q}_1$.

Proof

(i) Let $\mathcal{A} \otimes \mathcal{Q} = (V_1, E_1)$ and let $\mathcal{A} \otimes (\mathcal{Q} \otimes \mathcal{Q}) = (V_2, E_2)$. Since we know from the proof of Claim 8.7 that $\mathcal{Q}$ is isomorphic to $\mathcal{Q} \otimes \mathcal{Q}$, we have, by construction, that $\mathcal{A} \otimes \mathcal{Q}$ is isomorphic to $\mathcal{A} \otimes (\mathcal{Q} \otimes \mathcal{Q})$. Let h be this isomorphism. Let $\mathcal{T}_{\mathcal{A} \otimes \mathcal{Q}}(v_0, q_0)$ denote the unfolding of $\mathcal{A} \otimes \mathcal{Q}$ at any node $(v_0, q_0) \in V_1$ and let $\mathcal{T}_{\mathcal{A} \otimes (\mathcal{Q} \otimes \mathcal{Q})}(v_0, q_0, q_0)$ denote the unfolding of $\mathcal{A} \otimes (\mathcal{Q} \otimes \mathcal{Q})$ at any node $(v_0, q_0, q_0) \in V_2$.

Suppose α is stable in $\mathcal{A} \otimes \mathcal{Q}$. Then, by definition, for any node $t = (v, q, \mathbf{u}) \in \mathcal{T}_{\mathcal{A} \otimes \mathcal{Q}}(v_0, q_0)$, $(v_0, q_0) \in V_1$, we have $t \models \alpha$. We have to show that $\langle h^{-1}(v, q), \mathbf{u} \rangle \models \alpha$ for the node $\langle h^{-1}(v, q), \mathbf{u} \rangle \in \mathcal{T}_{\mathcal{A} \otimes (\mathcal{Q} \otimes \mathcal{Q})}(h^{-1}(v_0, q_0))$. The only interesting case is when $\alpha \equiv \langle \mathbf{a} \rangle^+ \alpha$. $(v, q, \mathbf{u}) \models \langle \mathbf{a} \rangle^+ \alpha$ iff there exists $(v', q', \mathbf{ua})$ such that $(v, q) \xrightarrow{\mathbf{a}} (v', q')$ and $(v', q', \mathbf{ua}) \models \alpha$ iff $(v', q', q', \mathbf{ua}) \models \alpha$, that is, iff $\langle h^{-1}(v, q), \mathbf{u} \rangle \models \alpha$. The other direction is similar.

(ii) Let $\mathcal{A} \otimes \mathcal{Q}_1 \otimes \mathcal{Q}_2 = (V_1, E_1)$ and let $\mathcal{A} \otimes \mathcal{Q}_2 \otimes \mathcal{Q}_1 = (V_2, E_2)$. Since we know from the proof of Claim 8.6 that $\mathcal{Q}_1 \otimes \mathcal{Q}_2$ is isomorphic to $\mathcal{Q}_2 \otimes \mathcal{Q}_1$, we have, by construction, that $\mathcal{A} \otimes \mathcal{Q}_1 \otimes \mathcal{Q}_2$ is isomorphic to $\mathcal{A} \otimes \mathcal{Q}_2 \otimes \mathcal{Q}_1$. Let h be this isomorphism. Let $\mathcal{T}_{\mathcal{A} \otimes \mathcal{Q}_1 \otimes \mathcal{Q}_2}(v_0, q_0^1, q_0^2)$ denote the unfolding of $\mathcal{A} \otimes \mathcal{Q}_1 \otimes \mathcal{Q}_2$ at any node $(v_0, q_0^1, q_0^2) \in V_1$ and let $\mathcal{T}_{\mathcal{A} \otimes \mathcal{Q}_2 \otimes \mathcal{Q}_1}(v_0, q_0^2, q_0^1)$ denote the unfolding of $\mathcal{A} \otimes \mathcal{Q}_2 \otimes \mathcal{Q}_1$ at any node $(v_0, q_0^2, q_0^1) \in V_2$.

Suppose α is stable in $\mathcal{A} \otimes \mathcal{Q}_1 \otimes \mathcal{Q}_2$. Then, by definition, for any node $t = (v, q_1, q_2, \mathbf{u}) \in \mathcal{T}_{\mathcal{A} \otimes \mathcal{Q}_1 \otimes \mathcal{Q}_2}(v_0, q_0^1, q_0^2)$, $(v_0, q_0^1, q_0^2) \in V_1$, we have $t \models \alpha$. We have to show that $\langle h^{-1}(v, q_1, q_2), \mathbf{u} \rangle \models \alpha$ for the node $\langle h^{-1}(v, q_1, q_2), \mathbf{u} \rangle \in \mathcal{T}_{\mathcal{A} \otimes \mathcal{Q}_2 \otimes \mathcal{Q}_1}(h^{-1}(v_0, q_0^1, q_0^2))$. Once again, the only

interesting case is when $\alpha \equiv \langle \mathbf{a} \rangle^+ \alpha$. $(v, q_1, q_2, \mathbf{u}) \models \langle \mathbf{a} \rangle^+ \alpha$ iff there exists $(v', q_1', q_2', \mathbf{ua})$ such that $(v, q_1, q_2) \xrightarrow{\mathbf{a}} (v', q_1', q_2')$ and $(v', q_1', q_2', \mathbf{ua}) \models \alpha$ iff $(v', q_2', q_1', \mathbf{ua}) \models \alpha$, that is, iff $\langle h^{-1}(v, q_1, q_2), \mathbf{u} \rangle \models \alpha$. The other direction is symmetrical.

□

When is the construction worthwhile?

We thus see from Proposition 8.11 that when the players are of a fixed number t of types, we can carry out our stability analyses using the FST specifications of only these types and infer correctly about the outcome of the entire game; instead of dealing with the strategy FSTs of all of the n players. This might be helpful in situations when $t << n$ which is usually the case. However, there is a price to pay. One can construct simple counterexamples to show that the type FSTs necessarily need to be deterministic for the above analysis to go through. Hence there is an exponential blowup in the size of the FSTs that are used in the analysis. So a natural question to ask is when is it worthwhile to carry out the above construction.

Let $\mathcal{A} = (V, E)$ be an arena with $|V| = m$. Let there be a total of n players divided into t types and let χ be the mapping of players to their types. Let the ith type be given by the nondeterministic FST $\mathcal{R}_i$, having state set R_i. Let $p = \max_i |R_i|$. Let $\mathcal{Q}_i$ be the determinisation of $\mathcal{R}_i$. $\mathcal{Q}_i$ has a state set of size at most 2^p.

The number of nodes in the graph $\mathcal{A} \otimes \mathcal{R}_{\chi(1)} \otimes \ldots \mathcal{R}_{\chi(n)}$ is at most $m \cdot \underbrace{p \cdot \ldots \cdot p}_{n} = m \cdot p^n$. On the other hand, the number of nodes in the graph $\mathcal{A} \otimes \mathcal{Q}_1 \otimes \ldots \otimes \mathcal{Q}_t$ is at most $m \cdot \underbrace{2^p \cdot \ldots \cdot 2^p}_{t} = m \cdot 2^{tp}$. Hence the construction is worthwhile only when $m \cdot 2^{tp} < m \cdot p^n$. That is when $n \log_2 p > tp$. That is when $n > 0.693 \cdot t \cdot \pi(p)$, where $\pi(p)$ is the number of primes less than or equal to p.

8.2 Other extensions

Though we have tried to be comprehensive and address what we think are the relevant issues in the context of strategy switching in structured strategies, what we have actually done, is only to scratch the surface of an extremely rich area. Strategies, inspite of being the most important objects in games, are complicated. We understand surprisingly little about them.

There are many directions one can take from here. Below we elucidate some of them.

We must admit that we have taken a short cut route in the presentation of our game arenas. We have presented them as finite directed graphs without deadends. However, probably the most intuitive and common-sensical way to present game arenas is rule-based. When we teach someone chess, we tell them that the bishop can move diagonally and the rook can move horizontally and vertically and so on. We tell them when a piece can capture another and when the game ends. Listing out all the *finite* set of rules of chess is enough to describe the entire game and also the strategies available to the players. As the players play the game, they construct the arena dynamically. We do not present them with the entire game tree! That would be stupid!

Developing a uniform framework for rule-based presentation and analysis of games is a challenging but interesting task. There is very little literature in the area. [KS10] is definitely a progress in the right direction but a lot more is still to be done.

The theory of 'Epistemic Logic' is a rich and vast theory to reason about knowledge and beliefs of players in a game. The most important modality in epistemic logic is $K_a\alpha$ which says that player a knows that α is true. The knowledge modality can be applied iteratively as in $K_aK_b\alpha$ which means that player a knows that player b knows that α is true. It might seem that in most of our discussion, epistemic modalities play a fundamental part and that is true to a lot of extent. However, as the central theme of our presentation has been large games where the players are uncertain about many things, including the identity and the knowledge of other players, the classical epistemic modalities with their usual semantics are not applicable per se. Developing new epistemic modalities, taking into account the players' uncertainties and bounded resources, and carrying out a similar analysis would not only be an immensely interesting but also a challenging task.

In Chapter 5 we have studied the consequences of charging players for each strategy switch they make. We have studied the quantitative model of discounted repeated strategic form games. We wish to extend our analysis to the case where the strategies of the players are logically specified, like that in Chapter 4. Typical strategies of players in that case would be as follows: "Do not switch if the cost of switching is greater than c", "Switch only if the cost can be broken even within the next k rounds", "Do not switch if the other player switches to an opposing strategy" and so on.

We wish to do an analysis similar to Section 8.1 for our games with neighbourhood structure of Chapter 7. More precisely, as the players are of

a fixed number of types, it is worthwhile to investigate if working with only the types of the players helps us conclude about the dynamics and outcome of the entire game. Moreover, the logic we have introduced for specifying the types of players is quite limited. What would be more interesting is a logic with the power of quantification over various cliques of the neighbourhood graph. Since the cliques of the graph keep changing, such a logic would be highly expressive and may lead to better logical foundation for this study.

In this study we have mainly focussed on unbounded duration games on finite graphs. Such games have a rich structure and hence the strategies that arise from them are structured as well. However, repeated single-shot games of unbounded duration are interesting in their own right. It would be a fruitful exercise to carry out our entire analysis for such games and possibly gain new intuitions and insights.

Also, the strategies and the logics of strategies we have introduced and studied in this work deal mostly with pure strategies. However, one can carry out a parallel development with mixed strategies. Such a development would be closer to the way players actually play in the real world. Mixed strategies naturally arise when players have expectations about the strategies of the other players and strategise based on those expectations. One would then talk about expected outcomes rather than sure outcomes and the treatment would be probabilistic. We think such an analysis would be highly interesting and illuminating.

Finally, perhaps the most important question, the one underlying all the discussions in the entire exposition is to develop the right logic for specifying strategies. We hope to have made a small contribution in this regard through this work. However there is a long way to go. This logic should take into account a player's expectations of the other players, it should be probabilistic (to model mixed strategies) and should be expressive enough to specify any strategy (whether finite or infinite memory). Although there are many logics to talk about strategies (see, for instance, [STR] for a list of references), the fact remains that we are not yet there. And rightly so, because strategies are complicated objects: players strategise based on: beliefs, expectations, experience, bias, likes, dislikes, challenges, constraints, risks, hopes... and maybe even love and enmity! A logic that takes into account so many facets of the human psyche will perhaps be not simple. Or will it?

Bibliography

[AD05] R. J. Aumann and J. H. Dreze. When all is said and done, how should you play and what should you expect? Discussion Paper Series dp387, Center for Rationality and Interactive Decision Theory, Hebrew University, Jerusalem, March 2005.

[AFBH08] Heiner Ackermann, Simon Fischer, Petra Berenbrink, and Martin Hoefer. Concurrent imitation dynamics in congestion games, 2008.

[Ago06] T. Agotnes. Action and knowledge in alternating time temporal logic. *Synthese*, 149(2):377–409, 2006.

[AHK02] R. Alur, T.A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal of ACM*, 49(5):672–713, 2002.

[AJ94] S. Abramsky and R. Jagadeesan. New foundations for the geometry of interactions. *Information and Computation*, 111(1):53–119, 1994.

[ARSvS10] D. Avis, G. Rosenberg, R. Savani, and B. von Stengel. Enumeration of Nash equilibria for two-player games. *Economic Theory*, 42:9–37, 2010.

[ARV06] Heiner Ackermann, Heiko Röglin, and Berthold Vöcking. On the impact of combinatorial structure on congestion games. In *In Proc. of the 47th Ann. IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 613–622, 2006.

[Ban92] Abhijit V. Banerjee. A simple model of herd behaviour. *The Quarterly Journal of Economics*, 107(3):797–817, 1992.

[BFH09] F. Brandt, F. Fischer, and M. Holzer. Symmetries and the complexity of pure nash equilibrium. *Journal of computer and system sciences*, 75(3):163–177, 2009.

[BL69] J. R. Büchi and L. H. Landweber. Solving sequential conditions by finite-state strategies. *Transactions of the American Mathematical Society*, 138:295–311, 1969.

[Blo99] Matthias Blonski. Anonymous games with binary actions. *Games and Economic Behaviour*, 28:171–180, 1999.

[Blo00] Matthias Blonski. Characterisation of pure strategy equilibria in finite anonymous games. *Journal of Mathematical Economics*, 34:225–233, 2000.

[Bon01] G. Bonano. Branching time logic, perfect information games and backward induction. *Games and Economic Behaviour*, 36(1):57–73, 2001.

[Bor07] S. Borgo. Coalitions in action logic. In *Proceedings of IJCAI'07*, pages 1822–1827, 2007.

[Cha90] Subir K. Chakrabarti. Characterizations of the equilibrium payoffs of inertia supergames. *Journal of Economic Theory*, 51(1):171–183, June 1990.

[Cha07] Krishnendu Chatterjee. *Stochastic ω-regular Games*. PhD thesis, University of California at Berkeley, 2007.

[CJM04] K. Chatterjee, M. Jurdzinski, and R. Majumdar. On Nash equilibria in stochastic games. In *Proceedings of the 13th Annual Conference of the European Association for Computer Science Logic*, volume 3210 of *LNCS*, pages 26–40. Springer-Verlag, 2004.

[dAH00] L. de Alfaro and T. A. Henzinger. Concurrent omega-regular games. In *LICS 2000: 15th International IEEE Symposium on Logic in Computer Science*, pages 141–154. IEEE Press, 2000.

[dAHK98] L. de Alfaro, T. A. Henzinger, and O. Kupferman. Concurrent reachability. In *FOCS 98*, pages 564–575. IEEE, 1998.

[dAM01] L. de Alfaro and R. Mazumdar. Quantitative solution of omega-regular games. In *STOC'01*, pages 675–683. ACM, 2001.

[DGP06] C. Daskalakis, P. W. Goldberg, and C. H. Papadimitriou. The complexity of computing a Nash equilibrium. In *STOC'06*. ACM, 2006.

[DH05] A. Dawar and P. Hunter. Complexity bounds for regular games. In *Proceedings of the 30th International Symposium on Mathematical Foundations of Computer Science, MFCS'05*, volume 3618 of *Lecture Notes in Computer Science*, pages 495–506. Springer-Verlag, 2005.

[DJW97] S. Dziembowski, M. Jurdzinski, and I. Walukiewicz. How much memory is needed to win infinite games? *LICS'97*, pages 99–110, 1997.

[DP04] Jayasri Dutta and Kisalaya Prasad. Imitation and long run outcomes. *The B.E. Journal of Theoretical Economics*, topics.4(1):7, 2004.

[DP07] C. Daskalakis and C. H. Papadimitriou. Computing equilibria in anonymous games. In *Proceedings of the 48th symposium on Foundations of Computer Science (FOCS)*, pages 83–93. IEEE Computer Society Press, 2007.

[Dut95] P. Dutta. A folk theorem for stochastic games. *Journal of Economic Theory*, 66:1–32, 1995.

[EF95] Glenn Ellison and Drew Fudenberg. Word-of-mouth communication and social learning. *The Quarterly Journal of Economics*, 110(1):93–125, 1995.

[EGG06] Edith Elkind, Leslie A. Goldberg, and Paul W. Goldberg. Nash equilibria in graphical games on trees revisited. In *Proceedings of the 7th ACM conference on electronic commerce (ACM EC)*. ACM, 2006.

[EGG07] E. Elkind, L. A. Goldberg, and P. W. Goldberg. Computing good Nash equilibria in graphical games. In *Proceedings of the 8th ACM conference on Electronic Commerce (ACM-EC)*, pages 162–171. ACM Press, 2007.

[EJ91] E. A. Emerson and C. S. Jutla. Tree automata, mu-calculus and determinacy. In *Proceedings of the 32nd Annual Symposium on Foundations of Computer Science, FoCS'91*, pages 368–377. IEEE Computer Society Press, 1991.

[EM79] A. Ehrenfeucht and J. Mycielski. Positional strategies for mean payoff games. *International Journal of Game Theory*, 8:109–113, 1979.

[GH82] Y. Gurevich and L. Harrington. Trees, automata and games. In *Proceedings of the 14th Annual Symposium on Theory of Computing*, pages 60–65. ACM Press, 1982.

[Gho08] S. Ghosh. Strategies made explicit in dynamic game logic. In Johan van Benthem and Eric Pacuit, editors, *Proceedings of the Workshop on Logic and Intelligent Interaction, ESSLLI*, pages 74–81, Hamburg, 2008.

[GU08] E. Grädel and M. 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.

[Har77] John C. Harsanyi. *Rational Behaviour and Bargaining Equilibrium in Games and Social Situations*. Cambridge University Press, London and New York, 1977.

[HO94] J. M. E. Hyland and C. H. L. Ong. On full abstraction for pcf: I, ii and iii. *Information and Computation*, 1994.

[Hor05] Ulrich Horst. Dynamic systems of social interactions. In *NSF/CEME Mathematical Economics Conference at Berkeley*, 2005.

[Hor08] F. Horn. Explicit Muller games are PTIME. In *Proceedings of the Foundation of Software Technology and Theoretical Computer Science, FSTTCS*, Dagstuhl Research Online, pages 235–243, 2008.

[HvdHMW03] Paul Harrenstein, Wiebe van der Hoek, John-Jules Meyer, and Cees Witteven. A modal characterisation of Nash equilibrium. *Fundamenta Informaticae*, 57(2-4):281–321, 2003.

[JPZ06] Marcin Jurdzinski, Mike Paterson, and Uri Zwick. A deterministic subexponential algorithm for solving parity games. In *Proceedings of SODA*, pages 117–123. ACM/SIAM, 2006.

[JS96] Mark Jerrum and Alistair Sinclair. The markov chain monte carlo method: an approach to approximate counting and integration. pages 482–520. PWS Publishing, 1996.

[JvdH04] W. Jamroga and W. van der Hoek. Agents that know how to play. *Fundamenta Informaticae*, 63(2-3):185–219, 2004.

[KLS01a] Michael J. Kearns, Michael L. Littman, and Satinder P. Singh. An efficient, exact algorithm for solving tree-structured graphical games. In *NIPS*, pages 817–823, 2001.

[KLS01b] Michael J. Kearns, Michael L. Littman, and Satinder P. Singh. Graphical models for game theory. In *UAI*, pages 253–260, 2001.

[KS10] Łukasz Kaiser and Łukasz Stafiniak. Playing structure rewriting games. In *Proceedings of AGI '10*. Atlantis Press, 2010.

[LP07] David K. Levine and Wolfgang Pesendorfer. The evolution of cooperation through imitation. *Games and Economic Behaviour*, 58(2):293–315, 2007.

[LW97] Bart L. Lipman and Ruqu Wang. Switching costs in frequently repeated games. Working Papers 955, Queen's University, Department of Economics, September 1997.

[LW09] Barton L. Lipman and Ruqu Wang. Switching costs in infinitely repeated games. *Games and Economic Behavior*, 66(1):292–314, May 2009.

[Mar75] D. A. Martin. Borel determinacy. *Annals of Mathematics*, 102:363–371, 1975.

[Mar98] D. A. Martin. The determinacy of blackwell games. *The Journal of Symbolic Logic*, 63(4):1565–1581, 1998.

[Mos91] A. W. Mostowski. Games with forbidden positions. Technical report, Instytut Matematyki, Uniwersytet Gdanski, Poland, 1991.

[MS96] Dov Monderer and L. S. Shapley. Potential games. *Games and Economic Behaviour*, 14:124–143, 1996.

[Nas50] John F. Nash. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences*, 36(1):48–49, 1950.

[Par85] Rohit Parikh. The logic of games and its applications. *Annals of Discrete Mathematics*, 24:111–140, 1985.

[PR10] Soumya Paul and R. Ramanujam. Imitation in large games. In *Proceedings of the first international symposium on games, automata, logics and verification (GandALF)*, Electronic proceedings in theoretical computer science, pages 162–172, 2010.

[PR11] Soumya Paul and R. Ramanujam. Neighbourhood structure in large games. In *Proceedings of the 13th conference on the Theoretical Aspects of Rationality and Knowledge (TARK 2011)*, 2011. To appear.

[PRS09a] Soumya Paul, R. Ramanujam, and Sunil Simon. Dynamic restriction of choices: A preliminary logical report. In *Proceedings of the 12th Conference on Theoretical Aspects of Rationality and Knowledge (TARK)*, pages 218–226, 2009.

[PRS09b] Soumya Paul, R. Ramanujam, and Sunil Simon. Stability under strategy switching. In Klaus Ambos-Spies, Benedict Löwe, and Wofgang Merkle, editors, *Proceedings of the 5th Conference on Computability in Europe (CiE)*, volume 5635 of *LNCS*, pages 389–398, 2009.

[PS09] Soumya Paul and Sunil Simon. Nash equilibrium in generalised Muller games. In *Proceedings of the Conference on Foundation of Software Technology and Theoretical Computer Science, FSTTCS*, volume 4 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 335–346. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2009.

[PY93] H Peyton Young. The evolution of conventions. In *Econometrica*, volume 61, pages 57–84. Blackwell Publishing, 1993.

[PY00] H Peyton Young. The diffusion of innovations in social networks. Economics Working Paper Archive 437, The Johns Hopkins University,Department of Economics, May 2000.

[RS06] R. Ramanujam and Sunil Simon. Axioms for composite strategies. In *Proceedings of Logic and the Foundations of Game and Decision Theory (LOFT06)*, pages 189–198, Liverpool, 2006.

[RS08a] R. Ramanujam and Sunil Simon. Dynamic logic on games with structured strategies. In *Proceedings of the 11th Conference on Principles of Knowledge Representation and Reasoning*, pages 49–58, 2008.

[RS08b] R. Ramanujam and Sunil Simon. A logical structure for strategies. In *Logic and the Foundations of Game and Decision Theory (LOFT 7)*, volume 3 of *Texts in Logic and Games*, pages 183–208. Amsterdam University Press, 2008.

[Sch98] Karl S. Schlag. Why imitate, and if so, how? a boundedly rational approach to multi-armed bandits. *Journal of Economic Theory*, pages 130–156, January 1998.

[Sch08] Sven Schewe. An optimal strategy improvement algorithm for solving parity and payoff games. In *Proceedings of the 17th Annual Conference of the European Association for Computer Science Logic (CSL 2008), 15–19 September, Bertinoro, Italy*, volume 5213 of *Lecture Notes in Computer Science*, pages 368–383. Springer-Verlag, 2008.

[SP00] Brian Skyrms and Robin Pemantle. A dynamic model of social network formation. *Proceedings of the National Academy of Sciences*, 97(16):9340–9346, 2000.

[STR] STRATMAS. http://www.ai.rug.nl/ sujata/documents.html.

[Tho90] Wolfgang Thomas. Automata on infinite objects. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 133–192. Elsevier Science Publishers, Amsterdam, 1990.

[Tho97] Wolfgang Thomas. Languages, automata and logic. In G. Rosenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 3, pages 389–455. Springer, 1997.

[Umm06] M. Ummels. Rational behaviour and strategy construction in infinite multiplayer games. In *Proceedings of the Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 4337 of *LNCS*, pages 212–223. Springer, 2006.

[vB01] Johan van Benthem. Games in dynamic epistemic logic. *Bulletin of Economic Research*, 53(4):219–248, 2001.

[vB02] Johan van Benthem. Extensive games as process models. *Journal of Logic Language and Information*, 11:289–313, 2002.

[vB07] Johan van Benthem. In praise of strategies. In J. van Eijck and R. Verbrugge, editors, *Foundations of Social Software*, Studies in Logic, pages 283–317. College Publications, 2007.

[vdHJW05] W. van der Hoek, W. Jamroga, and M. Wooldridge. A logic for strategic reasoning. In *Proceedings of the Fourth International*

Joint Conference on Autonomous Agents and Multi-Agent Systems, pages 157–164, 2005.

[vdHW02] Wiebe van der Hoek and Michael Wooldridge. Tractable multiagent planning for epistemic goals. In *Proceedings of AAMAS*, pages 1167–1174, 2002.

[vNM44] John von Neumann and Oskar Morgenstern. *Theory of Games and Economic Behaviour*. Princeton University Press, Princeton NJ, 1944.

[vS10] B. von Stengel. Computation of Nash equilibria in finite games: introduction to the symposium. *Economic Theory*, 42:1–7, 2010.

[WvdHW07] D. Walther, W. van der Hoek, and M. Wooldridge. Alternating time temporal logic with explicit strategies. In *Proceedings of the Theoretical Aspects of Rationality and Knowledge*, pages 269–278, 2007.

[Zie98] W. Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theoretical Computer Science*, 200(1-2):135–183, 1998.

[ZP96] U. Zwick and M. Paterson. The complexity of mean payoff games on graphs. *Theoretical Computer Science*, 158:343–359, 1996.