# Stochastic games and their complexities

Marcin Przybylko

University of New Caledonia
Institut des Sciences Exactes
et Appliquées

University of Warsaw
Faculty of Mathematics, Informatics
and Mechanics

Marcin Przybyłko

# Stochastic games and their complexities

*PhD dissertation*

Supervisors

prof. dr hab. Damian Niwiński

Institute of Informatics
University of Warsaw

prof. dr hab. Teodor Knapik

Institut des Sciences Exactes et Appliquées
Université de la Nouvelle Caledonie

Auxiliary Supervisor

dr Michał Skrzypczak

Institute of Informatics
University of Warsaw

Thèse soutenue le 03/04/2019

October 2018

Author's declaration:
aware of legal responsibility I hereby declare that I have written this dissertation myself and all the contents of the dissertation have been obtained by legal means.

October 15, 2018 .....

*date* *Marcin Przybylko*

Supervisors' declaration:
the dissertation is ready to be reviewed

October 15, 2018 .....

*date* *prof. dr hab. Damian Niwiński*

.....

*prof. dr hab. Teodor Knapik*

# Contents

# Abstract

We study a class of games introduced by Mio to capture the *probabilistic $\mu$-calculi* called *branching games*. They are a subclass of stochastic two-player zero-sum turn-based infinite-time games of imperfect information. Branching games extend *Gale-Stewart* games by allowing players to split the execution of a play into new concurrent sub-games that continue their execution independently. In consequence, the play of a branching game has a tree-like structure, as opposed to linearly structured plays of Gale-Stewart games.

In this thesis, we focus our attention on *regular branching games*. Those are the branching games whose pay-off functions are the indicator functions of *regular sets* of infinite trees, i.e. the sets recognisable by *finite tree automata*. We study the problems of *determinacy*, *game value computability*, and the related problem of computing a *measure* of a regular set of infinite trees.

Determinacy is a property of a game that guarantees that none of the players gains or loses an advantage by revealing their strategy at the start of the game. In general, branching games are not determined: not even under *mixed strategies* nor when the winning sets are *topologically simple*. On the positive side, we show that regular branching games with *open* winning sets are determined under *mixed strategies*. Moreover, we show that *game automata* definable winning sets guarantee a stronger version of determinacy – the determinacy under *pure strategies*. Both results are accompanied by examples showing the limits of used techniques.

We give an answer to the problem of computing a value of a regular branching game. We show that a *mixed value* of a non-stochastic branching game is uncomputable and that a *pure value* of a stochastic branching game is also uncomputable. On the other hand, we provide an algorithm that computes all pure values of a given non-stochastic regular branching game.

We partially solve the problem of computing measures of regular sets of trees. We provide an algorithm that computes the *uniform* measure of a regular winning set in two cases. Either when it is defined by a *first-order* formula with no descendant relation or when it is defined by a Boolean combination of *conjunctive queries*.

Finally, we use real-life data to show how to incorporate game-theoretic techniques in practice. We propose a general procedure that given a *time series* of data extracts a *reactive model* that can be used to predict the evolution of the system and advise on the strategies to achieve predefined goals. We use the procedure to create a game based on *Markov decision processes* that is used to predict and control level of pest in a tropical fruit farm.

# Streszczenie

W pracy badamy klasę gier zwanych *grami rozgałęziającymi*. Gry rozgałęziające zostały wprowadzone przez Mio w celu uchwycenia semantyki *probabilistycznego rachunku* $\mu$. Stanowią one podklasę stochastycznych dwuosobowych gier turowych o sumie zerowej i nieskończonym czasie rozgrywki. Gry rozgałęziające rozszerzają gry *Gale'a-Stewarta* poprzez to, że pozwalają podzielić rozgrywkę, tworząc nowe podgry, które są rozgrywane równolegle i niezależnie. Z tego powodu *rozgrywka* gry rozgałęziającej ma strukturę drzewiastą, w przeciwieństwie do liniowej struktury rozgrywek w grach Gale'a-Stewarta.

W niniejszej rozprawie doktorskiej skupiamy się na regularnych grach rozgałęziających. Są to te gry rozgałęziające, w których funkcja wypłaty jest funkcją charakterystyczną regularnego zbioru nieskończonych drzew, to jest zbioru nieskończonych drzew, który jest rozpoznawany przez skończony automat na drzewach. W pracy skupiamy się głównie na problemie determinacji, na obliczalności wartości gier oraz na powiązanym z grami problemie obliczania miary regularnych zbiorów drzew.

Determinacja jest to własność gry, która gwarantuje, że żaden z graczy nie zyska ani nie straci przewagi poprzez ujawnienie swojej strategii na początku gry. Pokazujemy, że gry rozgałęziające nie muszą być zdeterminowane: ani gdy zbiory wygrywające są topologicznie proste, ani w przypadku, gdy dopuścimy strategie mieszane. Z drugiej strony, pokazujemy, że regularne gry rozgałęziające z *otwartymi* zbiorami wygrywającymi są zdeterminowane w strategiach mieszanych. Co więcej, pokazujemy, że te gry są zdeterminowane w strategiach czystych, jeśli zbiory wygrywające są rozpoznawalne przez *automaty growe*. Dla obu rezultatów konstruujemy przykłady pokazujące granice użytych technik.

Rozwiązujemy także problem obliczalności wartości gier pokazując, że mieszane wartości niestochastycznych gier rozgałęziających nie są obliczalne oraz, że czyste wartości gier stochastycznych także nie są obliczalne. Z drugiej strony, opisujemy algorytm, który wylicza czyste wartości niestochastycznych gier rozgałęziających.

Częściowo rozwiązujemy problem obliczania miar regularnych zbiorów drzew. Przedstawiamy algorytm, który oblicza miarę zbioru drzew, względem jednorodnej miary na drzewach, w dwu przypadkach: kiedy zbiór jest zdefiniowany poprzez formułę pierwszego rzędu nie używającą relacji potomka oraz gdy zbiór jest zdefiniowany poprzez boolowską kombinację zapytań koniunkcyjnych.

Wreszcie, pokazujemy w jaki sposób możemy zastosować bogate techniki teorii gier w praktyce. Proponujemy procedurę, która na podstawie szeregu czasowego tworzy reaktywny model pozwalający przewidzieć ewolucję modelowanego systemu i tworzyć strategie pozwalające zrealizować uprzednio zdefiniowane cele, jeśli takie strategie istnieją. Używamy wyżej wymienionej procedury, by stworzyć grę bazującą na procesach decyzyjnych Markowa, która pozwala przewidzieć i kontrolować poziomy obecności szkodników w tropikalnym sadzie owocowym.

# Résumé

Nous étudions les jeux ramifiés introduits par Mio pour définir la sémantique du $\mu$-calcul modal stochastique. Ces jeux stochastiques infinis à information imparfaite joués tour à tour par deux joueurs forment une sous-classe des jeux infinis à somme nulle. Elles étendent les jeux de Gale-Stewart en ce que chaque partie peut se scinder en sous-parties qui se déroulent indépendamment et simultanément. En conséquence, chaque partie a une structure arborescente, contrairement à la structure linéaire des parties des jeux de Gale-Stewart.

Dans cette thèse, nous étudions les jeux ramifiés réguliers. Ceux-ci ont pour caractéristique d'avoir leurs ensembles gagnants régulières, c'est à dire, des ensembles d'arbres infinis reconnus par automates finis d'arbres. Nous nous intéressons aux problèmes de détermination, de calcul des valeurs de jeux ramifiés réguliers et de calcul effectif de la mesure d'un ensemble régulier d'arbres.

La détermination est une propriété qui garantit qu'aucun des joueurs n'acquiert ou ne perd un avantage en révélant sa stratégie au début d'une partie. En général, les jeux ramifiés réguliers ne sont pas déterminés, pas même dans les stratégies mixtes, ni lorsque les ensembles gagnants sont topologiquement simples. Du côté positif, nous montrons que les jeux ramifiés réguliers ayant pour ensembles gagnants des ouverts sont déterminés en stratégies mixtes. De plus, dans le cas où les ensembles gagnants sont reconnus par ce qu'on appelle des automates de jeu, nous montrons la détermination en stratégies pures. Ces deux résultats sont accompagnés d'exemples qui montrent les limites des techniques utilisées.

Nous donnons une réponse au problème de calcul des valeurs des jeux ramifiés réguliers. Nous montrons que les valeurs mixtes des jeux ramifiés non stochastiques ne sont pas calculables et que les valeurs pures des jeux ramifiés stochastiques ne sont pas calculables. En revanche, nous donnons un algorithme qui calcule les valeurs pures des jeux ramifiés non stochastiques.

Nous donnons une réponse partielle au problème de calcul des mesures d'un ensemble régulier d'arbres. En particulier, nous donnons un algorithme qui calcule la mesure uniforme dans les deux cas suivants : lorsque l'ensemble est défini par une formule de la logique du premier ordre qui n'utilise pas la relation de descendant, ou lorsque l'ensemble est défini par une combinaison booléenne de requêtes conjonctives.

Finalement, nous utilisons des données réelles pour présenter comment on peut employer des techniques de la théorie des jeux stochastiques en pratique. Nous proposons une procédure générale qui à partir d'une série temporelle crée un modèle réactif capable de prédire l'évolution du système. Ce modèle facilite aussi les choix des stratégies permettant d'atteindre certains objectifs prédéfinis. La procédure nous sert ensuite à créer un jeux basé sur les processus décisionnels de Markov. Le jeu obtenu peut être utilisé pour prédire et contrôler le niveau d'infestation d'un verger expérimental.

# Acknowledgements

First of all, I would like to thank my supervisors Damian Niwiński and Teodor Knapik. I owe you much for introducing me to the topics contained in this thesis and for allowing me to choose my research subject freely. I thank Damian especially for his patience, knowledge, and almost pedantic approach to the written word. Each time I had to rewrite my definitions or proofs made my writing simpler, more precise, and easier to understand. I thank Teodor especially for the abundance of new ideas and research problems that he would give me every time we spoke. Every new problem resulted in new intuitions and ideas, which presence can be seen throughout this document.

I thank my auxiliary supervisor Michał Skrzypczak, especially for the final months of the process when he brought a new wave of enthusiasm and motivation that pushed me through the finish lane. Thank you for many fruitful discussion, annoying corrections, and words of encouragement.

I am also very grateful to Filip Murlak, my master's thesis supervisor, who introduced me to the scientific world and taught me how to think like a researcher. You helped me build the foundation on which this thesis firmly rests.

Many thanks to my family and friends, who many times offered me a welcome respite from the hardships and problems related to the laborious process of writing-up the manuscript.

I cannot not mention Anne Rouault, who made my transition and communication between the universities as smooth as possible.

I also want to express my gratitude to the reviewers of the thesis, whose suggestions helped me to improve several parts of the text.

Finally, special thanks to my fellow PhD students and various collogues, for many inspiring talks and moments which made this journey unforgettable.

# Chapter 1

# Introduction

From its very beginning *game theory* has been used to discover, understand, model, and predict the behaviour of naturally occurring systems. Game theory is especially useful when the systems in question are defined by an interaction of a number of agents that, not necessarily in cooperation, try to achieve their individual goals, e.g. a group of processes in an operating system competing for resources, a group of investment bankers trading shares, or a pack of predators hunting prey. Systems like those can be found in almost every branch of modern computer science, economy, or natural sciences. In computer science, games are used in semantics, verification, logic, and automata theory, to name a few, where they are used to define and formalise the notions of interaction. In economics, game theory is often associated with the *rational choice* in which we assume that the agents behave rationally. Lastly, in natural sciences games are often used to model complex events and ecosystems, where a number of competing parties try to achieve the best possible outcome, e.g. predator-prey equilibria.

**Games** The central notion in game theory are games. Those considered in this thesis are an extension of so-called *games on graphs*. Games on graphs are played on possibly infinite graphs with vertices distributed between the players. The players move a token, initially placed in one of the vertices, along the edges of the graph and in accordance to the ownership of the vertices. If a vertex is owned by a single player, then this player decides where to move the token. If a vertex is shared, then the players simultaneously and independently choose an action each; the chosen tuple of actions indicates the next placement of the token. The outcome of the game, called a *play*, is the trace of the token. After the game is played, every player achieves a score defined by a specific to the player *pay-off function*.

Games on graphs are often enhanced with probability. Such games, called *stochastic games*, introduce the uncertainty with a new type of vertices, called *random* vertices, in which the next position of the token is not decided by the players, but by the value of an associated random variable. The addition of random vertices is often encoded as an additional, ficti-

tious, player that chooses its moves at random. In the case of stochastic games, the score is the expected value of the pay-off function over the set of possible outcomes. Games with no random vertices are called *non-stochastic* or *pure* games.

The abundance of possible applications and areas of relevance of game theory gave birth to many classes of games which are often defined by some of their properties and require different tools to be analysed efficiently. The properties defining those classes include, but are not limited to, the duration of the game, the progress of time, the number of the players, the shape of the arena, the presence of uncertainties, players' knowledge, and the form of the objectives. Considering duration of the game, we can distinguish *one-shot games*, e.g. matrix games, matching pennies, rock-paper-scissors; *finite time games*, e.g. chess, tic-tac-toe; and (potentially) *infinite time games*, like *reachability games*, *system-user* interaction, or *Gale-Stewart* games [19]. Note that from the technical point of view, one-shot games can be seen as (in)finite time games, and (in)finite time games can be seen as one-shot games. Indeed, we can either add some inconsequential moves or demand that players declare all their future decisions at the start of the game.

The progress of time leads to distinction between turn based games, which are played in rounds, and continuous time games, see e.g. [2]. In the discrete time setting, we have concurrent games, where some vertices can be shared, e.g. Blackwell games [31], and turn-based games, where every vertex has at most one owner, e.g. Gale-Stewart games [19], or *parity games* [38, 16]. The *objectives* of games are usually given by families of pay-off functions, one for each player. An important class of games are *zero-sum* games, where the pay-off functions are chosen so that the sum of individual scores is zero. A game has a *winning set* if the possible scores are binary: *win* or *lose*. We say that a game is *regular* if it has a regular winning set, i.e. the inverse image of win is a *regular set*. By regular set we understand a set defined by a monadic second-order formula, see e.g. [54] for details.

**Determinacy** One of the most important notions in game theory is *determinacy*. Intuitively, a game is *determined* if no player gains an advantage knowing the *strategies* of the other players.

The exact definition of determinacy depends on the type of the game and the class of allowed strategies, e.g. in concurrent games or in matrix games with real valued pay-off functions the determinacy is defined in terms of equilibria, while in zero-sum turn-based games with winning sets, like Gale-Stewart games, in terms of *winning strategies*.

The celebrated result of Martin [30] states that Gale-Stewart games with Borel winning sets are determined under *pure* strategies. On the other hand, since the seminal work of Gale and Stewart [19], we know that not every game is determined under pure strategies. Therefore, broader classes of strategies are considered. Nash theorem [41] states that in one-shot games with finitely many strategies, there exists at least one point of equilibrium. This is an extension of the works of von Neumann and Morgenstern [57], who have shown the

determinacy of two-payer zero-sum games, and was later improved by Glicksberg [21] who showed that in two-player zero-sum games the equilibrium exists if the space of possibly infinite number of strategies is compact and the pay-off function is continuous. The equilibria are expressed in terms of *mixed strategies*, i.e. probability distributions on the set of pure strategies. A similar result by Martin [31] states that Blackwell games, a class of infinite duration discrete time concurrent games with a finite number of possible actions per turn, are determined under mixed strategies. Note that both results by Martin hold in the stochastic set-up, see [31] for details.

**Branching games** In this thesis we study a special extension of stochastic two-player zero-sum turn-based games on graphs called *branching games* [33]. The novel addition of branching games [33] is yet another kind of vertices, as opposed to *players' vertices* and random vertices, called *branching vertices*. A token placed in one of those vertices is split into a number of indistinguishable new copies of the token. The copies are placed in the successor vertices of the current vertex, one in each, and moved with no information on whereabouts of the other copies. This new type of vertices can be seen as a delegation process, where the players delegate the resolution of the rest of the game to independent parties that cannot communicate. Note that branching games are games of imperfect information: we assume that when players decide where to move a copy of the token, they are unaware of the positions of other copies.

Traditionally, the two players playing the game are called *Adam* and *Eve*, and the fictitious player encoding randomness is called *Nature*.

**Complexities of games** The main theoretical focus of this thesis is placed on the *computational complexity* of computing the values of the *regular branching games*. This can be seen as a natural extension of the work of Mio, who introduced branching games [35] and studied some of their properties [33].

We are interested in this family of games for two reasons:

- regular sets are a robust class with strongly developed theory and many good properties, e.g. closure properties, effective representation, and many decision procedures;
- regular sets are complex enough to not trivialise the problems and showcase interesting properties of branching games, e.g. lack of perfect information or perfect recall, for the definition of perfect recall see e.g. [27].

Considering the scope and theme of the theoretical part of this thesis, we continue the work of Mio by considering branching games with regular winning sets and studying their computational complexity. In greater scope, this research inscribes itself into a rich literature describing the complexity of $\omega$*-regular* games, for a survey see e.g. [10].

**Applications** As we mentioned at the start, using games to model complex ecosystems has always been an important motivation in the development of game theory. We con-

tribute to this part of the research by creating a framework that allows an easy incorporation of game-theoretic methods. We propose a general procedure that given a time series of data, extracts a reactive model that can be used to predict the evolution of the system and advise on the strategies to achieve predefined goals.

This is a case study, in which we were presented a data set to work with. Due to the nature of the data, we have decided to use *Markov decision processes* as our models of choice and *Baum-Welch* procedure to teach our models. Nevertheless, the described procedure is general and, if the data would allow, both the model and the teaching procedure can be replaced effortlessly.

**Organisation of the thesis and main results** This thesis consists of three main parts. The first part, Chapters 2 to 4, introduces basic notions. The second part, Chapters 5 to 8, studies *branching games* with regular winning objectives. The last part, Chapter 9, shows how game theory in conjunction with machine learning can be used in real-life applications in modern agriculture.

In Chapter 2 we introduce the basic notions used throughout this document and recall a number of definitions of classical games. Chapter 3 consists of the central definitions of this thesis. In this chapter, we define the *branching games, strategies, game values*, and the *formal notion of determinacy*. Moreover, we discuss how various classes of games presented (informally) in Chapter 2 can be uniformly represented as branching games.

In the short Chapter 4, we define the problems of interest in this thesis and recall some of the important and previously known results concerning the *computational complexity* of those problems.

In Chapter 5 we study the family of pure branching games, i.e. branching games with no random vertices. Moreover, when considering pure branching games we allow *pure strategies* only.

We start by recalling the notion of a *winning strategy*. This allows us to associate the *pure values* of the game and the determinacy with the existence of a winning strategy. Then we recall that branching games are not necessarily determined under pure strategies and discuss the complexity of computing the values of a given game. We start the discussion with the case of single-player games, which are necessarily determined under pure strategies. Then we inspect the case of finite two-player games and show that the sets of winning strategies are regular sets of trees. From this, we conclude that there is an algorithm that decides whether a given game is determined under pure strategies. Moreover, we provide an alternative proof of the existence of an undetermined branching game. The proof is based purely on computational complexity.

Finally, we present one of the interesting properties of the branching games: the *dealternation* of the winning objective. We show that if the winning objective is given as an alternating automaton, then we can modify in polynomial time both the board and the representation

of the winning objective so that the pure values are unchanged and the winning set in the new game is given by a non-deterministic automaton instead of an alternating one. Because of complexity reasons, such an operation is impossible without the branching elements of the arena.

The main results of this chapter are as follows.

- In the case of finite single-player games, we reduce the problem of computing the value of a game with no Adam's vertices to the non-emptiness of a regular set of trees; similarly, the problem of computing the value of a game with no Eve's vertices to the universality of a regular set of trees. Moreover, we provide matching upper bounds; in particular, we show that if the winning set is given by a non-deterministic automaton then computing the pure values is in $NP \cap coNP$ with no Adam's vertices and is *EXP*-complete with no Eve's vertices.
- In the case of finite two-player games, we show that the set of wining strategies of either player is regular and, thus, the values are computable. Moreover, we provide matching upper bounds, for instance if the winning set is given by a non-deterministic automaton then computing Eve's pure value is *2-EXP*-complete and computing Adam's pure value is *EXP*-complete.
- There is an algorithm that decides whether a game is determined; the algorithm works in doubly exponential time.

Results in this chapter are based on [45, 47].

In Chapter 6 we lift the restrictions on the types of vertices and the types of strategies. Here we study branching games with stochastic elements, i.e. we allow random vertices, and both behavioural and mixed strategies. We start by showing that branching games with regular objectives are not necessarily determined even under mixed strategies. On the other hand, we show that if the winning objective is topologically, relatively, simple, i.e. is an open (or a closed) set, then the game is determined under mixed strategies. Later, we discuss the computational complexity of deciding the value of a branching game with a regular winning set. We show that even in the single-player case there is no algorithm that can compute any of the values of an arbitrary branching game. In particular, we show that deciding whether a value of a branching game with effectively encoded regular winning set is strictly greater than a certain threshold is undecidable.

Finally, we present another interesting property of the branching games: the *derandomisation* property. We show that we can modify in polynomial space both the board and the winning set so that the mixed values remain unchanged and the new regular game has no random vertices.

The main results of this chapter are:

- branching games with open winning sets are determined under mixed strategies;

- there exists a branching game, with a winning set being a difference of two open sets, that is not determined under mixed strategies;
- there is no algorithm that computes a value of an arbitrary branching game;
- there is no algorithm that computes a mixed value of a non-stochastic branching game.

Results in this chapter are based on [47].

In the short Chapter 7 we study branching games with winning sets given by so-called game automata. Game automata are a syntactic restriction of the alternating automata on trees. We show that those games reduce in polynomial time to meta-parity games introduced by Mio [35]. Since stochastic meta-parity games are determined [33] and their value is computable (an unpublished result by Mio), branching games with winning conditions given by game automata are determined under pure strategies and their value is computable.

The main results of this chapter are:

- branching games with winning sets given by so-called game automata are determined under pure strategies;
- there is an algorithm that computes the value of a given branching game with a winning set given by a game automaton;
- there is an algorithm that computes the value of a given non-stochastic branching game with a winning set given by a game automaton, the algorithm belongs to the class $UP \cap co\text{-}UP$.

In Chapter 8 we attack the problem of computing the uniform measure of a regular set of trees. This problem can be seen as a special case of computing a value of a given *half-player* game, i.e. a game with only branching and random vertices. It turns out that, in some restricted classes of first-order definable sets of trees, we can use *Gaifman locality* to show that the measure of a set of trees is rational and computable. We leave the general problem unsolved, but we give an example, inspired by Potthoff's example [44], of a first-order definable set of trees with an irrational, but algebraic, measure. Moreover, we conjecture that the measures of regular sets of trees are algebraic.

The main results of this chapter are the following.

- If a set of trees is defined by a first-order formula using unary predicates and the successor relations only, then the measure is rational and computable in triple exponential space.
- If a set of trees is defined by a Boolean combination of conjunctive queries using unary predicates, the successor relations, and the ancestor relation, then the measure is rational and computable in exponential space. Moreover, deciding whether the measure is positive is *NEXP*-complete.

- There is a first-order definable set of trees whose uniform measure is an irrational number.

Results in this chapter are based on [46].

In Chapter 9 we show how game theory, and stochastic games in particular, can be used to support modern agriculture. We propose a *plantation game* framework, where we show how using *time series* of data describing a plantation one can create a tool that can *model and predict* the behaviour of the plantation and *advise* the owner. This is a case study where we take a time series describing a real fruit plantation and, using machine learning methods, create a model and, later, a game that can represent the interactions between the different elements of the fruit farm. Then, we show how the game can be used to predict the evolution of the system and how to use the game to create an artificial advisor, connecting the theory with real life applications.

# Chapter 2

# Vital definitions

*Automata theory*, *game theory*, and *logic* are rich and spacious disciplines of modern science. While we are aware of the vast literature that tackle each of them separately, we prefer to see them as necessarily entwined areas of research, with a great number of similarities in definitions and techniques. This approach is motivated by the nature of the problems considered in this thesis, which in natural way mix those areas of research.

Hence, we will introduce the needed definitions in a uniform manner. Let us begin with the basic building blocks used in this thesis.

## 2.1 Words and trees

**Basic sets** By $\mathbb{R}$ we denote the set of real numbers with the standard topology, standard order ($\leq$), and its standard Lebesgue measure. By $\mathbb{R}_{\geq 0}$ we denote the set of non-negative real numbers. By $\mathbb{R}_\infty$ we denote the set of real numbers augmented by the infinity, i.e. the unique element $\infty$ such that for every real number $x$ we have that $x < \infty$. By $\mathbb{N} = \{0, 1, 2, \ldots\}$ we denote the set of natural numbers with the standard discrete topology and standard order on its elements. By $\omega$ we denote the first infinite ordinal number. Technically, $\mathbb{N}$ and $\omega$ denote the same sets of numbers, but we use $\omega$ to emphasise the order on the natural numbers.

**Functions** By $f\colon X \to Y$ we denote a (total) function from $X$ to $Y$, by $f\colon X \rightharpoonup Y$ we denote a partial function from $X$ to $Y$. If $f\colon X \rightharpoonup Y$ is a (partial) function, then by $\mathit{dom}(f)$ we denote its domain and by $\mathit{range}(f)$ its range, i.e. the set $\{y \in Y \mid \exists x \in X.\ f(x) = y\}$. For a subset $S \subseteq Y$ by $f^{-1}(S)$ we denote the pre-image of $S$, i.e. the set $\{x \in X \mid \exists y \in S.\ f(x) = y\}$. If $S = \{y\}$ is a singleton, then we simply write $f^{-1}(y)$. Recall that, often it is convenient to see any partial function $f\colon X \rightharpoonup Y$ as a relation $f \subseteq X \times Y$, especially in the context of relational structures.

Let $V, S$ be a pair of sets such that $S \subseteq V$. By $i_V$ we denote the identity function, i.e. the function $i_V\colon V \to V$ such that $i_V(v) = v$ for every $v \in V$, and by $\chi_S$ the characteristic

function of the set $S$, i.e. the function $\chi_S \colon V \to \{0,1\}$ such that $\chi_S(v) = 1$ if $v \in S$ and $\chi_S(v) = 0$ if $v \notin S$.

**Letters** An *alphabet* $\Gamma$ is any finite non-empty set, the elements of an alphabet are called *letters*. Given an alphabet, a letter is called *fresh* if it does not belong to the alphabet. We assume that the family of all letters is infinite and for every alphabet we can find a fresh letter. We fix a special letter $\flat$ and call it *blank*. We write $\Gamma \cup \{\flat\}$ to emphasise that the alphabet in consideration contains blank (even if $\flat \in \Gamma$), and $\Gamma \setminus \{\flat\}$ to emphasise that it does not contain blank (even if $\flat \notin \Gamma$).

Let $f \colon S \to \Gamma \cup \{\flat\}$ be a function, by $\mathit{nodes}(f)$ we denote the set $\mathit{nodes}(f) \stackrel{\text{def}}{=} \{s \in S \mid f(s) \neq \flat\}$, i.e. the complement of the pre-image of the letter $\flat$. The intuitive meaning of the symbol $\flat$ is that the object "marked with" this symbol does not exist.

**Words** A *word* $w$ over an alphabet $\Gamma$ is any function $w \colon \mathbb{N} \to \Gamma \cup \{\flat\}$ with a $\leq$-closed set of nodes[1]. The symbol $\varepsilon$ stands for the unique empty sequence, i.e. the function $\varepsilon \colon \mathbb{N} \to \{\flat\}$, called the *empty word*. The *length* of a word, denoted $|w|$, is the cardinality of its set of nodes. If the word $w$ has finite length then we call it *finite*, if not then we call it *infinite*. The set of all finite words over an alphabet $\Gamma$ is denoted $\Gamma^*$, the set of all infinite words by $w \in \Gamma^\omega$. The set of all words over $\Gamma$ is denoted $\Gamma^{\leq\omega}$. Moreover, if $\bowtie$ is one of $\{<, \leq, =, \geq, >\}$, then $\Gamma^{\bowtie k} \stackrel{\text{def}}{=} \{w \in \Gamma^{\leq\omega} \mid |w| \bowtie k\}$, e.g. the set $\{a,b\}^{=5}$ is the set of all words of length 5 over the alphabet $\{a,b\}$.

Let $w_1 \in \Gamma^*$ be a finite word, and $w_2 \in \Gamma^{\leq\omega}$ be a word, then $w_1 \cdot w_2$ is the standard *concatenation* of $w_1$ and $w_2$, i.e. the word defined as follows.

$$(w_1 \cdot w_2)(x) \stackrel{\text{def}}{=} \begin{cases} w_1(x) & \text{if } x \in \mathit{nodes}(w), \\ w_2(x - |w_1|) & \text{otherwise.} \end{cases} \tag{2.1}$$

We often drop the symbol $\cdot$ and write $w_1 w_2$ instead of $w_1 \cdot w_2$.

The standard *prefix order* on words is denoted $\sqsubseteq$, i.e. we write $w_1 \sqsubseteq w_2$ if $w_1$ is finite and there is a word $w_3$ such that $w_2 = w_1 \cdot w_3$ or $w_1$ is infinite and $w_2 = w_1$.

**Trees** A *binary tree* over an alphabet $\Gamma$ is a function $t \colon \{\mathtt{L}, \mathtt{R}\}^* \to \Gamma \cup \{\flat\}$ with a prefix-closed set of nodes $\mathit{nodes}(t) \subseteq \{\mathtt{L}, \mathtt{R}\}^*$. The symbols $\mathtt{L}, \mathtt{R}$ are called *left* and *right*, respectively, and describe directions in the tree. The elements of the set $\{\mathtt{L}, \mathtt{R}\}^*$ are called *positions*.

The height of a tree $t$ is defined as $h(t) \stackrel{\text{def}}{=} \sup\{|w| \mid w \in \mathit{nodes}(t)\}$. A tree $t$ is

- *finite* if the set of nodes of $t$ is finite (i.e. $|\mathit{nodes}(\mathrm{t})| < \omega$),
- *infinite* if the set of nodes is infinite,
- of height $h$ if $h = h(t)$,

[1]Note, that the above implies that if $w(i) = \flat$ for some natural number $i$ then for every $j \geq i$ we have that $w(j) = \flat$.

- a *complete tree of height h* if $\mathit{nodes}(t) = \{\mathtt{L}, \mathtt{R}\}^{\leq h}$
- a *full binary tree* if $\mathit{nodes}(t) = \{\mathtt{L}, \mathtt{R}\}^*$.

Let $\Gamma$ be an alphabet, then the set $\mathcal{T}_\Gamma$ is the set of all binary trees over $\Gamma$, the set $\mathcal{T}_\Gamma^\infty$ is the set of all full binary trees over $\Gamma$, and the set $\mathcal{T}_\Gamma^k$ is the set of all complete binary trees of height $k$ over $\Gamma$. In general, one can consider trees of higher arity, e.g. quaternary trees $t\colon \{0,1,2,3\}^* \to \Gamma \cup \{\flat\}$ or even unranked trees $t\colon \mathbb{N}^* \to \Gamma \cup \{\flat\}$. In this work we consider binary trees only, thus, from now on, when we write a tree we mean a binary tree. Let $s_d(u) \stackrel{\text{def}}{=} u \cdot d$ for a position $u$ and a direction $d$. We say that a node $u$ is a *child* of $v$ in a tree $t$ if $u, v \in \mathit{nodes}(t)$ and $u = s_d(v)$ for some $d \in \{\mathtt{L}, \mathtt{R}\}$. Conversely, $v$ is called the *parent* of $u$. We write $\mathit{parent}(u)$ for the parent of the node $u$. We call the node $s_\mathtt{L}(u)$ *left child* of $u$ and the node $s_\mathtt{R}(u)$ *right child* of $u$.

A node $u$ of a tree $t$ is *branching* if it has at least one child in the tree, is *fully branching* if it has two children in the tree, and is *uniquely branching* if it has exactly one child in the tree.

**Distance** The *distance* between two positions is the function $d\colon \{\mathtt{L}, \mathtt{R}\}^* \times \{\mathtt{L}, \mathtt{R}\}^* \to \mathbb{N}$ defined as $d(u, v) = |u| + |v| - 2|x|$, where $x$ is the longest common prefix of $u$ and $v$. Equivalently, the distance between two different positions is the length of the shortest undirected path connecting the two nodes in the graph $\langle \{\mathtt{L}, \mathtt{R}\}^*, s_\mathtt{L} \cup s_\mathtt{R} \rangle$. In other words, if $u, v \in \{\mathtt{L}, \mathtt{R}\}^*$ are in distance $n$, then there is a sequence of nodes $u_0, \ldots, u_n$ such that $u_0 = u$, and $u_n = v$, and for every $0 \leq i < n$ we have that either $\mathit{parent}(u_i) = u_{i+1}$ or $\mathit{parent}(u_{i+1}) = u_i$. A sequence of nodes connected by a child relation is called a *walk*.

Let $t \in \mathcal{T}_\Gamma$ be tree, a *path* $p$ starting at node $u \in \mathit{dom}(t)$ is a possibly infinite sequence of positions $\{u_i\}_{0 \leq i < \gamma}$, for some $\gamma \leq \omega$, such that $u_0 = u$ and for every $0 \leq i < \gamma$ we have that $u_i$ is the parent of $u_{i+1}$. Let $S \subseteq \Gamma$ be a non-empty set of letters, we say that a path is an *$S$-labelled path* if the set of labels is contained in $S$, i.e. if $\{a \in \Gamma \mid \exists i.\ t(u_i) = a\} \subseteq S$. If $S$ is a singleton, i.e. $S = \{a\}$ for some $a \in \Gamma$ we may write *$a$-labelled path* instead.

**Prefixes and sub-trees** We say that a tree $t_1$ is a *prefix* of a tree $t_2$, denoted $t_1 \sqsubseteq t_2$, if $\mathit{nodes}(t_1) \subseteq \mathit{nodes}(t_2)$ and for every $u \in \mathit{nodes}(t_1)$ we have $t_1(u) = t_2(u)$. The set of all trees $t \in \mathcal{T}_\Gamma$ such that $t_1$ is a prefix of $t$ is denoted $\mathbb{B}_{t_1}$.

For a tree $t$ and a node $u \in \mathit{nodes}(t)$, by $t \vartriangleleft u$ we denote the unique tree such that for every position $v \in \{\mathtt{L}, \mathtt{R}\}^*$ the following holds.

$$t \vartriangleleft u(v) \stackrel{\text{def}}{=} t(uv) \tag{2.2}$$

The tree $t \vartriangleleft u$ is called the *sub-tree of $t$ at node $u$*.

If a node $u$ is not fully branching in a tree $t$ and for a direction $d$ we have that $t(ud) = \flat$, then we say that the sub-tree $t \vartriangleleft ud$ was *cut*.

**Words as trees** Note that words may be seen as trees. A word $w\colon \mathbb{N} \to \Gamma \cup \{\flat\}$ can be seen as a function $t\colon \{\mathtt{L},\mathtt{R}\}^* \to \Gamma \cup \{\flat\}$ such that $\mathit{nodes}(t) \subseteq \{\mathtt{L}\}^*$ and for $i \in \{0,1,\ldots\}$ we have that $w(i) = t(\mathtt{L}^i)$.

## 2.2 Topology and measure

Let $\Gamma$ be a finite set. Then, the set of all functions $f\colon \{\mathtt{L},\mathtt{R}\}^* \to \Gamma \cup \{\flat\}$ can naturally be enhanced with a topology in such a way that it becomes a homeomorphic copy of the Cantor set, see e.g. [55]. From now on, whenever we refer to a topology we mean this topology.

Note that the family of the sets of the form

$$\{f\colon \{\mathtt{L},\mathtt{R}\}^* \to \Gamma \mid t \sqsubseteq f\}, \tag{2.3}$$

where $t \in \mathcal{T}_\Gamma$ is a complete binary tree of some finite height, constitutes a *base of the topology*. A set from the basis is called a *base set*.

A set is *open* if it is a union, possibly empty, of some base sets; *closed* if it is a complement of an open set; *clopen* if it is both open and closed. Note, that our chosen basis consists of clopen sets.

**Fact 2.2.1.** *The set of all trees over the alphabet $\Gamma$ is a closed subset of all functions $f\colon \{\mathtt{L},\mathtt{R}\}^* \to \Gamma \cup \{\flat\}$.*

A set $S$ is *compact* if from every open cover of $S$ one can choose a finite cover, i.e. if for every family of open sets $\{S_i\}_{i\in I}$ such that $S \subseteq \bigcup_{i\in I} S_i$ there is a finite family of sets $\{S\}_{j\in J}$ such that $J \subseteq I$ and $S \subseteq \bigcup_{j\in J} S_j$.

**Fact 2.2.2.** *The set of all functions $f\colon \{\mathtt{L},\mathtt{R}\}^* \to \Gamma \cup \{\flat\}$ is compact.*

Recall the following basic lemma.

**Lemma 2.2.3** (Folklore)**.** *If $X$ is a compact topological space and a set $S \subseteq X$ is closed, then $S$ is compact.*

A family of subsets of a non-empty set $X$ is called a *$\sigma$-algebra* on $X$ if it is closed under complement, countable unions, countable intersections, and contains the empty set. A set is *Borel* if it belongs to the smallest $\sigma$-algebra containing all open sets.

The following fact is folklore and will be stated without a proof.

**Fact 2.2.4.** *The set of all trees over the alphabet $\Gamma$ is a closed subset of all functions $f\colon \{\mathtt{L},\mathtt{R}\}^* \to \Gamma \cup \{\flat\}$.*

**Continuous functions** Let $X$ and $Y$ be topological spaces. We say that a function $f\colon X \to Y$ is *continuous* if for every open set $S \subseteq Y$ the pre-image $f^{-1}(S)$ is an open set. We say that a real valued function $f\colon X \to \mathbb{R}$ is *lower semi-continuous* (resp., *upper*) if for every $y \in \mathbb{R}$ the set $\{x \in X \mid f(x) \leq y\}$ $\big($resp., $\{x \in X \mid f(x) \geq y\}\big)$ is closed.

**Measures** For a comprehensive introduction to topology and measure theory we refer to [25, Chapter 17]. Here we will introduce definitions and properties that are required for this work to be self-contained. Though some of the following properties will be unproven, the adequate references will be provided.

A *measurable space* is a pair $\langle X, \mathcal{X}\rangle$, where $X$ is a non-empty set and $\mathcal{X}$ is a $\sigma$-algebra on $X$. Let us fix a measurable space $\langle X, \mathcal{X}\rangle$. A set $S \subseteq X$ is called *measurable* if $S \in \mathcal{X}$.

A *measure* $\mu$ is a function $\mu\colon \mathcal{X} \to \mathbb{R}_\infty$ such that

- for every $S \in \mathcal{X}$ we have that $\mu(S) \geq 0$,
- $\mu(\varnothing) = 0$,
- and for every countable collection of disjoint measurable sets $\{S_i\}_{i\geq 0}$ we have that $\mu\big(\bigcup_{i\geq 0} S_i\big) = \sum_{i\geq 0} \mu(S_i)$.

For a given measure $\mu$, we say that a set $S \subseteq X$ is *$\mu$-measurable* if $S \in dom(\mu)$. We say that the measure $\mu$ is a *probability measure* if $\mu(X) = 1$, a *Borel measure* if its domain contains the $\sigma$-algebra of Borel sets, and *complete* if for every pair of sets $S, S' \subseteq X$ such that $S \subseteq S'$ and $S' \in \mathcal{X}$, we have the following: if $\mu(S') = 0$ then $S \in \mathcal{X}$ and $\mu(S) = 0$.

Let $\langle X, \mathcal{X}\rangle$ be a measurable space, by $dist(X)$ we denote the set of all complete Borel probability measures on $\langle X, \mathcal{X}\rangle$. Note that if $X$ is countable, then the power set of $X$ is a $\sigma$-algebra and every probability measure can be seen as a function induced by some function $f\colon X \to \mathbb{R}_{\geq 0}$ such that $\sum_{x\in X} f(x) = 1$. In that case, we often write write $f(x)$ instead of $f(\{x\})$.

**Integrals** Let $X, Y$ be measurable spaces and $\mu$ be a measure on $X$. A function $f\colon X \to Y$ is *measurable* if the pre-image of any measurable set in $Y$ is measurable in $X$. A function $f\colon X \to \mathbb{R}$ is *$\mu$-measurable* if the pre-image of any measurable set in $\mathbb{R}$ is measurable in $\langle X, dom(\mu)\rangle$. We say that $f\colon X \to Y$ is *universally measurable* if it is measurable for every complete Borel measure $\mu$ on $X$.

If $\mu$ is a measure on the space $X$ and $f\colon X \to \mathbb{R}$ is $\mu$-measurable, then by $\int_X f(x)\ \mathrm{d}\mu(x)$ we denote the *Lebesgue integral* of $f$ with respect to the measure $\mu$ over $X$. We say that a function is *(Lebesgue) integrable* if it has the Lebesgue integral.

**Fact 2.2.5.** *Let $X$ be a measurable space of finite measure, i.e. $\mu(X) < \infty$, and $f$ be bounded on $X$. Then, $f$ is integrable over $X$ if and only if it is measurable.*

If $f\colon X \to Y$ is measurable then $f\#\mu$ is the *pushforward* measure, i.e. $f\#\mu$ is the measure on $Y$ defined by the equation

$$f\#\mu(S) = \mu(f^{-1}(S)) \tag{2.4}$$

where $S$ ranges over the measurable sets in $Y$. Moreover, if $\Phi\colon Y \to [0,1]$ is an integrable function then the following equation holds.

$$\int_Y \Phi(y)\ \mathrm{d}f\#\mu(y) = \int_X \Phi\circ f(x)\ \mathrm{d}\mu(x) \tag{2.5}$$

By convention, if the variable over which we integrate, e.g. $x$, is clear from the context, we may drop it and write $\int_X f\ \mathrm{d}\mu$ instead of $\int_X f(x)\ \mathrm{d}\mu(x)$.

## 2.3 Game theory

Games often considered in the literature can be seen as objects played in a linear-time manner, i.e. games in which players declare their moves in turns. The sequence of their moves is recorded as a (possibly) infinite word and, then, checked against a rulebook – a set containing all the winning sequences. Those games are usually played by a finite number of players, on a (possibly rudimentary) board, with an established order of movements in every turn.

We will now recall a number of informal definitions of classic games. In the following chapter, we will show how to encode those games in our set-up, providing, in consequence, formal definitions.

### 2.3.1 Games on graphs

*Games on graphs* is a common name for a fairly large family of games that can be encoded as games that are played by moving a token alongside the edges of a possibly infinite graph.

A game on graph is a pair $G = \langle \mathtt{B}, W\rangle$, where $\mathtt{B} = \langle V, E, \rho, \lambda, v_{\mathrm{I}}\rangle$ is a graph, called a *board*, that the game is played on and $W$ is a winning set. The set $V$ is a set of *vertices*, $E \subseteq V \times V$ is a set of *edges*, $v_{\mathrm{I}}$ is an *initial vertex*, $\rho\colon V \to \{E, A\}$ is a division between players' positions (Eve's and Adam's positions), and $\lambda\colon V \to \Lambda$ is a labelling of vertices with a possibly infinite set of labels $\Lambda$. For technical reasons, here and in the future definitions of games, we assume that every vertex has at least one outgoing edge, i.e. for every $v \in V$ there is $v' \in V$ such that $\langle v, v'\rangle \in E$. The winning set $W \subseteq \Lambda^\omega$ is the set of the labellings of infinite paths which are winning for Eve.

Intuitively, the game starts with the token placed in the initial vertex $v_{\mathrm{I}}$ and is played in turns. Each turn, a player moves the token alongside the edges of the graph. The move is made by the player that owns the vertex on which the token lies. Eve owns all the vertices in the set $\rho^{-1}(\{E\})$ and Adam owns all the vertices in the set $\rho^{-1}(\{A\})$.

The sequence of the moves of each player is defined by their *strategies*. A strategy is a function mapping the path the token took in the unfolding of the arena to the desired displacement, i.e. a function $s\colon V^*V \to V$ such that for every $v \in V$ and $u \in V^*$ we have that $\langle v, s(uv)\rangle \in E$.

After an infinite number of moves, a labelled path, called *play*, is created. Eve wins the play if the sequence of labels on the path belongs to the set $W$. If not, Adam wins.

### 2.3.2 Reachability games

*Reachability games*, see e.g. [4], are one of the basic games considered in game theory. A reachability game is a tuple $G = \langle V, E, \rho, F, v_{\mathrm{I}}\rangle$, where $V$ is a set of *vertices*, $E \subseteq V \times V$ is a set of *edges*, $v_{\mathrm{I}}$ is an *initial vertex*, $\rho\colon V \to \{E, A\}$ is a division between Eve's and Adam's positions, and $F \subseteq V$ is a set of *final vertices*. The winning condition is defined as follows: Eve wins a play if at least one of the visited vertices belongs to $F$.

Reachability games are those games on graphs in which the board is the tuple $G = \langle V, E, \rho, i_V, v_{\mathrm{I}}\rangle$ and the winning set $W \subseteq V^\omega$ is the set of infinite sequences in which appears an element belonging to the set $F$. Recall that $i_V$ stands for the identity function.

### 2.3.3 Parity games

*Parity games*, see e.g. [38, 16], are one of the crucial concepts in the theory of regular languages of infinite trees. Parity games are a generalisation of reachability games, with the set of final vertices $F$ replaced by the *ranking function* $\alpha$ and the winning condition replaced by the so-called *parity condition*.

A parity game is a tuple $G = \langle V, E, \rho, \alpha, v_{\mathrm{I}}\rangle$, where $V$ is a set of vertices, $E \subseteq V \times V$ is a set of edges, $v_{\mathrm{I}}$ is an initial vertex, $\rho\colon V \to \{E, A\}$ is a division between Eve's and Adam's positions, and $\alpha\colon V \to \{i, \ldots, j\}$ is a ranking function assigning priorities from a finite set of natural numbers $\{i, \ldots, j\} \subseteq \mathbb{N}$ to vertices. Eve wins a play if the smallest priority visited infinitely often is even. This is the so-called *parity condition*.

Parity games are those games on graphs in which the board is the tuple $\langle V, E, \rho, \alpha, v_{\mathrm{I}}\rangle$ and the winning set $W \subseteq \mathit{range}(\alpha)^\omega$ is the set of infinite sequences satisfying the parity condition.

### 2.3.4 Gale-Stewart games

A *Gale-Stewart game*, see e.g. [19, 30], is a tuple $G = \langle S, W\rangle$, where $S$ is a non-empty set and $W \subseteq S^\omega$ is a winning set.

The game is played in turns numbered $0, 1, \ldots$ In $i$th turn first Eve chooses an element $e_i \in S$ and then Adam chooses an element $a_i \in S$. After an infinite number of moves,

an infinite sequence $e_0 a_0 \cdots e_n a_n \cdots$, called play, is created. Eve wins a play if it belongs to the winning set $W$.

Gale-Stewart games can be seen as games on graphs in the following way. The winning set is $W$ and the board is a tuple $\langle V, N, \rho, \lambda, v_{\mathrm{I}} \rangle$, where the graph structure consists of a set of vertices $V = S_A \cup S_E \cup \{v_{\mathrm{I}}\}$ containing the initial vertex $v_{\mathrm{I}}$ and two sets of vertices $S_A, S_E$, where $S_A = S \times \{A\}$, $S_E = S \times \{E\}$, between which the players alternate by choosing an edge from the set $N = (\{v_{\mathrm{I}}\} \times S_A) \cup (S_A \times S_E) \cup (S_E \times S_A)$, $N$ may be understood as "next move". The first coordinate of a vertex defines the labelling $\lambda(v_{\mathrm{I}}) = \varepsilon$, $\lambda(\langle a, x \rangle) = a$, the second defines the ownership $\rho(v_{\mathrm{I}}) = E$, $\rho(\langle a, x \rangle) = x$.

### 2.3.5 Simple stochastic games

*Simple stochastic games* are an extension of reachability games that introduce probability, see e.g. [11].

A simple stochastic game is a tuple $G = \langle V, E, \rho, F, v_{\mathrm{I}} \rangle$, where $V$ is a set of *vertices*, $E \subseteq V \times V$ is a set of *edges*, $v_{\mathrm{I}}$ is an *initial vertex*, $\rho \colon V \to \{E, A, \mathcal{N}\}$ is a division between Eve's, Adam's, and *Nature*'s positions, and $F \subseteq V$ is a set of *final vertices*.

Intuitively, the game starts with the token placed in the initial vertex and is played in turns. Each turn, one of the players moves the token alongside the edges of the graph. If the vertex is owned by *Nature*, a fair dice is rolled, and the next position of the token is chosen with uniform probability. If the vertex is not owned by *Nature*, then, as before, the move is made by the player that owns the vertex.

The winning condition is defined as follows: Eve wins a play if at least one of the visited vertices belongs to $F$.

Simple stochastic games cannot be expressed in terms of the above definition of games on graphs. Still, by a standard construction that introduces a third player, called *Nature*, and enhances the board with a new function describing random choice, simple stochastic games can be seen as a stochastic extension of games on graphs. This extension will be exploited in the following chapter, where we define *branching games*.

### 2.3.6 Blackwell games

The *Blackwell games*, see e.g. [31], are a subclass of *concurrent* games, i.e. games in which players simultaneously decide the next displacement of the token.

A Blackwell game is a tuple $G = \langle S, W \rangle$, where $S$ is a non-empty finite set and $W \subseteq S^\omega$ is a winning set. The game is played by Eve and Adam in turns numbered $0, 1, \ldots$. Every turn both players *simultaneously* choose an element from the set $S$. In $i$th turn Eve chooses $e_i$ and Adam chooses $a_i$.

As before, after an infinite number of moves, an infinite sequence $e_0 a_0 \cdots e_n a_n \cdots$, called a play, is created. Eve wins a play if it belongs to the winning set $W$.

As in the case of simple stochastic games, the concurrent nature of Blackwell games cannot be expressed using our definition of games on graphs. Again, by allowing a shared ownership of nodes we could recover concurrent behaviour. We will not define this extension because, as we will see, branching games can recover the concurrency without shared ownership of the vertices.

## 2.4 Finite automata

The following definition of *finite automata* is valid for both finite and infinite trees.

**The automata** An *alternating tree automaton* is a tuple $\mathcal{A} = \langle Q, \Gamma, \delta, \alpha, q_{\mathrm{I}} \rangle$ where $Q$ is a finite set of *states*; $\Gamma$ is a finite alphabet; $\alpha$ is a *ranking function* that assigns a *priority* $\alpha(q) \in \{i, i+1, \ldots, j\}$ to a state $q \in Q$; $q_{\mathrm{I}}$ is an *initial state*; and $\delta$ is a *transition function* mapping pairs $\langle q, a \rangle \in Q \times \big(\Gamma \cup \{\flat\}\big)$ to formulae $\varphi$ built using the following grammar

$$\varphi ::= \varphi \wedge \varphi \mid \varphi \vee \varphi \mid (p, d) \mid \top \mid \bot, \tag{2.6}$$

where $p \in Q$ and $d \in \{\mathtt{L}, \mathtt{R}\}$.

We say that a formula $\varphi$ is

- *atomic* if $\varphi$ is of the form $(p, d)$ where $p \in Q, d \in \{\mathtt{L}, \mathtt{R}\}$,
- an $\wedge$*-split* if $\varphi$ is of the form $(p, \mathtt{L}) \wedge (q, \mathtt{R})$ where $p, q \in Q$,
- an $\vee$*-split* if $\varphi$ is of the form $(p, \mathtt{L}) \vee (q, \mathtt{R})$ where $p, q \in Q$.

Note that the split formulae necessarily utilise both directions in the atomic sub-formulae.

An automaton $\mathcal{A}$ is called

- a *deterministic* automaton, abbreviated DTA, if the formulae are $\wedge$-splits,
- a *game* automaton, abbreviated GA, if the formulae are either $\vee$- or $\wedge$-splits,
- a *non-deterministic* automaton, abbreviated NTA, if the formulae are disjunctions of $\wedge$-splits.
- an *alternating* automaton, abbreviated ATA, if the formulae are arbitrary positive Boolean combinations, as in the grammar (2.6).

**Acceptance game** Let $t$ be a tree over an alphabet $\Gamma \cup \{\flat\}$ and let $\Delta$ be the set of all sub-formulae of transitions of an alternating tree automaton $\mathcal{A}$. The automaton $\mathcal{A}$ *accepts*

the tree $t$ if Eve has a winning strategy in the parity game $\mathrm{G}(\mathcal{A}, t) = \langle V, s_{\mathtt{L}} \cup s_{\mathtt{R}}, \rho, \alpha', v_{\mathrm{I}} \rangle$ defined as:

- $V = \Delta \times \{\mathtt{L}, \mathtt{R}\}^*$,
- $s_{\mathtt{L}}, s_{\mathtt{R}} \colon V \to V$,
- $v_{\mathrm{I}} = \langle \delta(q_{\mathrm{I}}, t(\varepsilon)), \varepsilon \rangle$.
- If $m = \max_{q \in Q} \alpha(q)$, then $\rho$, $\alpha'$, $s_{\mathtt{L}}$, and $s_{\mathtt{L}}$ are defined as follows: for each $v = \langle \psi, w \rangle \in V$
  - if $\psi = \psi_{\mathtt{L}} \vee \psi_{\mathtt{R}}$, then $\rho(v) = E$, $\alpha'(v) = m$, and $s_d(v) = \langle \psi_d, w \rangle$ for $d \in \{\mathtt{L}, \mathtt{R}\}$;
  - if $\psi = \psi_{\mathtt{L}} \wedge \psi_{\mathtt{R}}$, then $\rho(v) = A$, $\alpha'(v) = m$, and $s_d(v) = \langle \psi_d, w \rangle$ for $d \in \{\mathtt{L}, \mathtt{R}\}$;
  - if $\psi = (d, q)$, then $\rho(v) = E$, $s_{\mathtt{L}}(v) = s_{\mathtt{R}}(v) = \big(\delta(q, t(wd)), wd\big)$, and $\alpha'(v) = \alpha(q)$;
  - if $\psi = \top$, then $\rho(v) = E$, $s_{\mathtt{L}}(v) = s_{\mathtt{R}}(v) = v$, and $\alpha'(v) = 0$;
  - if $\psi = \bot$, then $\rho(v) = E$, $s_{\mathtt{L}}(v) = s_{\mathtt{R}}(v) = v$, and $\alpha'(v) = 1$.

We denote the set of trees accepted by $\mathcal{A}$ as $\mathrm{L}(\mathcal{A})$ and call it the *language recognised* by $\mathcal{A}$.

**Nondeterminism** For the non-deterministic automata we can define an equivalent notion of accepting a tree, the acceptance by a *run*. A *run* of a non-deterministic automaton $\mathcal{A}$ on the tree $t$ is a labelled tree $\mathfrak{r} \colon \{\mathtt{L}, \mathtt{R}\}^* \to Q$ such that $\mathfrak{r}(\varepsilon) = q_{\mathrm{I}}$ and for every $u \in \{\mathtt{L}, \mathtt{R}\}^*$ and every $d \in \{\mathtt{L}, \mathtt{R}\}$ there are states $p_{\mathtt{L}}, p_{\mathtt{R}} \in Q$ and a formula $\psi$ such that $\mathfrak{r}(u\mathtt{L}) = p_{\mathtt{L}}$, $\mathfrak{r}(u\mathtt{R}) = p_{\mathtt{R}}$ and $\delta(\mathfrak{r}(u), t(u)) \equiv \big((p_{\mathtt{L}}, \mathtt{L}) \wedge (p_{\mathtt{R}}, \mathtt{R})\big) \vee \psi$. We say that $\mathfrak{r}$ is accepting if for every path in the tree $\mathfrak{r}$ the limes inferior of the priorities of the labels on the path is even.

**Automata on words** Now we will introduce the *automata on words* (or finite automata) as a special case of automata on trees. A word automaton is an alternating tree automaton that uses atomic formulae with the direction $\mathtt{L}$ only. This is consistent with the previous observation that any word can be seen as a tree with the nodes contained in $\{\mathtt{L}\}^*$. An ATA $\mathcal{A}$ is called

- a *deterministic finite automaton*, abbrev. DFA, if the formulae are atomic with $d = \mathtt{L}$,
- *non-deterministic finite automaton*, abbrev. NFA, if the formulae are disjunctions of atomic formulae with $d = \mathtt{L}$.
- *alternating finite automaton*, abbrev. AFA, if the formulae are arbitrary positive Boolean combinations of atomic formulae with $d = \mathtt{L}$, as defined in the grammar (2.6).

We say that a set of trees (resp. words) $L$ is regular if it is recognised by an alternating automaton on trees (resp. words), i.e. if $L = \mathrm{L}(\mathcal{A})$ for some automaton $\mathcal{A}$.

## 2.5 Logic

**Relational structure** A *relational structure* is a tuple $\mathcal{R} = \langle V, R_i \rangle_{i=1}^{n}$, where $V$ is called the *universe* of the structure $\mathcal{R}$ and $R_i \subseteq V^{a_i}$ are some relations. The number $a_i \in \mathbb{N}$ is called the arity of the relation $R_i$. The set of "names" of the relations $R_i$ is called *signature* of the structure $\mathcal{R}$.

**Gaifman graph** Let $\mathcal{R}$ be a relational structure. The *Gaifman graph* of $\mathcal{R}$ is the undirected graph $G^{\mathcal{R}}$ where the set of vertices is the universe of $\mathcal{R}$ and there is an edge between two vertices $u, v$ in $G^{\mathcal{A}}$ if there is a relation $R$ in $\mathcal{R}$ and a tuple $x \in R$ that has both $u$ and $v$ on some coordinates. The distance $d(u, v)$ between two elements $u, v$ of the universe of $\mathcal{R}$ is the distance between $u, v$ in the Gaifman graph of $\mathcal{R}$. The $r$-neighbourhood of an element $v \in V$ is the set of elements $S \subseteq V$ that are in the distance at most $r$ from $v$, i.e. for every $u \in S$ we have that $d(u, v) \leq r$.

**Trees and logic** A tree $t$ over a finite alphabet $\Gamma$ can be seen as a relational structure $t' = \langle \{\mathtt{L}, \mathtt{R}\}^*, s_{\mathtt{L}}, s_{\mathtt{R}}, s, \sqsubseteq, (a^t)_{a \in \Gamma \cup \{\flat\}} \rangle$, where

- $\{\mathtt{L}, \mathtt{R}\}^*$ is the universe of $t$;
- $s_{\mathtt{L}}, s_{\mathtt{R}} \subseteq \{\mathtt{L}, \mathtt{R}\}^* \times \{\mathtt{L}, \mathtt{R}\}^*$ are left child relation ($u\ s_{\mathtt{L}}\ u{\cdot}\mathtt{L}$) and right child relation ($u\ s_{\mathtt{R}}\ u{\cdot}\mathtt{R}$), respectively;
- $s$ is the child relation $s_{\mathtt{L}} \cup s_{\mathtt{R}}$;
- $\sqsubseteq$ is the ancestor relation, i.e. the reflexive, transitive closure of the relation $s$;
- $a^t = \lambda^{-1}(a) \subseteq \{\mathtt{L}, \mathtt{R}\}^*$ for $a \in \Gamma \cup \{\flat\}$.

Moreover, the family of sets $(a^t)_{a \in \Gamma \cup \{\flat\}}$ is a partition of $\{\mathtt{L}, \mathtt{R}\}^*$. Conversely, if $t$ is a tree seen as a relational structure, then the partition $(a^t)_{a \in \Gamma \cup \{\flat\}}$ induces a labelling function $\lambda_t \colon \{\mathtt{L}, \mathtt{R}\}^* \to \Gamma \cup \{\flat\}$ in the natural way, i.e. $\lambda_t(u) = a$ if and only if $u \in a^t$.

**Monadic second-order logic** Formulae of *monadic second-order logic*, abbrev. MSO, can quantify over elements of the universe ($\exists x, \forall x$) and over the subsets of the universe ($\exists X, \forall X$) of a relational structure. A sentence is a formula with no free variables. We use the standard notions of *free variables*, *quantifier rank*, *valuations*, etc.; for details see e.g. [54].

We say that a monadic second-order formula $\varphi$ is *over a signature* $\Sigma$ if $\varphi$ is a well-formed formula built from the symbols in $\Sigma$ together with the quantifiers and logical connectives. Let $\Gamma$ be an alphabet, in this document we consider only the formulae over the signatures $\Sigma$ such that $\Sigma \subseteq \{s_{\mathtt{L}}, s_{\mathtt{R}}, s, \sqsubsetneq, \sqsubseteq, \varepsilon\} \cup \big(\Gamma \cup \{\flat\}\big)$. The symbol $\varepsilon$ stands for the *root predicate*, i.e. $\varepsilon(u) \iff u = \varepsilon$, and the relation $\sqsubsetneq$ stands for the *strict prefix order*, i.e. $u \sqsubsetneq v \iff (u \sqsubseteq v \wedge u \neq v)$.

**MSO and tree languages** Let $t$ be a tree; $v_1, \ldots, v_n$ be a list of elements from the universe of $t$; $V_1, \ldots, V_k$ be a list of sets of elements from the universe; and $\varphi((X_1, \ldots, X_k, x_1, \ldots, x_n)$ be a monadic second-order formula with $n$ first-order free variables and $k$ second-order free variables. We write $t, V_1, \ldots, V_k, v_1, \ldots, v_n \vDash \varphi(X_1, \ldots, X_k, x_1, \ldots, x_n)$ if $\varphi$ is satisfied in a tree $t$ where every $X_i$ is interpreted as $V_i$ and every $x_i$ is interpreted as $v_i$. We call a formula $\varphi$ satisfiable (in a tree $t$) if it is satisfied for some $V_1, \ldots, V_k, v_1, \ldots, v_n$ as above. If a formula is not satisfiable, we call it unsatisfiable.

The language of trees defined by a monadic second-order sentence $\varphi$ over a signature containing an alphabet $\Gamma$, denoted $\mathrm{L}(\varphi)$, is the set $\{t \in \mathcal{T}_\Gamma \mid t \vDash \varphi\}$.

**Theorem 2.5.1** (Rabin [50])**.** *Let $\varphi$ be a monadic second-order formula. Then, the set $\mathrm{L}(\varphi)$ is regular.*

**First-order logic** A *first-order formula* is a monadic second-order formula that does not quantify over the sets. We say that a first-order formula $\varphi(x)$ is an $r$-local formula around $x$ if the quantifiers are restricted to $r$-neighbourhood of $x$, i.e. if $\varphi(x)$ uses the quantifiers $\forall^{\leq r}$ and $\exists^{\leq r}$ defined as follows: $\exists^{\leq r} y.\psi(y) \stackrel{\text{def}}{=} \exists y.\psi(y) \wedge d(x,y) \leq r$ and $\forall^{\leq r} y.\psi(y) \stackrel{\text{def}}{=} \forall y.(d(x,y) \leq r) \to \psi(y)$, where $d(x,y) \leq r$ is a first-order formula stating that the distance in the Gaifman graph between $x$ and $y$ is at most $r$.

We say that a first-order sentence $\varphi$ is a *basic $r$-local sentence* if it is of the form

$$\exists x_1, \ldots, x_s. \Big( \bigwedge_{1 \leq i \leq s} \varphi_i(x_i) \wedge \bigwedge_{1 \leq i < j \leq s} d(x_i, x_j) > 2r \Big) \tag{2.7}$$

where $\varphi_i(x)$ are $r$-local formulae around $x$ and $d(x,y) > z$ are first-order formulae stating that the distance between $x$ and $y$ in the Gaifman graph of the structure is strictly greater than $z$.

**Theorem 2.5.2** (Gaifman [18])**.** *Every first-order sentence $\varphi$ is equivalent to a Boolean combination of basic $r$-local sentences, i.e. sentences of the form defined by Equation* (2.7)*. Furthermore, one can compute a Boolean combination such that $r \leq 7^{qr(\varphi)}$ and $s \leq n + qr(\varphi)$, where $qr(\varphi)$ is the quantifier rank of the formula $\varphi$ and $n$ is the length of the formula $\varphi$.*

**Conjuntive queries** A *conjunctive query* (abbrv., CQ) over an alphabet $\Gamma$ is a sentence of first-order logic using only conjunction, existential quantification, unary predicates $a(x)$, for $a \in \Gamma$, the root predicate $\varepsilon(x)$, and the binary predicates $s_{\mathtt{L}}, s_{\mathtt{R}}, s, \subsetneq$.

**Patterns** An alternative way of looking at conjunctive queries is via graphs and graph homomorphisms. A pattern $\pi$ over an alphabet $\Gamma$ is a tuple $\pi = \langle V, V_\varepsilon, E_{\mathtt{L}}, E_{\mathtt{R}}, E_s, E_{\subsetneq}, \lambda_\pi \rangle$, where $V$ is a set of vertices, $\lambda_\pi \colon V \rightharpoonup \Gamma$ is a partial function assigning labels, $V_\varepsilon$ is the set

of root vertices, $E_{\mathtt{L}}, E_{\mathtt{R}}, E_s$, and $E_{\mathtt{\Yup}} \subseteq V \times V$ are a left child, a right child, a child, and an ancestor relations, respectively.

Since the patterns consist of relations of arity at most two, they can be represented as graphs. A graph of a pattern $\pi$ is the finite graph $G_\pi = \langle V, E_{\mathtt{L}} \cup E_{\mathtt{R}} \cup E_s \cup E_{\mathtt{\Yup}} \rangle$ whose set of vertices is the universe of $\pi$ and the set of edges comprises of left child edges $E_{\mathtt{L}}$, right child edges $E_{\mathtt{R}}$, child edges $E_s$, and ancestor edges $E_{\mathtt{\Yup}}$. By $|\pi|$ we mean the size of the pattern $\pi$, i.e. the cardinality of the set of vertices of the underlying graph.

A sub-pattern $\pi'$ of a pattern $\pi = \langle V, V_\varepsilon, E_{\mathtt{L}}, E_{\mathtt{R}}, E_s, E_{\mathtt{\Yup}}, \lambda_\pi \rangle$ is the pattern with the graph structure induced by some set of vertices $S \subseteq V$.

We say that a tree $t = \langle dom(t), s_{\mathtt{L}}, s_{\mathtt{R}}, \mathtt{\Yup}, (a^t)_{a \in \Gamma \cup \{\flat\}} \rangle$ satisfies a pattern $\pi = \langle V, V_\varepsilon, E_{\mathtt{L}}, E_{\mathtt{R}}, E_s, E_{\mathtt{\Yup}}, \lambda_\pi \rangle$, denoted $t \models \pi$, if there exists a homomorphism $h \colon \pi \to t$, that is a function $h \colon V \to dom(t)$ such that

1. $h \colon \langle V, E_{\mathtt{L}}, E_{\mathtt{R}}, E_s, E_{\mathtt{\Yup}} \rangle \to \langle dom(t), s_{\mathtt{L}}, s_{\mathtt{R}}, s_{\mathtt{L}} \cup s_{\mathtt{R}}, \mathtt{\Yup} \rangle$ is a homomorphism of relational structures,
2. for every $v \in V_\varepsilon$ we have that $h(v) = \varepsilon$,
3. and for every $v \in dom(\lambda_\pi)$ we have that $\lambda_\pi(v) = \lambda_t(h(v))$.

Every pattern can be seen as a conjunctive query and vice versa. Hence, we will use those terms interchangeably.

**Fact 2.5.3** (Folklore)**.** *There is a log-space procedure that inputs a conjunctive query $q$ (resp., a pattern $\pi$) over an alphabet $\Gamma$ and outputs a pattern $\pi$ (resp., a conjunctive query $q$) over the alphabet $\Gamma$ such that for every binary tree $t$ over an alphabet $\Gamma$ we have that*

$$t \models q \iff t \models \pi \tag{2.8}$$

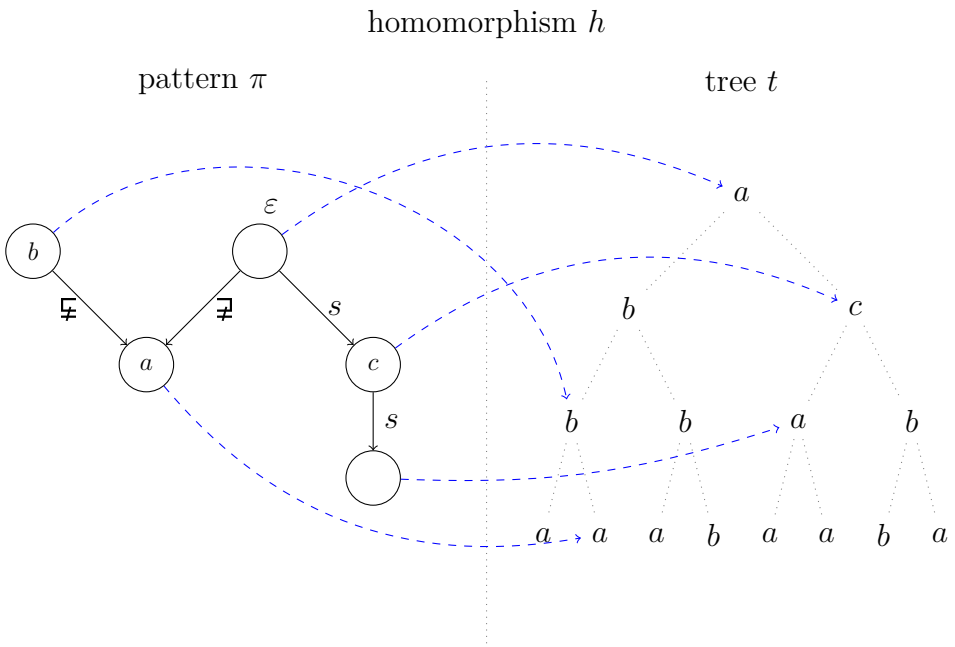



Figure 2.1 – A pattern $\pi$ (solid arrows) equivalent to the conjunctive query $\varphi \stackrel{\text{def}}{=} \exists x_1, x_2, x_3, x_4, x_5.\ \varepsilon(x_2) \wedge b(x_1) \wedge a(x_3) \wedge c(x_4) \wedge x_1 \not\subsetneq x_3 \wedge x_2 \not\subsetneq x_3 \wedge x_2 s x_4 \wedge x_4 s x_5$, and a homomorphism $h \colon \pi \to t$ (blue dashed arrows) between the pattern $\pi$ and the tree $t$ (dotted lines).

# Chapter 3

# Branching games

In this chapter we define branching games which constitute the central notion of the theoretical part of this thesis. In particular, we define the concepts of a branching board, a (mixed) strategy, a partial value, and determinacy.

Additionally, we show how the games on graphs defined in the previous chapter can be interpreted as branching games and state some previously known results about their determinacy.

## 3.1 Branching games

Branching games, also known as *tree games*, see [33], can be seen as an extension of *Gale-Stewart* games [19] or so-called games on graphs [53]. Those games are, in general, played by three players: Eve, Adam – two antagonistic abstractions of existential and universal choices – and *Nature* – the third, disinterested party representing random choice. The interactions between the players are defined by a *board* on which the game is played, by the objectives the players want to achieve, and are formalised as strategies.

The extension proposed in the branching games is formed by adding yet another type of positions to the board, called the *branching* positions, and adjusting the notions of an objective and a strategy to be consistent with the new features of the board.

Intuitively, the branching positions split the flow of the game creating new, separate threads, which continue the execution of the game independently. The only information shared between the threads is their common history, i.e. a path beginning at the initial position on the board and ending at the branching vertex that split the threads.

More specifically, a *branching game* is a pair $G = \langle \mathtt{B}, \Phi \rangle$ where $\mathtt{B}$ is a *branching board* and $\Phi$ is a universally measurable bounded real function $\Phi\colon \mathrm{plays}(\mathtt{B}) \to \mathbb{R}_{\geq 0}$ describing the objective of the game. The game is played by two adversaries called Eve and Adam (or shortly $E$ and $A$) on the *branching board* $\mathtt{B}$. The behaviour of the third player, who is denoted $\mathcal{N}$,

is randomised, consistent with the construction of the board, and visible to the players.

Now we will describe the formalism necessary to discuss the branching games in a precise manner, i.e. we will define the notions of a branching board, a play, a strategy, and a value of a game.

### 3.1.1 Branching board

A branching board is a tuple $\mathtt{B} = \langle V, \Gamma, s_{\mathtt{L}}, s_{\mathtt{R}}, \rho, \eta, \lambda, v_{\mathrm{I}} \rangle$, where

- $V$ is a set of *vertices*;
- $\Gamma$ is an *alphabet*;
- $s_{\mathtt{L}}, s_{\mathtt{R}} \colon V \to V$ are *successor functions*;
- $\rho \colon V \to \{E, A, \mathcal{N}, \mathcal{B}\}$ is a partition of the vertices between Eve's, Adam's, *Nature*'s, and branching vertices;
- $\eta \colon \rho^{-1}\big(\{\mathcal{N}\}\big) \to \mathit{dist}(\{\mathtt{L}, \mathtt{R}\})$ is a function that maps *Nature*'s vertices to random distributions over the successors;
- $\lambda \colon V \to \Gamma$ is a *labelling* of the vertices with the alphabet $\Gamma$;
- and $v_{\mathrm{I}} \in V$ is an *initial vertex*.

**Unfolding** We extend the successor functions $s_d$, $d \in \{\mathtt{L}, \mathtt{R}\}$, to arbitrary sequences of directions in the natural way. Let $v \in V$ be a vertex, $u \in \{\mathtt{L}, \mathtt{R}\}^*$ be a sequence of directions, and $d \in \{\mathtt{L}, \mathtt{R}\}$ be a direction. Then, $s_\varepsilon(v) \stackrel{\text{def}}{=} v$ and $s_{u \cdot d}(v) \stackrel{\text{def}}{=} s_d\big(s_u(v)\big)$.

The *unfolding* (or unravelling) of a branching board $\mathtt{B}$ in a vertex $v$ is the function $t_{\mathtt{B}} \colon \{\mathtt{L}, \mathtt{R}\}^* \to V$ such that $t_{\mathtt{B}}(u) \stackrel{\text{def}}{=} s_u(v)$. If $v = v_{\mathrm{I}}$ we call it simply the unfolding of the board $\mathtt{B}$. Moreover, every board $\mathtt{B}$ defines a tree $t^{\lambda}_{\mathtt{B}} \colon \{\mathtt{L}, \mathtt{R}\}^* \to \Gamma$ as the unfolding of the adequate labelled sub-graph of the board, i.e. $t^{\lambda}_{\mathtt{B}} \stackrel{\text{def}}{=} \lambda \circ t_{\mathtt{B}}$. Similarly, we can define the tree of position assignments $t^{\rho}_{\mathtt{B}} \stackrel{\text{def}}{=} \rho \circ t_{\mathtt{B}}$. Note that $t^{\lambda}_{\mathtt{B}}$ (resp., $t^{\rho}_{\mathtt{B}}$) is a full binary tree over the alphabet $\Gamma$ (resp. the set $\{E, A, \mathcal{N}, \mathcal{B}\}$).

For $P \in \{E, A, \mathcal{N}, \mathcal{B}\}$, by $V_P$ we denote the set of vertices belonging to $P$, i.e. $\rho^{-1}\big(\{P\}\big)$. For $\mathcal{P} \subseteq \{E, A, \mathcal{N}, \mathcal{B}\}$ we say that $\mathtt{B}$ is *$\mathcal{P}$-branching* if $\mathit{range}(\rho) \subseteq \mathcal{P}$, e.g. a board $\mathtt{B}$ is $\{\mathcal{N}, \mathcal{B}\}$-branching if it has no Eve's nor Adam's positions.

We say that a board $\mathtt{B}$ is

- *non-stochastic* if it is $\{E, A, \mathcal{B}\}$-branching;
- *single player* if it is either $\{E, \mathcal{N}, \mathcal{B}\}$-or $\{A, \mathcal{N}, \mathcal{B}\}$-branching;
- *finitary* if $V$ is finite and every distribution in $\eta$ has rational values;
- *simple* if every distribution in $\eta$ has value $\frac{1}{2}$.

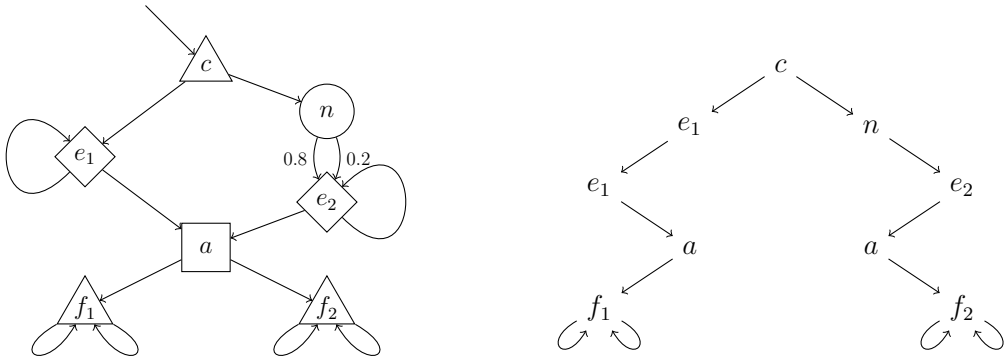



Figure 3.1 – An example of a branching board and a play on this board. We denote Eve's, Adam's, *Nature*'s, and branching vertices by diamonds, squares, circles, and triangles respectively. *Nature*'s vertices are equipped with a probability distribution over the successors. The initial vertex is the only vertex with an arrow not having a source vertex. The successors $\mathtt{R}$ and $\mathtt{L}$ are drawn in the clockwise order, i.e. $\mathtt{R}$ moves to the right and is drawn first in the clockwise order.

**Flow of the game** With the branching board defined, we can intuitively explain how the players play the game. The proper mathematical description will be presented in the next few paragraphs. Intuitively, a single play over a branching board $\mathtt{B}$ proceeds in threads, each thread has exactly one token, located in a vertex of the board. Initially, there is a single thread with the token located in $v_{\mathrm{I}}$.

Consider a thread with a token located in a vertex $v$:

- if $\rho(v) = \mathcal{B}$ then the thread is duplicated into two separate threads with tokens located in $s_{\mathtt{L}}(v)$ and $s_{\mathtt{R}}(v)$;
- if $\rho(v) = \mathcal{N}$ then the token is moved either to $s_{\mathtt{L}}(v)$ or to $s_{\mathtt{R}}(v)$ with probability given by the distribution $\eta(v)$;
- if $\rho(v) \in \{E, A\}$ then the respective player moves the token either to $s_{\mathtt{L}}(v)$ or to $s_{\mathtt{R}}(v)$ depending on the history of the current thread.

The threads are executed in a non-specified order, independently, and simultaneously. Furthermore, the players cannot take into account positions of the tokens from the other threads in the current play when moving the tokens.

After all the threads moved the tokens infinitely many times, a tree-like structure, called a *play* $t$, has been created.

We will now formalise the notion of a play.

**Branching** Consider a non-empty set $\mathcal{P} \subseteq \{E, A, \mathcal{N}, \mathcal{B}\}$. We say that a tree $t \sqsubseteq t^{\lambda}_{\mathtt{B}}$ is $\mathcal{P}$-*branching* if it is fully branching in the nodes $u \in \{\mathtt{L}, \mathtt{R}\}^*$ such that $\rho\big(s_u(v_{\mathrm{I}})\big) \in \mathcal{P}$ and

uniquely branching in the remaining nodes.

**Plays** A *play* on a board $\mathtt{B}$ is a tree $t \sqsubseteq t^{\lambda}_{\mathtt{B}}$ that is $\{\mathcal{B}\}$-branching. The set of all plays on a board $\mathtt{B}$ is denoted by $\mathrm{plays}(\mathtt{B})$. The value $val_G(t)$ of a play $t$ is the value of the pay-off function $\Phi$, that is $val_G(t) = \Phi(t)$, recall that a branching game is a pair $G = \langle \mathtt{B}, \Phi \rangle$. Figure 3.1 depicts a branching board and a play on this board.

**Strategy** For $P \in \{E, A, \mathcal{N}\}$ we say that a tree $t \subseteq t^{\lambda}_{\mathtt{B}}$ is a *pure strategy* of $P$ over $\mathtt{B}$ if $t$ is $\big(\{E, A, \mathcal{N}, \mathcal{B}\} \setminus \{P\}\big)$-branching. The set of pure strategies of $P$ over $\mathtt{B}$ is denoted $\Sigma^P_{\mathtt{B}}$. Notice that the sets $\mathrm{plays}(\mathtt{B})$ and $\Sigma^P_{\mathtt{B}}$ for $P \in \{E, A, \mathcal{N}\}$ are closed sets of $\Gamma$-labelled trees. Moreover, if $V$ is finite then all these sets are regular. Indeed, by Proposition 3.1.2 the set of plays is regular. Additionally, a set of pure strategies of a player $P$ can be seen as a set of plays on a modified board $\mathtt{B}'$ where every vertex not belonging to $P$ is now branching.

Given three pure strategies $\sigma \in \Sigma^E_{\mathtt{B}}$, $\pi \in \Sigma^A_{\mathtt{B}}$, and $\eta \in \Sigma^{\mathcal{N}}_{\mathtt{B}}$ the *play resulting from $\sigma$, $\pi$, and $\eta$* (denoted $\mathrm{eval}_{\mathtt{B}}(\sigma, \pi, \eta)$) is the unique tree $t \in \mathrm{plays}(\mathtt{B})$ such that

$$nodes(t) = nodes(\sigma) \cap nodes(\pi) \cap nodes(\eta).$$

Thus, $\mathrm{eval}_{\mathtt{B}}$ can be seen as a function $\mathrm{eval}_{\mathtt{B}} \colon \Sigma^E_{\mathtt{B}} \times \Sigma^A_{\mathtt{B}} \times \Sigma^{\mathcal{N}}_{\mathtt{B}} \to \mathrm{plays}(\Gamma)$ mapping triplets of pure strategies to the unique play they generate.

**Claim 3.1.1.** *The function* $\mathrm{eval}_{\mathtt{B}}$ *is continuous.*

**Mixed strategies** A mixed strategy of a player $P \in \{E, A, \mathcal{N}\}$ is a complete Borel probability measure over the set $\Sigma^P_{\mathtt{B}}$. The set of all mixed strategies of $P$ is denoted by $\Sigma^{MP}_{\mathtt{B}}$.

**Behavioural strategies** There is a natural family of mixed strategies, strongly associated with the standard games on graphs, called *behavioural* strategies. Let $f \colon \{\mathtt{L}, \mathtt{R}\}^* \rightharpoonup dist(\{\mathtt{L}, \mathtt{R}\})$ be a partial function supplying some of the positions with a probability distribution over the successors and let $t \in \mathcal{T}^{\infty}_{\Gamma}$ be a full binary tree. The function $f$ and the tree $t$ induce, in a natural way, a measure $\mu^f_t$ on the set $\mathcal{T}_{\Gamma}$.

The intuition behind the measure $\mu^f_t$ is that in a position $u$ belonging to the set $dom(f)$ the measure "chooses which sub-tree should be cut" with probability given by the distribution $f(u)$.

The measure is concentrated on the set of prefixes of $t$, i.e. $\mu^f_t\big(\{t' \in \mathcal{T}_{\Gamma} \mid t' \sqsubseteq t\}\big) = 1$, and defined as follows. Let $t'$ be a tree of height $k$, then we define $\mu^f_t(\mathbb{B}_{t'})$ as follows.

$$\mu^f_t(\mathbb{B}_{t'}) = \begin{cases} 0 & \text{if there is a node } u \in dom(f) \text{ such that} \\ & \text{the node } u \text{ has both children in } t'; \\ \alpha^f_t(t') & otherwise. \end{cases} \tag{3.1}$$

The number $\alpha_t^f(t')$ is the probability of the particular choices of directions within the tree $t'$ and can be computed as follows.

$$\alpha_t^f(t') = \prod_{\substack{u \in dom(f),\\ ud \in nodes(t')}} f(u)(d) \tag{3.2}$$

We say that a mixed strategy $\sigma_m$ of a player $P$ is *behavioural* if it is a measure induced by some partial function $f\colon \{\mathtt{L},\mathtt{R}\}^* \rightharpoonup dist(\{\mathtt{L},\mathtt{R}\})$ such that $dom(f) = (t_{\mathtt{B}}^{\rho})^{-1}(\{P\})$ and the tree $t_{\mathtt{B}}^{\lambda}$, i.e. if $\sigma_m = \mu^f_{t_{\mathtt{B}}^{\lambda}}$. The set of all behavioural strategies of $P$ is denoted by $\Sigma_{\mathtt{B}}^{BP}$.

Clearly, we can treat every pure strategy in $\Sigma_{\mathtt{B}}^{P}$ as a Dirac delta function in $\Sigma_{\mathtt{B}}^{MP}$ (in fact in $\Sigma_{\mathtt{B}}^{BP}$). Thus, we can assume that $\Sigma_{\mathtt{B}}^{P} \subseteq \Sigma_{\mathtt{B}}^{BP} \subseteq \Sigma_{\mathtt{B}}^{MP}$.

**Strategy of *Nature*** With every branching board $\mathtt{B}$ we associate a special mixed strategy $\eta_{\mathtt{B}}^*$ of *Nature*. This strategy represents the intuition, that after a sequence of displacements of the token, described by a sequence of directions $u \in \{\mathtt{L},\mathtt{R}\}$, ending in a vertex $v = s_u(v_{\mathrm{I}}) \in V$ such that $\rho(v) = \mathcal{N}$, *Nature* chooses to move the token in a direction $d \in \{\mathtt{L},\mathtt{R}\}$ with the probability $\eta(v)(d)$. The strategy $\eta_{\mathtt{B}}^*$ is defined as follows.

$$\eta_{\mathtt{B}}^* \stackrel{\text{def}}{=} \mu^{t_{\mathtt{B}}^{\eta}}_{t_{\mathtt{B}}^{\lambda}} \tag{3.3}$$

**Strategies as functions** There is a different way to define the three types of strategies that may give more intuition to the behaviour and expressive power of the strategies, we can call those strategies *functional strategies*. A pure strategy $\sigma \in \Sigma_{\mathtt{B}}^{P}$ can be seen as a function $\sigma\colon \{\mathtt{L},\mathtt{R}\}^* \to \{\mathtt{L},\mathtt{R}\}$; a behavioural strategy $\sigma_b \in \Sigma_{\mathtt{B}}^{BP}$ as a function $\sigma_b\colon \{\mathtt{L},\mathtt{R}\}^* \to dist(\{\mathtt{L},\mathtt{R}\})$; and a mixed strategy $\sigma_m \in \Sigma_{\mathtt{B}}^{MP}$ as a measure $\sigma_m \in dist(\Sigma_{\mathtt{B}}^{P})$.

In other words, the three types of strategies utilise different levels of randomness when deciding where to move the token. A pure strategy deterministically decides the next move taking into account only the path the token took from the initial position to the current one, a behavioural strategy flips a biassed coin to determine the next move, the bias is determined by the path, and a mixed strategy simply chooses a pure strategy at the start of the game, according to some probability distribution.

This interpretation also allows us to justify stating that a pure strategy chooses a vertex of the board. When we say that a pure strategy chooses a vertex $v$ of the board, we mean that the strategy moves the token from its current position to the vertex $v$.

Let $G = \langle \mathtt{B}, \Phi \rangle$ be a branching game. For a player $P$ and a strategy type $X$, we will often write $\Sigma_G^{XP}$ instead of $\Sigma_{\mathtt{B}}^{XP}$ or, if the board is clear from the context, we will simply drop the subscript and write $\Sigma^{XP}$. Similarly, we will write $\mathrm{eval}_G$ or simply eval instead of $\mathrm{eval}_{\mathtt{B}}$.

### 3.1.2 Branching boards and their expressive power

In order to better understand the formal definitions of a branching board and of a branching game let us discuss which sets of plays a branching game can produce.

We say that a tree language $L \subseteq \mathcal{T}_\Gamma$ is *game definable*[1] if there is an $\{E, \mathcal{B}\}$-branching finitary board $\mathtt{B}$, with a finite set of vertices, such that $L$ is the set of possible plays on the board $\mathtt{B}$, i.e. $L = \mathrm{plays}(\mathtt{B})$.

**Proposition 3.1.2.** *Every game definable language $L$ is a closed regular set of trees. Moreover, given a finitary board $\mathtt{B}$ one can compute a non-deterministic automaton recognising $L$ in time linear in the size of the board.*

*Proof.* If a tree language $L$ is game definable, then there exists a board $\mathtt{B}$ such that $L = \mathrm{plays}(\mathtt{B})$. A tree $t$ is not a play if there is a finite prefix $p \sqsubseteq t$ such that it either has a node that should be fully branching and it is not, or should be uniquely branching and it is not. Therefore, the set of trees that are not plays is open, as it is a union of open sets $\mathbb{B}_p$. In consequence, the set of plays is the complement of an open set and, thus, is closed.

Now we will describe the construction of the automaton. It is defined as follows. The alphabet is the set of positions on the board and the blank symbol. The states of the automaton are the positions on the board and a rejecting state. The initial state is the initial vertex.

The transition function works as follows. If the state is a branching position, then the automaton assumes that both the left child and the right child are not blank and are labelled with the left successor and the right successor, respectively; and verifies that. If the state is not a branching position then the automaton guesses which sub-tree was cut and if guesses correctly, then continues the run. If not, then it enters the rejecting state. If the automaton is in a rejecting state, then it stays in the rejecting state.

Rejecting state has priority 1 and every other state has priority 0. Thus, a run is accepting if and only if it avoids the rejecting state.

It is not hard to see that this automaton recognises the language $L$. Moreover, it is clear that the above construction can be done in linear time, which concludes the proof. □

The converse statement is not true, e.g. the language $\{f_a, f_b\}$ where $f_a$ (resp., $f_b$) is the full binary tree with all nodes labelled with letter $a$ (resp., $b$) is not game definable. Indeed, in game definable languages there is a single letter that labels the roots trees in the language. Clearly, this is not the case for the language $\{f_a, f_b\}$.

[1] The similarity of terms "game definable" and "game automata" is purely coincidental.

## 3.2 Values of a branching game

With the notions of a board, a play, and a strategy properly defined, we can define what a value of a game is.

**Values of strategies** Assume that $\sigma_m \in \Sigma_{\mathtt{B}}^{ME}$ and $\pi_m \in \Sigma_{\mathtt{B}}^{MA}$ are two mixed strategies of the respective players. Our aim is to define the value $val_{\mathrm{G}}(\sigma_m, \pi_m)$. Intuitively, $val_{\mathrm{G}}(\sigma_m, \pi_m)$ should be the expected value of $\Phi\big(\mathrm{eval}_{\mathrm{G}}(\sigma, \pi, \eta)\big)$ where the pure strategies $\sigma$, $\pi$, and $\eta$ are chosen according to the probability distributions $\sigma_m$, $\pi_m$, and $\eta_{\mathtt{B}}^*$ respectively. This is formalised as follows.

$$val_{\mathrm{G}}(\sigma_m, \pi_m) \stackrel{\text{def}}{=} \int_{\Sigma_{\mathtt{B}}^E, \Sigma_{\mathtt{B}}^A, \Sigma_{\mathtt{B}}^N} \Phi\big(\mathrm{eval}_{\mathrm{G}}(\sigma, \pi, \eta)\big)\, \mathrm{d}\sigma_m(\sigma)\, \mathrm{d}\pi_m(\pi)\, \mathrm{d}\eta_{\mathtt{B}}^*(\eta) \tag{3.4}$$

If strategies $\sigma$ and $\pi$ are pure strategies of Adam and Eve, respectively, and the board $\mathtt{B}$ is non-stochastic, then $val_{\mathrm{G}}(\sigma, \pi) = \Phi(\mathrm{eval}_{\mathrm{G}}(\pi, \sigma, t_{\mathtt{B}}^{\lambda}))$.

**Values of a game** The aim of Eve in a branching game is to maximise the value $val_{\mathrm{G}}(\sigma, \pi)$. Let us define the *partial values* of the game. Consider $X \in \{\varepsilon, B, M\}$ (i.e. $X$ stands for respectively *pure*, *behavioural*, and *mixed* strategies). The $X$ *value of* G *for Eve* (resp. Adam) is defined as

$$val_{\mathrm{G}}^{XE} \stackrel{\text{def}}{=} \sup_{\sigma \in \Sigma_{\mathtt{B}}^{XE}} val_{\mathrm{G}}(\sigma) \quad \text{where} \quad val_{\mathrm{G}}(\sigma) \stackrel{\text{def}}{=} \inf_{\pi \in \Sigma_{\mathtt{B}}^{A}} val_{\mathrm{G}}(\sigma, \pi),$$
$$val_{\mathrm{G}}^{XA} \stackrel{\text{def}}{=} \inf_{\pi \in \Sigma_{\mathtt{B}}^{XA}} val_{\mathrm{G}}(\pi) \quad \text{where} \quad val_{\mathrm{G}}(\pi) \stackrel{\text{def}}{=} \sup_{\sigma \in \Sigma_{\mathtt{B}}^{E}} val_{\mathrm{G}}(\sigma, \pi).$$

Note that the second inf/sup is taken over the pure strategies of the opponent. This is explained by the following simple lemma.

**Lemma 3.2.1.** *Let* G *be a branching game. If $\sigma_m$ is Eve's mixed strategy then*

$$\inf_{\pi_m \in \Sigma_{\mathtt{B}}^{MA}} val_{\mathrm{G}}(\sigma_m, \pi_m) = \inf_{\pi_b \in \Sigma_{\mathtt{B}}^{BA}} val_{\mathrm{G}}(\sigma_m, \pi_b) = \inf_{\pi \in \Sigma_{\mathtt{B}}^{A}} val_{\mathrm{G}}(\sigma_m, \pi)$$

*The same holds for mixed strategies of Adam if we replace* inf *with* sup *and $A$ with $E$.*

*Proof.* Since $\Sigma_{\mathtt{B}}^A \subseteq \Sigma_{\mathtt{B}}^{BA} \subseteq \Sigma_{\mathtt{B}}^{MA}$, we immediately get

$$\inf_{\pi_m \in \Sigma_{\mathtt{B}}^{MA}} val_{\mathrm{G}}(\sigma_m, \pi_m) \leq \inf_{\pi_b \in \Sigma_{\mathtt{B}}^{BA}} val_{\mathrm{G}}(\sigma_m, \pi_b) \leq \inf_{\pi \in \Sigma_{\mathtt{B}}^{A}} val_{\mathrm{G}}(\sigma_m, \pi).$$

To conclude the proof we only need to show that

$$\inf_{\pi \in \Sigma_{\mathtt{B}}^{A}} val_{\mathrm{G}}(\sigma_m, \pi) \leq \inf_{\pi_m \in \Sigma_{\mathtt{B}}^{MA}} val_{\mathrm{G}}(\sigma_m, \pi_m). \tag{3.5}$$

Let $\sigma_m \in \Sigma_{\mathtt{B}}^{ME}$ and $\pi_m \in \Sigma_{\mathtt{B}}^{MA}$ be arbitrary mixed strategies, then

$$\begin{aligned} val_{\mathrm{G}}(\sigma_m, \pi_m) &= \int_{\Sigma_{\mathtt{B}}^E, \Sigma_{\mathtt{B}}^A, \Sigma_{\mathtt{B}}^{\mathcal{N}}} \Phi\big(\mathrm{eval}_{\mathrm{G}}(\sigma, \pi, \eta)\big)\, \mathrm{d}\sigma_m(\sigma)\, \mathrm{d}\pi_m(\pi)\, \mathrm{d}\eta_{\mathtt{B}}^*(\eta) \\ &= \int_{\Sigma_{\mathtt{B}}^A} val_G(\sigma_m, \pi)\ \mathrm{d}\pi_m(\pi) \\ &\geq \int_{\Sigma_{\mathtt{B}}^A} \big(\inf_{\pi' \in \Sigma_{\mathtt{B}}^A} val_G(\sigma_m, \pi')\big)\ \mathrm{d}\pi_m(\pi) \\ &= \inf_{\pi' \in \Sigma_{\mathtt{B}}^A} val_G(\sigma_m, \pi'). \end{aligned}$$

Since $val_{\mathrm{G}}(\sigma_m, \pi_m) \geq \inf_{\pi \in \Sigma_{\mathtt{B}}^A} val_{\mathrm{G}}(\sigma_m, \pi)$ holds for every pair of strategies $\sigma_m \in \Sigma_{\mathtt{B}}^{ME}$, $\pi_m \in \Sigma_{\mathtt{B}}^{MA}$; we infer that Equation (3.5) holds and conclude the proof. □

**Determinacy** As a simple consequence of Lemma 3.2.1 we obtain the following inequalities

$$val_{\mathrm{G}}^A \geq val_{\mathrm{G}}^{BA} \geq val_{\mathrm{G}}^{MA} \geq val_{\mathrm{G}}^{ME} \geq val_{\mathrm{G}}^{BE} \geq val_{\mathrm{G}}^E. \tag{3.6}$$

The first two (resp. the last two) inequalities hold by the fact that we take inf (resp. sup) over greater (reps. smaller) sets of strategies. The third inequality holds by Lemma 3.2.1 and the fact that $\inf_x \sup_y f(x,y) \geq \sup_y \inf_x f(x,y)$.

We will say that a branching game G is *determined*

- *under pure strategies* if $val_{\mathrm{G}}^A = val_{\mathrm{G}}^E$,
- *under behavioural strategies* if $val_{\mathrm{G}}^{BA} = val_{\mathrm{G}}^{BE}$,
- *under mixed strategies* if $val_{\mathrm{G}}^{MA} = val_{\mathrm{G}}^{ME}$.

Clearly, Equation (3.6) shows that pure determinacy implies behavioural determinacy and behavioural determinacy implies mixed determinacy. In general, the opposite implications may not hold, as seen in the following example inspired by Mio, see [33].

**Example 3.2.2.** *Let* $\mathtt{B}$ *be as in Figure 3.2 and* $L$ *be defined as*

$$L \stackrel{def}{=} \big\{t \in \mathrm{plays}(\mathtt{B}) \mid x_1(t) = x_2(t) = x_3(t) = x_4(t) \ \vee\ x_3(t) \neq x_4(t)\big\}, \tag{3.7}$$

*where* $x_i(t)$ *is the label of the non-blank child of the* $x_i$*-labelled node in a tree* $t$*. Then, the game* $\mathrm{G} = \langle \mathtt{B}, \chi_L \rangle$ *has the following partial values:*

$$val_{\mathrm{G}}^A = 1; \qquad val_{\mathrm{G}}^{BA} = \frac{3}{4}; \qquad val_{\mathrm{G}}^{MA} = \frac{1}{2} = val_{\mathrm{G}}^{ME}; \qquad val_{\mathrm{G}}^{BE} = \frac{1}{4}; \qquad val_{\mathrm{G}}^E = 0.$$

*Computing values.* Observe that each player has exactly four pure strategies. The set of pure strategies of Eve is the set $\Sigma_G^E = \{\sigma_{0,0}, \sigma_{1,0}, \sigma_{0,1}, \sigma_{1,1}\}$, where $\sigma_{i,j}$ is the strategy such that in the resulting play $t$ we have that $x_1(t) = i$ and $x_2(t) = j$. Similarly, the set of pure

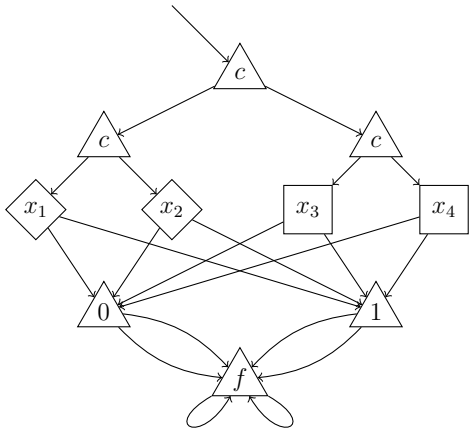



Figure 3.2 – A branching board that is not determined under behavioural strategies.

strategies of Adam is the set $\Sigma_G^A = \{\pi_{0,0}, \pi_{1,0}, \pi_{0,1}, \pi_{1,1}\}$, where $\pi_{i,j}$ is the strategy such that in the resulting play $t$ we have that $x_3(t) = i$ and $x_4(t) = j$. The set of possible outcomes is given in the following table.

| $val(\sigma, \pi)$ | $\sigma_{0,0}$ | $\sigma_{1,0}$ | $\sigma_{0,1}$ | $\sigma_{1,1}$ |
|---|---|---|---|---|
| $\pi_{0,0}$ | 1 | 0 | 0 | 0 |
| $\pi_{1,0}$ | 1 | 1 | 1 | 1 |
| $\pi_{0,1}$ | 1 | 1 | 1 | 1 |
| $\pi_{1,1}$ | 0 | 0 | 0 | 1 |

(3.8)

Since in every row (resp., column) in (3.8) there is at least one value 1 (resp., 0), we have that $val_G^A = 1$ (resp., $val_G^E = 0$).

For the mixed value of Eve, observe that, without loss of generality, Eve can restrict her set of mixed strategies to the set $dist(\{\sigma_{0,0}, \sigma_{1,1}\})$. Then, any mixed strategy $\sigma_m$ of Eve is a distribution such that $\sigma_m(\sigma_{0,0}) = 1 - \sigma_m(\sigma_{1,1}) = a$, for some $0 \leq a \leq 1$. Thus,

$$\begin{aligned} val_G^{ME} &= \sup_{\sigma_m \in \Sigma_G^{ME}} \inf_{\pi \in \Sigma_G^A} val_G(\sigma_m, \pi) \\ &= \sup_{\sigma_m \in dist(\{\sigma_{0,0}, \sigma_{1,1}\})} \inf_{\pi \in \Sigma_G^A} val_G(\sigma_m, \pi) \\ &= \sup_{0 \leq a \leq 1} \inf_{\pi \in \Sigma_G^A} a \cdot val_G(\sigma_{0,0}, \pi) + (1-a) \cdot val_G(\sigma_{1,1}, \pi) \\ &= \sup_{0 \leq a \leq 1} \min(\{a, 1, 1, 1-a\}) = \tfrac{1}{2}. \end{aligned}$$

Similarly, we show that $val_G^{MA} = \frac{1}{2}$.

Now we compute the behavioural values. The strategy $\sigma_{\alpha,\beta}$ is the behavioural strategy of Eve that chooses left child of the node labelled $x_1$ with probability $\alpha$ and chooses left child

of the node labelled $x_2$ with probability $\beta$. Strategies $\pi_{\alpha,\beta}$ of Adam, where $0 \leq \alpha, \beta \leq 1$, are defined *mutatis mutandis*. Note that those two definitions organically extend the definitions of the available pure strategies. Moreover, observe that any behavioural strategy of Eve (resp., of Adam) is of the form $\sigma_{\alpha,\beta}$ (resp., $\pi_{\alpha,\beta}$) where $0 \leq \alpha, \beta \leq 1$.

To compute $val_G^{BE}$ we use the following sequence of basic equalities.

$$\begin{aligned}
val_G^{BE} &= \sup_{\sigma_b \in \Sigma_G^{ME}} \inf_{\pi \in \Sigma_G^A} val_G(\sigma_b, \pi) \\
&= \sup_{0 \leq \alpha,\beta \leq 1} \inf_{\pi \in \Sigma_G^A} val_G(\sigma_{\alpha,\beta}, \pi) \\
&= \sup_{0 \leq \alpha,\beta \leq 1} \inf_{\pi \in \Sigma_G^A} \big(\alpha\beta \cdot val_G(\sigma_{0,0}, \pi) + \alpha(1-\beta) \cdot val_G(\sigma_{0,1}, \pi) + \\
&\qquad (1-\alpha)\beta \cdot val_G(\sigma_{1,0}, \pi) + (1-\alpha)(1-\beta) \cdot val_G(\sigma_{1,1}, \pi)\big) \\
&= \sup_{0 \leq \alpha,\beta \leq 1} \min\left(\{\alpha\beta,\ 1,\ 1,\ (1-\alpha)(1-\beta)\}\right) \\
&= \sup_{0 \leq \alpha,\beta \leq 1} \min\left(\{\alpha\beta, (1-\alpha)(1-\beta)\}\right)
\end{aligned}$$

Now, if $\alpha \leq 1-\beta$ then $\alpha\beta \leq (1-\alpha)(1-\beta)$ and

$$\sup_{\substack{0 \leq \alpha,\beta \leq 1 \\ \alpha \leq 1-\beta}} \min\left(\{\alpha\beta, (1-\alpha)(1-\beta)\}\right) = \sup_{\substack{0 \leq \alpha,\beta \leq 1 \\ \alpha \leq 1-\beta}} \alpha\beta \leq \sup_{\substack{0 \leq \alpha,\beta \leq 1 \\ \alpha \leq 1-\beta}} (1-\beta)\beta = \frac{1}{4}.$$

On the other hand, if $\alpha \geq 1-\beta$ then $\alpha\beta \geq (1-\alpha)(1-\beta)$ and

$$\sup_{\substack{0 \leq \alpha,\beta \leq 1 \\ \alpha \geq 1-\beta}} \min\left(\{\alpha\beta, (1-\alpha)(1-\beta)\}\right) = \sup_{\substack{0 \leq \alpha,\beta \leq 1 \\ \alpha \geq 1-\beta}} (1-\alpha)(1-\beta) \leq \sup_{\substack{0 \leq \alpha,\beta \leq 1 \\ \alpha \geq 1-\beta}} (1-\alpha)\alpha = \frac{1}{4}.$$

Hence, $val_G^{BE} \leq \frac{1}{4}$. To show that $val_G^{BE} = \frac{1}{4}$, we observe that

$$val_G(\sigma_{\frac{1}{2},\frac{1}{2}}) = \min(\{\frac{1}{2} \cdot \frac{1}{2}, (1-\frac{1}{2})(1-\frac{1}{2})\}) = \frac{1}{4}.$$

We show that $val_G^{BA} = \frac{3}{4}$ in the same way. □

**Regular branching games** The following theorem gives a large class of games, where the pay-off function $\Phi$ is a characteristic function of a regular set of trees $L$, i.e. $\Phi(t) = 1$ if $t \in L$ and $\Phi(t) = 0$ otherwise. In the case of a game G such that the pay-off function is a characteristic function of a set of trees, we say that the game G has $L$ *as a winning set* and we write $G = \langle \mathtt{B}, L \rangle$ instead of $G = \langle \mathtt{B}, \Phi \rangle$.

**Theorem 3.2.3** (Gogacz et al. [22])**.** *Every regular set $L$ of infinite trees is universally measurable, i.e. for every complete Borel measure $\mu$ on the set of trees, we know that $L$ is $\mu$-mea-*

*surable.*

We say that a branching game is a *regular branching game* if the pay-off function is a characteristic function of a regular set of trees.

## 3.3 The standard games collections

We now show how the games defined in Section 2.3 can be expressed in our framework. For more "standard" definitions of the below games see Section 2.3 on page 21.

**Positional strategies** In the following games the notions of pure strategies for the respective players are the same as in the case of general branching games. Additionally, we call a pure strategy $\tau$ of $P \in \{E, A\}$ *positional* if the decision made by $P$ depends only on the current vertex $v \in V$. In other words, for every two $u, w \in \mathit{dom}(\tau)$ such that $s_u(v_\mathrm{I}) = s_w(v_\mathrm{I}) \in V_P$, either $u_\mathtt{L}, w_\mathtt{L} \in \mathit{dom}(\tau)$ or $u_\mathtt{R}, w_\mathtt{R} \in \mathit{dom}(\tau)$. Thus, a positional strategy of $P$ can be represented as a function $\tau' \colon V_P \to \{\mathtt{L}, \mathtt{R}\}$.

### 3.3.1 Simple stochastic games

A *simple stochastic game* $G = \langle \mathtt{B}, \Phi \rangle$ is an $\{E, A, \mathcal{N}\}$-branching game where the labelling of the vertices is a function $\lambda \colon V \to \{0, 1\}$ and the distributions in *Nature*'s vertices given by $\eta$ are uniform distributions, i.e. for every $v \in V_\mathcal{N}$ we have that $\eta(v)(\mathtt{L}) = \eta(v)(\mathtt{R}) = \frac{1}{2}$. A play $t$ of a simple stochastic game has the shape of a unique infinite branch. Such a play is won by Eve (i.e. $\Phi(t) = 1$) if there is a node $u \in \mathit{nodes}(t)$ labelled 1. Otherwise, $\Phi(t) = 0$.

**Theorem 3.3.1** (Condon [11])**.** *Let $G$ be a simple stochastic game. Then, $G$ is determined under pure positional strategies.*

Note that any reachability game can be seen as an $\{E, A\}$-branching simple stochastic game.

### 3.3.2 Parity games

A *parity game* $G = \langle \mathtt{B}, \Phi \rangle$ is an $\{E, A\}$-branching game where the labelling of the vertices is a function $\lambda \colon V \to \{i, i+1, \ldots, j\} \subseteq \omega$. Those labels are called *priorities*. For technical purposes we associate with those priorities a ranking function $\alpha \colon V \to \{i, i+1, \ldots, j\} \subseteq \omega$ such that $\alpha(v) = \lambda(v)$. A play $t$ of a parity game has the shape of a unique infinite branch that corresponds to a sequence of vertices $v_\mathrm{I} v_1 v_2 \cdots$. Such a play is won by Eve (i.e. $\Phi(t) = 1$) if the limes inferior of the values $\alpha(v_i)$ is even. Otherwise, $\Phi(t) = 0$.

**Theorem 3.3.2** (Emerson and Jutla [16], Mostowski [38])**.** *Let $G$ be a parity game. Then, $G$ is determined under pure positional strategies.*

### 3.3.3 Games with $\omega$-regular winning sets

Those are the games on graphs with winning condition defined by a regular language $W$ of infinite words. An *$\omega$-regular game* $G = \langle \mathtt{B}, \Phi \rangle$ is an $\{E, A\}$-branching game where the pay-off function is the characteristic function of a special class of regular languages of trees. Every language from this class satisfies the following property "the only infinite path in the tree is a word belonging to the set $W$".

**Theorem 3.3.3** (Büchi and Landweber [6]). *Let $G$ be an $\omega$-regular game. Then, $G$ is determined under pure strategies.*

### 3.3.4 Gale-Stewart games

As mentioned in Section 2.3.4 on page 22, Gale-Stewart games can be seen as games on graphs with a winning set $W$ of infinite words. The only problem with that representation is that any vertex of the games on graphs representation of a Gale-Stewart game $G = \langle S, W \rangle$ can have an arbitrary number of successors.

To remedy that observe that if the alphabet $S$ is binary then every vertex has exactly two successors and a *Gale-Stewart* game $G$ can be seen as an $\{E, A\}$-branching game $G' = \langle \mathtt{B}, \Phi \rangle$ where the pay-off function is the characteristic function of the language of trees defined as: "the only infinite path in the tree is an infinite word belonging to the set $W$".

On the other hand, if the alphabet $S$ is not binary but finite, then a standard construction can make it binary. If $S = \{0\}$ is unary then simply add a new letter 1 and change the winning set $W$ by adding the condition that the player who first plays the letter 1 looses. If $S = \{0, 1, 2, 3, \ldots, n-1\}$ then change the alphabet to the binary alphabet $\{c, d\}$ and change the winning set $W$ to $f_S^{-1}(W)$, where $f_S \colon \{c, d\}^\omega \to W$ maps the plays over the binary alphabet to the plays over the original alphabet $S$ as follows. Let $p = e'_0 a'_0 \ldots e'_i a'_i \cdots$ be a play in the new game, then $p = w_1 v_1 w_2 v_2 \cdots w_i v_i \cdots$, for some words $w_i, v_i \in \{0, 1\}^{2n}$. Now $f_S$ on the play $p$ is defined as follows: $f_S(p) = e_0 a_0 \ldots e_i a_i \cdots$, where $e_i$ is the position of the first occurrence of the letter $d$ on even position in the word $w_i$ and $a_i$ is the position of the first occurrence of the letter $d$ on odd position in the word $v_i$. In other words, a single letter of the original play is encoded as a word of length $2|S|$; in every encoding of a players original letter, we ignore bits chosen by their opponent and signify the original letter by the first occurrence of letter $d$ in the encoding; the words encoding Adam's and Eve's original letters alternate.

**Theorem 3.3.4** (Martin [30]). *Let $G$ be a Gale-Stewart game. If the winning set is Borel, then $G$ is determined under pure strategies.*

### 3.3.5 Meta-parity games

A *meta-parity game* $G = \langle \mathtt{B}, \Phi \rangle$ is an $\{E, A, \mathcal{N}, \mathcal{B}\}$-branching game where the labelling of the vertices is a function $\lambda \colon V \to \{i, i+1, \ldots, j\} \times \{E, A\} \times \{\top, \bot\} \times \{\top, \bot\}$, for some $i \leq j < \omega$.

A play $t$ of a meta-parity game is interpreted as a parity game. The set of vertices of the parity games is the set $\mathit{nodes}(t) \cup \{\bot, \top\}$ with $\bot$ and $\top$ being auto-loss and auto-win, respectively. The transition relations in the nodes of the tree $t$ are induced by the child relation and the third and fourth coordinates of the labelling, if the respective children are missing. More precisely, for a node $u \in \mathit{nodes}(t)$ if the position $u\mathtt{L} \notin \mathit{nodes}(t)$, then $s_{\mathtt{L}}(u)$ is the third coordinate of $\lambda(u)$. Similarly, if the position $u\mathtt{R} \notin \mathit{nodes}(t)$, then $s_{\mathtt{R}}(u)$ is the fourth coordinate of $\lambda(u)$. The priorities for the nodes of the tree $t$ are given by the first coordinate of the labels, and the partition of the positions by the second coordinate. The vertices $\bot, \top$ satisfy $\alpha(\bot) = 1$, $\alpha(\top) = 0$, $s_{\mathtt{L}}(\bot) = s_{\mathtt{R}}(\bot) = \bot$, and $s_{\mathtt{L}}(\top) = s_{\mathtt{R}}(\top) = \top$. A play $t$ is won by Eve (i.e. $\Phi(t) = 1$) if she has a winning strategy in the resulting parity game. Otherwise, $\Phi(t) = 0$.

**Theorem 3.3.5** (Mio [33])**.** *Let $G$ be a stochastic meta-parity game. Then, $G$ is determined under pure strategies.*

### 3.3.6 Blackwell games

The encoding of Blackwell games was proposed by Mio in his thesis, see [33, Section 4.2, pages 138-148] for details. The construction can be easily adapted to express any Blackwell game with a regular winning set as a regular branching game.

**Theorem 3.3.6** (Martin [31])**.** *Let $G$ be a Blackwell game over a finite alphabet $\Gamma$. If the winning set is Borel, then $G$ is determined under mixed strategies.*

# Chapter 4

# Problems of interest

In this short chapter we define problems that will be discussed in the following chapters of this thesis. We focus on two groups of problems in general:

- the first group, called *classification problems*, asks to define families of boards and/or families of pay-off functions admitting "good" properties, like determinacy, existence of equilibria, existence of winning strategies, existence of optimal strategies, etc.;
- the second group, called *complexity problems*, asks about the computational complexity aspects of deciding the above properties and the algorithms to compute the values of branching games.

## 4.1 Classification problems

The classification problems are mostly purely theoretical problems. The solutions of such problems take the form of mathematical proofs that assure certain properties, or the form of counterexamples that invalidate such properties.

The main classification problem we consider in this thesis can be described as follows.

**Problem 4.1.1** (Determinacy)**.** *For which families of winning sets, regular branching games are determined under pure (mixed, behavioural) strategies?*

A collection of classic results concerning determinacy of non-branching games can be found in the previous chapter.

## 4.2 Computational complexity problems

The complexity problems come in two flavours. The first one consists in the *computational problems*, for which the expected answer is a number, e.g. compute the value of a game,

compute the number of structures satisfying a first-order sentence, or compute the measure of a set of infinite words. The other flavour of the complexity problems are the *decision problems*. Here, the expected output is binary: the answer is either *yes* or *no*.

Those two kinds are often strongly related: every computational problem can be made a decision problem by establishing a threshold.

The basic computational complexity problem considered in this thesis is the *value problem*. Let *val* be one of the partial values, i.e. $val \in \{val^A, val^{BA}, val^{MA}, val^{ME}, val^{BE}, val^E\}$.

**Problem 4.2.1** (Compute *val* problem)**.**

*Input:* *A regular finitary branching game.*
*Output:* *The value val.*

Note that the above problem, and all the below problems, is, in fact, a family of problems. A family parametrised by the partial value that we are asked to compute and by the encodings of the branching games, pay-off functions, and thresholds.

With the value *val* problem we associate the appropriate decision problem.

**Problem 4.2.2** (Threshold for *val* problem)**.**

*Input:* *A regular finitary branching game and a rational number c.*
*Output:* *Does* val $\geq c$*?*

We also consider its simpler version, where the threshold is $\frac{1}{2}$ and the inequality is strict.

**Problem 4.2.3** (Simple threshold for *val* problem)**.**

*Input:* *A regular finitary branching game.*
*Output:* *Does* val $> \frac{1}{2}$*?*

**Problem 4.2.4** (Determinacy problem)**.**

*Input:* *A regular finitary branching game.*
*Output:* *Is the game determined?*

Note that deciding determinacy is not harder than computing values of a game. Indeed, if we can compute the values of a game, we can decide whether that game is determined.

## 4.3 Known complexity results

Some of the results concerning classification problems are presented in the previous chapter in Section 3.3. Here, we list some important, and useful in the scope of this thesis, results

concerning the computational complexity problems and games on graphs. For an introduction to computational complexity see [42].

The following results will refer only to those games that are determined under pure strategies. For such games all partial values coincide, thus we will present the results only for the case of simple threshold for $val^E$ problem.

We start with reachability games, for which we can state the following.

**Theorem 4.3.1** (Folklore)**.** *Let $G$ be a reachability game. Then, simple threshold for $val^E$ problem is P-complete.*

The above theorem is folklore: the polynomial time algorithm simply saturates the set of vertices reachable from the initial vertex. The stochastic version of reachability games poses a bigger algorithmic challenge, as can be seen in the following theorem.

**Theorem 4.3.2** (Condon [11])**.** *Let $G$ be a simple stochastic game. Then, simple threshold for $val^E$ problem is in UP∩co-UP.*

In the case of simple stochastic games, one can associate certain values with the vertices of the board. The value associated with a vertex is the Eve's value of the game where that vertex is the initial vertex. Guessing those values allows to verify in polynomial time that they are indeed the game values and to solve the problem in the stated complexity, see [11]. The exact complexity is open and strongly connected to parity games, for details see e.g. [9].

**Theorem 4.3.3** (Jurdziński [23])**.** *Let $G$ be a parity game. Then, simple threshold for $val^E$ problem is in UP∩co-UP.*

Since parity-games are determined under positional strategies, the simplest algorithm guesses a special unique winning strategy and verifies it in polynomial time. More direct approach results in exponential time algorithms like the strategy improvement algorithm, see e.g. [23], or the fix-point iteration algorithm, see e.g. [5]. In recent works, an interesting line of quasi-polynomial time algorithms has been presented, see [8, 12, 28] for details.

Parity games are a special, but important, case of $\omega$-regular games: they can be seen as $\omega$-regular games with the pay-off function given by deterministic automata on words. Solving $\omega$-regular games has been widely studied, for a survey see e.g. [10].

In this thesis we are especially interested in automata based pay-off functions. For those, we know the following.

**Theorem 4.3.4.** *Let $G$ be an $\omega$-regular game with a winning set given by a non-deterministic word automaton. Then, simple threshold for $val^E$ problem is EXP-complete.*

The upper bound is a consequence of the fact that automata on words can be determinised, i.e. for a given non-deterministic automaton on words one can compute a deterministic automaton that recognises the same language. The construction can be done in exponential time, see e.g. [51] for details. The resulting $\omega$-regular game can be solved in exponential

time with respect to the size of the original game: the new game is, in essence, a parity game.

The exponential blow-up is unavoidable, and is, partially, the reason behind the lower bound: already the universality of a non-deterministic automaton is *EXP*-complete, see e.g. [54].

Since we will work with alternating automata, we need also to mention the following result.

**Theorem 4.3.5.** *Let $G$ be an $\omega$-regular game with a winning set given by an alternating automaton. Then, simple threshold for $val^E$ problem is 2-EXP-complete.*

As before, the upper bound is a consequence of the determinisation procedure, which is doubly exponential in the case of alternating automata on words. The lower bound follows from the fact that alternating automata are succinct. It can be inferred from e.g. [1].

As stated by the celebrated theorem of Rabin, any set of trees that is defined by a monadic second-order formula is regular, see e.g. [50]. Moreover, since the construction of the automaton recognising the language defined by the formula is effective, we have the following.

**Theorem 4.3.6.** *Let $G$ be an $\omega$-regular game with a winning set given by a monadic second-order formula. Then, simple threshold for $val^E$ problem is decidable.*

Lastly, we give an unpublished result by Mio.

**Remark 4.3.7** (Mio, personal communication)**.** *Let $G$ be a stochastic meta-parity game. Then, simple threshold for $val^E$ problem is decidable.*

The result is a consequence of Mio's work. In [33], Mio shows that the denotational and the game semantics for the probabilistic $\mu$-calculus coincide. In particular, he defines a translation that takes a formula of the probabilistic $\mu$-calculus and returns a meta-parity game such that the value of the formula is equal to the value of the game. Later, in collaboration with Michalewski, Mio proves that the uniform measure of a game automata definable language is computable, see [32] which builds on the previous collaboration of Mio with Simpson [36, 37].

To prove the computability Michalewski and Mio translate the automaton and the probability distribution into a *Markov branching play*, which is a play of a meta-parity game.[1] The Markov branching play is, then, translated into an appropriate system of (least and greatest) fixed-point equations, which is solved using *Tarski's quantifier elimination procedure* [52].

In personal communication, Mio stated that using the equivalence between the game and the denotational semantics, one can easily extend the above result to a result that translates a meta-parity game into an appropriate system of (least and greatest) fixed-point equations, which, again, can be solved using *Tarski's quantifier elimination procedure*.

[1] Equivalently, Markov branching plays can be seen as instances of meta-parity games in which neither Adam nor Eve owns any vertices.

# Chapter 5

# Pure branching games

In this chapter we consider the branching games with no probabilistic elements involved. In this set-up computing Eve's (resp., Adam's) pure value is equivalent to deciding whether there exists Eve's (resp., Adam's) winning strategy.

We show that for a finite game, the sets of winning strategies are regular and their finite representation, in the form of an alternating tree automaton, computable.

Hence, we conclude that

- the pure values are computable
- we can decide whether a pure branching game is determined.

Every branching game considered in this chapter will be a regular $\{E, A, \mathcal{B}\}$-branching game.

## 5.1 Winning strategies

Since there are no vertices belonging to *Nature*, there ultimately is a single strategy of *Nature*, namely the full binary tree $t_{\mathtt{B}}^{\lambda}$ that is the unravelling of the board. Let $\sigma$ be a pure strategy of Eve and $\pi$ be a pure strategy of Adam. Then, the measure induced by the strategies $\sigma, \pi$ and the board is concentrated on a single tree, denoted $\mathit{play}(\sigma, \pi) \stackrel{\text{def}}{=} \mathrm{eval}(\sigma, \pi, t_{\mathtt{B}}^{\lambda})$, and the value of that play belongs to the binary set $\{0, 1\}$. Moreover, $\mathit{val}_{\mathrm{G}}(\sigma, \pi) = 1$ if and only if $\mathit{play}(\sigma, \pi) \in L$.

This observation allows us to give an alternative way to define pure determinacy. We say that a strategy $\sigma \in \Sigma_G^E$ of Eve is *winning* if for every strategy $\pi$ of Adam we have that $\mathrm{eval}(\sigma, \pi, t_{\mathtt{B}}^{\lambda}) \in L$. Dually, a strategy $\pi \in \Sigma_G^A$ of Adam is *winning* if for every strategy $\sigma$ of Eve we have that $\mathrm{eval}(\sigma, \pi, t_{\mathtt{B}}^{\lambda}) \notin L$. With the established notion of a winning strategy we can state.

**Fact 5.1.1.** *An $\{E, A, \mathcal{B}\}$-branching game $G$ is determined under pure strategies if and only if either Adam or Eve has a winning strategy. Moreover,*

$$val_G^E = 1 \iff \exists \sigma \in \Sigma_G^E . \forall \pi \in \Sigma_G^A .\ play(\sigma, \pi) \in L \tag{5.1}$$

*and*

$$val_G^A = 0 \iff \exists \pi \in \Sigma_G^A . \forall \sigma \in \Sigma_G^E .\ play(\sigma, \pi) \notin L, \tag{5.2}$$

*Proof.* We start with proving the equivalences (5.1) and (5.2). The equivalence (5.1) is inferred as follows.

$$\begin{array}{rcl} val_G^E = 1 & \overset{1}{\iff} & \sup_{\sigma \in \Sigma_{\mathcal{B}}^E} \inf_{\pi \in \Sigma_{\mathcal{B}}^A} val_G(\sigma, \pi) = 1 \\ & \overset{2}{\iff} & \exists \pi \in \Sigma_A^G . \forall \sigma \in \Sigma_G^E . val_G(\sigma, \pi) = 1 \\ & \overset{3}{\iff} & \exists \pi \in \Sigma_A^G . \forall \sigma \in \Sigma_G^E . play(\sigma, \pi) \in L. \end{array}$$

The first equivalence is the definition of the value $val^E$. The second one follows from the fact that the values $val_{\mathrm{G}}(\sigma, \pi)$ belong to a finite set, which implies that the infima and the suprema are always realised. The last one is simply the observation that $val_{\mathrm{G}}(\sigma, \pi) = 1$ if and only if $play(\sigma, \pi) \in L$. The equivalence (5.2) is inferred similarly.

To prove the main statement let us remind that, by the definition, a game $G$ is determined under pure strategies if $val_G^E = val_G^A$. Moreover, by Equation (3.6) on page 37 we know that $val_G^E \leq val_G^A$.

Therefore, if Eve has a winning strategy then

$$1 = val_G^E \leq val_G^A \leq 1$$

and the game is determined under pure strategies. Similarly, if Adam has a winning strategy then

$$0 = val_G^A \geq val_G^E \geq 0$$

and, again, the game is determined.

On the other hand, let us assume that a game $G$ is determined, i.e. $val_G^E = val_G^A$. Then, either

- $val_G^A = 0$ and by (5.2) Adam has a winning strategy,
- or $val_G^E = 1$ and by (5.1) Eve has a winning strategy.

This concludes the proof. □

The notion of a winning strategy is important and widely used in the non-stochastic setting. It introduces a combinatorial structure that allows us to forget about the underlying

continuous nature of the partial values of branching games and to adapt techniques from the discrete setting.

From Example 3.2.2 we know that not every $\{E, A, \mathcal{B}\}$-branching game is determined and, thus, not every branching game has a winning strategy. Therefore, there are at least two natural directions of research.

The first one asks whether there are natural families of winning sets for which every $\{E, A, \mathcal{B}\}$-branching game is determined under pure strategies.

The other one tasks us with developing algorithms that allow us to decide whether a given game is determined under pure strategies, to decide which of the players has a winning strategy, or to compute a winning strategy, if any exists.

## 5.2 Single player winning strategies

One of the simplest examples of branching games that are necessarily determined under pure strategies are *single player* branching games, i.e. the family of games where only one of the players, either Eve or Adam, is present. Nevertheless, even if a game is determined, deciding which player has a winning strategy can be computationally hard.

**Theorem 5.2.1.** *Let $G = \langle \mathtt{B}, L \rangle$ be a finite branching game. If the game is $\{E, \mathcal{B}\}$-branching, then deciding whether Eve has a winning strategy*

- *is in UP∩co-UP, if $L$ is given by a non-deterministic automaton,*
- *is EXP-complete, if $L$ is given by an alternating automaton.*

*If the game is $\{A, \mathcal{B}\}$-branching then deciding whether Eve has a winning strategy*

- *can be done in UP∩co-UP, if $L$ is given by a game automaton,*
- *is EXP-complete, if $L$ is given by a non-deterministic or an alternating automaton.*

To prove the theorem, we use the fact that both the tree language $\mathrm{plays}(\mathtt{B})$ and the winning set $L$ are regular.

*Proof.* We start with the following observation. If $G$ is $\{E, \mathcal{B}\}$-branching then Eve wins if and only if $\mathrm{plays}(\mathtt{B}) \cap L$ is not empty. Dually, if $G$ is $\{A, \mathcal{B}\}$-branching then Eve wins if and only if $\mathrm{plays}(\mathtt{B}) \subseteq L$.

Now, the upper bounds follow from this observation and known results about tree automata. Let $\mathcal{A}$ be the automaton recognising $L$, i.e. the input automaton. Let $\mathcal{B}$ be a non-deterministic automaton recognising the set $\mathrm{plays}(\mathtt{B})$. Since $\mathcal{B}$ can be computed in polynomial time, cf. Proposition 3.1.2, we have the following: we can check whether $\mathrm{L}(\mathcal{B}) \cap \mathrm{L}(\mathcal{A}) = \emptyset$ in exponential time in general and in *UP∩co-UP* if $\mathcal{A}$ is a non-deterministic automaton. Similarly, we can check whether $\mathrm{L}(\mathcal{B}) \subseteq \mathrm{L}(\mathcal{A})$ in exponential time in general and in *UP∩co-UP*

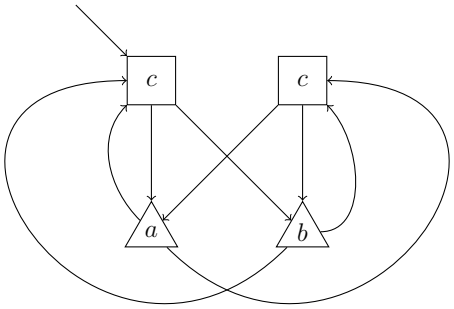



Figure 5.1 – The board used in the proof of Lemma 5.2.2, with $\mathcal{X} = A$.

if $\mathcal{A}$ is a game automaton. For the details concerning alternating automata see e.g. [54], for game-automata case see e.g. [15, 17].

All we need to conclude the proof is to show the lower bounds. For that, we formulate the following lemma.

**Lemma 5.2.2.** *Let $\mathcal{X} \in \{E, A\}$. There exists a logarithmic space reduction that inputs an alternating tree automaton $\mathcal{A}$ over the two letter alphabet $\Gamma = \{a, b\}$ and constructs a finite $\{\mathcal{X}, \mathcal{B}\}$-branching game $G = \langle \mathtt{B}, \mathrm{L}(\mathcal{C})\rangle$ with the winning set given by an alternating tree automaton $\mathcal{C}$ such that*

- $\mathrm{plays}(\mathtt{B}) \subseteq \mathrm{L}(\mathcal{C})$ *if and only if* $\mathcal{T}_\Gamma^\infty = \mathrm{L}(\mathcal{A})$,
- *and* $\mathrm{plays}(\mathtt{B}) \cap \mathrm{L}(\mathcal{C}) = \emptyset$ *if and only if* $\mathrm{L}(\mathcal{A}) = \emptyset$.

*Moreover, if $\mathcal{A}$ is a non-deterministic tree automaton, then so is $\mathcal{C}$.*

*Proof.* The board $\mathtt{B}$ does not depend on the automaton and is defined as in Figure 5.1. Formally, the board has four vertices, $V = \{0, 1, 2, 3\}$, and successors are defined as follows: $s_{\mathtt{R}}(0) = s_{\mathtt{R}}(1) = 2, s_{\mathtt{R}}(2) = s_{\mathtt{R}}(3) = 0$, $s_{\mathtt{L}}(0) = s_{\mathtt{L}}(1) = 3, s_{\mathtt{L}}(2) = s_{\mathtt{L}}(3) = 1$, and the initial vertex is $v_{\mathrm{I}} = 0$. The labelling is defined as $\lambda(2) = a, \lambda(3) = b, \lambda(0) = \lambda(1) = c$, and the partition as $\rho(0) = \rho(1) = A, \rho(2) = \rho(3) = \mathcal{X}$.

The automaton $\mathcal{C} = \langle Q', \Gamma', \delta', \alpha', q_{\mathrm{I}}'\rangle$ does depend on the automaton $\mathcal{A}$ and is constructed as follows. For a given automaton $\mathcal{A} = \langle Q, \Gamma, \delta, \alpha, q_{\mathrm{I}}\rangle$, the automaton $\mathcal{C}$ on odd depth (nodes labelled $a$ or $b$) behaves like $\mathcal{A}$ and on even depth (nodes labelled $c$) transits to the non-blank child without change of the current state. More formally, $\mathcal{C}$ is an NTA such that

- $\Gamma' = \Gamma \cup \{\flat\}$,
- $Q' = Q \times \{o, e\} \sqcup \{f, \top, \bot\}$,
- $q_{\mathrm{I}}' = \langle q_{\mathrm{I}}, e\rangle$,

- $\alpha'(q) = \begin{cases} \alpha(p), & \text{if } q = \langle p, l\rangle, \text{ where } l \in \{o, e\}; \\ 0, & \text{if } q \in \{f, \top\}; \\ 1, & \text{otherwise.} \end{cases}$

- $\delta'(q, x) = \begin{cases} \delta(p, x), & \text{if } q = \langle p, o\rangle, x \neq \flat \\ (\langle p, \mathtt{L}\rangle \wedge \langle f, \mathtt{R}\rangle) \vee (\langle f, \mathtt{L}\rangle \wedge \langle p, \mathtt{R}\rangle), & \text{if } q = \langle p, e\rangle, x \neq \flat \\ \langle \top, \mathtt{L}\rangle \wedge \langle \top, \mathtt{R}\rangle, & \text{if } q = f, x = \flat \\ \langle \bot, \mathtt{L}\rangle \wedge \langle \bot, \mathtt{R}\rangle, & \text{if } q = f, x \neq \flat \\ \langle q, \mathtt{L}\rangle \wedge \langle q, \mathtt{R}\rangle, & \text{otherwise.} \end{cases}$

It is easy to see that $\mathcal{A}$ accepts every full binary tree over the alphabet $\{a, b\}$ if and only if $\mathcal{C}$ accepts every play from the set plays$(\mathtt{B})$. Moreover, since we add only non-deterministic transitions, if the original automaton is a non-deterministic tree automaton then the resulting automaton is a non-deterministic tree automaton as well. □

Now we can conclude the proof of Theorem 5.2.1. To show *EXP*-hardness in the $\{A, \mathcal{B}\}$ case we will use the *EXP*-complete problem of the *universality of an NTA over a two letter alphabet*. Let $\mathcal{A}$ be an NTA as in Lemma 5.2.2, $\mathcal{X} = A$, and $G = \langle \mathtt{B}, \mathrm{L}(\mathcal{C})\rangle$ be the branching game produced by Lemma 5.2.2. Since the board has no existential positions, by the definition Eve has a winning strategy if and only if plays$(\mathtt{B}) \subseteq \mathrm{L}(\mathcal{C})$. In consequence, we have that plays$(\mathtt{B}) \subseteq \mathrm{L}(\mathcal{C})$ if and only if $\mathcal{T}_\Gamma^\infty \subseteq \mathrm{L}(\mathcal{A})$. This implies that Eve has a winning strategy if and only if $\mathcal{A}$ is universal which concludes the proof of this case.

The proof of the $\{E, \mathcal{B}\}$ case is similar. We will use the *EXP*-complete problem of the *emptiness of an ATA over a two letter alphabet*. As before, let $\mathcal{A}$ be an ATA as in Lemma 5.2.2 and $\mathcal{X} = E$. Again, let $G = \langle \mathtt{B}, \mathrm{L}(\mathcal{C})\rangle$ be the branching game constructed by Lemma 5.2.2. Since the board has no universal positions, by the definition Eve has a winning strategy if and only if the set plays$(\mathtt{B}) \cap \mathrm{L}(\mathcal{C})$ is not empty. In consequence, we have that plays$(\mathtt{B}) \cap \mathrm{L}(\mathcal{C}) \neq \varnothing$ if and only if $\mathrm{L}(\mathcal{A}) \neq \varnothing$. This implies that Eve has a winning strategy if and only if $\mathcal{A}$ accepts a tree.

Since the upper bounds were already shown in the first part of the proof, this concludes the proof of Theorem 5.2.1. □

## 5.3 Two player winning strategies

We now move to games where both Eve and Adam are allowed to move the token. Here, the determinacy is not guaranteed, not even in non-branching set-up – cf. e.g. [26], and the game values are even harder to compute. Nevertheless, the following lemma allows us to decide whether a given finite game is determined under pure strategies.

**Lemma 5.3.1.** *Let $G = \langle \mathtt{B}, \mathrm{L}(\mathcal{A})\rangle$ be an $\{E, A, \mathcal{B}\}$-branching finite game with a regular winning set given by an alternating tree automaton $\mathcal{A}$, and $P \in \{E, A\}$ denotes a player. Then, the set of winning strategies of the player $P$ is regular and recognisable by an alternating tree automaton $\mathcal{C}$ of size polynomial in the size of the board $\mathtt{B}$ and exponential in the size of the automaton $\mathcal{A}$. Moreover, if $\mathcal{A}$ is non-deterministic and $P = A$, then $\mathcal{C}$ is of polynomial size in the size of the board $\mathtt{B}$ and in the size of the automaton $\mathcal{A}$.*

*Proof.* Every strategy $t \in \mathcal{T}_\Gamma$ belonging to the set $\Sigma^E_{\mathtt{B}}$ of all strategies of Eve is a prefix $t$ of the tree $t^\lambda_{\mathtt{B}}$ such that $t$ is uniquely branching in Eve's nodes and fully branching in the remaining nodes. We claim that the language $L_E \subseteq \Sigma^E_{\mathtt{B}}$ of all winning strategies of Eve is regular and show how to construct the automaton $\mathcal{C}$.

Let $\overline{L_E} \subseteq \mathcal{T}_\Gamma$ be the complement of the language $L_E$. Let $L^E_{loosing}$ be the language of Eve's strategies that are not winning, i.e. $L^E_{loosing}$ consists of strategies $\sigma$ for which there is a play $p \in \mathrm{plays}(\mathtt{B})$ consistent with this tree, i.e. $p \sqsubseteq \sigma$, such that $p$ does not belong to the winning set $L$. Then, $\overline{L_E} = L^E_{loosing} \cup \overline{\Sigma^E_{\mathtt{B}}}$.

An automaton $\mathcal{A}_1$ recognising the language $\overline{\Sigma^E_{\mathtt{B}}}$ simply guesses where is the inconsistency between the tree and the board, and is of polynomial size with respect to the board. Checking that a tree is not a winning strategy is a bit more complicated. The automaton $\mathcal{A}_2$ recognising this language has to guess a play $p$ and check on-the-fly that this play does not belong to $L$. Since the check is performed on-the-fly, it can be done by a non-deterministic automaton $\mathcal{A}_3$ that recognises the complement of the language $L$. Indeed, given an NTA recognising $\overline{L}$ the automaton simply assumes that the tree is a proper strategy of Eve and decides on-the-fly which sub-trees Adam cuts and what is the run on such a pruned tree. It is not a problem that the automaton $\mathcal{A}_2$ may accept trees that are not Eve's valid strategies, in the end we will take the union of the languages $\mathrm{L}(\mathcal{A}_2)$ and $\mathrm{L}(\mathcal{A}_1)$.

The automaton $\mathcal{A}_3$ is exponential in the size of $\mathcal{A}$, therefore we can construct an ATA $\mathcal{B}'$ recognising the complement of the language $L_E$ that is of size exponential in the size of $\mathcal{A}$ and polynomial in the size of the board. Indeed, using basic operations on automata, we construct an automaton $\mathcal{B}'$ such that $\mathrm{L}(\mathcal{B}') = \overline{L_E} = \mathrm{L}(\mathcal{A}_1) \cup (\mathrm{L}(\mathcal{A}_2) \cap \overline{\mathrm{L}(\mathcal{A}_1)})$. To obtain the automaton $\mathcal{B}$, we simply take an ATA that recognises the complement of $\mathrm{L}(\mathcal{B}')$. Such an automaton is of the same size as $\mathcal{B}'$. This ends the proof of the first part of the lemma.

Now we will prove the second part. Let $\mathcal{A}$ be an NTA. Let $L^A_{loosing}$ be the language of Adam's strategies that are not winning. Then the complement of the language of Adam's winning strategies $\overline{L_A}$ is the union of $\overline{\Sigma^A_{\mathtt{B}}}$ and $L^A_{loosing}$. This time, the automaton recognising the language $L^A_{loosing}$ guesses a play $p$ and checks on-the-fly that this play *does* belong to $L$. Since we do not need the complement of the language $L$, we can simply use the automaton $\mathcal{A}$. Therefore, one can construct a non-deterministic automaton $\mathcal{B}''$, of size polynomial in the size of $\mathcal{A}$ and polynomial in the size of the board, that recognises the complement of the language $L_A$. Again, to obtain the automaton $\mathcal{B}$ one can simply take the alternating

automaton that recognises the complement of the language $\mathrm{L}(\mathcal{B}^{II})$. Such an automaton is of the same size as $\mathcal{B}^{II}$. □

The above lemma allows us to compute a finite representation of a winning strategy, or decide that one does not exists. In conjunction with Fact 5.1.1, this allows us to compute the pure partial values $val^E$ and $val^A$ of any $\{E, A, \mathcal{B}\}$-branching game with a regular winning set.

**Theorem 5.3.2.** *Let $G$ be an $\{E, A, \mathcal{B}\}$-branching finite game with a winning set given by an alternating tree automaton and $val \in \{val^E, val^A\}$. Then, simple threshold for val problem is 2-EXP-complete.*

*Proof.* In both cases the upper bound is a straightforward implication of Lemma 5.3.1. We will discuss the $val^E$ case for succinctness. We know that $val^E \in \{0, 1\}$ and, by Fact 5.1.1, we have that $val^E = 1$ if and only if Eve has a winning strategy. Since the non-emptiness of an ATA is *EXP*-complete, we infer the desired upper bound.

The lower bounds can be obtained from previously known results. Solving two-player games on graphs, i.e. $\{E, A\}$-branching games, with objectives defined by an LTL formulae is *2-EXP*-complete (cf. e.g. [1], [43]). Since an LTL formula can be translated into an ATA in polynomial time, cf. [56], the lower bound for simple threshold for $val^E$ problem immediately follows. To obtain the lower bound for simple threshold for $val^A$ problem we recall that such games are determined under pure strategies, thus $val^A = val^E$, cf. Martin's theorem on page 41. □

Note that, to witness determinacy under pure strategies of a branching game, we need to compare the pure values only. Thus, as a simple consequence we observe.

**Corollary 5.3.3.** *Let $G$ be an $\{E, A, \mathcal{B}\}$-branching finite game with a winning set given by an alternating tree automaton. Then, there is an algorithm that in doubly-exponential time decides whether the game $G$ is determined.*

We can refine Theorem 5.3.2 in the following way.

**Theorem 5.3.4.** *Let $G$ be an $\{E, A, \mathcal{B}\}$-branching game with a winning set given by a non-deterministic automaton. Then, simple threshold for $val^E$ problem is 2-EXP-complete and simple threshold for $val^A$ problem is EXP-complete.*

To prove the above theorem we will introduce an interesting property of branching games called *dealternation*. This property will be defined and proved in the following section, but before that, let us point out that the above theorem gives us a bit roundabout but interesting way to show that not every branching game with a regular winning set is determined under pure strategies. That is, we can infer the existence of such a branching game from Theorem 5.3.4 by a reasoning based purely on complexity theory.

**Corollary 5.3.5.** *There is a finite $\{E, A, \mathcal{B}\}$-branching game with a regular winning set that is not determined under pure strategies.*

Indeed, if branching games with regular winning sets were determined under pure strategies then for every $\{E, A, \mathcal{B}\}$-branching game with a regular winning set we would have that $val^E = val^A$. Since simple threshold for $val^A$ problem with a winning set defined by an NTA is *EXP*-complete, in exponential time we can check whether $val^A = 0$ or $val^A \neq 0$, solving the *2-EXP*-complete problem of deciding simple threshold for $val^E$ problem in exponential time. Since this is impossible, by the *time hierarchy theorem*, e.g. see [24], the corollary holds.

## 5.4 Dealternation

The entirety of this section is devoted to proving that regular branching games have a dealternation property. This property can be defined by the following lemma.

**Lemma 5.4.1.** *There exists a procedure such that given an $\{E, A, \mathcal{B}\}$-branching finite game* G *with a winning set given by an alternating tree automaton outputs an $\{E, A, \mathcal{B}\}$-branching finite game* $\mathrm{G}'$ *with a winning set given by a non-deterministic tree automaton, such that $val^E_{\mathrm{G}} = val^E_{\mathrm{G}'}$. This procedure runs in polynomial time.*

Intuitively, the dealternation property allows us to remove alternation from an automaton recognising the winning set and put it into the branching board, with only polynomial cost to the size of the game.

*Proof.* To prove the lemma, we will show the construction of the new game from the old one, and prove its correctness. We start with a simple observation about the nature of the regular branching games.

**Two-phase game** Consider a finite $\{E, A, \mathcal{B}\}$-branching game $\mathrm{G} = \langle \mathtt{B}, \mathrm{L}(\mathcal{A}) \rangle$ where the winning set $\mathrm{L}(\mathcal{A})$ is given by an alternating tree automaton $\mathcal{A}$. Simple threshold for $val^E$ problem for a game G asks whether

$$\exists \sigma \in \Sigma^E_G.\ \forall \pi \Sigma^A_G.\ play(\sigma, \pi) \in \mathrm{L}(\mathcal{A}). \tag{5.3}$$

Recall that the acceptance of a tree $t$ by an ATA $\mathcal{A}$ is defined in terms of a game, see Section 2.4 on page 24 for the definition of the acceptance game $\mathrm{G}(\mathcal{A}, t)$. Thus, the above equation can be seen as a two-phase game, where in the first phase the players play the original branching game and after that they play a parity game.

Since the automaton $\mathcal{A}$ is alternating and the parity game $H \stackrel{\text{def}}{=} \mathrm{G}(\mathcal{A}, play(\sigma, \pi))$ is determined under pure strategies, the question whether $play(\sigma, \pi) \in \mathrm{L}(\mathcal{A})$ can be written equivalently as:

- $\exists\bar{\sigma}\,\forall\bar{\pi}.\ \bar{\sigma}$ wins against $\bar{\pi}$ in $\mathrm{G}(\mathcal{A}, \mathit{play}(\sigma,\pi))$,
- $\forall\bar{\pi}\,\exists\bar{\sigma}.\ \bar{\sigma}$ wins against $\bar{\pi}$ in $\mathrm{G}(\mathcal{A}, \mathit{play}(\sigma,\pi))$,

where $\bar{\sigma}$ and $\bar{\pi}$ range over the adequate sets of pure strategies of the respective players in the game $\mathrm{G}(\mathcal{A}, \mathit{play}(\sigma,\pi))$.

Since parity games are positionally determined, in both cases we can restrict to positional pure strategies $\bar{\sigma}$ and $\bar{\pi}$. By incorporating the latter condition, Equation (5.3) transforms into:

$$\exists\sigma \in \Sigma_G^E.\ \forall\pi \in \Sigma_G^A.\ \forall\bar{\pi} \in \Sigma_H^A.\ \exists\bar{\sigma} \in \Sigma_H^E.\ \bar{\sigma} \text{ wins against } \bar{\pi} \text{ in } \mathrm{G}(\mathcal{A}, \mathit{play}(\sigma,\pi)). \tag{5.4}$$

**The construction** Now we will construct a board $\mathtt{B}'$ and a non-deterministic tree automaton $\mathcal{A}'$ such that a pure strategy of Adam over $\mathtt{B}'$ will encode both, a pure strategy $\pi$ over $\mathtt{B}$ and a positional strategy $\bar{\pi}$ in $\mathrm{G}(\mathcal{A}, \mathit{play}(\sigma,\pi))$. Intuitively, the non-deterministic automaton $\mathcal{A}'$ will guess a positional strategy $\bar{\sigma}$ of Eve in $\mathrm{G}(\mathcal{A}, \mathit{play}(\sigma,\pi))$. Thus, simple threshold for $\mathit{val}^E$ problem for the game $\mathrm{G}'$ will be equivalent to asking whether:

$$\exists\sigma \in \Sigma_{G'}^E.\ \forall(\pi,\bar{\pi}) \in \Sigma_{G'}^A.\ \mathit{play}(\sigma,(\pi,\bar{\pi})) \in \mathrm{L}(\mathcal{A}'). \tag{5.5}$$

Assume that the set of states of $\mathcal{A}$ is $Q = \{q_0, q_1, \ldots, q_n\}$, its initial state is $q_\mathrm{I}$ and $\Delta$ is the set of the all sub-formulae in the transitions of $\mathcal{A}$. The alphabet $\Gamma'$ of the new board $\mathtt{B}'$ will be the disjoint union of the original alphabet $\Gamma$, the set $Q \times \Gamma$, the set $\Delta$, and a special symbol $f$. Our aim is to replace each vertex $v \in V$ on the board $\mathtt{B}$ with $\lambda(v) = a$ by a gadget $\mathcal{G}(v)$ that simulates all the possible transitions of $\mathcal{A}$ over $a$.

First let us define inductively $\mathcal{F}(\delta)$ for $\delta \in \Delta$ as depicted in Figure 5.2 – the atomic transitions lead to trivial sub-games looping in a vertex labelled by $f$, the disjunctions are replaced by branching vertices, and the conjunctions are replaced by Adam's vertices.

**New board** Now, $\mathtt{B}'$ is obtained from $\mathtt{B}$ by performing the replacement depicted in Figure 5.3 – each vertex $v$ of $\mathtt{B}$ that was labelled by $a$ is replaced by a gadget listing all the states of $\mathcal{A}$ with all the transitions from these states over the letter $a$.

**New automaton** We will now describe the automaton $\mathcal{A}'$. Its aim is to simulate a play of $\mathrm{G}(\mathcal{A}, t)$ over the extended arena $\mathtt{B}'$. The only essential non-determinism of $\mathcal{A}'$ will be available when a letter of the form $\phi \vee \psi$ is read. In that case the automaton will guess a choice of Eve (from the strategy $\bar{\sigma}$) and follow the respective sub-tree generated by the sub-boards $\mathcal{F}(\phi)$ or $\mathcal{F}(\psi)$. After resolving the current transition $\delta(q,a)$ (represented as $\mathcal{F}(\delta(q,a))$) when the automaton finally reaches a vertex labelled by an atomic formula $\psi$ (e.g. $(q,d)$) it needs to pass this knowledge to the successive node denoted by the hexagon in Figure 5.3. This will be achieved by guessing in advance that the resolved atomic formula will be $\psi$.

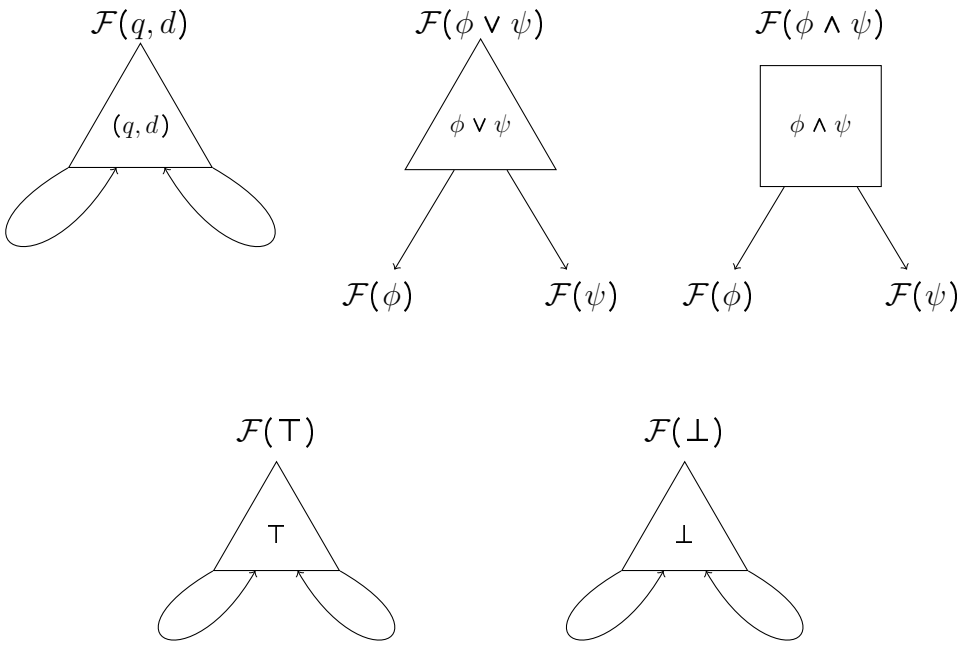



Figure 5.2 – The inductive construction of the sub-board $\mathcal{F}(\delta)$ for $\delta \in \Delta$.

Let the set of states of $\mathcal{A}'$ be $Q \cup Q \times \{\mathtt{L}, \mathtt{R}\} \cup \{\top, \bot\}$. The states from $Q$ will denote the fact that a transition of $\mathcal{A}$ has not been resolved yet, while the states from $Q \times \{\mathtt{L}, \mathtt{R}\} \cup \{\top, \bot\}$ will denote the fact that we already have played a finite game over the formulae in $\Delta$. The initial state of $\mathcal{A}'$ is $q_{\mathrm{I}}$. Let $m = \max_{q \in Q} \alpha(q)$ and let $\alpha'(q,d) = \alpha'(\top) = \alpha'(\bot) = m$. For $q \in Q$ let $\alpha'(q) = \alpha(q)$.

**A run description** Before we list the transitions of $\mathcal{A}'$ we describe the operation of $\mathcal{A}'$ informally:

- It enters a component $\mathcal{G}(v)$ as depicted in Figure 5.3 in a state $q$.
- It passes along the left-most branch of $\mathcal{G}(v)$ until a letter $(q,a)$ is reached.
- It guesses the atomic formula that will be reached when resolving the transition $\delta(q,a)$ (i.e. the sub-board $\mathcal{F}(\delta(q,a))$).
- It passes the guessed atomic formula along the left-most branch of $\mathcal{G}(v)$ until reaching the final node labelled by a letter $a \in \Gamma$.
- At the same time it passes the guessed atomic formula down the sub-tree of $\mathcal{F}(\delta(q,a))$.
- Since the conjunctive formulae of the form $\phi \wedge \psi$ were translated into Adam's positions, they are already resolved – i.e. the nodes of the tree labelled by them are uniquely branching.

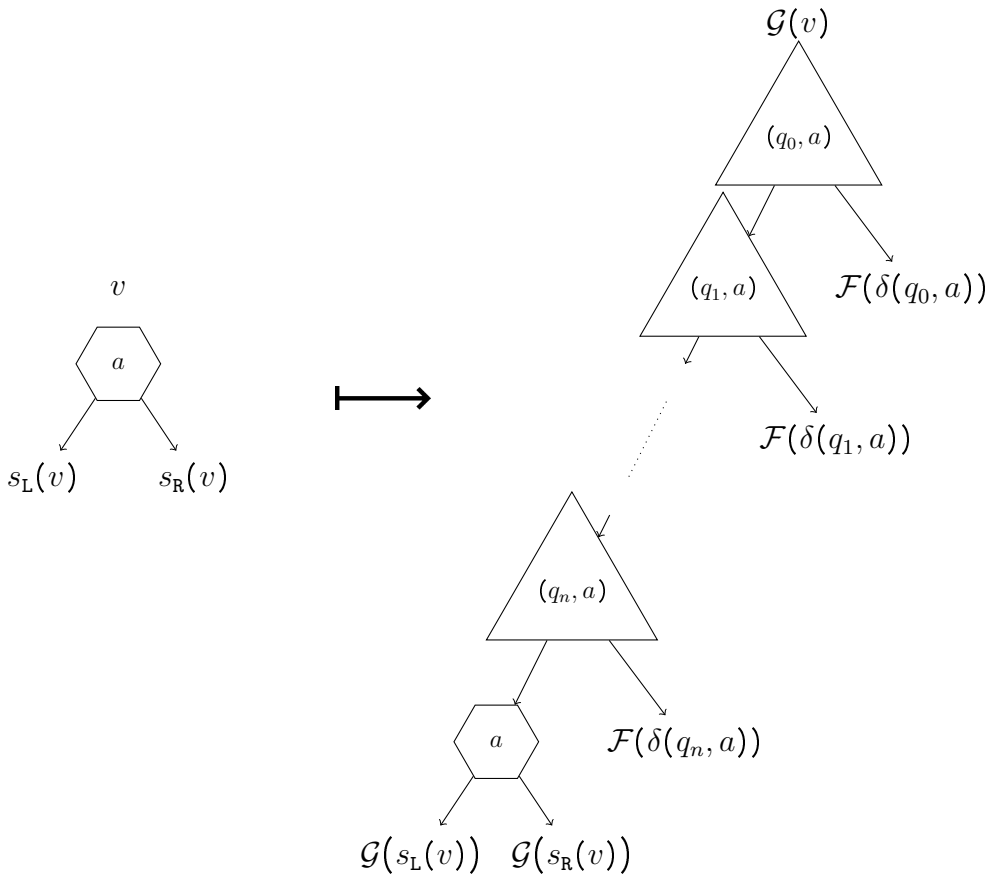



Figure 5.3 – The transformation on the board $\mathtt{B}$ to obtain $\mathtt{B}'$. The hexagon represents an arbitrary vertex $v$ of $\mathtt{B}$ that is labelled by $a \in \Gamma$. This vertex is replaced by a gadget $\mathcal{G}(v)$ in $\mathtt{B}'$, as depicted on the right-hand side of the figure. The gadget lists all the states $q_0, \ldots, q_n$ of $\mathcal{A}$ and for each state contains the sub-board $\mathcal{F}\big(\delta(q_i, a)\big)$ that represents the formula of the transition of $\mathcal{A}$ over $a$ from $q$. Finally, the left-most branch of the gadget $\mathcal{G}(v)$ reaches a copy of the vertex $v$ that leads to the gadgets corresponding to the successors of $v$ in $\mathtt{B}$.

- The disjunctive formulae of the form $\phi \vee \psi$ are not resolved yet and the automaton resolves them using non-determinism.
- When it reaches a node labelled by an atomic formula, it checks that the formula is the same as its state.

**Transitions** The automaton will have the following transitions:

- From a state $q$ over a letter $(q', a)$ with $q' \neq q$ it will deterministically take the transition $(q, \mathtt{L})$, following the left-most path in the gadget depicted in Figure 5.3.
- From a state $q$ over a letter $(q, a)$ it will non-deterministically take one of the transitions $\big(\theta, \mathtt{L}\big) \wedge \big(\theta, \mathtt{R}\big)$ for $\theta \in Q \times \{\mathtt{L}, \mathtt{R}\} \cup \{\top, \bot\}$. Such a transition corresponds to guessing that the formula $\delta(q, a)$ will be resolved to an atomic formula $\theta$ (the $\big(\theta, \mathtt{R}\big)$ part) and continuing the computation on the left-most path in the state $\theta$ (the $\big(\theta, \mathtt{L}\big)$ part).
- From a state $\theta \in Q \times \{\mathtt{L}, \mathtt{R}\} \cup \{\top, \bot\}$ over a letter $(q', a')$ it will deterministically take the transition $\big(\theta, \mathtt{L}\big)$, following the left-most path in the gadget depicted in Figure 5.3.
- From a state $\theta \in Q \times \{\mathtt{L}, \mathtt{R}\} \cup \{\top, \bot\}$ over a letter $\phi \vee \psi$ it will non-deterministically take one of the transitions $\big(\theta, d\big)$ for $d \in \{\mathtt{L}, \mathtt{R}\}$.
- From a state $\theta \in Q \times \{\mathtt{L}, \mathtt{R}\} \cup \{\top, \bot\}$ over a letter $\phi \wedge \psi$ it will non-deterministically take one of the transitions $\big(\theta, d\big)$ for $d \in \{\mathtt{L}, \mathtt{R}\}$ – the node labelled $\phi \wedge \psi$ has exactly one child in a play and we need to follow that successor.
- From a state $\theta \in Q \times \{\mathtt{L}, \mathtt{R}\} \cup \{\top, \bot\}$ over a letter $\theta' \in Q \times \{\mathtt{L}, \mathtt{R}\} \cup \{\top, \bot\}$ with $\theta \neq \theta'$ the automaton takes the transition to $\bot$.
- From a state $\theta \in Q \times \{\mathtt{L}, \mathtt{R}\} \cup \{\top, \bot\}$ over the letter $\theta$ the automaton takes the transition $\top$.
- From a state $(q, d) \in Q \times \{\mathtt{L}, \mathtt{R}\}$ over a letter from the original alphabet $a \in \Gamma$ the automaton deterministically takes the transition $(q, d)$, moving in the direction $d$ to the state $q \in Q$.
- From a state $\theta \in \{\top, \bot\}$ over a letter from the original alphabet $a \in \Gamma$ the automaton deterministically takes the transition $\theta$.
- For states and letters not listed above, the automaton takes the rejecting transition.

**Correctness** Let $\mathrm{G}' = \langle \mathtt{B}', \mathrm{L}(\mathcal{A}') \rangle$. Now we will prove the correctness of the construction of $\mathrm{G}'$, as expressed by the following claim.

**Claim 5.4.2.** *Eve has a winning strategy in the original branching game* G *if and only if Eve has a winning strategy in the game* $\mathrm{G}'$.

*Proof.* First, assume that $\sigma$ is a pure winning strategy of Eve in the original game G. Since Eve has no additional choices over the board $\mathtt{B}'$, the strategy $\sigma$ can be naturally interpreted as a pure strategy $\sigma'$ over the board $\mathtt{B}'$. Consider a pure strategy $\pi'$ of Adam over the board $\mathtt{B}'$. We will prove that $\mathit{play}(\sigma', \pi') \in \mathrm{L}(\mathcal{A}')$.

Notice that the strategy $\pi'$ consists of two parts:

- first part encodes a pure strategy $\pi$ of Adam over the board $\mathtt{B}$,
- the second, i.e. the choices made by $\pi'$ in the vertices labelled by the elements of $\Delta$, encodes a positional strategy $\bar{\pi}$ of Adam in the game $\mathrm{G}(\mathcal{A}, \mathit{play}(\sigma, \pi))$.

Since the strategy $\sigma$ is winning, we know that $\mathit{play}(\sigma, \pi) \in \mathrm{L}(\mathcal{A})$. Therefore, there exists a positional strategy $\bar{\sigma}$ of Eve that wins against $\bar{\pi}$ in the game $\mathrm{G}(\mathcal{A}, \mathit{play}(\sigma, \pi))$. We can use $\bar{\sigma}$ to define an accepting run of $\mathcal{A}'$ over the tree $\mathit{play}(\sigma', \pi')$. Therefore, we have proven that $\mathit{play}(\sigma', \pi') \in \mathrm{L}(\mathcal{A}')$.

Now consider a pure winning strategy $\sigma'$ of Eve in the new game $\mathrm{G}'$. Because of the lack of additional choices of Eve in $\mathtt{B}'$ and the fact that the choices of Eve over $\mathtt{B}'$ cannot depend on the choices made by Adam on the sub-boards $\mathcal{F}(\psi)$, we know that $\sigma'$ corresponds to a pure strategy $\sigma$ over the board $\mathtt{B}$. We will prove that $\sigma$ is winning in G. Consider any pure strategy $\pi$ of Adam over $\mathtt{B}$. Our aim is to prove that $\mathit{play}(\sigma, \pi) \in \mathrm{L}(\mathcal{A})$. Assume to the contrary, that $\mathit{play}(\sigma, \pi) \notin \mathrm{L}(\mathcal{A})$ and let $\bar{\pi}$ be a positional winning strategy of Adam in the game $\mathrm{G}(\mathcal{A}, \mathit{play}(\sigma, \pi))$. Similarly as above, the pair of strategies $\pi$ and $\bar{\pi}$ can be combined into one pure strategy $\pi'$ of Adam over $\mathtt{B}'$.

We will now prove that $\mathit{play}(\sigma', \pi') \notin \mathrm{L}(\mathcal{A}')$, contradicting the assumption that $\sigma'$ was winning in the game $\mathrm{G}'$. If it was the case that $\mathit{play}(\sigma', \pi') \in \mathrm{L}(\mathcal{A}')$ then a winning strategy of Eve in $\mathrm{G}(\mathcal{A}', \mathit{play}(\sigma', \pi'))$ would translate into a strategy $\bar{\sigma}$ in $\mathrm{G}(\mathcal{A}, \mathit{play}(\sigma, \pi))$ that would win against the positional strategy $\bar{\pi}$ of Adam. Thus, $\mathit{play}(\sigma', \pi') \notin \mathrm{L}(\mathcal{A}')$. This concludes the proof of the claim. □

By Claim 5.4.2 the construction is correct. This ends the proof of Lemma 5.4.1. □

With the dealternation procedure we can finally prove Theorem 5.3.4.

*Proof of Theorem 5.3.4.* The lower bound for Adam's value follows from Theorem 5.2.1. The lower bound for Eve's value follows from Lemma 5.4.1. Indeed, Lemma 5.4.1 provides us with a reduction from the *2-EXP*-complete problem from Theorem 5.3.2.

The upper bound for Adam's value is inferred from Lemma 5.3.1: we can check in exponential time whether the resulting polynomial-in-size alternating automaton recognises a non-empty set of trees. Finally, the upper bound for Eve's value follows directly form Theorem 5.3.2. □

We end this chapter with the observation that the dealternation is an inherent property of branching games.

**Claim 5.4.3.** *The dealternation cannot be performed in polynomial time without the presence of branching elements in the outputted game.*

*Proof.* If such an operation would be possible, then in polynomial time we could transform a game on graphs with a winning set defined by an ATA into a game on graphs with a winning set defined by an NTA. Since the problem of computing the value of the former is *2-EXP*-complete, and the problem of computing the value of the latter is *EXP*-complete, by the time hierarchy theorem, such a translation is impossible. □

# Chapter 6

# Stochastic branching games

In the previous chapter we have limited our interest to non-stochastic boards and pure strategies. This restriction on two basic aspects of the game, the shape of the board and the permitted strategies, allowed us to use a number of combinatorial and automata based techniques to identify a class of branching games with computable pure values.

As we will see in this chapter, the above class is in a sense maximal: restoring the full power of any of those aspects with no restriction on the regular winning sets yields undecidability.

In this chapter, we also discuss the determinacy of stochastic regular branching games. In particular, we show that those games do not have to be determined under mixed strategies, and that the class of branching games with open winning sets is determined under mixed strategies.

## 6.1 Regular objectives vs determinacy

As mentioned before, branching games with regular objectives are undetermined under pure strategies. In fact, this is true even for bounded depth families of trees or clopen sets, for such a game see e.g. Example 3.2.2. On the other hand, Nash Existence Theorem, see e.g. [41], implies that for those types of winning sets a mixed equilibrium exists.

**Proposition 6.1.1.** *Let $G$ be a branching game with a winning set $W$ that is clopen or there is a common bound on the height of trees in $W$. Then, $G$ is determined under mixed strategies.*

We do not include the proof of the above proposition as it will be a simple corollary of Theorem 6.1.8.

Intuitively, when a game allows one of the players to limit the number of valid strategies to a finite number, Nash Existence Theorem infers the mixed determinacy. On the other hand, if the number of valid strategies is infinite, a game is not always determined.

**Example 6.1.2** (Choose a Number)**.** Choose a Number *is a game in which two players, Eve and Adam, simultaneously and independently choose a natural number each. The number chosen by Eve is* $n \in \mathbb{N}$ *and the number chosen by Adam is* $m \in \mathbb{N}$*. The outcome of the game is either* 0*, Adam wins, or* 1*, Eve wins. Eve wins the game if* $n > m$*.*

It is folklore that Choose a Number is not determined under mixed strategies. Intuitively, each player prefers to play as large number as possible. Since the set of the allowed numbers is unbounded, for any player and their chosen mixed strategy $\xi_m$, there will be a number $N$ such that the probability that the number chosen by the strategy $\xi_m$ is greater than $N$ is arbitrarily close to 0. Hence, the opponent, knowing the strategy $\xi_m$, can always choose a number that is larger than $N$, and win with a probability arbitrarily close to 1.

Now we will encode Choose a Number in our framework and show that the resulting game is a branching game that is not determined under mixed strategies.

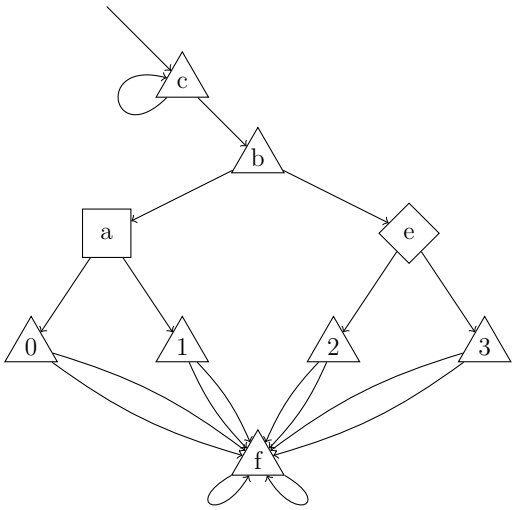



Figure 6.1 – A branching board that is not determined under mixed strategies.

**Example 6.1.3.** *Let* $\mathtt{B}_{CN}$*, the index CN stands for "Choose a Number", be an arena as depicted in Figure 6.1. For a tree* $t$*, let* $words(t)$ *be the set of all infinite words in* $t$ *starting from the root, i.e. the set* $\{w \in \Gamma^\omega \mid \exists u \in \{\mathtt{L},\mathtt{R}\}^\omega.\ \forall i \geq 0.\ w(i) = t\big(u(0)u(1)\cdots u(i-1)\big)\}$*.*

*Observe that, for every* $t \in \mathrm{plays}(\mathtt{B}_{CN})$ *we have that* $words(t) \subseteq c^\omega + c^*ba(0+1)f^\omega + c^*be(2+3)f^\omega$*. Moreover, for every tree* $t \in \mathrm{plays}(\mathtt{B}_{CN})$ *and* $n \geq 1$ *either* $c^nba0f^\omega$ *or* $c^nba1f^\omega$ *belongs to the set* $words(t)$*, but not both. Similarly, either* $c^nbe2f^\omega$ *or* $c^nbe3f^\omega$ *belongs to the set* $words(t)$*, but not both.*

*We define* $L_{CN}$ *as the language of trees* $t$ *such that the tree* $t$ *is a proper play, i.e.* $t \in \mathrm{plays}(\mathtt{B}_{CN})$*, and there is a number* $n$ *such that*

- $c^nbe2f^\omega \in words(t)$*,*
- *for every* $1 \leq k < n$ *we have that* $c^kbe3f^\omega \in words(t)$*, and*
- *if there is* $m$ *such that* $c^mba0f^\omega \in words(t)$ *then* $m < n$ *holds for the smallest such* $m$*.*

This game encodes Choose a Number in the following manner. In the game $\langle \mathtt{B}_{\mathrm{CN}}, L_{\mathrm{CN}} \rangle$ the board allows to simulate the choice of numbers. For a play $p \in \mathrm{plays}(\mathtt{B}_{\mathrm{CN}})$, the number chosen by Eve is the smallest number $n$ such that $c^nbe2f^\omega \in words(p)$ or $-1$ if such a number

does not exist. Similarly, the number chosen by Adam is the smallest number $m$ such that $c^m ba0f^\omega \in \mathit{words}(p)$ or $-1$ if such a number does not exist.

The language $L_{\mathrm{CN}}$ assures that Eve wins if and only if she chooses a number and her number is strictly greater than the number chosen by Adam.

**Lemma 6.1.4.** *The game $G = \langle \mathtt{B}_{CN}, L_{CN} \rangle$ is an $\{E, A, \mathcal{B}\}$-branching game with a regular winning set. Moreover, the mixed game values are:*

- $val_G^{ME} = 0$,
- $val_G^{MA} = 1$.

Before we prove the above lemma, let us see some intuition. We say that a player plays an even (an odd) number if the player moves a token into a vertex labelled by an even (an odd) number. Intuitively, we demand that both Adam and Eve will eventually play an even number, 2 for Eve and 0 for Adam. Moreover, whoever plays an even number at larger depth, wins the game. Therefore, the best strategy for either player is to choose an even number at larger depth than the opponent. Unfortunately, since the players cannot observe each other, postponing playing the even number may result in not playing it at all and, in consequence, forfeiting the game.

*Proof of Lemma 6.1.4.* Since it is easy to see that the language $L_{CN}$ is regular, we will focus our attention on computing the mixed values.

Let us fix a mixed strategy $\sigma_m \in \Sigma^{ME}_{\mathtt{B}_{\mathrm{CN}}}$ and a number $\varepsilon > 0$. We will show that there is a pure strategy $\pi \in \Sigma^A$ such that $val_G(\sigma_m, \pi) < \varepsilon$. This will immediately imply that $val_G^{ME} = 0$.

Let $E_n \subseteq \mathrm{plays}(\mathtt{B}_{\mathrm{CN}})$ be the set of plays where Eve plays an even number before the depth $n$, i.e. $E_n = \{t \in \mathrm{plays}(\mathtt{B}_{\mathrm{CN}}) \mid \text{there is } n' < n \text{ such that } c^{n'} be2f^\omega \in \mathit{words}(t)\}$.

It is easy to see that $E_n$ is a monotone sequence, i.e. $E_n \subseteq E_{n+1}$, and that the set $E = \bigcup E_n$ is the set of trees where "Eve eventually plays an even number". In consequence, we have that $L_{NC} \subseteq E$.

The family of sets $\{E_n\}_{n\in\mathbb{N}}$ has a special property: for a fixed mixed strategy of Eve $\sigma_m$ the probability that the resulting play belongs to $E_n$ does not depend on the strategy of Adam. More formally, let $\mathtt{B}'$ be a branching board, $L'$ be a regular set of trees, and $\sigma'_m \in \Sigma^{ME}_{\mathtt{B}'}, \pi'_m \in \Sigma^{MA}_{\mathtt{B}'}$, then by $\mu^{\sigma'_m, \pi'_m}(L')$ we denote the value $\mu^{\sigma'_m, \pi'_m}(L') \stackrel{\mathrm{def}}{=} val_{\langle \mathtt{B}', L' \rangle}(\sigma'_m, \pi'_m)$. Now, the special property can be expressed as follows. For every $n$ and any two strategies of Adam $\pi'_m, \pi''_m \in \Sigma^{MA}_{\mathtt{B}_{\mathrm{CN}}}$ we have that

$$\mu^{\sigma_m, \pi'_m}(E_n) = \mu^{\sigma_m, \pi''_m}(E_n). \tag{6.1}$$

Thus, the value $p_n \stackrel{\text{def}}{=} \mu^{\sigma_m,\pi_m}(E_n)$ is well defined for every $n$ and does not depend on the choice of $\pi_m$. Since the sequence $E_n$ is ascending, the sequence $p_n$ converges to $p = \mu^{\sigma_m,\pi_m}(E)$. Intuitively, $p$ is the probability that "Eve eventually plays an even number".

Now, we can define the pure strategy $\pi \in \Sigma^A_{\mathtt{B}_{\mathrm{CN}}}$ of Adam. Since $p_n$ converges to $p$, there is a number $N \geq 1$ such that for every $k \geq N$ we have $p - p_k < \varepsilon$. Let $\pi \in \Sigma^A_{\mathtt{B}_{\mathrm{CN}}}$ be a strategy such that $\pi(\mathtt{L}^N\mathtt{RLL}) = 0$ and for every $1 \leq k < N$ we have that $\pi(\mathtt{L}^k\mathtt{RLR}) = 1$, i.e. "Adam plays an even number at the depth $N$".

To end the proof it is enough to observe that

$$\begin{aligned} val_G(\sigma_m,\pi) &\stackrel{1}{=} \mu^{\sigma_m,\pi}(L_{CN}) \\ &\stackrel{2}{=} \mu^{\sigma_m,\pi}(E_N \cap L_{CN}) + \mu^{\sigma_m,\pi}((E \setminus E_N) \cap L_{CN}) \\ &\stackrel{3}{=} \mu^{\sigma_m,\pi}((E \setminus E_N) \cap L_{CN}) \\ &\stackrel{4}{\leq} \mu^{\sigma_m,\pi}(E \setminus E_N) \stackrel{5}{=} p - p_N \stackrel{6}{<} \varepsilon. \end{aligned}$$

The first equality holds from the definition, the second from the fact that $L_{CN} \subseteq E$. The third from the fact that "Adam plays an even number at the depth $N$", i.e. we have that $\mu^{\sigma_m,\pi}(E_N \cap L_{CN}) = 0$. The fourth inequality follows from the fact we increase the set while the fifth from the fact that $E_N \subseteq E$. Finally, the last inequality stems from the choice of $N$.

Since for every strategy $\sigma_m$ we can find $\pi$ such that the value $val_G(\sigma_m,\pi)$ is arbitrarily close to 0, we infer that $val^{ME}_G = 0$.

The argument implying that $val^{MA}_{\mathtt{B}_{\mathrm{CN}}} = 1$ is analogous. Given an arbitrary mixed strategy $\pi_m$ of Adam and a positive number $\varepsilon$, we construct a pure strategy $\sigma$ of Eve such that $val_G(\sigma,\pi_m) > 1 - \varepsilon$. To construct such a strategy, we define a family of sets $A_n$ where $A_n$ is the set of plays where for some $1 \leq k \leq n$ "Adam plays an even number at the depth $k$". When the family $A_n$ is chosen, the rest of the proof follows the same path. We take as $p_n$ the measure, defined as in Equation (6.1), of the set $A_n$ and then choose $N$ sufficiently big to approximate the limit measure $p$. □

From the above lemma we get immediately.

**Proposition 6.1.5.** *The game $\langle \mathtt{B}_{CN}, L_{CN}\rangle$ is not determined under mixed strategies.*

A closer inspection of the winning set reveals that it can be expressed as a difference of two open sets. This allows us to claim the following.

**Theorem 6.1.6.** *There is an $\{E, A, \mathcal{B}\}$-branching game with a winning set being a difference of two open sets that is not determined under mixed strategies.*

*Proof.* The set $L_{\mathrm{CN}}$ is a difference of two open sets $L_1$ and $L_2$. The set $L_1$ is the set of trees where "Eve plays an even number". That is, $L_1 = \bigcup_{i\geq 1} \mathbb{B}_{t_i}$, where $t_i$ is the greatest common prefix of the plays $t$ for which $c^i be 2 f^\omega \in words(t)$. The set $L_2$ is the union of the

complement of the set of plays plays($\mathtt{B}_{\mathrm{CN}}$) and the set $L_A$ of trees in which Adam plays his first even number after Eve has already played an even number. That is, $L_A \stackrel{\text{def}}{=} \bigcup_{j \geq i \geq 0} \mathbb{B}_{t_{i,j}}$ where every $t_{i,j}$ is the greatest common prefix of the plays $t$ for which "Adam plays an even number at the depth $j$", i.e. $c^j ba0f^\omega \in \mathit{words}(t)$, "Eve plays an even number at the depth $i$", i.e. $c^i be2f^\omega \in \mathit{words}(t)$, and for all $1 \leq k < j$ "Adam does not play an even number at the depth $k$", i.e. $c^k ba1f^\omega \in \mathit{words}(t)$.

$L_1$ is open because it is a union of sets of trees having a common finite prefix. $L_2$ is open because it is a union of a complement of a closed set, see Proposition 3.1.2 for details, and of a union of sets of trees having a common finite prefix. □

As an interesting corollary we notice that branching games cannot be simulated by Blackwell games.

**Corollary 6.1.7.** *There is no function $f\colon \Gamma^* \to \Gamma^*$ that takes a representation of a branching game $G$ with a regular winning set and produces a representation of a Blackwell game $G'$ over a finite set with a Borel winning set, both encoded as finite words over an alphabet $\Gamma$, such that for every partial value $val \in \{val^{BA}, val^{MA}, val^{ME}, val^{BE}\}$ we have that $val_G = val_{G'}$.*

*Proof.* If such a function would exist, it would preserve the determinacy, i.e. the image of an undetermined branching game would be an undetemined Blackwell game. Martin's determinacy for Blackwell games theorem, see [31], states that Blackwell games with Borel winning sets are determined under mixed strategies. However, by Theorem 6.1.6, we know that there are branching games with Borel winning sets that are not determined under mixed strategies. Therefore, there cannot exist such a function $f$. □

Theorem 6.1.6 states that any branching game with a winning set being a difference of two open sets is not necessarily determined under mixed strategies. We cannot provide similar examples with topologically simpler winning sets, because branching games with open winning sets are determined under mixed strategies, as expressed in the following theorem.

**Theorem 6.1.8.** *Branching games with open (resp., closed) winning sets are determined under mixed strategies.*

Note that the above theorem does not place many restrictions on the structure of the game. It does not assume that the winning set is regular nor that the board is finite. All it requires is that the board is finitely branching and that the pay-off function is semi-continuous. When those requirements are met, the theorem follows immediately from Glicksberg's minimax theorem.

**Theorem 6.1.9** (Glicksberg, [21])**.** *Let $X, Y$ be compact metric spaces and $f\colon X \times Y \to [0,1]$ be a lower (resp., an upper) semi-continuous function. Then, the following equality holds*

$$\sup_{\sigma \in dist(X)} \inf_{\pi \in dist(Y)} \int_{X \times Y} f \, \mathrm{d}\sigma \mathrm{d}\pi = \inf_{\pi \in dist(Y)} \sup_{\sigma \in dist(X)} \int_{X \times Y} f \, \mathrm{d}\sigma \mathrm{d}\pi \tag{6.2}$$

*Proof of Theorem 6.1.8.* Let $G = \langle \mathtt{B}, L \rangle$ be a branching game with an open winning set $L$. Let $X = \Sigma^E_{\mathtt{B}}, Y = \Sigma^A_{\mathtt{B}}$, and $f(\sigma, \pi) = \int_{\Sigma^{\mathcal{N}}_{\mathtt{B}}} \chi_L(\mathrm{eval}_{\mathtt{B}}(\sigma, \pi, \eta)) \, \mathrm{d}\eta^*_{\mathtt{B}}(\eta)$.

The sets $\Sigma^E_{\mathtt{B}}, \Sigma^A_{\mathtt{B}}$ are compact metric spaces, they are closed subsets of the compact set of all trees $\mathcal{T}_\Gamma$. By Corollary 17.21 in [25] on page 112, the function

$$\Sigma^{ME}_{\mathtt{B}} \times \Sigma^{MA}_{\mathtt{B}} \times \Sigma^{M\mathcal{N}}_{\mathtt{B}} \ni \langle \sigma_m, \pi_m, \eta_m \rangle \mapsto \mathrm{eval}_{\mathtt{B}} \#(\sigma_m \times \pi_m \times \eta_m)(L) \in \mathbb{R}.$$

is lower semi-continuous. Thus, by fixing $\eta_m$ as $\eta^*_{\mathtt{B}}$, we obtain that lower semi-continuous is the function

$$\Sigma^{ME}_{\mathtt{B}} \times \Sigma^{MA}_{\mathtt{B}} \ni \langle \sigma_m, \pi_m \rangle \mapsto \mathrm{eval}_{\mathtt{B}} \#(\sigma_m \times \pi_m \times \eta^*_{\mathtt{B}})(L) \in \mathbb{R}.$$

Now, as the identity injection $i \colon \Sigma^E_{\mathtt{B}} \times \Sigma^A_{\mathtt{B}} \to \Sigma^{ME}_{\mathtt{B}} \times \Sigma^{MA}_{\mathtt{B}}$ is continuous, the following function is also lower semi-continuous

$$\Sigma^E_{\mathtt{B}} \times \Sigma^A_{\mathtt{B}} \ni \langle \sigma, \pi \rangle \mapsto \mathrm{eval}_{\mathtt{B}} \#(\sigma \times \pi \times \eta^*_{\mathtt{B}})(L) = \int_{\Sigma^{\mathcal{N}}_G} \chi_L(\mathrm{eval}_{\mathtt{B}}(\sigma, \pi, \eta)) \, \mathrm{d}\eta^*_{\mathtt{B}}(\eta) = f(\sigma, \pi).$$

Hence, the chosen sets $X, Y$ and the function $f$ satisfy the assumptions of Glicksberg's minimax theorem. In consequence, Equation (6.2), describing the mixed determinacy holds. □

## 6.2 Values of stochastic regular branching games

Now we discuss the problem of computing the values of stochastic branching games. In contrary to the complexity results presented in the previous chapter, even in the single-player case the values of stochastic branching games with regular winning sets are uncomputable.

**Theorem 6.2.1.** *Simple threshold for val problem of a regular branching game $G = \langle \mathtt{B}, L \rangle$ is undecidable for every partial value $val \in \{val^A, val^{BA}, val^{MA}, val^{ME}, val^{BE}, val^E\}$. The problem is undecidable even for a fixed single player simple finitary board, i.e. the board $\mathtt{B}$ is a simple finitary $\{P, \mathcal{B}, \mathcal{N}\}$-branching board where $P \in \{E, A\}$.*

To prove the undecidability we will reduce the following undecidable problem.

**Problem 6.2.2** (Word problem of VSNA)**.**

*Input:* *A very simple non-deterministic automaton $\mathcal{A}$ on finite words over the alphabet $\{a, b\}$.*

*Output:* *Does there exist a finite word such that more than half of the runs of $\mathcal{A}$ on this word is accepting?*

A very simple non-deterministic automaton is an NFA such that for every state $q \in Q$ and every letter $a \in \Gamma$ the transition function leads to a disjunction of two atomic formulae, i.e. $\delta(q, a) = \langle q_0, \mathtt{L}\rangle \vee \langle q_1, \mathtt{L}\rangle$, where $q_0, q_1 \in Q$ and $q_0 \neq q_1$.

The undecidability of the above problem is a refinement of Corollary 1 in [20]. In [20] the corollary does not assume that the automaton is very simple, but it is clear that the proof can be easily improved (by cloning necessary states) to show the required refinement.

*Proof of Theorem 6.2.1.* By Lemma 3.2.1, a $\{P, \mathcal{N}, \mathcal{B}\}$-branching game is determined under pure strategies. Thus, all the six partial values are the same for such games. Hence, without loss of generality we can assume that $P = E$ and $val = val^E$.

To prove the theorem, we will reduce an instance of word problem of VSNA to an instance of simple threshold for $val^E$ problem.

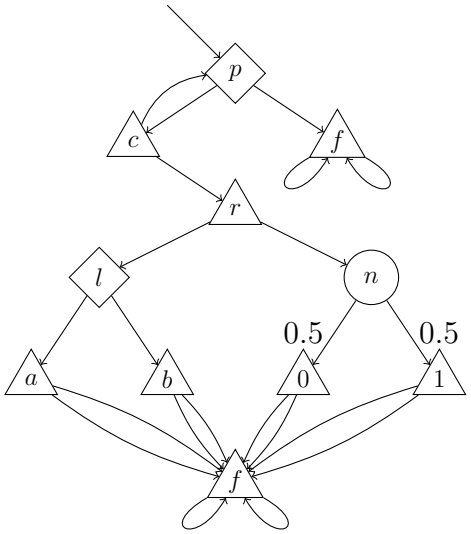
Let $\mathcal{A}$ be a very simple non-deterministic automaton. The board $\mathtt{B}$ does not depend on the automaton $\mathcal{A}$ and is depicted in Figure 6.2a.

A play $t \in \text{plays}(\mathtt{B})$, which can be seen in Figure 6.2b, on this board consists of a sequence of decisions made by Eve, whether to move from the vertex labelled $l$ to the vertex labelled $a$ or to the vertex labelled $b$. At every moment Eve can stop this sequence by choosing the right successor of the vertex labelled $p$. For every choice of $a$ or $b$ by Eve, *Nature* simultaneously and independently chooses a number 0 or 1, by choosing the appropriate vertex. Thus, any play $t \in \text{plays}(\mathtt{B})$ results in two finite sequences of the same length, or in two infinite sequences: $l_0, l_1, \ldots$ with $l_i \in \{a, b\}$ and $n_0, n_1, \ldots$ with $n_i \in \{0, 1\}$. We will call those sequences $t_E$ and $t_{\mathcal{N}}$, respectively.

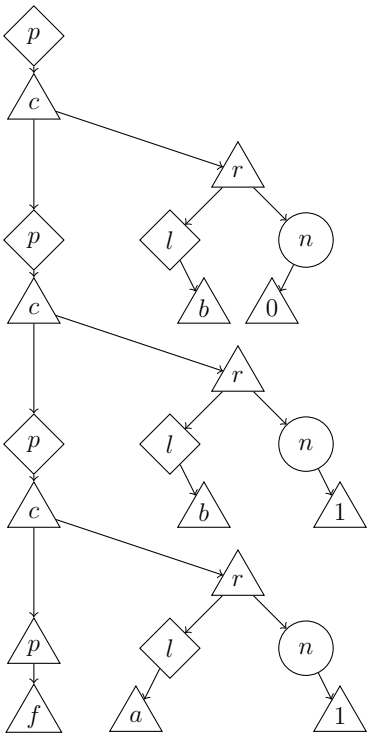
The reduction of word problem for VSNA to simple threshold for $val^E$ problem is defined as follows. Let $\mathcal{A}$ be a very simple NFA and let us assume that the two transitions over a letter $l \in \{a, b\}$ from a state $q \in Q$ lead to the states $\delta_0(q, l)$ and $\delta_1(q, l)$. Now, we will show how to construct an $\{E, \mathcal{N}, \mathcal{B}\}$-branching game $G = \langle \mathtt{B}, L\rangle$ with a regular winning set $L$ such that $val^E_G > \frac{1}{2}$ if and only if there exists a finite word such that more than half of the runs of $\mathcal{A}$ on this word is accepting.

Let $t \in \text{plays}(\mathtt{B})$ have a finite sequence $t_E$. Then, the two sequences $t_E$ and $t_{\mathcal{N}}$ allow us to naturally define a sequence $t_\rho \in Q^*$ that follows the respective transitions of $\mathcal{A}$ over the word $t_E$: $t_\rho(0) = q_\mathrm{I}$ and $t_\rho(i + 1) = \delta_{t_{\mathcal{N}}(i)}(t_\rho(i), t_E(i))$.

Figure 6.2 – Boards used in the undecidability proofs.




(a) A branching board used in the proof of Theorem 6.2.1.




(b) A play $t$ representing a run of a very simple non-deterministic automaton. The play encodes the word $t_E$ = *bba* and the sequence of bits $t_{\mathcal{N}}$ = 011. The non-essential nodes labelled $f$ are omitted.

The winning set is defined in the following way.

$$L \stackrel{\text{def}}{=} \{t \in \text{plays}(\mathtt{B}) \mid \text{the sequence } t_E \text{ is finite and } t_\rho \text{ is an accepting run on } t_E\}. \qquad (6.3)$$

It is easy to see that the winning set $L$ can be represented as a regular language of trees. Indeed, the appropriate automaton has to assert two facts: that the tree is a proper play and that the combined choices of Eve and *Nature* encode an accepting run. To check the latter, the automaton simulates the original automaton: it chooses letters according to Eve's choices and transitions according to *Nature*'s choices.

To conclude the proof we need to show the correctness of the reduction, i.e. we need to show that $val_G^E > \frac{1}{2}$ if and only if there exists a finite word $w \in \{a,b\}^*$ such that more than half of the runs of $\mathcal{A}$ on this word is accepting. By the definition, $val_G^E > \frac{1}{2}$ if and only if there exists a pure strategy $\sigma \in \Sigma_G^E$ of Eve such that $val_G(\sigma) > \frac{1}{2}$. Now, a pure strategy of Eve in the game G either never moves from the vertex labelled $p$ to the vertex labelled $f$ (in that case its value is 0) or it corresponds to a finite word $w \in \{a,b\}^*$. The value of such a strategy $\sigma$ is the probability that the choices of *Nature* represent an accepting run of $\mathcal{A}$ over the word $w$. Since every such run is equiprobable, $val_{\mathrm{G}}^E(\sigma) > \frac{1}{2}$ if and only if more than half of the runs of $\mathcal{A}$ on $w$ is accepting. This concludes the correctness of the reduction and the proof of Theorem 6.2.1. □

## 6.3 Derandomisation

We can strengthen Theorem 6.2.1 by removing the need of *Nature*'s vertices in the game. This can be obtained by *derandomisation*, i.e. by a procedure that removes stochastic positions from the board. Such a procedure was already introduced by Mio in his thesis, for details see [33, Section 4.4]. Our construction improves on Mio's in two aspects. First, the pay-off function remains a characteristic function of a regular set. Second, our construction is effective. Unfortunately, those improvements come at a price. We need to slightly modify the structure of the board, in particular both Adam's and Eve's vertices will be present after the derandomisation, and we may not preserve the pure values. The exact statement of the result is formulated in Theorem 6.3.5.

We start the description of the derandomisation with the standard construction allowing us to encode arbitrary rational numbers as branching games. To be more precise, the number will be encoded as a Markov chain.

**Lemma 6.3.1.** *There exists a polynomial space procedure that inputs a rational number* $r = \frac{x}{y} \in [0,1]$ *where* $x$ *and* $y$ *are represented in binary and outputs a simple finitary* $\{\mathcal{N}\}$*-branching game* $G = \langle \mathtt{B}_r, \mathrm{L}(\mathcal{A}_r)\rangle$ *with a regular winning set given by an NTA* $\mathcal{A}_r$ *such that* $val = r$ *for every partial value* $val$ *of the game* $G$.

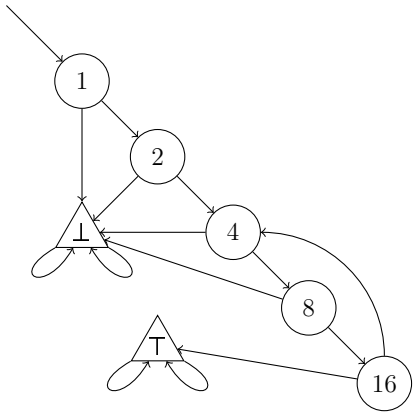



Figure 6.3 – The game $G_r$ encoding the number $r = \frac{1}{28}$, i.e. the number $0.00(001)^\omega$ in binary. Only reachable vertices are shown.

*Proof.* We start by constructing an $\{\mathcal{N}, \mathcal{B}\}$-branching game $G_r = \langle \mathtt{B}_r, \mathrm{L}(\mathcal{A}_r)\rangle$ for a rational number $r = \frac{x}{y}$. Later, we will show how to remove the branching vertices.

The board $\mathtt{B}_r = \langle V, \{0,1\}, s_{\mathtt{L}}, s_{\mathtt{R}}, \rho, \eta, \lambda, v_{\mathrm{I}}\rangle$ consists of the set of $y + 3$ vertices $V = \{0, \ldots, y\} \cup \{\top, \bot\}$, with the initial vertex $v_{\mathrm{I}} = x$. The vertices $\top$ and $\bot$ are branching and the rest belongs to *Nature*. The successor functions are defined as follows. For every vertex $v \in V \setminus \{\bot, \top\}$,

- if $\frac{v}{y} < \frac{1}{2}$ then $s_{\mathtt{L}}(v) = \bot$ and $s_{\mathtt{R}}(v) = 2v$,
- if $\frac{v}{y} \geq \frac{1}{2}$ then $s_{\mathtt{L}}(v) = \top$ and $s_{\mathtt{R}}(v) = 2v - y$,

and for $v \in \{\bot, \top\}$ $s_{\mathtt{L}}(v) = s_{\mathtt{R}}(v) = v$. The distribution on *Nature*'s vertices chooses successors with uniform probability. The alphabet is the set $\{0,1\}$ and the labelling function $\lambda$ assigns 1 to the vertex $\top$ and 0 to the rest of the vertices.

The winning set does not depend on the number $r$ nor the board $\mathtt{B}$ and is the language of all trees that contain a node labelled 1. The automaton $\mathcal{A}_r$ simply guesses the path to a 1-labelled node of the binary tree.

We claim that the game $G_r = \langle \mathtt{B}_r, \mathrm{L}(\mathcal{A}_r)\rangle$ has the desired partial values.

Since the game $G_r$ has only *Nature*'s and branching positions, it is determined and, thus, all the partial values coincide. Hence, it is enough to show that $val^E_{G_r} = r$.

The set of possible plays in the game $G_r$ is the set $\mathrm{plays}(G_r) = \{t_i\}_{1 \leq i \leq \omega}$, where $t_i$ is a tree that either contains an ancestor monotone sequence of nodes labelled 0 that is infinite or every such sequence is finite and the maximal one ends in a node $u$ such that $t_i{\vartriangle}u$ is a full binary tree with every node labelled 1. More precisely, for every $j$ such that $0 \leq j \leq i-2$ we have that $t_i(\mathtt{R}^j\mathtt{L}) = \mathtt{b}$, $t_i(\mathtt{R}^i) = \mathtt{b}$, and the sub-tree rooted in the node $t_i(\mathtt{R}^{i-1}\mathtt{L})$ is a full binary tree with every node labelled either 0 or 1. In particular, the tree $t_\omega$ has no 1-labelled nodes and every left child is labelled $\mathtt{b}$.

By the definition, see Equation (3.3) on page 34, the measure $\eta^*_{\mathtt{B}_r}$ satisfies

$$\eta^*_{\mathtt{B}_r}(\{t_i\}) = 2^{-i}, \tag{6.4}$$

with $\eta^*_{\mathtt{B}_r}(\{t_\omega\}) = 0$. Moreover, let $w = b_1 b_2 \cdots b_n \cdots \in \{0,1\}^\omega$ be a word such that for every $i > 0$ we have that $b_i = 1$ if and only if $t_i \in \mathrm{L}(\mathcal{A}_r)$. It is easy to see that $w$ is a binary expansion of $r$ i.e. the following holds.

$$r = \sum_{i \geq 1} b_i 2^{-i} \tag{6.5}$$

Therefore, if $\Phi$ is the characteristic function of the set $\mathrm{L}(\mathcal{A}_r)$, then the following holds.

$$val^E_{G_r} = \int_{\Sigma^{\mathcal{N}}_{G_r}} \Phi(t)\ \mathrm{d}\eta^*_{\mathtt{B}_r}(t) = \sum_{i \geq 1} \Phi(t_i)\eta^*_{\mathtt{B}_r}(\{t_i\}) = \sum_{i \geq 1} b_i 2^{-i} = r$$

To conclude the proof, observe that the branching positions can be changed to *Nature*'s without changing the values of the game.

The procedure can clearly be done in polynomial space: we iterate over the set $V$ that is exponential in the size of the encoding of $r$. □

**Expressive power** Which real numbers can be represented? Lemma 6.3.1 gives a procedure that for a predefined rational number $r$ constructs a simple finitary branching game $G_r$ such that $val^E_{G_r} = r$. The procedure simply computes a binary expansion of $r$: the tree $t_i$ represents the $i$th bit in the expansion. Thus, the procedure, if run *ad infinitum*, defines a branching-game representation for every real number $r$ from the interval $[0,1]$. However, the resulting board is infinite.

Still, it is impossible to represent every real number from the interval $[0,1]$ as a partial value of a finitary branching game with a regular winning set, since there is only countably many such games. On the other hand, we know that some algebraic irrational numbers can be represented, e.g. see [32] or Proposition 8.3.1 from page 94. This seems to be the limit, we are not aware of any finitary branching game with a regular winning set and a non-algebraic partial value. Thus, we conjecture the following.

**Conjecture 6.3.2.** *Let $G$ be a finitary regular branching game. Then, all partial values are algebraic.*

**Technical note** Before we proceed with more complex stochastic branching games, we need to adjust some notions. For the sake of simplicity and readability of the proofs, in the remainder of this chapter, we will partially diverge from viewing pure strategies as trees, see the definition of a pure strategy in Section 3.1.1 on page 33, and we will use the functional point of view, see *Strategies as functions* on page 34. In other words, in the proof of Lemma 6.3.3

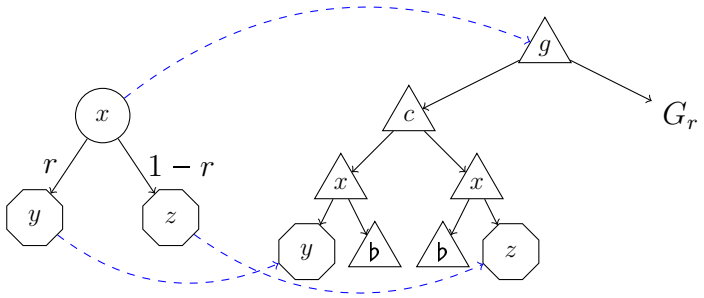



Figure 6.4 – The gadget used to transform a finitary game to a simple finitary game. The ownership of the hexagon nodes is undefined and depends on the original board. The blue dashed arrows depict the function $f$.

and in the proof of Theorem 6.3.5, a pure strategy $\xi$ is a function $\xi\colon \{\mathtt{L},\mathtt{R}\}^* \to \{\mathtt{L},\mathtt{R}\}$, a behavioural strategy $\xi_b$ is a function $\xi_b\colon \{\mathtt{L},\mathtt{R}\}^* \to \mathit{dist}(\{\mathtt{L},\mathtt{R}\})$, and, as before, a mixed strategy $\xi_m$ is a probability measure over the set of pure strategies.

The functional approach simply adds information on the behaviour of the strategies in unreachable positions. That superfluous information is not a problem, as it does not change the behaviour in reachable positions and we can easily recover the adequate tree-like strategy from the functional one, by forgetting the choices in the sub-trees cut by the functional strategy in the vertices owned by the player.

The main advantage given by the change of the notion of a pure strategy are simpler definitions of functions combining pure strategies, e.g. see page 76 for the definition of the function $f_{\mathcal{N},\mathcal{N}}$ that combines two strategies of *Nature* into one.

**Simple branching games** Returning to deradomisation, we use the construction from Lemma 6.3.1 to transform branching games into simple branching games, i.e. branching games where the probability distributions in *Nature*'s vertices are uniform: for $v \in V_{\mathcal{N}}$ we have that $\eta(v)(\mathtt{L}) = \eta(v)(\mathtt{R}) = \frac{1}{2}$.

**Lemma 6.3.3.** *Let $S \subseteq \{E, A, \mathcal{N}, \mathcal{B}\}$ be a non-empty set. There exists a polynomial space procedure that inputs a finitary $S$-branching game $G = \langle \mathtt{B}, \mathrm{L}(\mathcal{A})\rangle$, with a regular winning set given by a non-deterministic tree automaton and outputs a simple finitary $(S \cup \{\mathcal{N}, \mathcal{B}\})$-branching game $G' = \langle \mathtt{B}', \mathrm{L}(\mathcal{A}')\rangle$ with the winning set given by a non-deterministic tree automaton $\mathcal{C}$ such that for every partial value $val \in \{val^A, val^{BA}, val^{MA}, val^{ME}, val^{BE}, val^E\}$ we have that $val_G = val_{G'}$.*

*Proof.* By Lemma 6.3.1, for every rational number $r \in [0,1]$ there is a simple finitary $\{\mathcal{N}\}$-branching game $G_r = \langle \mathtt{B}_r, \mathrm{L}(\mathcal{A}_r)\rangle$ such that all partial values coincide and are equal to $r$. Moreover, since the automaton $\mathcal{A}_r$ does not depend on the number $r$ nor on the

board $\mathtt{B}_r$ we can compute in constant time an automaton $\mathcal{B}_r$ that recognises the complement of the language $\mathrm{L}(\mathcal{A}_r)$.

The new board $\mathtt{B}'$ is the original board $\mathtt{B}$ where every *Nature*'s vertex $x$, with successors labelled $y$, $z$, is replaced by the gadget depicted in Figure 6.4. We assume that $g$ and $c$ are fresh letters.

The automaton $\mathcal{C}$ is defined as follows. The automaton $\mathcal{C}$ on a play $p \in \mathrm{plays}(\mathtt{B}')$ behaves like the automaton $\mathcal{A}$ until it reaches a root of a copy of the gadget, i.e. a node $v$ labelled $g$. Then, the automaton $\mathcal{C}$ stores the current state $q$ of the automaton $\mathcal{A}$ and non-deterministically guesses whether the sub-tree in the right child $v\mathtt{R}$ of the node $v$ belongs to the language $\mathrm{L}(\mathcal{A}_r)$.

If the automaton guesses that $p \vartriangle v\mathtt{R} \in \mathrm{L}(\mathcal{A}_r)$ then in the node $v\mathtt{R}$ the automaton simulates $\mathcal{A}_r$ to compute the probability $r$ and continues the run of the automaton $\mathcal{A}$ in the node $v\mathtt{LL}$. If the automaton guesses that $p \vartriangle v\mathtt{R} \notin \mathrm{L}(\mathcal{A}_r)$, then in the node $v\mathtt{R}$ the automaton simulates $\mathcal{B}_r$ to compute the probability $r - 1$ and continues the run of the automaton $\mathcal{A}$ in the node $v\mathtt{LR}$.

The construction of the new board from the original one defines a function $f\colon \{\mathtt{L},\mathtt{R}\}^* \to \{\mathtt{L},\mathtt{R}\}^*$ that maps positions in the unfolding of the board $\mathtt{B}$ to their respective positions in the unfolding of the new board $\mathtt{B}'$, with the *Nature*'s positions mapped to the root of the adequate copy of the gadget.

Since the construction of the board $\mathtt{B}'$ does not introduce new positions of neither Adam nor Eve, the above function $f$ induces a continuous bijection $f_E\colon \Sigma^E_{G'} \to \Sigma^E_G$, and a continuous bijection $f_A\colon \Sigma^A_{G'} \to \Sigma^A_G$, defined by collapsing the nodes, in the respective trees, that originate from the gadget.

The transformation defines also a measurable surjection of *Nature*'s strategies $f_{\mathcal{N}}\colon \Sigma^{\mathcal{N}}_{G'} \to \Sigma^{\mathcal{N}}_G$ in the following manner. If $\eta' \sqsubseteq t^{\lambda}_{\mathtt{B}'}$ is a pure strategy of *Nature* in the game $\mathrm{G}'$, then $f_{\mathcal{N}}(\eta') \sqsubseteq t^{\lambda}_{\mathtt{B}}$ is a pure strategy of *Nature* in the game G such that for every *Nature*'s position $u$ in $t^{\lambda}_{\mathtt{B}}$ we have the following. If in the gadget rooted in the node $f(u)$ the sub-tree in the node $f(u)\mathtt{R}$ belongs to the language $\mathrm{L}(\mathcal{A}_r)$, then the strategy $f_{\mathcal{N}}(\eta')$ in the node $u$ chooses the left child, if not then the strategy $f_{\mathcal{N}}(\eta')$ in the node $u$ chooses the right child.

Since regular languages of trees are universally measurable, see Theorem 3.2.3 on page 39, the function $f_{\mathcal{N}}$ is measurable and we can use it to push forward the measure $\eta^*_{\mathtt{B}'}$. Observe that the pushforward measure $f_{\mathcal{N}}\#\eta^*_{\mathtt{B}'}$, see Equation (2.4) on page 21 for definition, is defined on the set $\Sigma^{\mathcal{N}}_G$ and coincides with the board measure $\eta^*_{\mathtt{B}}$ in the original game, as expressed in the following claim.

**Claim 6.3.4.** *The following equation holds.*

$$f_{\mathcal{N}}\#\eta^*_{\mathtt{B}'} = \eta^*_{\mathtt{B}} \tag{6.6}$$

*Proof.* The strategy $\eta^*_{\mathtt{B}}$ is a behavioural strategy, i.e. $\eta^*_{\mathtt{B}}$ is defined by a function $f_{\eta^*_{\mathtt{B}}}\colon \{\mathtt{L},\mathtt{R}\}^* \to$

$dist(\{\mathtt{L}, \mathtt{R}\})$ that maps *Nature*'s positions to distributions over successors. The same function induces the measure $f_{\mathcal{N}} \# \eta^*_{\mathtt{B}'}$. Indeed, if in a node $u$ the strategy $\eta^*_{\mathtt{B}}$ chooses left child with probability $r$ then the strategy $f_{\mathcal{N}} \# \eta^*_{\mathtt{B}'}$ chooses left child with probability $val^E_{G_r}$, where $G_r$ is the sub-game in the game $G'$ that starts at node $f(u)\mathtt{R}$. Since $val^E_{G_r} = r$, the claim holds. □

Hence, for every pair of strategies $\sigma' \in \Sigma^E_{G'}$ and $\pi' \in \Sigma^A_{G'}$ we have that

$$\begin{aligned}
val_{G'}(\sigma', \pi') &\overset{1}{=} \int_{\Sigma^{\mathcal{N}}_{\mathtt{B}'}} \Phi'\big(\mathrm{eval}_{\mathrm{G}'}(\sigma', \pi', \eta')\big)\, \mathrm{d}\eta^*_{\mathtt{B}'}(\eta') \\
&\overset{2}{=} \int_{\Sigma^{\mathcal{N}}_{\mathtt{B}'}} \Phi\big(\mathrm{eval}_{\mathrm{G}}(f_E(\sigma'), f_A(\pi'), f_{\mathcal{N}}(\eta'))\big)\, \mathrm{d}\eta^*_{\mathtt{B}'}(\eta') \\
&\overset{3}{=} \int_{\Sigma^{\mathcal{N}}_{\mathtt{B}}} \Phi\big(\mathrm{eval}_{\mathrm{G}}(f_E(\sigma'), f_A(\pi'), \eta)\big)\, \mathrm{d} f_{\mathcal{N}} \# \eta^*_{\mathtt{B}'}(\eta) \\
&\overset{4}{=} \int_{\Sigma^{\mathcal{N}}_{\mathtt{B}}} \Phi\big(\mathrm{eval}_{\mathrm{G}}(f_E(\sigma'), f_A(\pi'), \eta)\big)\, \mathrm{d}\eta^*_{\mathtt{B}}(\eta) \\
&\overset{5}{=} val_G(f_E(\sigma'), f_A(\pi')).
\end{aligned}$$

The first equation is the definition applied to the pure strategies $\sigma'$ and $\pi'$. The second is the consequence of point-wise equality. The third equation is the integration by substitution of the pushforward measure $f_{\mathcal{N}} \# \eta^*_{\mathtt{B}'}(\eta)$, see Equation (2.5) on page 21 and the fourth stems from Equation (6.6). The last one is, again, the definition of the value in the original game.

Since $f_E$ and $f_A$ are bijections, this proves that the pure values in the game $G$ coincide with the respective pure values in the game $G'$. The equivalence of the mixed values follows immediately from pushing forward the mixed strategies while the equivalence of the behavioural values follows from the fact that pushing forward a behavioural strategy through the considered bijections yields a behavioural strategy. □

We use the above lemma to show that, when considering stochastic partial values, we can entirely eliminate *Nature*'s positions.

**Theorem 6.3.5.** *There exists a logarithmic space procedure that inputs a simple finitary branching game $G = \langle \mathtt{B}, \mathrm{L}(\mathcal{A}) \rangle$ with a regular winning set given by an alternating tree automaton and outputs a finitary non-stochastic $\{E, A, \mathcal{B}\}$-branching game $G' = \langle \mathtt{B}', \mathrm{L}(\mathcal{A}') \rangle$ with the winning set given by an alternating tree automaton $\mathcal{A}'$ such that for every partial value val from the set $\{val^{BA}, val^{MA}, val^{ME}, val^{BE}\}$ we have that $val_G = val_{G'}$.*

*Proof.* We start by defining the game $G' = \langle \mathtt{B}', \mathrm{L}(\mathcal{A}') \rangle$, then we will show that the values of the old game and the new game coincide.

The new board $\mathtt{B}'$ is obtained by substituting the vertices of *Nature* with the gadget depicted in Figure 6.5. An $x$-labelled vertex $v$ in the original game, with the $y$-labelled vertex $s_{\mathtt{L}}(v)$ and the $z$-labelled vertex $s_{\mathtt{R}}(v)$, is replaced by a copy of the gadget so that the

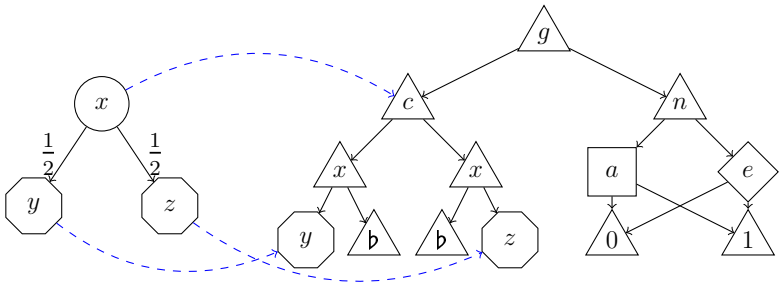



Figure 6.5 – The gadget used to transform a simple stochastic game to a non-stochastic game. The ownership of the hexagon nodes is undefined and depends on the original board. The blue dashed arrows depict the function $f$.

$g$-labelled vertex $v'$ is put in the place of $v$, $s_{\mathtt{LLL}}(v') = s_{\mathtt{L}}(v)$, and $s_{\mathtt{LRR}}(v') = s_{\mathtt{R}}(v)$. We assume that $g$ and $c$ are fresh letters.

The new winning set is defined as follows. The automaton $\mathcal{A}'$ recognising the new winning set simulates the original one until it reaches the gadget, i.e. a $g$-labelled node $v$. Then, the automaton $\mathcal{A}'$ stores the current state $q$ of the automaton $\mathcal{A}$, and guesses whether Eve and Adam chose the same direction in the vertices $v_{\mathtt{RL}}$ and $v_{\mathtt{RR}}$. If it guesses that they did, the automaton resumes the simulation of the automaton $\mathcal{A}$ in the node $v_{\mathtt{LL}}$ and verifies the guess in the sub-tree rooted in the node $v_{\mathtt{R}}$. If the automaton $\mathcal{A}'$ guesses that they did not choose the same direction, the automaton $\mathcal{A}'$ resumes the simulation of the automaton $\mathcal{A}$ in the node $v_{\mathtt{LR}}$ and verifies the guess in the sub-tree rooted in the node $v_{\mathtt{R}}$.

The new game is defined and the procedure is clearly definable in log-space. Thus, all we need to prove is that the construction preserves the values, i.e. that the following holds.

**Claim 6.3.6.** *For* $val \in \{val^{BA}, val^{MA}, val^{ME}, val^{BE}\}$ *we have that* $val_G = val_{G'}$.

The proof of the claim is technical, thus we start with some intuition. Intuitively, we will show that for Eve and Adam playing with uniform probability in vertices belonging to the gadget does not worsen the game values. Thus, every use of the gadget simulates a *Nature*'s vertex in a simple branching game. Since every *Nature*'s vertex has been replaced by an instance of the gadget, the values coincide.

Observe that the construction of the new board from the original one defines a function $f\colon \{\mathtt{L},\mathtt{R}\}^* \to \{\mathtt{L},\mathtt{R}\}^*$ that maps positions in the unfolding of the board $\mathtt{B}$ to their respective positions in the unfolding of the new board $\mathtt{B}'$, with *Nature*'s positions mapped to the left child of the root of the adequate copy of the gadget, i.e. the $c$-labelled node of the gadget.

Note that the structure of the gadget replaces a position of *Nature* with a pair of positions. One of the positions belongs to Eve, the other one to Adam. This induces the following set of functions in the natural way.

- A continuous bijection $f_E\colon \Sigma^E_{G'} \to \Sigma^E_G \times \Sigma^{\mathcal{N}}_G$,

- a continuous bijection $f_A \colon \Sigma^A_{G'} \to \Sigma^A_G \times \Sigma^{\mathcal{N}}_G$,
- and a continuous surjection $f_p \colon \mathrm{plays}(G') \to \mathrm{plays}(G)$.

The functions $f_E$ and $f_A$ take as an input a pure strategy $\xi' \in \Sigma^X_{G'}$, $X \in \{E, A\}$, and output a pair of strategies $\langle \xi, \eta \rangle \in \Sigma^X_G \times \Sigma^{\mathcal{N}}_G$ such that the strategy $\xi$ in a position $u$ belonging to the player $X$ in the game $G$ chooses the same node as the strategy $\xi'$ in the node $f(u)$, and the strategy $\eta$ in a position $v$ belonging to *Nature* in the game $G$ chooses the same node as the strategy $\xi'$ in the node of the player $X$ in the copy of the gadget whose $c$-labelled node is $f(v)$. The function $f_p$ simply maps a play of the game $G'$ to its interpretation in the game $G$.

In the reverse direction, the transformation defines a measurable function $f^*_E \colon \Sigma^{ME}_G \to \Sigma^{ME}_{G'}$ and a measurable function $f^*_A \colon \Sigma^{MA}_G \to \Sigma^{MA}_{G'}$, obtained by pushing forward the mixed strategies in the following way.

$$f^*_E(\sigma_m) = f_E^{-1} \# (\sigma_m \times \eta^*_{\mathtt{B}}) \tag{6.7}$$

$$f^*_A(\pi_m) = f_A^{-1} \# (\pi_m \times \eta^*_{\mathtt{B}}) \tag{6.8}$$

The function $f^*_E$ (resp., $f^*_A$) combines a mixed strategy of Eve (resp., of Adam) in the game $G$ with the strategy of *Nature* in $G$ that chooses moves in the gadget with uniform distribution.

In the sequel we will transfer strategies of the players between the games $G$ and $G'$. Equation (6.7) and Equation (6.8) show how we transfer strategies form the game $G$ to the game $G'$. The transfer in the reverse direction is given by the functions $f_E$ and $f_A$. Observe that the transfer of strategies from $G'$ to $G$ yields a pair of strategies of *Nature*. To merge those strategies into a single strategy of *Nature* we use the following function $f_{\mathcal{N},\mathcal{N}} \colon \Sigma^{\mathcal{N}}_G \times \Sigma^{\mathcal{N}}_G \to \Sigma^{\mathcal{N}}_G$. The function $f_{\mathcal{N},\mathcal{N}}$ takes two strategies $\eta_1, \eta_2$ of *Nature* and returns a pure strategy $\eta$ of *Nature* such that for every position $u$ of *Nature*, the strategy $\eta$ chooses the left child, $\eta(u_{\mathtt{R}}) = \flat$, if the strategies $\eta_1, \eta_2$ agree on that position, i.e. $\eta_1(u_{\mathtt{L}}) = \eta_2(u_{\mathtt{L}})$ and $\eta_1(u_{\mathtt{R}}) = \eta_2(u_{\mathtt{R}})$, and chooses the right child if they do not agree.

**Claim 6.3.7.** *For every mixed strategy $\eta_m \in \Sigma^{M\mathcal{N}}_G$ of Nature the following holds.*

$$f_{\mathcal{N},\mathcal{N}} \# (\eta_m \times \eta^*_{\mathtt{B}}) = \eta^*_{\mathtt{B}} = f_{\mathcal{N},\mathcal{N}} \# (\eta^*_{\mathtt{B}} \times \eta_m) \tag{6.9}$$

*Proof.* By the symmetry argument, we only need to prove $f_{\mathcal{N},\mathcal{N}} \# (\eta_m \times \eta^*_{\mathtt{B}}) = \eta^*_{\mathtt{B}}$.

The strategy $\eta^*_{\mathtt{B}}$ is behavioural, i.e. the strategy $\eta^*_{\mathtt{B}}$ is defined by a function $f_{\eta^*_{\mathtt{B}}} \colon \{\mathtt{L}, \mathtt{R}\}^* \to \mathit{dist}(\{\mathtt{L}, \mathtt{R}\})$ that maps *Nature*'s positions to distributions over successors. The same function induces the measure $f_{\mathcal{N},\mathcal{N}} \# (\eta_m \times \eta^*_{\mathtt{B}})$. Indeed, let $\eta \in \Sigma^{\mathcal{N}}_G$ be a pure strategy of *Nature*. If

in a node $u$ the strategy $\eta$ chooses left child, then the strategy $f_{\mathcal{N},\mathcal{N}}\#(\eta \times \eta^*_{\mathtt{B}})$ chooses left child with probability $\frac{1}{2}$. Thus Equation (6.9) holds for any pure strategy of *Nature*.

Since Equation (6.9) holds for every pure strategy and mixed strategies are probability measures on the set of pure strategies, Equation (6.9) holds for every mixed strategy $\eta_m \in \Sigma_G^{M\mathcal{N}}$. □

Intuitively, the above claim simply states that when one of the players plays randomly in the game $G'$ in the positions originating from the gadget, then the strategy of the other player in those positions does not matter. Moreover, playing randomly in those nodes results in that the positions originating from the gadget simulate behaviour of *Nature* in the game $G$.

Let fst denote the first element of a pair and snd denote the other one. Now we are ready to prove the equality of the Eve's mixed values in Claim 6.3.6. We start by showing the following.

**Claim 6.3.8.** *The following inequality holds.*

$$val^{ME}_{G'} \geq val^{ME}_{G} \tag{6.10}$$

*Proof.* Recall that since the board $\mathtt{B}'$ has no vertices of *Nature*, the strategy $\eta^*_{\mathtt{B}'}$ induced by the board $\mathtt{B}'$ is the tree $t^{\lambda}_{\mathtt{B}'}$. Thus, for any mixed strategy $\sigma'_m \in \Sigma^{ME}_{G'}$ and any pure strategy $\pi' \in \Sigma^{A}_{G'}$ we have that $val_{G'}(\sigma'_m, \pi') = \int_{\Sigma^E_{G'}} \Phi'\big(\mathrm{eval}_{G'}(\sigma', \pi', \eta^*_{\mathtt{B}'})\big)\, \mathrm{d}\sigma'_m(\sigma')$. Now, the claim follows from the following sequence of (in)equalities.

$$\begin{aligned}
val^{ME}_{G'} &\overset{1}{=} \sup_{\sigma'_m \in \Sigma^{ME}_{G'}} \inf_{\pi' \in \Sigma^{A}_{G'}} val_{G'}(\sigma'_m, \pi') \\
&\overset{2}{\geq} \sup_{\sigma_m \in \Sigma^{ME}_{G}} \inf_{\pi' \in \Sigma^{A}_{G'}} val_{G'}(f^*_E(\sigma_m), \pi') \\
&\overset{3}{=} \sup_{\sigma_m \in \Sigma^{ME}_{G}} \inf_{\pi' \in \Sigma^{A}_{G'}} \int_{\Sigma^E_{G'}} \Phi'\big(\mathrm{eval}_{G'}(\sigma', \pi', \eta^*_{\mathtt{B}'})\big)\, \mathrm{d}f^*_E(\sigma_m)(\sigma') \\
&\overset{4}{=} \sup_{\sigma_m \in \Sigma^{ME}_{G}} \inf_{\pi' \in \Sigma^{A}_{G'}} \int_{\Sigma^E_G \times \Sigma^{\mathcal{N}}_G} \Phi'\big(\mathrm{eval}_{G'}(f_E^{-1}(\sigma, \eta), \pi', \eta^*_{\mathtt{B}'})\big)\, \mathrm{d}f_E\#f^*_E(\sigma_m)(\sigma, \eta) \\
&\overset{5}{=} \sup_{\sigma_m \in \Sigma^{ME}_{G}} \inf_{\pi' \in \Sigma^{A}_{G'}} \int_{\Sigma^E_G \times \Sigma^{\mathcal{N}}_G} \Phi'\big(\mathrm{eval}_{G'}(f_E^{-1}(\sigma, \eta), \pi', \eta^*_{\mathtt{B}'})\big)\, \mathrm{d}\sigma_m \times \eta^*_{\mathtt{B}}(\sigma, \eta) \\
&\overset{6}{=} \sup_{\sigma_m \in \Sigma^{ME}_{G}} \inf_{\pi' \in \Sigma^{A}_{G'}} \int_{\Sigma^E_G \times \Sigma^{\mathcal{N}}_G} \Phi\big(\mathrm{eval}_{G}(\sigma, \mathrm{fst}(f_A(\pi')), f_{\mathcal{N},\mathcal{N}}(\eta, \mathrm{snd}(f_A(\pi'))))\big) \\
&\qquad\qquad \mathrm{d}\sigma_m \times \eta^*_{\mathtt{B}}(\sigma, \eta) \\
&\overset{7}{=} \sup_{\sigma_m \in \Sigma^{ME}_{G}} \inf_{\pi' \in \Sigma^{A}_{G'}} \int_{\Sigma^E_G \times \Sigma^{\mathcal{N}}_G} \Phi\big(\mathrm{eval}_{G}(\sigma, \mathrm{fst}(f_A(\pi')), \eta)\big)\, \mathrm{d}\sigma_m \times \eta^*_{\mathtt{B}}(\sigma, \eta)
\end{aligned}$$

$$\overset{8}{=} \sup_{\sigma_m \in \Sigma_G^{ME}} \inf_{\pi' \in \Sigma_{G'}^{A}} val_G(\sigma_m, \mathrm{fst}(f_A(\pi')))$$

$$\overset{9}{=} \sup_{\sigma_m \in \Sigma_G^{ME}} \inf_{\pi \in \Sigma_G^{A}} val_G(\sigma_m, \pi) = val_G^{ME}$$

The first (in)equality is the definition, the second follows from the fact that we restrict the set of strategies over which we take the supremum. The third is, again, the definition, see Equation (3.4) on page 36. The fourth follows from integration by substitution and the fifth from computing the pushforward measure. The sixth equality follows from point-wise equality of pay-off functions and the seventh from the fact that the function $t \mapsto f_{\mathcal{N},\mathcal{N}}(t, \mathrm{snd}(f_A(\pi'))$ does not change the measure $\eta^*_{\mathtt{B}}$, see Equation (6.9).

The eighth equality is the definition of $val_G$, while the ninth follows from the fact that $\mathrm{fst}(f_A(\Sigma_{G'}^{ME})) = \Sigma_G^{ME}$. The last equality is the definition of $val_G^{ME}$. □

Now we prove the reverse inequality.

**Claim 6.3.9.** *The following inequality holds.*

$$val_{G'}^{ME} \leq val_G^{ME} \tag{6.11}$$

*Proof.* The following sequence of (in)equalities holds.

$$\begin{aligned}
val_{G'}^{ME} &\overset{1}{=} \sup_{\sigma'_m \in \Sigma_{G'}^{ME}} \inf_{\pi'_m \in \Sigma_{G'}^{MA}} val_{G'}(\sigma'_m, \pi'_m) \\
&\overset{2}{\leq} \sup_{\sigma'_m \in \Sigma_{G'}^{ME}} \inf_{\pi_m \in \Sigma_G^{MA}} val_{G'}(\sigma'_m, f_A^*(\pi_m)) \\
&\overset{3}{=} \sup_{\sigma'_m \in \Sigma_{G'}^{ME}} \inf_{\pi_m \in \Sigma_G^{MA}} \int_{\Sigma_{G'}^{E} \times \Sigma_{G'}^{A}} \Phi'\big(\mathrm{eval}_{G'}(\sigma', \pi', \eta^*_{\mathtt{B}'})\big)\, \mathrm{d}\sigma'_m \times f_A^*(\pi_m)(\sigma', \pi') \\
&\overset{4}{=} \sup_{\sigma'_m \in \Sigma_{G'}^{ME}} \inf_{\pi_m \in \Sigma_G^{MA}} \int_{\Sigma_G^{E} \times \Sigma_G^{\mathcal{N}} \times \Sigma_G^{A} \times \Sigma_G^{\mathcal{N}}} \Phi'\big(\mathrm{eval}_{G'}(f_E^{-1}(\sigma, \eta_1), f_A^{-1}(\pi, \eta_2), \eta^*_{\mathtt{B}'})\big) \\
&\qquad\qquad \mathrm{d}f_E \# \sigma'_m \times f_A \# f_A^*(\pi_m)(\sigma, \eta_1, \pi, \eta_2) \\
&\overset{5}{=} \sup_{\sigma'_m \in \Sigma_{G'}^{ME}} \inf_{\pi_m \in \Sigma_G^{MA}} \int_{\Sigma_G^{E} \times \Sigma_G^{\mathcal{N}} \times \Sigma_G^{A} \times \Sigma_G^{\mathcal{N}}} \Phi\big(\mathrm{eval}_G(\sigma, \pi, f_{\mathcal{N},\mathcal{N}}(\eta_1, \eta_2))\big) \\
&\qquad\qquad \mathrm{d}f_E \# \sigma'_m \times f_A \# f_A^*(\pi_m)(\sigma, \eta_1, \pi, \eta_2) \\
&\overset{6}{=} \sup_{\sigma'_m \in \Sigma_{G'}^{ME}} \inf_{\pi_m \in \Sigma_G^{MA}} \int_{\Sigma_G^{E} \times \Sigma_G^{\mathcal{N}} \times \Sigma_G^{A} \times \Sigma_G^{\mathcal{N}}} \Phi\big(\mathrm{eval}_G(\sigma, \pi, f_{\mathcal{N},\mathcal{N}}(\eta_1, \eta_2))\big) \\
&\qquad\qquad \mathrm{d}\ \mathrm{fst}(f_E \# \sigma'_m) \times \mathrm{snd}(f_E \# \sigma'_m) \times \pi_m \times \eta^*_{\mathtt{B}}(\sigma, \eta_1, \pi, \eta_2) \\
&\overset{7}{=} \sup_{\sigma'_m \in \Sigma_{G'}^{ME}} \inf_{\pi_m \in \Sigma_G^{MA}} \int_{\Sigma_G^{E} \times \Sigma_G^{A} \times \Sigma_G^{\mathcal{N}} \times \Sigma_G^{\mathcal{N}}} \Phi\big(\mathrm{eval}_G(\sigma, \pi, f_{\mathcal{N},\mathcal{N}}(\eta_1, \eta_2))\big) \\
&\qquad\qquad \mathrm{d}\ \mathrm{fst}(f_E \# \sigma'_m) \times \pi_m \times \mathrm{snd}(f_E \# \sigma'_m) \times \eta^*_{\mathtt{B}}(\sigma, \pi, \eta_1, \eta_2)
\end{aligned}$$

$$
\begin{aligned}
&\overset{8}{=} \sup_{\sigma'_m \in \Sigma^{ME}_{G'}} \inf_{\pi_m \in \Sigma^{MA}_G} \int_{\Sigma^E_G \times \Sigma^A_G \times \Sigma^{\mathcal{N}}_G} \Phi\big(\mathrm{eval}_G(\sigma,\pi,\eta)\big) \\
&\qquad\qquad \mathrm{d}\ \mathrm{fst}(f_E \# \sigma'_m) \times \pi_m \times \eta^*_{\mathtt{B}}(\sigma,\pi,\eta) \\
&\overset{9}{=} \sup_{\sigma_m \in \Sigma^{ME}_G} \inf_{\pi_m \in \Sigma^{MA}_G} \int_{\Sigma^E_G \times \Sigma^A_G \times \Sigma^{\mathcal{N}}_G} \Phi\big(\mathrm{eval}_G(\sigma,\pi,\eta)\big) \\
&\qquad\qquad \mathrm{d}\sigma_m \times \pi_m \times \eta^*_{\mathtt{B}}(\sigma,\pi,\eta) \\
&\overset{10}{=} \sup_{\sigma_m \in \Sigma^{ME}_G} \inf_{\pi_m \in \Sigma^{MA}_G} val_G(\sigma_m,\pi_m) = val^{ME}_G.
\end{aligned}
$$

The first equality is the definition of the mixed value of Eve in the game $G'$ together with Lemma 3.2.1 on page 36. The following inequality stems from the fact that we limit the set of Adam's strategies for which we compute the infimum. The third follows from the definition of $val_{G'}(\sigma'_m, f^*_A(\pi_m))$. The fourth follows from the integration by substitution and pushing forward the measure $\sigma'_m \times f^*_E(\pi_m)$. The fifth is a point-wise equality on the values of the pay-off functions and the sixth is obtained by computing the pushforward measure. The equalities numbered seven and eight follow from Fubini's theorem and Equation (6.9), respectively. The ninth equality stems from the fact that $\sigma'_m$ ranges over all strategies of Eve in the new game, thus $\mathrm{fst}(f_E \# \sigma'_m)$ ranges over all strategies of Eve in the old game. The tenth equality is the definition of $val_G(\sigma_m, \pi_m)$, and the last one is the definition of the mixed value of Eve in the game $G$, again, together with Lemma 3.2.1. □

In consequence, from Claim 6.10 and Claim 6.11 we infer that $val^{ME}_{G'} = val^{ME}_G$.

To conclude the proof, note that the equality $val^{MA}_{G'} = val^{MA}_G$ is proven *mutatis mutandis* and that the equalities between the behavioural values stem from the simple observation that both $f^*_E$ and $f^*_A$ map behavioural strategies to behavioural strategies. □

Theorem 6.3.5, the derandomisation theorem, implies immediately three interesting corollaries.

**Corollary 6.3.10.** *There exists a polynomial space procedure that inputs a finitary branching game $G = \langle \mathtt{B}, \mathrm{L}(\mathcal{A})\rangle$ with a regular winning set given by an alternating tree automaton and outputs a finitary non-stochastic $\{E, A, \mathcal{B}\}$-branching game $G' = \langle \mathtt{B}', \mathrm{L}(\mathcal{C})\rangle$ with the winning set given by an alternating tree automaton $\mathcal{C}$ such that for every partial value val from the set $\{val^{BA}, val^{MA}, val^{ME}, val^{BE}\}$ we have that $val_G = val_{G'}$.*

*Proof.* This is an immediate consequence of Theorem 6.3.5 and Lemma 6.3.3. □

Moreover, if the original game is determined under pure strategies we get the following.

**Corollary 6.3.11.** *There exists a polynomial space procedure that inputs a determined under pure strategies, finitary branching game $G = \langle \mathtt{B}, \mathrm{L}(\mathcal{A})\rangle$ with a regular winning set given by an alternating tree automaton $\mathcal{A}$ and outputs a finitary non-stochastic $\{E, A, \mathcal{B}\}$-branching*

*game* $G' = \langle \mathtt{B}', \mathrm{L}(\mathcal{C}) \rangle$ *with the winning set given by an alternating tree automaton* $\mathcal{C}$ *such that* $val^E_G = val^{BE}_{G'}$.

*Proof.* Since in branching games that are determined under pure strategies all partial values coincide, this is an immediate consequence of Corollary 6.3.10. □

Last, but not least, there is no algorithm that computes partial non-pure values of a given regular branching game. Recall that by Theorem 5.3.2, we can compute pure partial values of non-stochastic games.

**Corollary 6.3.12.** *For every* $val \in \{val^{BA}, val^{MA}, val^{ME}, val^{BE}\}$*, simple threshold for val problem of a regular* $\{E, A, \mathcal{B}\}$*-branching game is undecidable.*

*Proof.* The corollary follows immediately from Theorem 6.2.1 and Corollary 6.3.10. □

# Chapter 7

# Game automata winning sets

In this short chapter, we simplify branching games by restricting the power of the automata defining the winning sets of games. We do that by virtually removing non-determinism from the structure of the tree automata. This is obtained by the requirement that every transition is a split and results in an automaton model called game automaton, see Section 2.4 for the definition. Game automata were introduced by Duparc et al. [15] and studied in e.g. [17].

In this chapter, we show that regular branching games with winning sets given by game automata are determined under pure strategies and that there is an algorithm that decides whether pure values of a given game exceed any given threshold.

## 7.1 Reduction to meta-parity games

When we restrict the winning sets of branching games to those sets which are recognisable by game automata, branching games reduce to meta-parity games. For a definition of meta-parity games in the setting of branching games see Section 3.3.5 on page 42, for the original definition see e.g. [34] and for a refined definition see [33].

**Theorem 7.1.1.** *There exists a logarithmic space procedure using a UP∩co-UP oracle that inputs a finitary branching game $G = \langle \mathtt{B}, \mathrm{L}(\mathcal{A}) \rangle$ with a regular winning set given by a game automaton $\mathcal{A}$ and outputs a stochastic meta-parity game $G'$ such that $val^E_{G'} = val^E_G$ and $val^A_{G'} = val^A_G$. Moreover, if $S \subseteq \{E, A, \mathcal{N}, \mathcal{B}\}$ and $G$ is $S$-branching then $G'$, as a branching game, is also $S$-branching.*

*Proof.* Let $\mathtt{B} = \langle V, \Gamma, s_{\mathrm{L}}, s_{\mathrm{R}}, \rho, \eta, \lambda, v_{\mathrm{I}} \rangle$ and $\mathcal{A} = \langle Q, \Gamma \cup \{\flat\}, \delta, \alpha, q_{\mathrm{I}} \rangle$. We define the new game as $G' = \langle \mathtt{B}', \Phi \rangle$, where $\Phi$ is the pay-off function used in meta-parity games, see Section 3.3.5 on page 42, and the new $\mathtt{B}'$ is defined as follows. Intuitively, the board $\mathtt{B}' = \langle V', \Gamma', s'_{\mathrm{L}}, s'_{\mathrm{R}}, \rho', \eta', \lambda', v'_{\mathrm{I}} \rangle$ is the synchronised product of the old board $\mathtt{B}$ and the automaton $\mathcal{A}$. More precisely, the set of new vertices is the set $V' = V \times Q$, the new initial vertex is the

vertex $v_{\mathrm{I}}' = \langle v_{\mathrm{I}}, q_{\mathrm{I}} \rangle$, and the alphabet is the set $\Gamma' = \{i, i+1, \ldots, j\} \times \{E, A\} \times \{\top, \bot\} \times \{\top, \bot\}$. Let $t_\flat$ be the blank tree, i.e. the tree $t_\flat \colon \{\mathtt{L}, \mathtt{R}\}^* \to \{\flat\}$, and let $f_{\mathcal{A}} \colon Q \to \{\bot, \top\}$ be the function defined as

$$f_{\mathcal{A}}(p) = \begin{cases} \top, & \text{if } \mathcal{A}[q_{\mathrm{I}} = p] \text{ accepts the tree } t_\flat, \\ \bot, & \text{otherwise} \end{cases}$$

where the automaton $\mathcal{A}[q_{\mathrm{I}} = p]$ is the automaton $\mathcal{A}$ with the initial state replaced by the state $p$, i.e. $\mathcal{A}[q_{\mathrm{I}} = p] \stackrel{\text{def}}{=} \langle Q, \Gamma, \delta, \alpha, p \rangle$.

Now, we define the remaining components of $\mathtt{B}'$. If $v' = \langle v, q \rangle$ is a vertex of the new board, then for some $q_{\mathtt{L}}, q_{\mathtt{R}} \in Q$ and $\diamond \in \{\wedge, \vee\}$ we have that $\delta(q, \lambda(v)) = q_{\mathtt{L}} \diamond q_{\mathtt{R}}$, every transition of a game automaton is a split, and

- the successor functions satisfy $s'_{\mathtt{L}}(v') = \langle s_{\mathtt{L}}(v), q_{\mathtt{L}} \rangle$ and $s'_{\mathtt{R}}(v') = \langle s_{\mathtt{R}}(v), q_{\mathtt{R}} \rangle$;
- the ownership of $v'$ is induced by the ownership in the original game, i.e. $\rho'(v') = \rho(v)$;
- the labelling is defined as $\lambda'(v') = \langle \alpha(q), \diamond, f_{\mathcal{A}}(q_{\mathtt{L}}), f_{\mathcal{A}}(q_{\mathtt{R}}) \rangle$.

Since $\mathtt{B}'$ is the synchronised product of $\mathtt{B}$ and $\mathcal{A}$, there is a natural continuous bijection between the strategies in the original game and the new game, i.e. a function $f \colon \Sigma^E_{\mathtt{B}} \times \Sigma^A_{\mathtt{B}} \times \Sigma^{\mathcal{N}}_{\mathtt{B}} \to \Sigma^E_{\mathtt{B}'} \times \Sigma^A_{\mathtt{B}'} \times \Sigma^{\mathcal{N}}_{\mathtt{B}'}$.

Therefore, to conclude the proof, it is enough to show that for every triple of strategies $\sigma \in \Sigma^E_{\mathtt{B}}, \pi \in \Sigma^A_{\mathtt{B}}, \eta \in \Sigma^{\mathcal{N}}_{\mathtt{B}}$ the resulting tree $\mathrm{eval}_G(\sigma, \pi, \eta)$ is accepted by the automaton $\mathcal{A}$ if and only if Eve wins the parity game induced by the play $\mathrm{eval}_{G'}(\sigma', \pi', \eta')$, where $\langle \sigma', \pi', \eta' \rangle = f(\sigma, \pi, \eta)$, i.e. that the following equation holds.

$$\chi_{\mathrm{L}(\mathcal{A})}(\mathrm{eval}_G(\sigma, \pi, \eta)) = \Phi(\mathrm{eval}_{G'}(\sigma', \pi', \eta')) \tag{7.1}$$

Indeed, if Equation (7.1) is true, then by pushing forward the appropriate measures it is true for every mixed strategy of *Nature* and the equations

$$val^E_{G'} = val^E_G, val^A_{G'} = val^A_G$$

simply follow.

To prove Equation (7.1), we observe that the game induced by the play $\mathrm{eval}_{G'}(\sigma', \pi', \eta')$ is, essentially, the parity game $\mathrm{G}(\mathcal{A}, \mathrm{eval}_G(\sigma, \pi, \eta))$ with the sub-games $G(\mathcal{A}[q_{\mathrm{I}} = p], t_\flat)$ replaced by the appropriate $\bot$- or $\top$-vertex, for the definition of the acceptance game $\mathrm{G}(\mathcal{A}, t)$ see Section 2.4 on page 24. □

From the above theorem we infer immediately the following.

**Corollary 7.1.2.** *Let $G$ be a finitary branching game with a regular winning set given by a game automaton. Then, the game $G$ is determined under pure strategies and simple threshold for val problem is decidable for every value $val \in \{val^{BA}, val^{MA}, val^{ME}, val^{BE}\}$.*

*Proof.* By Theorem 7.1.1 and Theorem 3.3.5 we obtain the determinacy under pure strategies. In particular, since $G$ is determined under pure strategies all partial values coincide. Moreover, by Theorem 7.1.1 and Remark 4.3.7, we conclude that the values are computable. □

Intuitively, when the winning set is recognisable by a game automaton, the game flattens. That is, the two-phase structure of the game collapses to a single phase. This is especially evident in the case of non-stochastic games, where the whole game collapses to a parity game.

**Corollary 7.1.3.** *Let $S \subseteq \{E, A, \mathcal{B}\}$. There exists a logarithmic space procedure using a UP∩co-UP oracle that inputs a finitary non-stochastic $S$-branching game $G = \langle \mathtt{B}, \mathrm{L}(\mathcal{A})\rangle$ with a regular winning set given by a game automaton $\mathcal{A}$ and outputs a parity game $G'$ such that $val^E_{G'} = val^E_G$ and $val^A_{G'} = val^A_G$.*

The corollary follows immediately from Theorem 7.1.1 and Remark 5.1.11 on page 175 in [33]. We omit the full proof of the above corollary, more straightforward approach can be found in [45] where we show a direct reduction.

We conclude this chapter with the following observation. With small alternations to the definition of meta-parity games we can remove the necessity of the oracles: instead of using $\bot$ and $\top$ as plug-ins for the missing children in the play of the resulting meta-parity game, one could substitute the missing nodes with arbitrary, albeit finitary, parity games.

Such meta-parity games would remain determined under pure strategies and their values computable.

# Chapter 8

# Measures

In this chapter we focus on the problem of computing the uniform measure of a regular set of infinite trees. This problem is a simplification of the problem of computing game values: computing the value of a regular $\{\mathcal{N}, \mathcal{B}\}$-branching game $G = \langle \mathtt{B}, L \rangle$ can be seen as computing the value $\eta^*_{\mathtt{B}}(L)$, for the definition of $\eta^*_{\mathtt{B}}$ see Equation (3.3) on page 34.

We show that for a regular set of full binary trees its uniform measure is rational and computable if the set is defined by a first-order formula not using descendant relation or if it is defined by a Boolean combination of conjunctive queries.

## 8.1 Computing measure and simple examples

When computing the value, e.g. Eve's pure value or Adam's mixed value, of a stochastic branching game, it is not hard to see that we can distinguish two separate sub-problems. The first one is to find a strategy that realises (or approximates) the optimal, in a predefined notion, value of the game. The second sub-problem is to compute (or, again, approximate) the value of the play which is the result of that strategy. When the strategies $\sigma_m, \pi_m$ are specified, the formula defining the values interprets the strategies as a measure over the set of plays and computes the expected value of the pay-off function $\Phi$ with the respect to this measure.

$$val_{\mathrm{G}}(\sigma_m, \pi_m) = \int_{\Sigma^E_{\mathtt{B}}, \Sigma^A_{\mathtt{B}}, \Sigma^{\mathcal{N}}_{\mathtt{B}}} \Phi\big(\mathrm{eval}_{\mathrm{G}}(\sigma, \pi, \eta)\big)\, \mathrm{d}\sigma_m(\sigma)\, \mathrm{d}\pi_m(\pi)\, \mathrm{d}\eta^*_{\mathtt{B}}(\eta)$$

If the pay-off function $\Phi$ is defined as a set of winning plays, i.e. $\Phi = \chi_L$, this translates to computing the measure of the winning set with respect to this specific measure.

$$val_{\mathrm{G}}(\sigma_m, \pi_m) = \mathrm{eval}_{\mathrm{G}} \# \sigma_m \times \pi_m \times \eta^*_{\mathtt{B}}(L)$$

In this chapter we will focus on the problem of computing the measure of a winning set

which can be expressed by the following question.

**Question 8.1.1.** *Let $L$ be a set of trees given by a computable description, what is the measure of that set?*

While the above question does not specify the measurable space in which we compute the measures, even the basic measures, e.g. a uniform measure, pose a challenge.

A *uniform* (or *standard*) measure $\mu^*$ defined on the set of full trees $\mathcal{T}_\Gamma^\infty$ is the probability measure satisfying that for every finite tree $t \in \mathcal{T}_\Gamma$ we have that $\mu^*(\mathbb{B}_t) = |\Gamma|^{-|nodes(t)|}$. In other words, this measure is such that for every node $u \in \{\mathtt{L},\mathtt{R}\}^*$ and a label $a \in \Gamma$ the probability that in a random tree $t$ the node $u$ is labelled with the letter $a$ is $\frac{1}{|\Gamma|}$.

The following problem of computing the uniform measure of a regular set of infinite trees is unsolved.

**Problem 8.1.2** (open)**.** *Does there exist an algorithm that given a non-deterministic tree automaton $\mathcal{A}$ computes $\mu^*(\mathrm{L}(\mathcal{A}))$?*

To better understand the properties of this special measure let us consider some simple sets of infinite trees. The presented examples not only give an insight into the behaviour of the uniform measures, but can, and will be, used in the proofs of the theorems given in this chapter.

**Lemma 8.1.3.** *Let $t$ be a binary tree over the alphabet $\Gamma$ and $u \in \{\mathtt{L},\mathtt{R}\}^*$ be a position.*

1. *If $t$ is finite and $L = \mathbb{B}_{t,u} \stackrel{def}{=} \{t' \in \mathcal{T}_\Gamma^\infty \mid t \sqsubseteq t' \vartriangle u\}$ then $\mu^*(L) = \Gamma^{-|nodes(t)|}$.*
2. *If $t$ is finite and $L = \{t' \in \mathcal{T}_\Gamma^\infty \mid \exists v.(u \not\sqsubseteq v) \wedge (t \sqsubseteq t' \vartriangle v)\}$ then $\mu^*(L) = 1$.*
3. *If $t$ is infinite and $L = \mathbb{B}_{t,u} = \{t' \in \mathcal{T}_\Gamma^\infty \mid t \sqsubseteq t' \vartriangle u\}$ then $\mu^*(L) = 0$.*

*Proof.* The proof of Item 1 is straightforward. To prove Item 2, let $L_i$ be the language $L_i \stackrel{\text{def}}{=} \mathbb{B}_{t,u\mathtt{L}^i\mathtt{R}} = \{t' \in \mathcal{T}_\Gamma^\infty \mid t \sqsubseteq t' \vartriangle (u\mathtt{L}^i\mathtt{R})\}$. Then, for every $j \geq 0$ we have that $L_j \subseteq L$ and, in consequence, $\overline{L} \subseteq \bigcap_{j\geq 0} \overline{L_j}$. Hence, for every $j \geq 0$ we have that

$$1 - \mu^*(L) = \mu^*(\overline{L}) \leq \mu^*\Big(\bigcap_{j>i\geq 0} \overline{L_i}\Big) = \big(1 - |\Gamma|^{-|nodes(t)|}\big)^j,$$

where the last inequality follows from the fact that the nodes $u\mathtt{L}^l\mathtt{R}$ and $u\mathtt{L}^k\mathtt{R}$ are incomparable for $k \neq l$, thus $L_i$ are independent and

$$\mu^*\Big(\bigcap_{j>i\geq 0} \overline{L_i}\Big) = \prod_{0\leq i<j} \mu^*(\overline{L_i}) = \prod_{0\leq i<j} \big(1 - |\Gamma|^{-|nodes(t)|}\big) = \big(1 - |\Gamma|^{-|nodes(t)|}\big)^j.$$

Taking the limit, we conclude Item 2.

To prove Item 3 let $t_i$ be a sequence of finite trees such that for every $i \geq 0$ we have that $t_i \sqsubseteq t_{i+1} \sqsubseteq t$ and $|nodes(t_i)| < |nodes(t_{i+1})|$. Since the sequence of sets $\mathbb{B}_{t_i,u}$ is decreasing and its limit contains the set $\mathbb{B}_{t,u}$, i.e. $\mathbb{B}_{t_i,u} \supseteq \mathbb{B}_{t_{i+1},u} \supseteq \mathbb{B}_{t,u}$, we have that $\mu^*(\mathbb{B}_{t,u}) \leq \lim_{i \to +\infty} \mu^*(\mathbb{B}_{t_i,u}) = \lim_{i \to +\infty} |\Gamma|^{-|nodes(t_i)|} = 0$. □

The above examples may suggest that the uniform measures enjoy a form of *Kolmogorov's zero-one law*: e.g. in a random tree a given finite structure exists with probability 1, whereas a given infinite structure exists with probability 0. It is not exactly the case, as can be seen by the following examples.

**Example 8.1.4.** *Let* $\Gamma = \{a, b, c\}$.

1. *If* $L_a$ *is the set of trees over the alphabet* $\Gamma$ *with arbitrarily long sequences of* $a$*-labelled nodes, i.e.* $L_a = \{t \in \mathcal{T}_\Gamma^\infty \mid \forall k \geq 0.\ \exists w, v \in \{\mathtt{L}, \mathtt{R}\}^*.\big(\,(|v| \geq k) \wedge \forall\ u \sqsubseteq v.\ t(wu) = a\big)\}$, *then* $\mu^*(L_a) = 1$.

2. *If* $L_{a3}$ *is the language of trees over the alphabet* $\Gamma$ *with an infinite* $\{a\}$*-labelled path starting at the root, i.e.* $L_{a3} = \{t \in \mathcal{T}_\Gamma^\infty \mid \exists w \in \{\mathtt{L}, \mathtt{R}\}^\omega.\forall u \not\sqsubseteq w.\ t(u) = a\}$, *then* $\mu^*(L_{a3}) = 0$.

3. *If* $L_{a2}$ *is the language of trees over the alphabet* $\{a, b\}$ *with an infinite* $\{a\}$*-labelled path starting at the root, i.e.* $L_{a2} = \{t \in \mathcal{T}_{\{a,b\}}^\infty \mid \exists w \in \{\mathtt{L}, \mathtt{R}\}^\omega.\forall u \not\sqsubseteq w.\ t(u) = a\}$, *then* $\mu^*(L_{a2}) = 0$.

4. *If* $L_{ab}$ *is the language of trees over the alphabet* $\Gamma$ *with an infinite* $\{a, b\}$*-labelled path starting at the root, i.e.* $L_{ab} = \{t \in \mathcal{T}_\Gamma^\infty \mid \exists w \in \{\mathtt{L}, \mathtt{R}\}^\omega.\forall u \not\sqsubseteq w.\ t(u) \in \{a, b\}\}$, *then* $\mu^*(L_{ab}) = \frac{1}{2}$.

*Calculating the measures.* To see Item 1, let $t^i$ be a complete tree of height $i$ such that every node in $nodes(t^i)$ is labelled $a$ and let $L^i$ be the language of trees having $t^i$ as a prefix of its sub-tree at some node $u$. Then, by Lemma 8.1.3 part 2 we have that $\mu^*(L^i) = 1$. Moreover, $\bigcap_{i \geq 1} L^i \subseteq L_a$ and $L^{i+1} \subseteq L^i$. Since any measure is monotonically continuous, we have that

$$\mu^*(L_a) \geq \mu^*\Big(\bigcap_{i \geq 1} L^i\Big) = \lim_{n \to +\infty} \mu^*\Big(\bigcap_{n \geq i \geq 1} L^i\Big) = 1.$$

Let $\phi(t)$ stand for "in the tree $t$ there is an infinite $\{a\}$-labelled path starting at root" then Item 2 follows from the fact that the language in question is regular, thus measurable,

and its measure satisfies the following equation.[1]

$$\mu^*(L_{a3}) = \mu^*(\varphi(t) \wedge t(\varepsilon) \neq a) + \\ \mu^*(t(\varepsilon)=a) \cdot \left(\mu^*(\phi(t\vartriangle_{\mathtt{L}})) + \mu^*(\phi(t\vartriangle_{\mathtt{R}})) - \mu^*(\phi(t\vartriangle_{\mathtt{L}}) \wedge \phi(t\vartriangle_{\mathtt{R}}))\right)$$

The equation states that the measure of $L_{a3}$ is equal to the sum of the measures of two sets of trees. The first set consists of all trees $t$ such that the root is not labelled $a$ and the tree $t$ satisfies $\phi$. The second set consist of all trees $t$ such that the root is labelled $a$, the sub-tree at the left child of the root satisfies $\phi$, i.e. $\phi(t\vartriangle_{\mathtt{L}})$, or the sub-tree at the right child of the root satisfies $\phi$, i.e. $\phi(t\vartriangle_{\mathtt{R}})$.

Since the set of all possible trees at left child (or, at right child) of the root is the set of all trees, we get the equation

$$\mu^*(L_{a3}) = \frac{1}{3} \cdot \left(2\mu^*(L_{a3}) - \mu^*(L_{a3})^2\right) = \frac{2}{3}\mu^*(L_{a3}) - \frac{1}{3} \cdot \mu^*(L_{a3})^2$$

implying that $\mu^*(L_{a3}) = 0$ or $\mu^*(L_{a3}) = -1$. Since the measure cannot be negative, we conclude that $\mu^*(L_{a3}) = 0$.

Similarly, in Item 3 we get the equation

$$\mu^*(L_{a2}) = \frac{1}{2} \cdot \left(2\mu^*(L_{a2}) - \mu^*(L_{a2})^2\right) = \mu^*(L_{a2}) - \frac{1}{2} \cdot \mu^*(L_{a2})^2 \tag{8.1}$$

implying that $\mu^*(L_{a2}) = 0$.

In Item 4, we get the equation

$$\mu^*(L_{ab}) = \frac{2}{3} \cdot \left(2\mu^*(L_{ab}) - \mu^*(L_{ab})^2\right) = \frac{4}{3}\mu^*(L_{ab}) - \frac{2}{3} \cdot \mu^*(L_{ab})^2 \tag{8.2}$$

implying that either $\mu^*(L_{ab}) = \frac{1}{2}$ or $\mu^*(L_{ab}) = 0$. Thus we need to look at this example a bit more carefully. Consider a sequence of languages $\{A^i\}_{i\geq 0}$, where $A^0 = \mathcal{T}_\Gamma^\infty$ and $A^i$ is the language such that there is an $\{a, b\}$-labelled path of length $i$ beginning at the root. Then, we claim the following.

**Claim 8.1.5.** $\bigcap_{i\geq 1} A^i = L_{ab}$

*Proof.* Since an infinite path contains sub-paths of arbitrary length, we have that $\bigcap_{i\geq 1} A^i \supseteq L_{ab}$. For the reverse inclusion, let $t \in \bigcap_{i\geq 1} A^i$. Then, for every $i \geq 0$ the tree $t$ has an $\{a, b\}$-labelled path of length $i$. Since $t$ is a binary tree, König's lemma assures that the tree $t$ has an infinite $\{a, b\}$-labelled path. This concludes the proof of the claim. □

[1] Here, we slightly abuse the notation for the sake of readability. We write $\mu^*(\psi)$ instead of $\mu^*(\{t \in \mathcal{T}_\Gamma^\infty \mid \psi\})$.

Now, for every $i \geq 0$ we have that $A^{i+1} \subseteq A^i$ and

$$\mu^*(A^{i+1}) = \frac{2}{3} \cdot \left(2\mu^*(A^i) - \mu^*(A^i)^2\right) = \frac{4}{3}\mu^*(A^i) - \frac{2}{3} \cdot \mu^*(A^i)^2. \tag{8.3}$$

Now, note that if $\mu^*(A^i) \geq \frac{1}{2}$, then $\mu^*(A^{i+1}) \geq \frac{1}{2}$. Indeed, the quadratic function $f(x) = \frac{2}{3}(2x - x^2)$ is monotonically increasing on the interval $[-\infty, 1]$ and we have that $f(1) = \frac{2}{3}$ and $f(\frac{1}{2}) = \frac{1}{2}$. Since $\mu^*(A^0) = 1 \geq \frac{1}{2}$, we conclude that $\mu^*(L_{ab}) = \frac{1}{2}$. □

## 8.2 First-order definable languages and their standard measures

The ideas presented in both Lemma 8.1.3 and Example 8.1.4 allow us to compute the measures of sets of full trees defined by some first-order formulae.

**Theorem 8.2.1.** *Let $\varphi$ be a first-order sentence over the signature $\Gamma \cup \{\varepsilon, s_{\mathtt{L}}, s_{\mathtt{R}}, s\}$. Then the measure $\mu^*(\mathrm{L}(\varphi))$ is rational and computable in three-fold exponential space.*

The proof utilises the *Gaifman locality*, see Section 2.5 for definitions, to partition the formula into two separate sub-formulae. Intuitively, one sub-formulae describes the neighbourhood of the root while the other one describes the tree "far away from the root".

Before we proceed, let us note that in this section we disallow the use of the $\not\trianglelefteq$ relation in formulae. Hence, the Gaifman graf of a tree $t$ is induced by the child relations only, and so is the notion of distance in that tree. For the definition of the distance in a tree see Section 2.1 on page 18, for the definition of the Gaifman graph see Section 2.5.

Now we define the idea of a *root formula*, i.e. a formula that necessarily describes the neighbourhood of the root. Let $\psi(x)$ be an $r$-local formula around $x$. We say that $\psi(x)$ is a root formula if for every tree $t \in \mathcal{T}_\Gamma^\infty$ and every position $u \in \{\mathtt{L}, \mathtt{R}\}^*$ if $t, u \vDash \psi(x)$ then $\mathrm{d}(u, \varepsilon) < r$. Note that every unsatisfiable formula is, by the definition, a root formula.

Let $\varphi$ be a basic $r$-local sentence, i.e. of the form

$$\varphi \stackrel{\text{def}}{=} \exists x_1, \ldots, x_n \left( \bigwedge_{i=1}^{n} \varphi_i^r(x_i) \wedge \bigwedge_{1 \leq i < j \leq n} d(x_i, x_j) > 2r \right), \tag{8.4}$$

where $\varphi_i^r$ are $r$-local and $d(x_i, x_j) > 2r$ is a first-order formula stating that $x_i$ and $x_j$ are at a distance strictly greater than $2r$. We say that $\varphi_i^r$, for $i \in \{1, \ldots, n\}$, is a *root* formula of $\varphi$ if it is a root formula.

**Fact 8.2.2.** *For every satisfiable basic $r$-local sentence there is at most one root formula.*

In other words, only one of $\varphi_i^r$s can describe the $r$-neighbourhood of the root. As, if two such formulas $\varphi_i^r$, $\varphi_j^r$ would be root formulae, then the variables $x_i$ and $x_j$ would be mapped in a distance at most $r-1$ from the root, i.e. in a distance strictly smaller than $2r$ from each other.

Note that, by the definition of satisfiability, for a tree $t \in \mathcal{T}_\Gamma$ and a basic $r$-local sentence $\varphi$ we have that $t \models \varphi$ if and only if there is a function $\tau\colon \{x_1, \ldots, x_n\} \to \{\mathtt{L}, \mathtt{R}\}^*$ mapping variables $x_1, \ldots, x_n$ to positions so that for every $i \in \{1, \ldots, n\}$ we have that $t, \tau(x_i) \models \varphi_i(x_i)$ and for every pair of indices $i \neq j$ we have that $d(\tau(x_i), \tau(x_j)) > 2r$.

**Lemma 8.2.3.** *Let $\varphi$ be a basic $r$-local sentence, i.e. as in Equation* (8.4)*. If $\varphi$ is*

- *not satisfiable, then $\mu^*(\mathrm{L}(\varphi)) = 0$,*
- *satisfiable and has no root formula, then $\mu^*(\mathrm{L}(\varphi)) = 1$,*
- *satisfiable and has a root formula $\varphi^*$,*
  *then for every $t^r$ that is a complete tree of height $2r+1$ we have that*

$$\mu^*(\mathrm{L}(\varphi) \cap \mathbb{B}_{t^r}) = \begin{cases} \mu^*(\mathbb{B}_{t^r}) & \text{if } \exists u \in \{\mathtt{L}, \mathtt{R}\}^{\leq r}.\ t^r, u \models \varphi^*(x); \\ 0 & \text{otherwise.} \end{cases}$$

*Proof.* If $\varphi$ is not satisfiable then $\mathrm{L}(\varphi) = \varnothing$ and $\mu^*(\mathrm{L}(\varphi)) = 0$. Therefore, let us assume that $\varphi$ is satisfiable. By Fact 8.2.2 we know that there is at most one root formula in $\varphi$. Let $I$ be the set of indices of not root formulae, i.e. for $i \in I$ we have that $\varphi_i$ is not a root formula. Since $\varphi$ is satisfiable, for every $i \in I$ there are a finite tree $t_i \in \mathcal{T}_\Gamma$ and a node $u_i \in \{\mathtt{L}, \mathtt{R}\}^*$ of length $|u_i| > r$, such that $t_i, u_i \models \varphi_i(x)$, and the set $\mathit{nodes}(t_i)$ contains the $r$-neighbourhood of $u_i$.

Let $W = \{v_i\}_{i=1}^n$ be a set of $n$ $\sqsubseteq$-incomparable nodes such that for $i \in I$ we have that $|v_i| > 2r$. Let $F = \bigcap_{i \in I} L_i$ where $L_i$ is the set of trees for which $t_i$ is a prefix of a sub-tree at some node below $v_i$, i.e. $L_i \stackrel{\text{def}}{=} \{t' \in \mathcal{T}_\Gamma^\infty \mid \exists u.(v_i \not\sqsubseteq u) \wedge (t_i \sqsubseteq t' \vartriangle u)\}$. By Lemma 8.1.3 every $L_i$ has measure 1, thus we have that $\mu^*(F) = 1$. Moreover, for every tree $t \in F$ and index $i \in I$ there is a node $v_i' \in \{\mathtt{L}, \mathtt{R}\}^*$ such that $\mathrm{d}(v_i, v_i') > r$, $v_i \sqsubseteq v_i'$ and $t, v_i' \models \varphi_i(x)$, see Figure 8.1 for a visual depiction.

Now, if there is no root formula in $\varphi$, i.e. $I = \{1 \ldots, n\}$, then $F \subseteq \mathrm{L}(\varphi)$. Indeed, let $t \in F$, then for $i \neq j$ we have that $\mathrm{d}(v_i', v_j') > 2r$ and we can infer that $t \models \varphi$. Hence, the sequence of inequalities

$$1 = \mu^*(F) \leq \mu^*(\mathrm{L}(\varphi)) \leq 1$$

is sound and proves the second bullet of Lemma 8.2.3.

On the other hand, let there be a root formula in $\varphi$. Without loss of generality, $\varphi_1$ is the root formula and $I = \{2, \ldots, n\}$. Moreover, let $F$ and $v_i'$s be as before, let $t^r$ be a complete

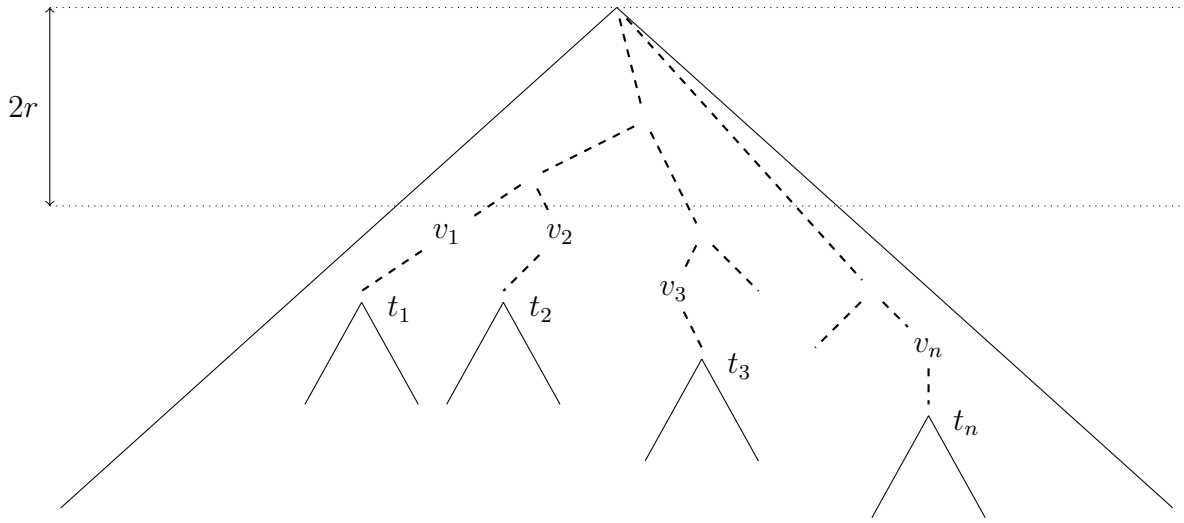
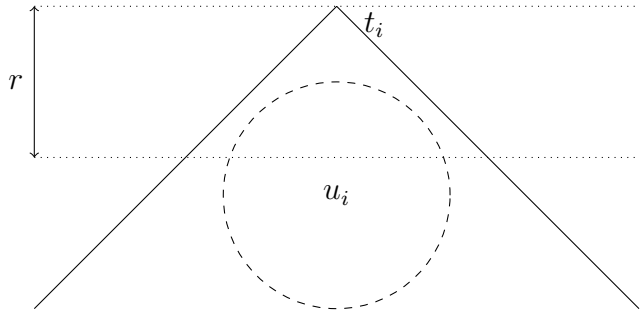



Figure 8.1 – The tree from family $F$ in the proof of Lemma 8.2.3.



tree of height $2r+1$, as stated in the lemma, and $t \in F \cap \mathbb{B}_{t^r}$ be a full binary tree. If there is $u_1 \in \{\mathtt{L},\mathtt{R}\}^{\leq r}$ such that $t^r, u_1 \models \varphi^*(x)$, then we take $v_1^! \stackrel{\text{def}}{=} u_1$. Now, again, for $i \neq j$ we have that $\mathrm{d}(v_i^!, v_j^!) > 2r$ and for all $i \in I$ we have that $t, v_i^! \models \varphi_i(x_i)$. In other words, if there is $u_1 \in \{\mathtt{L},\mathtt{R}\}^{\leq r}$ such that $t^r, u_1 \models \varphi^*(x)$ then $F \cap \mathbb{B}_{t^r} \subseteq \mathrm{L}(\varphi) \cap \mathbb{B}_{t^r}$. Moreover, since $F$ is of measure 1, the following sequence of inequalities is sound

$$\mu^*(\mathbb{B}_{t^r}) = \mu^*(F \cap \mathbb{B}_{t^r}) \leq \mu^*(\mathrm{L}(\varphi) \cap \mathbb{B}_{t^r}) \leq \mu^*(\mathbb{B}_{t^r}).$$

Furthermore, if there is no such $u_1$ then $\varphi_1$ is not satisfiable in $t^r$. Since $\varphi_1$ is a root formula, we have that $\mu^*(\mathrm{L}(\varphi) \cap \mathbb{B}_{t^r}) = 0$, which concludes the proof. □

Intuitively, the above lemma states that, when we consider the uniform measure and a basic $r$-local sentence, the behaviour of the sentence is almost surely defined by the neighbourhood of the root. This intuition can be formalised as follows.

**Lemma 8.2.4.** *Let $\varphi$ be a basic $r$-local sentence. Then there is a sentence $\varphi^*$ such that for every complete tree $t^r$ of height $2r+1$ we have that*

$$\mu^*(\mathrm{L}(\varphi) \cap \mathbb{B}_{t^r}) = \mu^*(\mathrm{L}(\varphi^*) \cap \mathbb{B}_{t^r}).$$

*Moreover, for every full tree $t \in \mathbb{B}_{t^r}$ we have that $t \models \varphi^*$ if and only if $t^r \models \varphi^*$.*

*Proof.* If $\varphi$ has a root formula $\varphi_i$, then we take $\varphi^* \stackrel{\text{def}}{=} \exists x.\varphi_i(x) \wedge \mathrm{d}(x,\varepsilon) < r$. If $\varphi$ has no root formulae, but is satisfiable then we take $\varphi^* \stackrel{\text{def}}{=} \exists x.\varepsilon(x)$. Otherwise, we take $\varphi^* \stackrel{\text{def}}{=} \bot$. □

The formula $\varphi^*$ is called the *reduction* of $\varphi$. Before we show how to compute the reduction of a basic $r$-local formula, we recall a known result.

**Lemma 8.2.5** (Folklore)**.** *There is an algorithm that given a first-order formula $\varphi$ and a finite tree $t$ decides whether $t \models \varphi$ in space polynomial with respect to the size of the formula $\varphi$ and with respect to the size of the set of nodes of the tree $t$.*

*Proof.* The lemma is folklore; the property can be easily verified using an *APTIME* algorithm. □

Now we show how to compute the reduction.

**Lemma 8.2.6.** *Given a basic $r$-local sentence $\varphi$ one can compute its reduction $\varphi^*$ in space polynomial in the size of the formula and doubly exponential in the unary encoding of $r$.*

*Proof.* An $r$-local formula $\psi(x)$ is satisfiable in some full binary tree if, and only if, it is satisfiable in a node $u$ of some tree of height $2r+1$, such that $|u| < r+1$. Thus, to check the satisfiability of any formula $\varphi_i$, we need to check the trees of height at most $2r+1$.

Moreover, to check whether $\varphi_i$ is a root formula, we need to check whether $\varphi_i$ is satisfiable and the formula $\varphi_i(x) \wedge \mathrm{depth}_{\geq r}(x)$, where $\mathrm{depth}_{\geq r}(x)$ stands for a first-order formula stating that $x$ is at depth $r + 1$ or more, is not satisfied in any full binary tree. This, again, can be check by iterating all trees of height at most $2r + 3$.

Thus, the reduction of $\varphi$ can be computed by the algorithm computeReduction presented in Algorithm 1. The complexity follows from Lemma 8.2.5. □

**Algorithm 1** computeReduction

**Require:** a first-order formula $\varphi$ in Gaifman normal form
  $\psi = \top$
  $S \leftarrow \{i \mid \varphi_i \text{ is not satisfiable}\}$
  **if** $|S| > 0$ **then**
    **return** $\psi$
  **end if**
  $S \leftarrow \{i \mid \varphi_i \text{ is a root formula}\}$
  **if** $|S| = 0$ **then**
    $\psi \leftarrow \exists x.\ \varepsilon(x)$
  **else if** $|S| = 1$ **then**
    $\psi \leftarrow \exists x.\varphi_i(x) \wedge \mathrm{d}(x, \varepsilon) < r$
  **else**
    $\psi \leftarrow \bot$
  **end if**
  **return** $\psi$

Lemma 8.2.4 can be extended to Boolean combinations of $r$-local basic formulae by the following property of measurable sets.

**Lemma 8.2.7.** *Let $M$ be a measurable space with measure $\mu$, $W$ be a measurable set and $\{S_i\}_{i \in I}$ be a family of measurable sets such that for every $i \in I$ either $\mu(W \cap S_i) = 0$ or $\mu(W \cap S_i) = \mu(W)$. Then, for every set $S$ in the Boolean algebra of sets generated by $\{S_i\}_{i \in I}$, we have that either $\mu(W \cap S) = 0$ or $\mu(W \cap S) = \mu(W)$.*

*Proof.* The proof goes by a standard inductive argument. □

Hence, by Lemma 8.2.3 and the above lemma, we obtain the following.

**Lemma 8.2.8.** *Let $\phi$ be a Boolean combination of basic $r$-local formulae and $t$ be a complete tree of height $2r+1$. Then, $\mu^*(\mathrm{L}(\phi) \cap \mathbb{B}_t) = \mu^*(\mathrm{L}(\phi^*) \cap \mathbb{B}_t)$, where $\phi^*$ is the reduction of $\phi$, i.e. the Boolean combination $\phi$ with its every basic $r$-local sentence $\varphi$ replaced by its reduction $\varphi^*$.*

*Moreover,* $\mu^*(\mathrm{L}(\phi^*) \cap \mathbb{B}_t) = \begin{cases} \mu^*(\mathbb{B}_t) & \text{if } t \vDash \phi^*; \\ 0 & \text{otherwise} \end{cases}$

*Proof.* For every complete tree $t^r$ of height $r$ we use Lemma 8.2.7 with $M$ being the set of full trees $\mathcal{T}_\Gamma^\infty$, $\mu$ being the uniform measure $\mu^*$, and $W$ being the set $\mathbb{B}_{t^r}$. The sets $S_i$ are the sets of trees defined by the basic $r$-local formulae and $S = \mathrm{L}(\phi)$. By Lemma 8.2.4, the assumptions of Lemma 8.2.7 are satisfied. □

With the above lemmas, we can finally prove Theorem 8.2.1.

*Proof of Theorem 8.2.1.* Let $\varphi$ be a first-order sentence as in the theorem. We utilise the Gaifman locality theorem (see Theorem 2.5.2 on page 27) to translate the sentence $\varphi$ into a Boolean combination $\phi$ of basic $r$-local sentences. Now, let $\phi^*$ be the reduction of $\phi$, as in Lemma 8.2.8, and let $S \subseteq \mathcal{T}_\Gamma$ be the set of all complete trees of height $h = 2r+1$. Then,

$$\begin{array}{lllll} \mu^*(\mathrm{L}(\phi)) & \overset{1}{=} & \mu^*(\mathrm{L}(\phi) \cap (\bigcup_{t\in S} \mathbb{B}_t)) & \overset{2}{=} & \mu^*(\bigcup_{t\in S} (\mathrm{L}(\phi) \cap \mathbb{B}_t)) \\ & \overset{3}{=} & \sum_{t\in S} \mu^*(\mathrm{L}(\phi) \cap \mathbb{B}_t) & \overset{4}{=} & \sum_{t\in S} \mu^*(\mathrm{L}(\phi^*) \cap \mathbb{B}_t) \\ & \overset{5}{=} & \sum_{t\in S \wedge t\vDash\phi^*} \mu^*(\mathbb{B}_t) & \overset{6}{=} & |\{t \in S \mid t \vDash \phi^*\}| \cdot \frac{1}{|\Gamma|^{2^{h+1}-1}}. \end{array}$$

The first equation follows from the fact that $\{\mathbb{B}_t \mid t \in S\}$ is a partition of the space. The second from operations on sets and the third is a simple property of measures. The fourth follows from the first part of Lemma 8.2.8, while the fifth follows from the second part of this lemma. The last equation is a consequence of the fact that $\mu^*(\mathbb{B}_t) = |\Gamma|^{-|dom(t)|}$, see e.g. Lemma 8.1.3.

Since $\mu^*(\mathrm{L}(\phi)) = \frac{|\{t\in S | t\vDash\phi^*\}|}{|\Gamma|^{2^{h+1}-1}}$, it is clearly rational and can be computed by counting how many complete trees of height $h = 2r+1$ satisfy the reduction of $\phi$. The pseudo-code of the algorithm, called computeMeasureFO, is presented in Algorithm 2.

The complexity upper bound comes from the fact that translating a first-order formula $\varphi$ into its Gaifman normal form can be done in three-fold exponential time and can produce a three-fold exponential formula $\phi$ in result, see [13] for details. The resulting formula $\phi$ is a Boolean combination of basic formulae, thus, we can compute its reduction in three-fold exponential space. The function computeReduction$^*$ computes the reduction $\phi^*$ of the Boolean combinations by replacing the formulae used in the Boolean combination with their reductions. This can be done in the required complexity, see Lemma 8.2.6 and note that the size of the formula dominates the constant $r$. Finally, the last part of the algorithm requires us to check the formula $\phi^*$ against three-fold exponential number of trees of size that is two-fold exponential in the size of the original formula. Since model checking of a first-order formula can be done in polynomial space with respect to the size of the tree and to the size of the formula, see Lemma 8.2.5, we get the upper bound. □

**Algorithm 2** computeMeasureFO

**Require:** a first-order formula $\varphi$ and a positive number $h$
$S \leftarrow$ the set of all complete trees of height $h$
$\phi \leftarrow$ computeGaifmanForm($\varphi$)
$\phi \leftarrow$ computeReduction$^*(\phi)$
$S \leftarrow \{t \in S \mid t \vDash \phi\}$
**return** $|S| \cdot |\Gamma|^{-2^{h+1}+1}$

## 8.3 First-order definable languages with descendant

The technique used to prove Theorem 8.2.1 cannot be extended to formulae utilising the descendant relation because when we allow the descendant relation, the diameter of the Gaifman graph of any tree is at most two. Additionally, as presented in Proposition 8.3.1 below, sets of full trees defined by such formulae can have irrational measures.

**Proposition 8.3.1.** *There is a set $L$ of full binary trees over an alphabet $\Gamma$ such that $L$ is definable by a first-order formula over the signature $\Gamma \cup \{s_{\mathtt{L}}, s_{\mathtt{R}}, \not\trianglelefteq\}$ and the standard measure of $L$ is irrational.*

*Proof.* Let $\Gamma = \{a, b\}$, we define a language $L$ in the following way $L \stackrel{\text{def}}{=} \{t \in \mathcal{T}^{\infty}_{\{a,b\}} \mid$ for every path the earliest node labelled $b$ (if exists) is at an even depth$\}$. Now, the measure $\mu^*(L)$ is irrational, and there is a language $L'$ definable by a first-order formula over the signature $\Gamma \cup \{s_{\mathtt{L}}, s_{\mathtt{R}}, \not\trianglelefteq\}$ such that $\mu^*(L') = \mu^*(L)$. We start by computing the measure of $L$, then we will define $L'$.

Observe that the measure $\mu^*(L)$ satisfies the following equation.

$$\mu^*(L) = \mu^*\big(\{t \in \mathcal{T}^{\infty}_{\{a,b\}} \mid t(\varepsilon){=}b\}\big) + \mu^*\big(\{t \in \mathcal{T}^{\infty}_{\{a,b\}} \mid t(\varepsilon){=}t(\mathtt{L}){=}t(\mathtt{R}){=}a\}\big) \cdot \mu^*(L)^4$$

After substituting the appropriate values, we obtain the equation

$$\mu^*(L) = \frac{1}{2} + \frac{1}{8}\mu^*(L)^4 \tag{8.5}$$

which, by the *rational root theorem*, see e.g. [7] page 116, has no rational solutions.

To conclude the proof, we will describe how to define the language $L'$. The crux of the construction comes from the beautiful example by Potthoff, see [44, Lemma 5.1.8]. We will use the following interpretation of the lemma: one can define in first-order logic over the signature $\{a, b, s_{\mathtt{L}}, s_{\mathtt{R}}, \not\trianglelefteq\}$ a language of finite trees over the alphabet $\{a, b\}$ where every node labelled $a$ has exactly two children and every node labelled $b$ is a leaf on an even depth.

To construct $L'$ we simply utilise the formula defining the language in the Potthoff's example to define $L'$ by substituting the formula describing a leaf with the formula describing

a first occurrence of the label $b$ on a path: $\varphi_{\text{leaf}}(x) \stackrel{\text{def}}{=} b(x) \wedge \forall y.\ (y \sqsubseteq x) \Longrightarrow a(y)$. Note that the set $L'$ agrees with $L$ on every tree that has a label $b$ on every infinite path from the root. On the other hand, the truth value of the modified formula on the trees that have an infinite path from the root with no nodes labelled $b$, i.e. on the set $L_{a2}$ from Example 8.1.4, is of no concern to us. Indeed, as previously shown, the standard measure of the set $L_{a2}$ is 0.

To be precise, for every tree $t \in \mathcal{T}^{\infty}_{\{a,b\}} \setminus L_{a2}$ we have that $t \in L \iff t \in L'$, where $L_{a2}$ is the language from Example 8.1.4. Therefore, we have that $L \cup L_{a2} = L' \cup L_{a2}$. Since $\mu^*(L_{a2}) = 0$, we have that

$$\mu^*(L) = \mu^*(L \cup L_{a2}) = \mu^*(L' \cup L_{a2}) = \mu^*(L'),$$

which concludes the proof. □

## 8.4 Conjunctives queries and standard measure

As shown in Proposition 8.3.1 allowing the descendant relation in first-order formulae permits irrational values of measures. Nevertheless, we can recover rational values and computability when we restrict the formulae to the positive existential formulae using only atomic formulae and conjunction, i.e. to the conjunctive queries. For a definition of conjunctive queries see Section 2.5.

Note that introducing the ancestor/descendant relation to the tree structure causes that every two nodes in the Gaifman graph are in distance at most two from each other. Thus, for the purpose of having a relevant definition of the distance in the tree, we retain the child related notion of distance, i.e. in this section, as before, the notion of the distance is induced by the child relations only.

**Theorem 8.4.1.** *Let $q$ be a conjunctive query over the signature $\Gamma \cup \{\varepsilon, s_{\mathtt{L}}, s_{\mathtt{R}}, s, \sqsubsetneq\}$. Then, the standard measure of the language $\mathrm{L}(q)$ is rational and computable in exponential space.*

To prove the theorem we will modify the concept of *firm* sub-patterns, used e.g. in [40]. Intuitively, a *firm sub-pattern* is a maximal part of a conjunctive query that has to be mapped in a tree in a small neighbourhood. To recall the notion of a pattern see Section 2.5, page 27.

A sub-pattern $\pi'$ is firm if it is a sub-pattern of a pattern $\pi$ induced by vertices belonging to a maximal strongly connected component in the graph $G_\pi = \langle V, E\rangle$ such that $\langle x, y\rangle \in E$ if either $xs_{\mathtt{L}}y$, $ys_{\mathtt{L}}x$, $xs_{\mathtt{R}}y$, $ys_{\mathtt{R}}x$, $xsy$, $ysx$, $x \sqsubsetneq y$, or $\varepsilon(x)$. In particular, a pattern is firm if it has a single strongly connected component. We say that a sub-pattern is *rooted* if it contains the predicate $\varepsilon$.

**Proposition 8.4.2.** *Let $\pi = \langle V, V_\varepsilon, E_{\mathtt{L}}, E_{\mathtt{R}}, E_s, E_{\sqsubsetneq}, \lambda_\pi\rangle$ be a firm pattern. Then, for every tree $t$ such that $t \models \pi$, for every two vertices $x, y$ in $V$, and for every homomorphism $h\colon \pi \to t$*

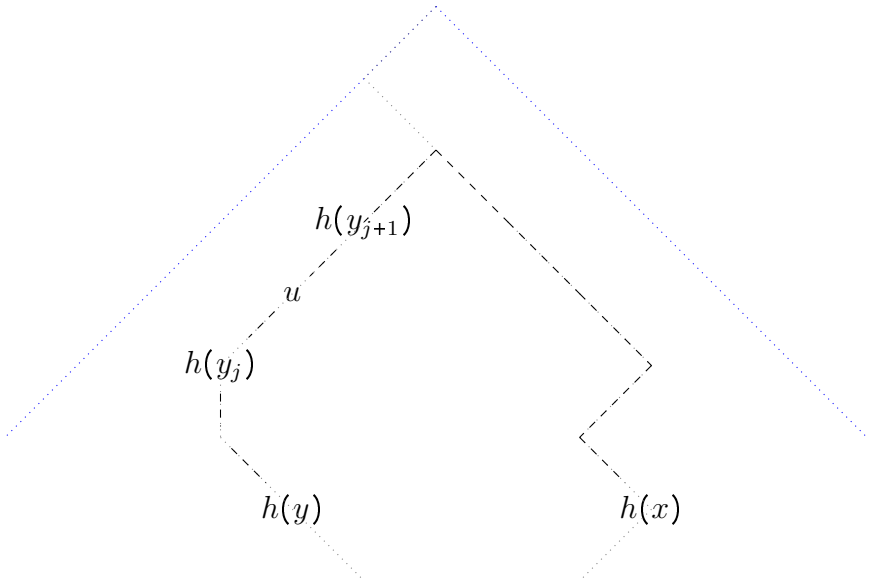



Figure 8.2 – A possible placement of nodes in the proof of Proposition 8.4.2.

*we have that* $d(h(x), h(y)) < |\pi|$*. Moreover, if* $\pi$ *is rooted then for every vertex* $x$ *we have that* $d(h(x), \varepsilon) < |\pi|$*.*

*Proof.* Let us assume otherwise, let $n = |\pi|$. Then, there is a tree $t$, a homomorphism $h$, and two vertices $x, y$ such that $t \models \pi$ and $d(h(x), h(y)) \geq n$. We claim that $x$ and $y$ cannot be in the same strongly connected component.

Since for some $m$ we have that $d(h(x), h(y)) = m - 1 \geq n$, there is a sequence of distinct nodes $u_1, u_2, \ldots u_m$ such that $u_1 = h(x)$, $u_m = h(y)$ and for every $i$, $u_i$ and $u_{i+1}$ are in a child relation, i.e. $s(u_i, u_{i+1})$ or $s(u_{i+1}, u_i)$. Moreover, there is a node $u$ such that $u = u_i$ for some $1 \leq i \leq m$, $u \notin h(\pi)$, and one of the nodes $h(x)$ or $h(y)$ is a descendant of $u$, i.e. $u \sqsubsetneq h(x)$ or $u \sqsubsetneq h(y)$. Without loss of generality, let us say that $u \sqsubsetneq h(y)$. Or, more precisely, that $u_{\text{L}} \sqsubseteq h(y)$. Figure **??** presents a possible arrangement of the nodes.

If $x$ and $y$ were in the same strongly connected component then there would be a path in the graph $G_\pi$ that connects $y$ to $x$, i.e. a sequence of vertices $y_1, y_2, \ldots, y_k$, for some $k$, such that $y_1 = y$, $y_k = x$, and for every $i = 1, \ldots, k-1$ there is an edge between $y_i$ and $y_{i+1}$ in $G_\pi$. In particular, this would imply that for every $i$ we have that $h(y_i)$ and $h(y_{i+1})$ are $\sqsubseteq$-comparable. Now, there would also exist an index $j \in \{1, \ldots, k-1\}$ such that $h(y_{j+1}) \sqsubsetneq u \sqsubsetneq h(y_j)$. Indeed, if there would be no such index, then all the vertices $y_i$ would satisfy $u_{\text{L}} \sqsubseteq h(y_i)$, as $y_i$ and $y_{i+1}$ are $\sqsubseteq$-comparable for every index $i$. But this is impossible because if $u_{\text{L}} \sqsubseteq h(y_i)$ for all $i$, then we would have that $u_{\text{L}} \sqsubseteq h(y_k) = h(x)$. Now, since $u_{\text{L}} \sqsubseteq h(y)$ and $u_{\text{L}} \sqsubseteq h(x)$, then, by the definition of the distance, $u$ would not belong to the sequence $u_1, \ldots, u_m$. Which is a contradiction with our assumption.

Therefore, there is an index $j$ such that $h(y_{j+1}) \not\sqsubseteq u \not\sqsubseteq h(y_j)$. Thus, by the definition of $G_\pi$ we have that either $y_j s_{\mathtt{L}} y_{j+1}$, $y_{j+1} s_{\mathtt{L}} y_j$, $y_j s_{\mathtt{R}} y_{j+1}$, $y_{j+1} s_{\mathtt{R}} y_j$, $y_j s y_{j+1}$, $y_{j+1} s y_j$, $y_j \not\sqsubseteq y_{j+1}$, or $\varepsilon(y_j)$. Neither of child relations is possible because the distance between $h(y_j)$ and $h(y_{j+1})$ is at least two. Similarly, both $y_j \not\sqsubseteq y_{j+1}$ and $\varepsilon(y_j)$ are impossible because we have that $u \not\sqsubseteq h(y_j)$. Hence, there is no such sequence $y_1, \dots y_k$, thus $x$ and $y$ cannot belong to the same strongly connected component. This proves the first part of the lemma.

Now, if $\pi$ is rooted then there is a vertex $y$ such that for every homomorphism $h$ we have that $h(y) = \varepsilon$. Hence, by the first part of the lemma, for all vertices $x \in \pi$ we have that $d(h(x), \varepsilon) = d(h(x), h(y)) < n$. □

**Graph of firm sub-patterns** Let $q$ be a conjunctive query. Consider a graph $\mathrm{G}_q^F = \langle V, E \rangle$ where $V$ is the set of firm sub-patterns of $q$ and there is an edge $\langle v_1, v_2 \rangle \in E \subseteq V \times V$ between two vertices $v_1, v_2$ if and only if there is an $\not\sqsubseteq$ labelled edge between some two vertices $w_1 \in v_1, w_2 \in v_2$. We call this graph the *graph of firm sub-patterns* of the conjunctive query $q$.

**Proposition 8.4.3.** *The directed graph $G_q^F$ of firm sub-patterns of a conjunctive query $q$ is acyclic and has at most one rooted firm sub-pattern. We call that sub-pattern the root sub-pattern.*

*Proof.* By the definition of the firm sub-patterns, every node with the predicate $\varepsilon$ ends up in the same maximal strongly connected component. The acyclicity follows directly from the fact that firm sub-patterns are the maximal strongly connected components. □

As in the case of root formulae, the root pattern decides the behaviour of a satisfiable conjunctive query, as expressed by the following lemma.

**Lemma 8.4.4.** *Let $q$ be a conjunctive query over the signature $\{a, b, \varepsilon, s_{\mathtt{L}}, s_{\mathtt{R}}, s, \not\sqsubseteq\}$. Then, either*

- *$q$ is not satisfiable and $\mu^*(\mathrm{L}(q)) = 0$,*
- *$q$ is satisfiable, has no root sub-pattern, and $\mu^*(\mathrm{L}(q)) = 1$,*
- *or $q$ is satisfiable, has a root sub-pattern $p$, and $\mu^*(\mathrm{L}(q)) = \mu^*(\mathrm{L}(p))$.*

*Proof.* If $q$ is not satisfiable then $\mathrm{L}(q) = \emptyset$ and so $\mu^*(\mathrm{L}(q)) = 0$. Let $q$ be satisfiable, i.e. there is a tree $t^q$ and a homomorphism $h \colon \mathrm{G}_q \to t^q$. Let $t^r$ be a finite tree such that $h(\mathrm{G}_q) \subseteq \mathit{nodes}(t^r)$ and let the set $S \subseteq \mathcal{T}_\Gamma^\infty$ be the set of all trees $t$ such that for every node $u \in \{\mathtt{L}, \mathtt{R}\}^{|q|+1}$ the tree $t^r$ is a prefix of $t \vartriangle u$. By Lemma 8.1.3, we have that $\mu^*(S) = 1$.

If $q$ has no root firm sub-pattern, then $S \subseteq \mathrm{L}(q)$ and we have that

$$\mu^*(\mathrm{L}(q)) \geq \mu^*(S) = 1.$$

On the other hand, if $q$ has a root sub-pattern $p$ then for every tree $t \in S$ we have that $t \models q$ if and only if $t \models p$. Thus, $\mathrm{L}(q) \cap S = \mathrm{L}(p) \cap S$ and since $\mu^*(S) = 1$, we have that

$$\mu^*(q) = \mu^*(\mathrm{L}(q) \cap S) = \mu^*(\mathrm{L}(p) \cap S) = \mu^*(p). \qquad \square$$

In other words, the problem of computing the uniform measure of a set of full binary trees defined by a conjunctive query reduces to the following problem of counting the models of a fixed height, see page 17.

**Problem 8.4.5** (Conjunctive queries counting)**.**

*Input:* *A conjunctive query $q$ and a natural number $n$.*
*Output:* *Number of complete binary trees of height $n$ that satisfy $q$.*

**Proposition 8.4.6.** *Problem 8.4.5 can be solved in space exponential in the unary encoding of $n$ and polynomial in the size of the query.*

*Proof.* All we need is to enumerate all the binary trees of height $n$ and check whether they satisfy the query. The number of such trees is exponential in the unary encoding of $n$ and the model checking can be done in polynomial space with respect to both the query and the size of the tree, see Lemma 8.2.5. $\square$

We infer Theorem 8.4.1 as an immediate consequence.

*Proof of Theorem 8.4.1.* In polynomial space we can check whether the query is satisfiable, see e.g. [3], and in polynomial time compute its root sub-pattern: it is folklore that one can compute all strongly connected components of a directed graph in polynomial time.

Now, by Lemma 8.4.4, if the query is not satisfiable then the measure is 0; if it is satisfiable, but there is no root sub-pattern then the measure is 1. Thus, the only case left is when the query is satisfiable and has a root sub-pattern $q'$.

If it is the case, then $\mu^*(\mathrm{L}(q)) = \mu^*(\mathrm{L}(q'))$. Since $q'$ is a root pattern, then by Proposition 8.4.2 for any tree $t$, any homomorphism $h\colon q' \to t$, and any vertex $v$ of the query we have that $|h(v)| < |q'|$. Thus, for any full binary tree $t \in \mathcal{T}_\Gamma^\infty$ to decide whether $t \models q'$ we only need to check the prefix of $t$ that is of height $|q'|$.

Since the sets $B_{t'}$, where $t'$ ranges over complete trees of height $|q'|$, form a partition of $\mathcal{T}_\Gamma^\infty$, to compute $\mu^*(\mathrm{L}(q'))$ it is enough to iterate over all such trees and compute how many of them satisfy $q'$.

Since every complete tree of height $|q'|$ is of exponential size with respect to $q'$ and they can be iterated in exponential space, we infer that the measure $\mu^*(\mathrm{L}(q'))$ can be computed in exponential space.

To conclude the proof,s we observe that the resulting measure is a finite sum of the measures of the sets $B_{t'}$. Those measures are rational by Lemma 8.1.3, hence the resulting measure is rational as well. □

As in the case of first-order formulae, we can lift Theorem 8.4.1 to Boolean combinations of conjunctive queries.

**Corollary 8.4.7.** *Let $\varphi$ be a Boolean combination of conjunctive queries. Then, the standard measure $\mu^*(\mathrm{L}(\varphi))$ is rational and can be computed in exponential space.*

*Proof.* This is an immediate consequence of Lemma 8.2.7 and Theorem 8.4.1.

Indeed, by Lemma 8.2.7 and Theorem 8.4.1, patterns in $\varphi$ can be replaced by their root patterns without the change of measure, or by the query $\exists x.\ a(x) \wedge b(x)$ if they are unsatisfiable. The rest of the reasoning follows as in the proof of Theorem 8.4.1. □

We conclude this chapter with a lower bound. We show that deciding whether the standard measure of a Boolean combination of conjunctive queries is positive is intractable, as expressed in the following theorem.

**Theorem 8.4.8.** *Let $\varphi$ be a Boolean combination of conjunctive queries. Then, deciding whether $\mu^*(\mathrm{L}(\varphi)) > 0$ is NEXP-complete.*

*Proof.* The upper bound is an immediate consequence of Lemma 8.2.7 and Theorem 8.4.1.

Again, by Lemma 8.2.7 and Theorem 8.4.1, patterns in $\varphi$ can be replaced by their root patterns without the change of the measure.

Now, to prove that the measure is positive, we only need to find a complete tree $t$ of height $|\pi|$ that satisfies the new combination of queries. Such a tree is of size exponential in the size of the original combination of queries and can be easily verified in non-deterministic exponential time: simply guess the homomorphism.

For the lower bound, we refer to the proof of Theorem 3 in Murlak et al. [40], the case of non-recursive schemas: it is shown there that verifying if a tree $t$ as above exists is *NEXP*-hard (the proof is given in a slightly different set-up but can be easily adapted to our needs). □

# Chapter 9

# Plantation game

One of the crucial directions of research, and science in general, is application of the theoretical models to the problems existing in nature. In this chapter, we will present the results of our cooperation with the *Institut Agronomique néo-Calédonien*, who kindly provided us with data concerning seven years of functioning of an experimental fruit farm situated in Pocqereux near La Foa, New Caledonia.

The main purpose of this chapter in not to find a perfect model emulating an ecosystem of a fruit farm situated in a tropical zone nor to use the rich theory of branching games, but to present a viable and verifiable way to use discrete stochastic game-based models in real-life applications.

Hence, we feel inclined to issue two comments. Firstly, due to not using the theory of branching games, we have made an effort to make this chapter self-contained. Nevertheless, while we do not use branching games explicitly, we use a form of Gale-Stewart games, or games on graphs, in our approach. Thus, the introductory chapters can be used as a reference when searching for definitions and occasional references to previous chapters are present in the text. Secondly, we are aware of the fact that many crucial improvements can be made. Thus, the state of the work described in this chapter should be considered as a work in progress.

Finally, we would like to point out that, though we do not use branching games as the game of choice in this chapter, the proof of Theorem 6.2.1 suggests that the *Markov decision processes*, that are used in this chapter, can be encoded as branching games.

## 9.1 Data overview, preparation, and selection

Data was being gathered between February 2000 and December 2006, twice a week, in irregular 3 or 4 days periods. In total, results of 719 separate measurements are listed in the dataset. The dataset contains information concerning

- methodology of the experiment, i.e. dates and descriptions of measurements,
- weather conditions, e.g. temperature, humidity, rainfall,
- secondary environmental observations, e.g. presence of fruits, evapotranspiration,
- quantities of flies, with particular emphasis on *Bactrocera psidii, B. tryoni, B. curvipennis*,
- and human activity, in form of pest control interventions.

The meteorological data was recorded once a week by a local meteorological station.The data concerning the chemical treatments has been recorded on the day of the spraying and consists of the dates, names of used chemicals and dosages. The pest activity in form of the number of fruit flies has been gathered biweekly and is recorded as a number of specimens collected from traps placed in the experimental orchard.

Some records are incomplete, with a substantial loss of data in September 2006, a month (10 consecutive entries) of pest levels is missing. We have dealt with the missing data by replacing it with averages taken from neighbouring measurements and rejecting any data collected after the 22.08.2006, i.e. we have deleted 35 entries, which contributes to less than 5% of the data. The difference in the data collection of weather conditions and insect presence was overcome by unifying the biweekly data and taking the averages. After the adjustments, the final dataset consists of 342 measurements gathered between the 11.02.2000 and the 22.08.2006.

After consultations with our colleagues from the *Institut Agronomique néo-Calédonien*, we have decided to select the following attributes to use in our model:

- relative humidity,
- temperature,
- evapotranspiration,
- fruit presence,
- treatments,
- number of *B. psidii*,
- and number of *B. tryoni*.

First five attributes are the environmental parameters, with only the treatments[1] being directly connected to human actions. The last two are our prediction objectives, i.e. the quantities that we want to forecast.

[1]Information about treatment is binary: we are interested only in the fact that a spraying occurred in given point of time, our model ignores the type of used chemicals.

## 9.2 Model description and extraction

To model interactions and dynamics reflected by the collected data, we use the *Partially Observed Markov Decision Process* (ab. *POMDP*) – a simple generalization of *Hidden Markov Models* which are a tool that has been used in time series analysis and forecasting to a great success.

A *POMDP* is a tuple $\mathcal{M} = \langle \pi, A, B \rangle$ where

- $\pi \in \mathit{dist}(Q)$ is the initial internal state of the system,
- $A : \{E \times P\} \to (Q \to \mathit{dist}(Q))$ is an indexed family of *transition* matrices,
- and $B : E \to (Q \to \mathit{dist}(P))$ is an indexed family of *emission* matrices,

with $Q, E, P$ being some finite sets. The set $Q$ is called the state space, elements of $P$ are called *observations*, and elements of $E$ are called *environmental types*.

The idea behind the use of a *POMDP* is as follows. A system we want to model can be seen as a discrete time process that can be arbitrarily complex and for which we cannot presume to possess the perfect information about its current state. Furthermore, the only information about the state of the system can be inferred from its visible behaviour, represented by a sequence of discrete signals $o$ emitted by the system, we call them *observations* and the set of all observations is denoted by $P$. We assume that the observations depend on both: the internal state of the system $q \in Q$ and the environment in which our system operates $e \in E$. Thus, from the sequence of pairs $(E \times P)^*$ we can assess our *belief* about the real state of the process. The belief is represented as a distribution over the internal, hidden states $\pi \in \mathit{dist}(Q)$ of the *POMDP*.

With a *POMDP* $\mathcal{M}$ and an auxiliary information in a form of a possibly infinite word $w = \langle e_i, o_i \rangle \in (E \times P)^\omega$, we associate two sequences of distributions $\mathcal{M}(w)$ and $\mathcal{M}^*(w)$.

The sequence $\mathcal{M}(w) = \langle \tau_i \rangle_{i=0}^{\omega}$ is defined in the following way:

- $\pi_0 = \pi$,
- $\pi_i = A(\langle e_i, o_i \rangle) \cdot \pi_{i-1}$ for $i > 0$,
- $\tau_i = B(e_i) \cdot \pi_i$ for $i \geq 0$.

The sequence $\mathcal{M}^*(w) = \langle \tau_i^* \rangle_{i=0}^{\omega}$ is also defined in an inductive way:

- $\pi_0^* = \pi$,
- $\tau_i^* = B(e_i) \cdot \pi_i^*$,
- $\pi_i^* = \sum_{o \in P} \tau_{i-1}^*(o) \cdot A(\langle e_{i-1}, o \rangle) \cdot \pi_{i-1}^*$ for $i > 0$.

Since function $\mathcal{M}^*$ ignores the second coordinate of the elements, we will sometimes abuse the notation and write $\mathcal{M}^*(w)$ for a sequence $w \in E^\omega$.

Any infinite sequence of distributions $d \in (dist(P))^{\omega}$ defines a measure over the family of infinite sequences of observations $P^{\omega}$. For a word $w$ and a POMDP $\mathcal{M}$, we will denote the measure defined by $\mathcal{M}(w)$ as $\mu_w$ and the measure defined by $\mathcal{M}^*(w)$ as $\mu^*_w$. Those two measures are connected in the following manner.

We say that a *POMDP* $\mathcal{M}$ is *faithful* to sequence $w = \langle e_i, o_i \rangle \in (E \times P)^{\omega}$ if $\mathcal{M}$ accurately predicts the visible part of the sequence, i.e if $\mathcal{M}(w)(i)(o_i) = 1$ for $i > 0$.

**Observation 9.2.1.** *If a partially observable Markov decision process $\mathcal{M}$ is faithful to sequence $w$, then $\mathcal{M}(w) = \mathcal{M}^*(w)$ and, thus, $\mu_w = \mu^*_w$.*

In general, we consider two versions of environmental sensitivity: one is *observation sensitive* where transitions between internal states of the model depend on both the environmental type and the observed infestation level. The other is *observation oblivious* where the *POMDP* uses the environmental type only, i.e. for a *POMDP* $\mathcal{M} = \langle \pi, A, B \rangle$ we have that $A(o_2, e) = A(o_1, e)$ for all infestation types $o_1, o_2 \in P$ and every environmental type $e \in E$. The discussion on advantages and weaknesses of both approaches can be found in Section 9.5.

**Model extraction** We do not assume to know the internal structure of the system. Hence, to use the underlying dependencies and hidden dynamics we need to extract them directly from the data. We achieve this goal by using *Data Mining* techniques and *Machine Learning* algorithms.

**Environmental and infestation types** Since our model, in a form of a *POMDP*, works with fine sets, we need to transform a possibly infinite sets of environmental conditions and observations into a finite discrete sets of environmental types $E$ and infestation types $P$, respectively.

We use the standard clustering methods to extract the environmental types present in our dataset. More specifically, we use the *k-centroids algorithm*, cf. e.g. [29], with the *euclidean distance* measure to generate a partition of the environment and use the resulting clusters as environmental types $e \in E$. Similarly, we use the $k$-centroids algorithm with the *Manhattan distance* to extract infestation types $o \in P$ from the data describing the quantity of pests. Each infestation type $o \in P$ is represented by the smallest convex area containing all points from the associated cluster. We consider this as an alternative to the *Voronoi diagrams*.

With obtained sets, we associate two classification functions $Env : N \times H \mapsto E$ and $Obs : \mathbb{N}^2_+ \mapsto P$, transforming environmental conditions into environmental types and pest numbers into infestation types, respectively. Those functions can be obtained as a result of the standard classification algorithms, in particular we use the *C4.5* algorithm [48].

**Model teaching** To teach a *POMDP*, we use a naive extension of the *Baum-Welch* (abbr., BW) algorithm which is a variation of the general *Expectation Maximisation* (abbr., EM)

procedure, for the EM procedure see e.g. [14]. A standard version of the Baum-Welch algorithm is used to teach *hidden Markov models*, see for example [58]. We modify the procedure in a way that it uses different transition and emission matrices, depending on the environmental conditions.

Since we use a naive modification of the *BW*, both the learning rate and the convergence properties are not well specified. Nevertheless, since the general *EM* procedure does not guarantee finding a global maximum, none of its variants can.The appealing factors of our approach are: iterative improvement procedure, intuitive soundness and relatively simple implementation.

Intuitively, given a sequence $s = \langle e_i, o_i \rangle_{i=0}^{N} \in (E \times P)^*$ we are searching for a model that maximises the accuracy of the forecast $\mathbb{P}(o_0 o_1 \cdots o_N | \mathcal{M}, s)$.

To measure the accuracy of a *POMDP* $\mathcal{M}$ on a sequence $s = \langle e_i, o_i \rangle_{i=0}^{N} \in (E \times P)^*$ we use the function

$$acc(\mathcal{M}, s) = \log_{||P||}(\mathbb{P}(o_0 o_1 \cdot o_N | s, \mathcal{M})) \tag{9.1}$$

where $\mathbb{P}(o_0 \cdots o_N | o_1 \cdots o_N, e_1 \cdots e_N, \mathcal{M})$ is the probability that $\mathcal{M}$ will emit sequence $o_0 \cdots o_N$ given the sequence of pairs $\langle e_i, o_i \rangle_{i=0}^{N}$, and $P$ is the set of possible infestation levels.

## 9.3 Game definition

In the data presented in Section 9.1 we can observe an interaction between *Nature*, *Human*, and *Pests* in an experimental fruit farm. In previous section, we have described how to create and teach a discrete stochastic model. Now, we will present how we can use such a system to predict future pest infestation levels and how to give strategy recommendations based on game-theoretic methods.

Since the only data regarding pests activity – their quantity – is exactly our prediction goal, the farm is to be seen as an arena witnessing the interactions between *N*ature and *H*uman farmers, i.e. the internal behaviour of insects $P$ will be completely embedded in the created model of the system.

Intuitively, players Human and Nature play in turns enumerated by natural numbers $i \in \mathbb{N}$. Each turn corresponds to a period of one week in which Nature chooses the environmental conditions $n_i$ and, then, Human prepares countermeasures in form of chemical treatments $h_i$. Consequently, players' actions result in a sequence $w \in E^{\omega}$ of environmental types and the final outcome, the play, is the sequence $\mathcal{M}(w)$ of distributions over the levels of infestation on which we evaluate the pay-off function defining our objectives.

More formally, the game is a tuple $G = \langle N, H, P, E, Env, \mathcal{M}, \Phi \rangle$ where

- $N$ is a set of Nature actions,
- $H$ is a set of Human actions,

- $P$ is a set of possible pest infestation types, called observations,
- $E$ is a set of environmental types,
- $Env : H \times N \to E$ is a function matching players actions into appropriate environmental types $E$,
- $\mathcal{M} = \langle \pi, A, B \rangle$ is a *POMDP*, with $dom(B) = E$ and $dom(A) = E \times P$,
- and $\Phi : P^\omega \to [0, 1]$ is a Borel pay-off function that describes the objectives.

Game $G$ is played in rounds and the notions of strategies and game value carry naturally from notions defined for games on graphs (see Section 3.3).

In our setting, standard definitions are interpreted in the following manner. Any play is a word $u = \langle n_i, h_i \rangle_{i=0}^{\omega} \in (N \times H)^\omega$. It defines a sequence of environmental types $w = \langle Env(n_i, h_i) \rangle_{i=0}^{\omega}$, and a sequence of infestation types distributions $\langle d_i \rangle_{i=0}^{\omega} = \mathcal{M}(w)$, where $d_i \in dist(O)$ for all $i > 0$. Last sequence, in a natural way, defines measure $\mu^u$ over the set of all possible observation sequences $O^\omega$. A visualization of a play can be seen in Table (9.2).

| *Round* | 1 | 2 | ... | $i$ | ... |
|---|---|---|---|---|---|
| $H$ | $h_1$ | $h_2$ | ... | $h_i$ | ... |
| $N$ | $n_1$ | $n_2$ | ... | $n_i$ | ... |
| $R$ | $Env(n_1, h_1)$ | $Env(n_2, h_2)$ | ... | $Env(n_i, h_i)$ | ... |
| *Obs* | $d_1(e)$ | $d_2(e)$ | ... | $d_i(e)$ | ... |

(9.2)

The set of Human's strategies is the set $\Sigma = (N \times H)^* \times N \to H$, and $\Pi = (N \times H)^* \to N$ is the set of Nature's strategies. The unique play defined by a pair of strategies $\sigma \in \Sigma$ and $\pi \in \Pi$ is denoted $u_{\sigma,\pi}$ and the value of a play $u \in (N \times H)^\omega$ is defined as $val(G, u) = \int_{O^\omega} \Phi d\mu^u$. Since $\Phi$ is Borel, the value of the game exists and is defined by the following equation.

$$val(G, \Phi) = \sup_{\sigma \in \Sigma} \inf_{\pi \in \Pi} val(G, u_{\sigma,\pi}) = \inf_{\pi \in \Pi} \sup_{\sigma \in \Sigma} val(G, u_{\sigma,\pi}) \tag{9.3}$$

With the definitions in place, a careful reader will notice a small discrepancy between the learning phase and the game phase of our procedure. In the learning phase we utilize a sequence $w = \langle e_i, o_i \rangle \in (E \times P)^\omega$ to find a model that is, in essence, an approximation of a POMDP faithful to $w$, while in the game phase we use a sequence $w^* = \langle e_i \rangle \in E^\omega$ to compute the sequence of observations. This approach is consistent by Observation 9.2.1.

### 9.3.1 Objectives

In general, there are two types of objectives we want to consider: infinite- and finite-horizon objectives. Intuitively, infinite-horizon objectives ask whether a certain situation can occur

somewhere in the future, whereas finite horizon ask what will happen in a fixed number of days.

Since a *POMDP* is essentially a probabilistic transducer, deciding whether one can achieve infinite horizon objectives, e.g. safety or reachability, is undecidable in general, see e.g. [10]. On the other hand, finite time objectives are decidable: we need to forecast a fixed number of rounds only.

The most basic problem we consider is *simulation*, which is essentially a strategy evaluation in a controlled environment. Since, in the real world, we cannot predict the environmental conditions with certainty, we need to be able to estimate the expected value of randomised strategies.

**Problem 9.3.1** (Forecast simulation)**.** *Given a game $G$, a set of forbidden observations $F \subseteq O$, two natural numbers $k, l$, a sequence of human actions $h_0, h_1, \ldots, h_{l-1} \in H^l$, and a forecast $n_1, n_2, \ldots n_{l-1} \in dist(N)^l$ in a form of a sequence of discrete distributions with finite support compute the probability that system will not emit forbidden observation between rounds $k$ and $l$, i.e. compute the value of the game with pay-off function*

$$\Phi(o_1 o_2 \cdots o_i \cdots) \stackrel{def}{=} \begin{cases} 1 & \text{if } o_i \notin F \text{ for } k \leq i \leq l \\ 0 & \text{otherwise.} \end{cases}$$

We call the above problem simulation because it does not take into account the interaction between players and tracks the evolution of the system only. To include the interactive nature of our model, we use two problems that ask about the existence of certain strategies.

Naturally, the use of chemicals should reduce the infestation levels, in near future[2]. Therefore, with an unrestricted set of human strategies, an optimal strategy for player $H$ is to perform treatment every round. On the other hand, toxicity, plant degradation, or fruit poisoning are a real concern in modern agriculture and require introducing constriction on the dosages and frequency of the chemical treatments. Therefore, for the real life applications we consider a version with the sets of strategies that restrict the number of chemical treatments.

**Problem 9.3.2** (Forecast recommendations)**.** *Given a game $G$, a fixed limit $t \in \mathbb{N}$ of the crucial actions from the set $A$, two natural numbers $k, l$, a forecast $n_1, n_2, \ldots n_{l-1} \in dist(N)^l$ in a form of a sequence of discrete distributions with finite support, and a threshold $c \in \mathbb{R}$, decide whether there is a strategy $\sigma \in \Sigma$ that uses at most $t$ crucial actions, and for which the probability that system does not emit observations from the set $F \subseteq O$ between the rounds $k$*

[2]We have no information concerning the overuse of the chemicals

*and $l$ is greater than $c$, i.e. does $val(G) \geq c$ hold with the pay-off function*

$$\Phi(o_1 o_2 \cdots o_i \cdots) := \begin{cases} 1 & \text{if } o_i \notin F \text{ for } k \leq i \leq l, \\ 0 & \text{otherwise.} \end{cases}$$

*and restricted set of player $H$ strategies $\Sigma_t$.*

Next problem asks about the existence of an optimal strategy with no information regarding the behaviour of the environment.

**Problem 9.3.3** (Advisor)**.** *Given a game $G$, a fixed limit $t \in \mathbb{N}$ of crucial actions from the set $A$, two natural numbers $k, l$, and a threshold $c \in \mathbb{R}$ decide whether there is a strategy $\sigma \in \Sigma$ that uses at most $t$ crucial actions and for which the probability that system does not emit observations from the set $F \subseteq O$ between the rounds $k$ and $l$ is greater than $c$, i.e. does $val(G) \geq c$ hold with the pay-off function*

$$\Phi(o_1 o_2 \cdots o_i \cdots) := \begin{cases} 1 & \text{if } o_i \notin F \text{ for } k \leq i \leq l, \\ 0 & \text{otherwise.} \end{cases}$$

*and restricted set of player $H$ strategies $\Sigma_t$.*

Finally, we consider the infinite-horizon *Safety* problem which ask whether there is a strategy that prevents certain levels of infestation.

**Problem 9.3.4** (Safety)**.** *Given a game $G$, a set of infestation levels $F \subseteq O$, and a threshold $c \in [0, 1]$, does $val(G) \geq c$ hold with the pay-off function*

$$\Phi(o_1 o_2 \cdots o_i \cdots) := \begin{cases} 1 & \text{if } o_i \notin F \text{ for } i \geq 0, \\ 0 & \text{otherwise.} \end{cases}$$

The complexity of the above problems varies, from polynomial to recursively enumerable, depending on the problem and the form of the input, see e.g. [10].

## 9.4 Use of the plantation game

The complexity follows directly from the choice of the model that we have used to mimic the changes in the system. Careful reader can notice immediately that our four main problems can be seen as variations of classical reachability problems for POMDPs.

As stated before the *Forecast simulation* problem can be solved via simple dynamic programming as shown in Algorith 3.

To solve the advisor problem we use the following observation.

**Algorithm 3** CALCULATESIMULATION

**Require:** $n \in dist(N)^l, h \in H^l$
  //$de$[i] – distribution of environmental types in round i
  //$do$[e][i] – distribution of infestation levels in round i, given environmental type $e$
  //$fo$[i] – distribution of infestation levels in round i
  $\pi[0] := \mathcal{M}.\pi$
  **for** $(i := 0;\ i \leq l;\ i := i + 1)$ **do**
    **for** $e \in E$ **do**
      $de[i][e] := Env(n_i, h_i)$
      $do[e][i] := \mathcal{M}.B(e) \cdot \pi[i]$
    **end for**
    $\pi[i + 1] := \sum_{e \in E} de[i][e] \cdot (\sum_{o \in O} do[e][i] \cdot (\mathcal{M}.A(e, o) \cdot \pi[i]))$
    **for** $e \in E$ **do**
      $fo[i][o] := \sum_{e \in E} de[i][e] \cdot do[e][i][o]$
    **end for**
  **end for**

**Observation 9.4.1.** *For every game $G$ there is at most $|H||E|$ essentially different actions that player $\mathcal{N}$ can perform each cycle.*

Indeed, let $r \subseteq N \times N$ be a relation, such that $\langle n_1, n_2 \rangle \in r$ if, and only if, $Env(n_1, h) = Env(n_2, h)$ for every action $h \in H$. Naturally, $r$ is an equivalence relation of index smaller than $|H||E|$.

The above observation implies that instead of working with an infinite space of possible actions, we can limit our attention to a finite case. If we are able to compute the classes of equivalence relation $r$, this reduces the problem to a finite-horizon problems for POMDPs. In other words, according to the above observations, both the lower and the upper complexity bounds follow from extensive work surveyed in [39].

As for the last problem, we recall the following result.

**Proposition 9.4.2.** *The Safety problem is undecidable even when the threshold $c$ is 1.*

The undecidability follows directly from the undecidability of the non-emptiness of a *probabilistic automaton* (cf. e.g. [20]).

### 9.4.1 Visible game definition

When using a *POMDP* as a model, we almost immediately encounter undecidability: every reasonable infinite-horizon objective is undecidable even for threshold 1. This seems to be the result of working with the unknown internal state of the system. Therefore, in order to verify infinite horizon properties we need to use a simplified model in which internal state is visible.

It is well-known that for stochastic games the border of decidability lies between partial and perfect information games, see e.g. [10].

Now, we propose a simplified version of the game. The simplified game starts with a token representing the state of the system placed in the initial observed state $o_I \in O$. Afterwards, players $H$ and $N$ in turns enumerated by natural numbers $i \in \mathbb{N}$ choose their actions. Each turn corresponds to a period of one week in which $N$ chooses the environmental conditions $n_i$ and $H$ prepares countermeasures in form of chemical treatments $h_i$. After players' actions are chosen the token is moved accordingly to the transition function $\delta$ and environmental type $Env(h_i, n_i)$.

More formally, the game is a tuple $G = \langle N, H, O, E, o_I, Env, \delta, \Phi \rangle$ where

- $N$ is the set of Nature actions,
- $H$ is the set of Human actions,
- $O$ is the set of possible pest infestation levels,
- $E$ is the set of environmental types,
- $o_I$ is the initial observed state of the system,
- $Env\colon H \times N \to E$ is a function matching players actions to appropriate environmental types belonging to the set $E$,
- $\delta\colon O \times E \to dist(O)$ is a transition function,
- and $\Phi\colon O^\omega \to [0, 1]$ is a Borel pay-off function that describes the objectives.

Game is played in rounds and the notions of strategies and game value, again, carry naturally from the games on graphs. In this setting, the standard definitions are interpreted in the following manner. A play is the path $u \in O^\omega$ that the token took. A visualization of a play can be seen in Table (9.4).

| *Round* | 1 | 2 | ... | $i$ | ... |
|---|---|---|---|---|---|
| $H$ | $h_1$ | $h_2$ | ... | $h_i$ | ... |
| $N$ | $n_1$ | $n_2$ | ... | $n_i$ | ... |
| $R$ | $Env(n_1, h_1)$ | $Env(n_2, h_2)$ | ... | $Env(n_i, h_i)$ | ... |
| $Obs$ | $u_1$ | $u_2$ | ... | $u_i$ | ... |

(9.4)

The set of strategies of player $H$ is the set $\Sigma_S = (O)^* \cdot N \to H$, and the set of player $N$ strategies is the set $\Pi = (O)^* \to N$.

A pair of strategies $\sigma \in \Sigma$ and $\pi \in \Pi$ in a natural way defines a measure $\mu^{\sigma,\pi}$ on the set $O^\omega$. Since $\Phi$ is Borel, the value of the game exists and is defined by the following equation.

$$val(G, \Phi) = \sup_{\sigma \in \Sigma} \inf_{\pi \in \Pi} \int_{O^\omega} \Phi(o) d\mu^{\sigma,\pi} = \inf_{\pi \in \Pi} \sup_{\sigma \in \Sigma} \int_{O^\omega} \Phi(o) d\mu^{\sigma,\pi}. \tag{9.5}$$

It is not hard to notice that game $G$ is, essentially, a simple stochastic game, see Section 2.3.5 on page 23, and, thus, the appropriate versions of the problems defined in Section 9.3.1 are computable.

For completeness, given a training data represented by a sequence $\langle e_i, o_i \rangle \in (E \times O)^*$, we define the transition function $\delta\colon O \times E \to \mathit{dist}(O)$ used in the simplified version of the game in the following way.

$$\delta(o,e)(\hat{o}) = \begin{cases} \frac{||\{i \in \mathbb{N} : o_i = o, e_i = e, o_{i+1} = \hat{o}\}||}{||\{i \in \mathbb{N} : o_i = o, e_i = e\}||} & \text{if } ||\{i \in \mathbb{N} : o_i = o, e_i = e\}|| > 0 \\ \delta_{o,\hat{o}} & \text{otherwise} \end{cases} \tag{9.6}$$

The function $\delta_{i,j}$ is the Kronecker delta.

## 9.5 Experimental results

We have divided our time series into two sets: the training set that consists of first 300 measurements and the test set constructed from the remaining 42 measurements.

Using the methods described in Section 9.2, we have taught a number of models with the number of internal states ranging between 4 and 15, using

- a fixed set of $e = 6$ environmental types,
- 4 sets of infestation levels, with $6, 7, 8$, and 9 different types of infestation levels, respectively,
- and both observation sensitive and observation oblivious semantics.

The diagrams of used infestation levels can be seen in Figure 9.1. The result of the teaching algorithms can be seen in Figure 9.2 which contains the accuracies of the models computed on the training set.

In the subsequent rows of the table showed in Figure (9.2), we have the number of internal states and the accuracy values with respect to the number of different infestation levels used in the teaching. The columns marked with the asterisk ($^*$) use the observation oblivious semantics; the remaining columns use the observation sensitive semantics. The first row of the table contains the benchmark values, i.e. the values that a model that emits infestation levels with the uniform probability achieves. The subsequent rows contain values that were achieved by the models obtained from our teaching algorithm.

As expected, with the increase of the number of internal states of the used *POMDP*, the accuracy of the forecast rises. The inconsistencies in the growth are, most probably, a consequence of the local nature of the learning algorithm, i.e. the of the fact that the algorithm can return a local optimum which is not necessarily a global optimum. Similarly, we observe that the increase of the number of possible infestation types results in a lower

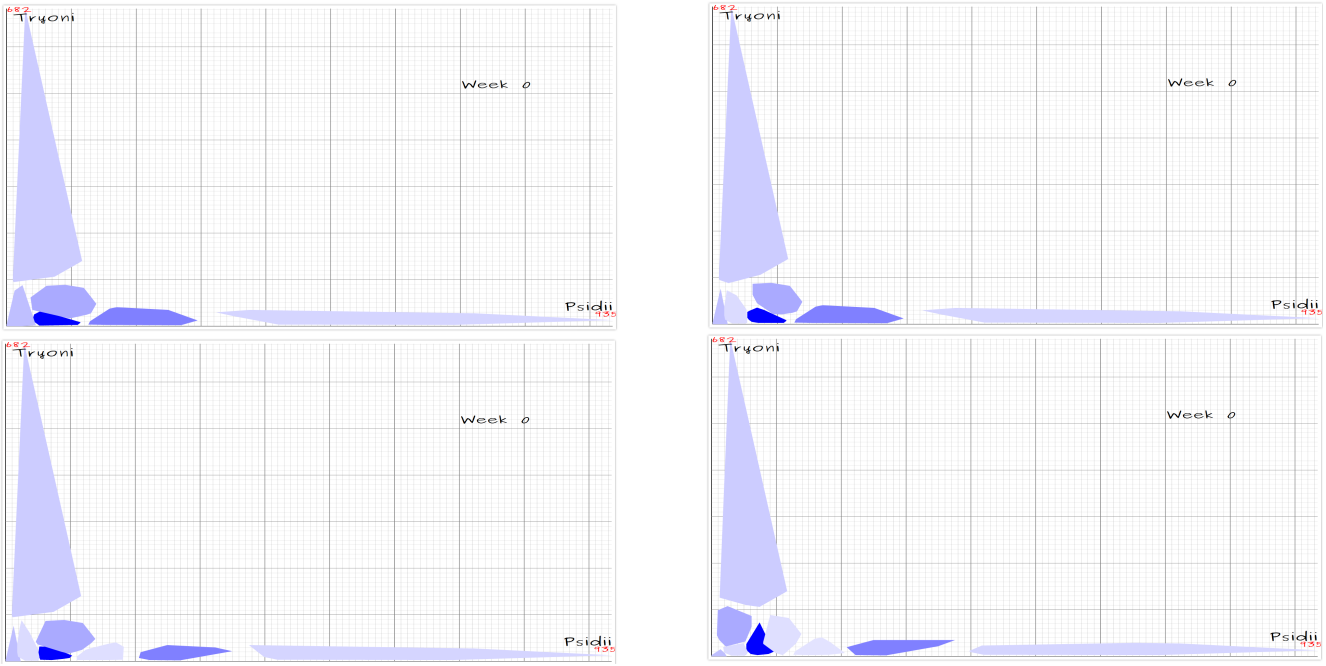



Figure 9.1 – Infestation levels used in the experiments.

accuracy of predictions. This, most probably, is a consequence of the small size of the dataset.

An interesting property of our approach can be drawn form the comparison of the accuracies of the *observation sensitive* and the *observation oblivious* models. While with low number of different infestation levels using the *observation sensitive* models increases the quality of teaching, high number of clusters reverses this property. We attribute this behaviour to the relation between the number of the independent parameters defining the model and the amount of available data. Of course, greater number of parameters allows to describe more complex dependencies, but it also reduces the number of available data per parameter. With $o$ different infestation levels, $e$ environmental types, and $q$ internal states an *observation sensitive* model uses $o \cdot e$ transition matrices of size $q^2$ and $e$ emission matrices of size $q \cdot o$, whereas a *observation oblivious* model uses only $e$ transition matrices of size $q^2$ and $e$ emission matrices of size $q \cdot o$. In other words, with $o = 9$ we have on average 9 times less data entries per matrix. In our experiment, this translates to difference between 50 and 5.56 training entry per transition matrix. Less training data results in worse parameter estimation and is not compensated by the rise of the expressive power of the model. On the other hand, when more training data is available the observation sensitive semantics should perform better than the observation oblivious semantics.

Moreover, since the clusterings of the pest quantities and the environmental conditions are done separately, it is possible that certain pairs of the infestation levels and of the en-

vironmental types are not present in the training data, and, thus, some of the transition matrices present in an observation sensitive model are not subjected to the teaching algorithm.[3] This can negatively influence the accuracy of the forecast performed on the test data, if some of the missing pairs are present in the test data, but not in the training data.

[3]Notice that every environmental type is present in the training data and, thus, this observation does not impact observation oblivious models.

| | 6 | $6^*$ | 7 | $7^*$ | 8 | $8^*$ | 9 | $9^*$ | Number of clusters |
|---|---|---|---|---|---|---|---|---|---|
| *base* | −300.0 | −300.0 | −300.0 | −300.0 | −300.0 | −300.0 | −300.0 | −300.0 | |
| 4 | −137.355 | −193.523 | −160.645 | −206.515 | −245.267 | −215.816 | −254.759 | −223.447 | |
| 5 | −130.953 | −191.449 | −150.875 | −204.058 | −249.847 | −205.735 | −251.889 | −215.211 | |
| 6 | −136.874 | −192.433 | −145.254 | −192.314 | −246.22 | −199.919 | −256.453 | −211.717 | |
| 7 | −126.404 | −179.869 | −133.513 | −191.579 | −247.775 | −198.515 | −254.8 | −204.212 | |
| 8 | −118.521 | −184.468 | −139.493 | −188.199 | −249.266 | −192.919 | −254.545 | −203.356 | |
| 9 | −123.575 | −181.93 | −140.341 | −183.608 | −249.512 | −191.406 | −253.077 | −203.251 | |
| 10 | −121.137 | −171.072 | −141.067 | −187.258 | −246.39 | −191.474 | −255.381 | −197.862 | |
| 11 | −118.045 | −172.56 | −140.666 | −184.119 | −249.012 | −187.848 | −254.65 | −198.524 | |
| 12 | −116.387 | −169.166 | −137.933 | −188.075 | −249.928 | −186.366 | −256.422 | −191.536 | |
| 13 | −119.667 | −174.54 | −133.172 | −186.587 | −247.615 | −184.768 | −255.987 | −191.382 | |
| 14 | −117.935 | −173.364 | −135.511 | −179.506 | −249.794 | −190.388 | −254.905 | −197.14 | |
| 15 | −118.575 | −172.955 | −131.855 | −173.512 | −247.585 | −190.826 | −254.677 | −192.216 | |
| States | | | | | | | | | |

Figure 9.2 – A table presenting achieved levels of accuracy when teaching a model.

# Chapter 10

# Conclusions and future work

In the theoretical part of the thesis we have studied the properties of regular branching games. We can differentiate three groups of results. The first one concerns the determinacy of branching games, the second one the computational complexity of computing game values, and the last one the computational complexity of computing the measures of regular sets of trees.

**Determinacy** In the case of determinacy we have shown that regular branching games with open (closed) sets are determined under mixed strategies, see Theorem 6.1.8. This is the limit in the terms of topological hierarchy of sets as we have also presented an example of a regular branching game which is not determined under mixed strategies, see Example 6.1.3. This game has a winning set that is a difference of two open sets, placing it at the level of Boolean combinations of open sets.

In the case of both pure and behavioural strategies even clopen sets or sets of trees of bounded depth do not guarantee determinacy. This is showcased in Example 3.2.2, which combines the classic game of "Matching Pennies" with the observation of Mio, see [33, Example 4.1.18]. This example stays contrary to Nash-like results in the perfect-information games with perfect recall, which state that finite duration games with finite set of actions are determined under behavioural strategies.

Still, those results do not characterise the classes of winning sets that guarantee determinacy. Indeed, in Chapter 7 we show that branching games with game automata definable winning sets are determined under pure strategies, see Corollary 7.1.2. Therefore, we think that it would be interesting to find new types of families of winning sets that guarantee determinacy.

**Computing game values** We have solved the general problem of computing values of regular branching games. In particular, we have shown that

- there is no algorithm that given a branching game with an arbitrary regular winning set computes any of the game's partial values, see Theorem 6.2.1;

- there is no algorithm that given a non-stochastic branching game with an arbitrary regular winning set computes any of the behavioural or mixed values, see Corollary 6.3.12;
- there is an algorithm that given a finite non-stochastic branching game with a regular winning set computes all of the pure values, see Theorem 5.3.2.

The exact computational complexity of the algorithm in the last bullet depends on the kinds of vertices on the board and the representation of the winning set.

The negative results are not necessarily surprising: endowing systems of imperfect information with probability often leads to undecidability, e.g. probabilistic automata [49]. The positive results give hope to find classes of games with computable partial values.

While the above results answer the question of computability in the general case, a smart restriction on the set of possible winning sets may yield a class of branching games with computable values. An example of such a class are branching games with winning sets defined by game automata, see Chapter 7, for which the values coincide and are computable in elementary time. Another promising class of regular winning sets may arise not from syntactic restrictions on automata, but by restrictions on the expressive power of monadic second-order logic. The class of special interest are Boolean combinations of conjunctive queries, for the reason mentioned in the following paragraph.

**Measures** In Chapter 8 we have tackled the problem of computing the standard measure of a given regular set of trees. We have shown that

- there is an algorithm that given a first-order formula $\varphi$ not using the descendant relation computes the standard measure of the set $\mathrm{L}(\varphi)$, see Theorem 8.2.1;
- there is an algorithm that given a Boolean combination of conjunctive queries $\varphi$ computes the standard measure of the set $\mathrm{L}(\varphi)$, see Theorem 8.4.1.

The involved techniques use the notion of locality and cannot be extended to the full power of monadic second-order logic. Even the full first-order logic is not captured in the scope of those results.

An obvious direction of future research is to try to find algorithms computing the measure of any arbitrary regular set or at least for a set definable by some logic subsumed by monadic second-order logic, e.g. *CTL*, or *modal $\mu$ calculus*. A less obvious direction of research would be to extend the techniques presented in this thesis to arbitrary measures generated by graphs[1].

**Applications** In Chapter 9 we have presented a generic way of using machine learning methods and game theory to create a simple advisor taught on a time series describing

[1] For instance, a measure generated by a graph can be defined in terms of behavioural strategies.

a closed ecosystem. The presented approach is simple yet robust, allowing easy exchange of used tools and techniques.

In this particular case we have used Markov decision processes and the Baum-Welch procedure to create a stochastic two-player game that represents a fruit farm. This game can be used to predict the presence of pests, in the form of fruit flies and to plan chemical treatments that will help with the management of the population of the flies.

The provided data did not allow us to use the rich theory we have developed in the study of branching games. Still, we think that designing and developing tools which incorporate branching games would be an interesting direction of future research. In our opinion, the systems that would benefit the most from the branching games representation are those, where two adversaries oversee a number of independent, non-communicating agents: e.g. virus outbreak, or breeding bacteria.

# Bibliography


[1] Rajeev Alur, Salvatore La Torre, and Parthasarathy Madhusudan. Playing games with boxes and diamonds. In *CONCUR*, pages 128–143. Springer Berlin Heidelberg, 2003.

[2] James Bergin and W. Bentley MacLeod. Continuous time repeated games. *International Economic Review*, 34(1):21–37, 1993.

[3] Henrik Bjørklund, Wim Martens, and Thomas Schwentick. Conjunctive query containment over trees. *Journal of Computer and System Sciences*, 77(3):450 – 472, 2011. Database Theory.

[4] Tomás Brázdil, Petr Jancar, and Antonín Kucera. Reachability games on extended vector addition systems with states. In *ICALP*, pages 478–489, 2010.

[5] Florian Bruse, Michael Falk, and Martin Lange. The fixpoint-iteration algorithm for parity games. In *GandALF 2014*, pages 116–130, 2014.

[6] Julius Richard Büchi and Lawrence H. Landweber. Solving sequential conditions by finite-state strategies. *Transactions of the American Mathematical Society*, 138:295–311, 1969.

[7] Lucas Bunt, Philip Jones, and Jack Bedient. *The Historical Roots of Elementary Mathematics*. Dover Books Explaining Science. Dover Publications, 1988.

[8] Cristian S. Calude, Sanjay Jain, Bakhadyr Khoussainov, Wei Li, and Frank Stephan. Deciding parity games in quasipolynomial time. In *STOC*, pages 252–263. ACM, 2017.

[9] Krishnendu Chatterjee and Nathanaël Fijalkow. A reduction from parity games to simple stochastic games. In *GandALF*, pages 74–86, 2011.

[10] Krishnendu Chatterjee and Thomas A. Henzinger. A survey of stochastic $\omega$-regular games. *J. Comput. Syst. Sci.*, 78(2):394–413, 2012.

[11] Anne Condon. The complexity of stochastic games. *Information and Computation*, 96:203–224, 1992.

[12] Laure Daviaud, Marcin Jurdziński, and Ranko Lazic. A pseudo-quasi-polynomial algorithm for mean-payoff parity games. In *LICS 2018*, pages 325–334, 2018.

[13] Anuj Dawar, Martin Grohe, Stephan Kreutzer, and Nicole Schweikardt. Model theory makes formulas large. In *ICALP*, pages 913–924, 2007.

[14] Arthur Dempster, Nan Laird, and Donald Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[15] Jacques Duparc, Alessandro Facchini, and Filip Murlak. Definable operations on weakly recognizable sets of trees. In *FSTTCS*, pages 363–374, 2011.

[16] Allen Emerson and Charanjit Jutla. Tree automata, mu-calculus and determinacy. In *FOCS*, pages 368–377, 1991.

[17] Alessandro Facchini, Filip Murlak, and Michał Skrzypczak. Index problems for game automata. *ACM Trans. Comput. Log.*, 17(4):24:1–24:38, 2016.

[18] Haim Gaifman. On local and non-local properties. In *Proceedings of the Herbrand Symposium*, pages 105 – 135. Elsevier, 1982.

[19] David Gale and Frank M. Stewart. Infinite games with perfect information. In *Contributions to the theory of games*, volume 2 of *Annals of Mathematics Studies, no. 28*, pages 245–266. Princeton University Press, 1953.

[20] Hugo Gimbert and Youssouf Oualhadj. Probabilistic automata on finite words: Decidable and undecidable problems. In *ICALP (2)*, pages 527–538, 2010.

[21] Irving Leonard Glicksberg. *Minimax Theorem for Upper and Lower Semi-continuous Payoffs*. Memorandum (Rand Corporation). Rand Corporation, 1950.

[22] Tomasz Gogacz, Henryk Michalewski, Matteo Mio, and Michał Skrzypczak. Measure properties of regular sets of trees. *Inf. Comput.*, 256:108–130, 2017.

[23] Marcin Jurdziński, Mike Paterson, and Uri Zwick. A deterministic subexponential algorithm for solving parity games. *SIAM J. Comput.*, 38(4):1519–1532, 2008.

[24] Juris Hartmanis and Richard Stearns. On the computational complexity of algorithms. *Transactions of the American Mathematical Society*, 1965.

[25] Alexander Kechris. *Classical descriptive set theory*. Springer-Verlag, New York, 1995.

[26] Eryk Kopczyński and Damian Niwiński. A simple indeterminate infinite game. In *Logic, Computation, Hierarchies*, pages 205–212. De Gruyter, 2014.

[27] Thomas Kuhn. Extensive games and the problem of information,. In *Annals of Mathematics Study 28*, pages 193–216. Princeton Univ. Press, Princeton, N.J., 1953.

[28] Karoliina Lehtinen. A modal $\mu$ perspective on solving parity games in quasi-polynomial time. In *LICS*, pages 639–648. ACM, 2018.

[29] James MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pages 281–297. University of California Press, 1967.

[30] Donald A. Martin. Borel determinacy. *Annals of Mathematics*, 102(2):363–371, 1975.

[31] Donald A. Martin. The determinacy of Blackwell games. *The Journal of Symbolic Logic*, 63(4):1565–1581, 1998.

[32] Henryk Michalewski and Matteo Mio. On the problem of computing the probability of regular sets of trees. In *FSTTCS*, pages 489–502, 2015.

[33] Matteo Mio. *Game Semantics for Probabilistic mu-Calculi*. PhD thesis, University of Edinburgh, 2012.

[34] Matteo Mio. On the equivalence of game and denotational semantics for the probabilistic mu-calculus. *Logical Methods in Computer Science*, 8(2), 2012.

[35] Matteo Mio. Probabilistic modal $\mu$-calculus with independent product. *Logical Methods in Computer Science*, Volume 8, Issue 4, November 2012.

[36] Matteo Mio and Alex Simpson. Łukasiewicz mu-calculus. In *FICS*, pages 87–104, 2013.

[37] Matteo Mio and Alex Simpson. Łukasiewicz $\mu$-calculus. *Fundam. Inform.*, 150(3-4):317–346, 2017.

[38] Andrzej W. Mostowski. Games with forbidden positions. Technical report, University of Gdańsk, 1991.

[39] Martin Mundhenk, Judy Goldsmith, Christopher Lusena, and Eric Allender. Complexity of finite-horizon Markov decision process problems, 2000.

[40] Filip Murlak, Michał Ogiński, and Marcin Przybyłko. Between tree patterns and conjunctive queries: Is there tractability beyond acyclicity? In *MFCS*, pages 705–717, 2012.

[41] John Nash. Non-cooperative games. *Annals of Mathematics*, 54(2):286–295, 1951.

[42] Christos H. Papadimitriou. *Computational complexity*. Addison-Wesley, 1994.

[43] Amir Pnueli and Roni Rosner. On the synthesis of a reactive module. In *Symposium on Principles of Programming Languages*, pages 179–190. ACM, 1989.

[44] Andreas Potthoff. *Logische Klassifizierung regulärer Baumsprachen*. PhD thesis, Universität Kiel, 1994.

[45] Marcin Przybyłko. Tree games with regular objectives. In *GandALF*, pages 231–244, 2014.

[46] Marcin Przybyłko. On computing the measures of first-order definable sets of trees. In *GandALF*, pages 206–219, 2018.

[47] Marcin Przybyłko and Michał Skrzypczak. On the complexity of branching games with regular conditions. In *MFCS 2016*, pages 78:1–78:14, 2016.

[48] J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[49] Michael Rabin. Probabilistic automata. *Information and Control*, 6(3):230 – 245, 1963.

[50] Michael Rabin. Decidability of second-order theories and automata on infinite trees. *Transactions of the American Mathematical Society*, 141:1–35, 1969.

[51] Samuel Safra. On the complexity of omega-automata. In *FOCS*, pages 319–327, 1988.

[52] Alfred Tarski. *A Decision Method for Elementary Algebra and Geometry*. University of California Press, 1951.

[53] Wolfgang Thomas. Finite-state strategies in regular infinite games. In *FSTTCS*, pages 149–158, 1994.

[54] Wolfgang Thomas. Languages, automata, and logic. In *Handbook of Formal Languages*, pages 389–455. Springer, 1996.

[55] Wolfgang Thomas and Helmut Lescow. Logical specifications of infinite computations. In *REX School/Symposium*, pages 583–621, 1993.

[56] Moshe Vardi. An automata-theoretic approach to linear temporal logic. In *Logics for Concurrency*, pages 238–266. Springer Berlin Heidelberg, 1996.

[57] John von Neumann and Oskar Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1944.

[58] Lloyd Welch. Hidden Markov Models and the Baum-Welch Algorithm. *IEEE Information Theory Society Newsletter*, 53(4), December 2003.